

System Administration Volume 2



0890430-100
August 2003

Copyright 2002 by Concurrent Computer Corporation. All rights reserved. This publication or any part thereof is intended for use with Concurrent Computer Corporation products by Concurrent personnel, customers, and end-users. It may not be reproduced in any form without the written permission of the publisher.

The information contained in this document is believed to be correct at the time of publication. It is subject to change without notice. Concurrent Computer Corporation makes no warranties, expressed or implied, concerning the information contained in this document.

To report an error or comment on a specific portion of the manual, photocopy the page in question and mark the correction or comment on the copy. Mail the copy (and any additional comments) to Concurrent Computer Corporation, 2881 Gateway Drive Pompano Beach, FL 33069. Mark the envelope **“Attention: Publications Department.”** This publication may not be reproduced for any other reason in any form without written permission of the publisher.

UNIX is a registered trademark, licensed exclusively by X/Open Company Ltd.

Other products mentioned in this document are trademarks, registered trademarks, or trade names of the manufactures or marketers of the product with which the marks or names are associated.

PowerMAX OS, Night Hawk, PowerMAXION, and Power Hawk are trademarks of Concurrent Computer Corporation.

Printed in U. S. A.

Revision History:	Level:	Effective With:
Original Release -- July 28, 1994	000	OS Release 1.1
Previous Release -- October 2001	080	OS Release 5.1
Previous Release -- October 2002	090	SAR 631
Current Release -- August 2003	100	SAR 849

Volume 2 Contents

Chapter 1 Introduction to Basic Administration

How to Use This Book	1-1
New Administrators	1-1
Experienced Administrators	1-2
How This Book Is Organized	1-2
What's In Volume 1	1-2
What's In Volume 2	1-4
Related Documents	1-5
Introduction to System Administration	1-7
Administrative Interfaces	1-7
Administration Procedures	1-8
Set Up	1-8
Administrative Privileges	1-8
Booting the System	1-9
System States	1-9
Users and Groups	1-10
Peripherals (Terminals, printers, networks, media)	1-10
Terminals	1-11
Printers	1-11
Networks	1-11
Media (Hard Disks, Tapes and CD-ROM)	1-11
Communicating with Users	1-12
Message of the Day	1-12
The wall Command	1-12
News	1-12
Electronic Mail	1-13
Collecting Users' Requests	1-13
Notation Conventions Used in This Book	1-14
Related Documentations	1-16

Part 1 File System and Storage Device Administration

Chapter 2 Managing Storage Devices

What Are Storage Devices?	2-1
Types of Devices	2-1
Physical Types of Devices	2-1
Logical Types of Devices	2-2
Supporting Software for Devices	2-2
Device Security Mechanism	2-3
Basic Device Access Control	2-4
Device Security Attributes	2-4
Device Driver Flags	2-6
Device Allocation	2-7

Using admalloc on Devices	2-7
Device Allocation During the Login Process	2-9
Devices Requiring Special Processing	2-9
Logical and Secure Device Aliases	2-10
Device Attributes	2-11
Suggestions for Managing Storage Devices	2-11
Managing Disk Drives	2-12
Physical Disk Organization	2-12
Logical Disk Address	2-13
Disk Capacity	2-14
Physical File Systems	2-14
Devices	2-14
Disk Partitions	2-15
Cylinder Groups	2-16
Blocks and Fragments	2-17
Components of the Device	2-18
System Block	2-18
Geometry Block	2-18
Super Block	2-19
Cylinder Group	2-20
Mounting and Dismounting	2-24
Linking Files and Directories	2-24
Creating New File Systems	2-26
Checking the Integrity of the File Systems	2-27
Formatting and Partitioning SCSI Disks	2-27
Formatting and Partitioning a SCSI Disk	2-28
Formatting HSA Disks with the Console Processor	2-32
Management of Media Flaws for SCSI Disks	2-33
SCSI Disk Flaw Management	2-33
Procedure For Identifying Flaws	2-34
Reformatting SCSI Disk	2-34
The Device Database: Adding and Removing Storage Devices	2-35
Using putdev to Add Storage Devices	2-35
Removing Storage Devices	2-35
Removing a Non-Critical Device from Service	2-36
Maintaining Storage Devices and Media	2-39
Identifying Existing Devices	2-39
Conventions Used to Position a Device File	2-40
Obtaining /dev Directory Listings	2-40
Formatting Floppy Diskettes and Hard Disks	2-41
Displaying Information About Storage Devices	2-42
Copying Data on Storage Media	2-42
Copying Files from Disk to Disk	2-43
Copying the Contents of a Directory from Hard Disk to Tape	2-43
Copying Files from Tape to Hard Disk	2-43
Checking a Corrupt File System	2-43
Erasing Storage Media	2-44
Defining Disk Partitions	2-44
Maintaining the Device Database	2-45
Using putdev to Create a Device Entry in the DDB	2-46
Recommended Attributes and Default Values	2-46
Standard Attributes	2-47
Using getdev to List Devices	2-51
Creating Customized Device Lists	2-51

Specifying Devices for a Listing	2-52
Specifying Criteria for a Listing	2-52
Sample Requests for Customized Lists	2-52
Listing Attributes	2-53
Using putdev to Change the Values of Attributes	2-54
Deleting Attributes from a Device Entry	2-54
Removing a Device Entry	2-55
Managing Devices in Groups	2-55
The Device Groups Database	2-55
Using putdgrp to Create a Device Group	2-55
Using putdgrp to Remove a Device Group	2-56
Using getdgrp and listdgrp to Maintain the Device Groups Database	2-56
Using getdgrp to Get a List of Groups	2-56
Using listdgrp to Get a List of Device Group Members	2-58
Managing Device Reservations	2-58
Using devreserv and devfree in Device Reservation	2-59
Using devreserv to Check Device Reservation Status	2-59
Using devreserv to Reserve a Device	2-60
Using devfree to Free a Reserved Device	2-60
Device Names and Default Partitions	2-61
Disks	2-61
Disk Partitions	2-61
Tapes	2-62
Virtual Partition	2-62
Striped VP	2-62
Mirrored VP	2-63
VP Driver	2-63
Configuration	2-63
Geometry	2-64
Initialization	2-64
Device Naming Convention	2-65
Vpstat Command	2-65
/etc/vptab	2-66
Example VP Set-Up	2-66
Example Mirrored VP Set-Up	2-67
Initial mirrored VP set-up	2-67
Variable Steps for Mirrored VP Set-Up	2-67
Final Steps for Mirrored VP Set-Up	2-68
Restoring Faulty Member of a Mirrored VP	2-69
Quick Reference Guide to Managing Devices	2-69
Adding a new device to the system:	2-70
Copying files from tape to disk:	2-71
Removing a noncritical device:	2-71
Verifying the usability of a disk:	2-72
Creating a device entry:	2-72
Listing devices:	2-73
Listing device attributes:	2-73
Modifying a device entry:	2-73
Removing a device entry:	2-73
Deleting an attribute from a device entry:	2-74
Creating a device group:	2-74
Listing device groups:	2-74
Listing the members of a device group:	2-74
Modifying a device group:	2-74

- Removing a device group: 2-75
- Reserving a device: 2-75
- Freeing a reserved device: 2-75
- Checking device reservation status: 2-75
- Example Procedures on Adding Devices 2-76
 - Adding a Disk Drive 2-76
 - Adding a CD-ROM Drive 2-79
 - Adding a Tape Drive 2-80
 - Adding Ethernet/FDDI Controller Device(s) 2-82
 - Adding New Controller Device and Software for First Time 2-82
 - Adding New Controller Device - Software Previously Installed 2-83

Chapter 3 Managing File System Types

- What Is a UNIX File System? 3-1
- The Relationship between the File System and the Storage Device 3-3
 - Formatting the Storage Device 3-3
 - Partitions on the Storage Device 3-3
 - Size Limitations on a File System 3-3
 - Avoiding File System Damage 3-4
 - Administering File Systems and Data Integrity 3-4
 - Do You Suspect File System Damage? 3-5
- The ufs File System Type 3-5
 - The ufs Cylinder Group Map 3-5
 - The ufs Super-Block 3-6
 - ufs Inodes 3-6
 - What Does a ufs Inode Contain? 3-6
 - ufs Storage Blocks 3-9
 - ufs Free Blocks 3-9
 - Quotas on the ufs File System 3-10
 - Using Quotas 3-10
 - What Effect Do Quotas Have on the User? 3-11
 - Time and Space Optimization 3-12
- The xfs File System Type 3-13
 - xfs File System Features 3-13
 - Fast File System Recovery 3-13
 - Block Size and Fragment Support 3-14
 - Space Allocation for Large Files 3-14
 - Contiguous Files 3-14
 - Large File Systems 3-15
 - Prioritized I/O 3-15
 - xfs File System Structure 3-15
 - Block 0 3-16
 - The xfs SuperBlock 3-16
 - xfs Free Space List 3-16
 - xfs Inode List 3-16
 - xfs Root Directory 3-16
 - xfs Cyclic Log 3-16
 - xfs Data Blocks 3-17
 - xfs Space Allocation 3-17
 - xfs File System Administration 3-18
 - Creating an xfs File System 3-19
- The Distributed xfs File System Type 3-20

xfsd Overview	3-20
Required Network Connection	3-21
Mounting an xfsd File System	3-21
Limitations on Client Access to the xfsd File System	3-22
System Failures	3-23
Example on How to Create an xfsd File System	3-24
xfsd Setup	3-24
The sfs File System Type	3-26
The sfs Inodes	3-26
The memfs File System Type	3-27
Administering a File System	3-28
The Generic Administrative Commands	3-28
How the Administrative Commands are Invoked	3-28
The vfstab File System Table	3-29
What Does the vfstab File System Table Do?	3-29
Commands for sfs File Systems	3-30
Listing Installed File System Types	3-31
Identifying the Type of an Unmounted File System	3-31
Creating a File System	3-31
Using mkfs to Create a File System	3-31
Choosing a Logical Block Size	3-32
What to Consider When Choosing a Logical Block Size	3-33
Using mkfs to Create a ufs File System	3-33
Using mkfs to Create an sfs File System	3-33
Moving and Copying File Systems	3-34
Step 1: Copying the Source File System	3-35
Steps 2 and 3: Removing the Source File System and Editing vfstab	3-36
Mounting and Unmounting File Systems	3-36
Example of Mounting File Systems	3-36
Using the mount Command	3-36
Unmounting a File System	3-37
Mounting ufs File System under sfs	3-37
Mounted File System Level Range	3-38
Device Level Range	3-38
Maintaining a File System	3-39
Monitoring the Percentage of Disk Space Used	3-39
Monitoring Files and Directories That Grow	3-39
Identifying and Removing Inactive Files	3-40
Identifying Large Space Users	3-41
The du Command	3-41
The find Command	3-41
Deleting /tmp Directories	3-41
Quotas	3-41
Quick Reference Guide to Managing File System Types	3-42

Chapter 4 File System Problems

What Is File System Checking?	4-1
Using the fsck Utility to Check File Systems	4-2
Generic fsck and Its Options	4-2
Parallel File System Checking with the -P Option	4-2
Checking ufs File Systems	4-3
ufs File System Components Checked by fsck	4-4

- Using fsck_ufs to Check Super-Blocks 4-4
 - Using fsck_ufs to Check File System Size and Inode List Size 4-4
 - Using fsck_ufs to Check Free Block List 4-4
 - Using fsck_ufs to Check Free Block Count 4-4
 - Using fsck_ufs to Check Free Inode Count. 4-5
- Using fsck_ufs to Check Inodes 4-5
 - Using fsck_ufs to Check Format and Type 4-5
 - Using fsck_ufs to Check Link Count 4-6
 - Using fsck_ufs to Check Duplicate Blocks 4-6
 - Using fsck_ufs to Check Bad Block Numbers 4-6
 - Using fsck_ufs to Check Inode Size 4-7
- Using fsck_ufs to Check Directory Data Blocks 4-7
 - Using fsck_ufs to Check Directory Unallocated. 4-7
 - Using fsck_ufs to Check Bad Inode Number 4-7
 - Using fsck_ufs to Check Incorrect “.” and “..” Entries 4-8
 - Using fsck_ufs to Check Disconnected Directories 4-8
- Running fsck on a ufs File System. 4-8
 - fsck_ufs Initialization Phase. 4-8
 - fsck_ufs Phase 1: Check Blocks and Sizes. 4-12
 - fsck_ufs Phase 1 Error Messages 4-13
 - fsck_ufs Phase 1B: Rescan for More DUP. 4-15
 - fsck_ufs Phase 2: Check Pathnames. 4-16
 - fsck_ufs Phase 2 Error Messages 4-16
 - fsck_ufs Phase 3: Check Connectivity 4-22
 - fsck_ufs Phase 3 Error Messages 4-22
 - fsck_ufs Phase 4: Check Reference Counts 4-24
 - fsck_ufs Phase 4 Error Messages 4-25
 - fsck_ufs Phase 5: Check Cylinder Groups 4-27
 - fsck_ufs Phase 5 Error Messages 4-28
 - fsck_ufs Cleanup Phase 4-28
- Checking sfs File Systems 4-29
 - fsck_sfs Phase 1: Check Blocks and Sizes 4-30
 - fsck_sfs Phase 1B: Rescan for More DUP 4-30
 - fsck_sfs Phase 2: Check Pathnames 4-30
 - fsck_sfs Phase 4: Check Reference Counts 4-31

Part 2 System Performance Administration

Chapter 5 Managing System Performance

- Introduction 5-1
- Improving and Controlling System Performance 5-1
 - Modifying the Tunable Configuration Parameters 5-2
 - Improving and Controlling File System Usage 5-2
 - Balancing File System Space 5-2
 - Procedure 5-2
 - When You Run out of Space 5-3
 - Selecting a ufs or sfs Logical Block Size 5-4
 - Selecting a xfs Logical Block Size 5-4
 - Controlling Directory Size. 5-5
 - Changing the Maximum File Size. 5-8
 - Procedure 5-8

Controlling System Work Loads	5-8
Controlling User PATH Variables	5-8
Controlling Runaway Processes	5-9
Memory-Based File System	5-9
Monitoring System Performance.	5-10
df and du Usage Reports.	5-10
System Performance Analysis Tools	5-10
Kernel Profiling	5-12
Loading the System Profiler.	5-13
Enabling and Disabling the Sampling Mechanism	5-13
Collecting Profiling Data	5-14
Formatting the Collected Data	5-14
System Activity Reporting	5-15
Automatically Collecting System Activity Data	5-16
Collecting System Activity Data on Demand	5-17
Checking File Access with sar -a	5-18
Checking Buffer Activity with sar -b	5-19
Checking System Calls with sar -c	5-20
Checking Disk Activity with sar -d	5-21
Checking Page-Out and Memory Freeing Activity with sar -g	5-22
Checking Kernel Memory Allocation Activity with sar -k	5-23
Checking Interprocess Communication with sar -m	5-24
Checking Page-In Activity with sar -p	5-25
Checking Queue Activity with sar -q	5-26
Checking Unused Memory with sar -r	5-27
Checking File System Usage with sar -t	5-28
Checking CPU Utilization with sar -u.	5-29
Checking System Table Status with sar -v	5-30
Checking Swapping and Switching Volume with sar -w	5-31
Checking Terminal Activity with sar -y	5-31
Checking Overall System Performance with sar -A	5-32
Reporting Application Turnaround with timex	5-39
Samples of Performance Management Procedures	5-40
Investigating Performance Problems	5-41
Checking for Excess Swapping.	5-42
Checking for Disk Slowdowns	5-42
Checking for Modem Interrupts	5-43
Checking for Table Overflows	5-43
Shifting the Workload to Off-Peak Hours.	5-43
Dynamic vs. Static Kernels	5-43
Quick Reference Guide to Managing Performance.	5-44

Chapter 6 Managing Dynamically Loadable Modules

What Are Dynamically Loadable Modules?	6-1
The Difference between Static Modules and Loadable Modules.	6-1
Types of Dynamically Loadable Modules	6-2
Determining If a Module Is a DLM	6-2
Procedure.	6-3
Determining If a DLM Needs to be Configured	6-3
Procedure.	6-3
Determining Which DLMs Are Currently Loaded	6-4
Procedure.	6-4

Obtaining More Information about a Specific DLM	6-4
Procedure	6-4
Configuring a Dynamically Loadable Module	6-5
Procedure	6-5
Reconfiguring All DLMs and the Entire Kernel	6-5
Procedure	6-5
Loading Modules	6-6
Overview of the Load Process	6-6
Automatic Load	6-6
Demand Loading a DLM	6-7
Procedure	6-7
Procedure	6-7
Resetting the Loadable Modules Search Path.	6-7
Procedure	6-7
Unloading Modules	6-8
Overview of the Unload Process	6-8
Automatic Unload	6-8
Demand Unloading a DLM	6-9
Procedure	6-9

Chapter 7 Tunable Parameters

Introduction	7-1
Tunable System Parameters	7-1
Reconfiguring the Operating System	7-2
Autotuning Parameters	7-3

Chapter 8 Configuring and Building the Kernel

Introduction	8-1
System Tunable Parameters	8-1
Configuring Kernel Modules	8-2
Hardware Adapters	8-2
Building and Installing the Kernel	8-3
Dynamic and Static Kernel Support	8-4
Kernel Configuration Utility	8-4
General Information	8-4
Kernel Config Utility	8-5
Terminal Attributes	8-5
Object Directory	8-6
How to Invoke Config	8-6
Current Configuration	8-6
Access Requirements	8-7
Menu Operations	8-7
Output Navigation	8-8
Kernel Config Utility Operations	8-9
Main Menu Functions	8-9
System Tunable Menu	8-9
Kernel Module Menu	8-11
Kernel Hardware Adapters Menu	8-13
Scheduling Class Dispatcher Parameter Tables	8-15
Kernel Rebuild Menu	8-16
Kernel Options Menu	8-17

Other Menu Functions	8-18
Minimizing Kernel Size	8-18
Configuring the Kernel	8-18
Required Modules	8-18
Optional Modules	8-19
Device Drivers	8-20
File Systems	8-20
Memory Management	8-21
Networking	8-21
Miscellaneous	8-21
Scheduling Classes	8-22
Security	8-22
Streams	8-22
VxVM Volume Manager	8-23
Tunable System Parameters	8-23
Minimum Bootable Configuration	8-26
Single User Mode (init S)	8-26
Multi User Mode (init 2)	8-27
Kernel Symbols	8-27
Loading Symbols	8-28
Removing Symbols from the Kernel Object File	8-28

Chapter 9 Process Scheduling

Introduction	9-1
Overview of the Process Scheduler	9-2
Time-Sharing Class and the Fixed Class	9-4
System Class	9-5
Fixed Priority Class	9-5
Configuring the Scheduler	9-5
Default Global Priorities	9-6
Tunable Parameters	9-7
Fixed Priority Parameter Table fp_dptbl	9-8
Fixed Class Parameter Table fc_dptbl	9-9
Kernel-Mode Parameter Table ts_kmdpris	9-9
Changing Scheduler Configuration	9-10
Procedure	9-10
Removing a Scheduler Class	9-11
Procedure	9-11
Installing a Time-Sharing Scheduler Class	9-11
Procedure	9-11
Installing a Fixed Class Scheduler Class	9-12
Procedure	9-12
Installing a Fixed Priority Scheduler Class	9-12
Procedure	9-12
Changing Scheduler Parameters with disadmin	9-13
Managing Processors and Processes	9-15
Processor Administration Information	9-15
Binding Processes to Processors	9-15

Part 3 Backup and Restore Services

Chapter 10 Archiving and Restoring Data

What Is A Backup Operation?	10-1
Archiving and Restoring Data On Your System	10-1
Comparison of fsdump/fsrestore & backup/restore	10-2
Tape Archiving Using tar	10-3
Making A Tape Archive	10-4
Create A New Tape Using Full Pathname(s)	10-5
Create A New Tape Using Partial Pathname(s)	10-5
Listing The Contents Of A Tape Archive	10-5
Restoring Files From A Tape Archive	10-6
Restore A Complete tar Tape with a Full Pathname	10-6
Restore Specific Files from a tar Tape with a Full Pathname	10-7
Restore a Complete tar Tape with a Partial Pathname	10-7
Restore Specific Files From tar Tape with a Partial Pathname	10-7
Tape Archiving Using cpio	10-7
Archiving and Restoring Files Using fsdump/fsrestore and xfdump/xfsrestore	10-9
Daily And Weekly Backups	10-9
Backing Up Selected System Files	10-10
The fsdump and xfdump Commands	10-10
Making the Daily Backup Tape	10-11
Making the Weekly Backup Tape	10-12
Monthly Backups	10-13
Saving root	10-13
Restoring Files	10-13
Restoring an Entire File System	10-14
Restoring an Individual File	10-15
Restoring an Individual Directory	10-15
Using the Interactive Restore	10-16
Restoring the root Partition	10-16
Additional Features of fsdump/fsrestore and xfdump/xfsrestore	10-16
File System Restore Utility	10-20
Introduction	10-20
Booting Up Procedure	10-21
Night Hawk Systems	10-21
Power Hawk Systems	10-21
File System Restore Operation	10-22

Chapter 11 Backup and Restore Services

What Is A Backup Operation?	11-1
Archiving and Restoring Data On Your System	11-1
Using the Basic Backup Service	11-2
Editing Backup and Ignore Files for Basic Backups	11-2
The Backup File	11-2
The Ignore File	11-2
Media Preparation and Handling for Basic Backups	11-3
Backing Up to Tape	11-3
Using the backup Command	11-3
Performing Complete Backups to Tape	11-3
Performing Partial Backups to Tape	11-3

Backing Up Selected Files to Tape	11-3
Backing Up User Home Directories to Tape.	11-4
Using the Extended Backup Service	11-4
Suggestions for Performing Extended Backup Operations	11-4
The System Administrator's Tasks	11-5
The Operator's Tasks	11-6
Establishing a System Backup Plan.	11-6
Backup Plan Information	11-6
Getting Backup Plan Information.	11-7
Using bkgreg to Set Up Backup Tables.	11-8
What Is a Backup Table?	11-8
Validating Backup Tables	11-8
Displaying the Contents of the Backup Tables.	11-9
Specifying Custom Backup Tables.	11-10
Using bkgreg to Customize Backup Tables	11-11
Adding or Changing Backup Table Entries	11-11
Adding an Operation Entry	11-11
Modifying an Existing Operation Entry	11-12
Removing an Operation Entry.	11-12
Specifying Backup Methods.	11-13
Determining Which Backup Method to Use.	11-13
Full File System Method	11-14
Incremental File System Method	11-14
Full Image Method	11-20
Full Disk Method	11-20
Full Data Partition Method	11-20
Requesting Migrations for Backed-Up Information	11-21
Requesting Core File System Backups	11-22
Specifying Originating Objects	11-22
Specifying Destination Devices	11-23
Specifying the Rotation Period.	11-24
Establishing Dependencies and Priorities	11-25
Creating Tables of Contents	11-26
Using backup to Perform Backup Operations	11-27
Selecting an Operator Mode.	11-28
Background Mode	11-28
Interactive Mode	11-29
Running Automatic Backups	11-30
Previewing Backup Operations	11-31
Requesting Limited Backups	11-32
Using bkoper to Monitor Backup Jobs	11-33
When Backup Jobs Need Assistance	11-34
Using bkstatus to Check Job Status	11-36
Displaying the Backup Status Table	11-37
Controlling Jobs in Progress.	11-38
Using bkhistory to Display the Backup History Log	11-39
Customizing the Backup History Log Display.	11-40
Customizing the Contents of the Display	11-40
Customizing the Format of the Display	11-41
Truncating the Backup History Log	11-42
Backing Up Data through OA&M Menus.	11-42
Basic Backup Service	11-42
Extended Backup Service.	11-43
Quick Reference to the Extended Backup Service	11-44

What Is a Restore Operation?	11-48
Before You Begin	11-48
Basic Restore Service.	11-48
Specifying Pattern Matching on the restore Command Line	11-48
Media Handling	11-49
Restore Examples	11-49
Displaying an Index of Archived Files	11-49
Restoring Everything from the Archive	11-49
Restoring Selected Files	11-50
Extended Restore Service	11-50
Who Is Responsible for Servicing Restore Requests?	11-50
Using rnotify to Arrange for Restore Requests	11-51
Deciding Which Restore Command to Use	11-51
How the Extended Restore Service Works	11-51
Using urestore to Restore a Directory or File	11-52
Using restore to Restore Other Disk Objects	11-52
Restoring a Specific Version of an Object	11-53
Restoring an Object to a New Location	11-53
Checking the Status of Restore Requests	11-54
Using restore or urestore to Check Status	11-54
Using rsstatus and ursstatus to Check Status.	11-54
Servicing Pending Restore Requests	11-56
Restricting Restores.	11-57
Removing and Canceling Restore Jobs	11-57
Restoring ffile and incfile Backup Archives Using cpio	11-58
Using the Restore Service Through OA&M Menus	11-60
Basic Restore Service	11-60
Extended Restore Service	11-60
Quick Reference to the Extended Restore Service	11-62
The Administrator's Tasks	11-62
The Operator's Tasks.	11-63
The User's Tasks	11-64

Part 4 Print Service Administration

Chapter 12 Basic Print Service

Introduction	12-1
Overview of the Print Service.	12-1
Functions	12-2
Getting Started	12-2
Choose Your Printer Sites.	12-2
Physically Connect Printers to Computers	12-4
Install the LP Print Service	12-5
Configure Printers for the Printer Service	12-5
Configuring a Directly Connected Printer	12-5
Choose the Device Level Range	12-6
Allocate the Device	12-7
Parallel Port Printer Setup (Power Hawk Only)	12-8
Step 2: Define Printer Attributes	12-8
Step 3: Notify the Printer to Accept Jobs	12-11
Step 4: Turn on the Printer (Logically)	12-11

Other Printer Configuration Options	12-12
Printer Port Characteristics	12-12
Printer Types	12-14
Content Types	12-14
Printer Interface	12-16
Character Sets, Fonts, or Print Wheels	12-16
Alerting to Mount a Print Wheel	12-19
Forms Allowed	12-20
Printer Fault Alerting	12-22
Printer Fault Recovery	12-24
User Access Restrictions	12-25
Inclusion of Banner Page in Output	12-26
Controlling Copying of Files	12-27
Printer Description	12-27
Default Printing Attributes	12-28
System Default Destination	12-29
Printer Class Membership	12-29
Putting It All Together	12-30
Example 1: Configuring a PostScript Printer on a Serial Port	12-30
Example 2: Configuring a PostScript Printer on a Parallel Port	12-30
Example 3: Adding a Printer with the Standard Interface	12-30
Example 4: Configuring an Alert	12-30
Example 5: Configuring a Secure Printer	12-31
Examining a Printer Configuration	12-31
Display Status of Printer Service	12-32
Making Printers Available	12-32
Accepting Print Requests for a New Printer	12-33
Enabling and Disabling a Printer	12-33
Allowing Users to Enable and Disable a Printer	12-34
Managing the Printing Load	12-35
Rejecting Requests for a Printer or Class	12-36
Accepting Requests for a Printer or Class	12-36
Moving Requests to Another Printer	12-36
Examples	12-37
Example 1	12-37
Example 2	12-37
Example 3	12-38
Starting and Stopping the LP Print Service	12-38
Manually Stopping the Print Service	12-38
Manually Starting the Print Service	12-38
Managing Classes of Related Printers	12-39
Adding a New Class	12-39
Listing Printers in Classes	12-39
Modifying Membership of a Class	12-39
Removing a Class	12-40
Managing Active Print Requests	12-40
Canceling Print Requests	12-40
Putting a Request on Hold	12-40
Releasing Held Print Requests	12-41
Moving Requests to A New Destination	12-41
Moving a Request Around in the Queue	12-42
Changing the Priority for a Request	12-42
Moving a Request to the Head of the Queue	12-42
Troubleshooting	12-42

Problem: No Output (Nothing Is Printed)	12-43
Is the Printer Connected to the Computer?	12-43
Is the Printer Enabled?	12-43
Is the Baud Rate Correct?	12-43
For a PostScript Printer: Do You Have Access to It?	12-43
Problem: A Print Request That Won't Cancel	12-44
Problem: No Output and No Notification from the lp Command	12-44
Problem: Printer Stops Working after Printing 2 or 3 Pages	12-44
Problem: Illegible Output	12-44
Is the Baud Rate Correct?	12-44
Is the Parity Setting Correct?	12-45
Are the Tabs Set Correctly?	12-45
Have You Selected the Correct Printer Type?	12-45
Problem: The Printing Is Legible but the Spacing Is Wrong	12-46
Double Spaced Text?	12-46
Text Has No Left Margin, Is Run Together, or Is Jammed Up?	12-46
Text Zig Zags Down the Page?	12-46
Letters Are Poorly Spaced?	12-46
A Combination of Problems?	12-46
Problem: Wrong Character Set or Font	12-47
Problem: An Attempt to Dial Out Fails	12-47
Problem: The Printer Is Idle	12-47
Problem: Warning Messages About Unrecognized Options	12-48
Problem: Error Messages About Options That Can't Be Handled	12-48
Administering LP through OA&M Menus	12-49
Quick Reference to LP Print Service Administration	12-50

Chapter 13 Advanced Print Service

Introduction	13-1
Configuring a Network Printer	13-2
Printer Server Setup	13-2
Configure the Server Printer	13-2
Advertise the LP Print Service	13-2
Define Attribute Mapping	13-3
Configure the Client Communication Parameters	13-4
Printer Client Setup	13-4
Advertise the LP Print Service	13-4
Define Attribute Mapping	13-5
Configure Server Communication Parameters	13-5
Configure the Printer Client Attributes	13-5
Notify the Printer to Accept Jobs	13-6
Turn on the Printer (Logically)	13-6
Putting It All Together	13-7
Example 1: Configuring a Remote Printer	13-7
Example 2: Configuring a Remote PostScript Printer	13-7
Configuring a Dial-up Printer	13-7
Sharing Printers	13-8
Providing Forms	13-9
Defining a Form	13-10
Removing a Form	13-12
Restricting User Access	13-12
Alerting to Mount a Form	13-13

Mounting a Form or Print Wheel	13-15
Unmounting a Form or Print Wheel	13-15
Setting the Default Destination	13-16
Examining a Form	13-16
Providing Filters	13-17
What Is a Filter?	13-17
Task 1: Converting Files	13-17
Task 2: Handling Special Modes	13-18
Task 3: Detecting Printer Faults	13-19
Will Any Program Make a Good Filter?	13-19
Defining a Filter	13-20
Defining Options with Templates	13-23
Command to Enter	13-28
Removing a Filter	13-28
Examining a Filter	13-28
Restoring Factory Defaults	13-29
A Word of Caution	13-29
Managing the Printer Queue	13-30
Assign Print Queue Priorities to Users	13-30
Setting A User's Priority Limits	13-30
Setting a Default Limit	13-30
Listing User's Priorities	13-31
Removing User's Priorities	13-31
Setting the System Priority Level	13-31
Cleaning Out the Request Log	13-31
PostScript Printers	13-33
How to Use a PostScript Printer	13-34
Support of Non-PostScript Print Requests	13-34
Additional PostScript Capabilities Provided by Filters	13-35
The Administrator's Duties	13-36
Installing and Maintaining PostScript Printers	13-36
Installing and Maintaining PostScript Filters	13-37
Installing and Maintaining PostScript Fonts	13-38
Managing Printer-Resident Fonts	13-39
Installing and Maintaining Host-Resident Fonts	13-40
Downloading Host-Resident Fonts	13-42
Customizing the Print Service	13-43
Adjusting the Printer Port Characteristics	13-45
Adjusting the Terminfo Database	13-46
How to Write an Interface Program	13-48
What Does an Interface Program Do?	13-48
How Is an Interface Program Used?	13-49
Customizing the Interface Program	13-50
An Example: Adding a Printer with a Customized Interface	13-52
Troubleshooting	13-52
Networking Problems	13-52
Jobs Backed Up in the Client Queue?	13-53
Jobs Backed Up in the Server Queue?	13-53
Conflicting Messages about the Acceptance of Jobs?	13-53
Administering LP through OA&M Menus	13-54
Quick Reference to LP Print Service Administration	13-55

Part 5 The sysadm Interface

Chapter 14 Using the sysadm Interface

Introduction	14-1
The Menu Interface Window	14-2
Types of Frames	14-3
Menus	14-4
Forms	14-4
Messages	14-4
Full Window Tasks	14-5
Frame Manipulation Tools	14-5
Navigation Keys	14-5
Function Keys	14-5
Alternate Keystrokes	14-6
Automatic Function Key Downloading	14-7
The Command Menu	14-8
The Command Line	14-8
Help for Forms: CHOICES	14-9
Getting Help	14-9
Using Express Mode	14-10
A Sample Session: Adding an Account for a New User	14-10
Step 1: Access the Menu Interface	14-10
Step 2: Select a Menu	14-11
Step 3: Select a Task	14-12
Step 4: Fill Out the Forms	14-12
Step 5: Exit from the Interface	14-14
System Administration Menu Map	14-14
Quick Reference to the sysadm Interface	14-20
Function Keys	14-20
Menus	14-20
Forms	14-20
Text Frames	14-20
Command Line	14-20
Exiting from sysadm	14-21

Chapter 15 Customizing the sysadm Interface

Overview of Customization	15-1
The Structure of the Menu Interface	15-1
Menu Modification Tools	15-2
Planning Interface Modifications	15-3
Selecting New Menus and Tasks	15-3
Naming New Menus and Tasks	15-4
Collecting Information for the Definition Form	15-4
Naming Requirements	15-5
How the System Handles Naming Collisions	15-5
Writing Help Messages	15-6
Item Help File Entries	15-7
The Menu Item Help Message Format	15-7
The Default Title Format	15-8
The Field Item Help Message Format	15-8
The Title Hierarchy	15-9

Setting Up for Item Help in an FMLI Object	15-10
Example Item Help Files	15-10
Creating or Changing a Menu Entry	15-10
Creating a Menu Entry	15-11
Changing a Menu Entry	15-13
Testing Your Menu Changes On-Line	15-15
The Menu Definition Form	15-15
Creating or Changing a Task Entry	15-16
Creating a Task Entry	15-16
Changing a Task Entry	15-18
The Task Definition Form	15-19
Deleting a Menu or Task Entry	15-20

Chapter 16 Modifying the sysadm Interface

Overview of sysadm Modification	16-1
Introduction to the Tools	16-1
The edsysadm Command	16-2
The delsysadm Command	16-2
The Data Validation Tools	16-3
Introduction to the Package Modification Files	16-3
Overview of the Interface Modification Process	16-4
Planning Your Interface Modifications	16-4
Deciding If You Should Modify the Interface	16-4
Planning the Location of Your Modifications	16-4
An Overview of the Interface Structure	16-5
Planning Your Administration Structure	16-5
Naming Your Interface Modifications	16-6
How to Name Your Modifications	16-6
Interface Naming Requirements	16-7
How the System Handles Naming Collisions	16-7
Writing Your Administration Actions	16-7
Privileged Command Execution in Task Action Files	16-8
Writing Your Help Messages	16-9
The Item Help File	16-10
The Menu Item Help Message Format	16-10
The Default Title Format	16-11
The Field Item Help Message Format	16-11
The Title Hierarchy	16-12
Setting Up for Item Help in an FMLI Object	16-13
Example Item Help Files	16-13
Packaging Your Interface Modifications	16-16
Basic Steps for Packaging Your Modifications	16-16
Creating or Changing the Packaging for a Menu Entry	16-16
Creating the Packaging for a Menu Entry	16-16
Changing the Packaging for a Menu Entry	16-17
Testing Your Menu Changes On-Line	16-19
The Menu Definition Form	16-19
Creating or Changing the Packaging for a Task Entry	16-20
Creating the Packaging for a Task Entry	16-20
Changing the Packaging for a Task Entry	16-21
The Task Definition Form	16-22
Preparing Your Package	16-24

Deleting Interface Modifications	16-25
Data Validation Tools	16-25
Types of Tools	16-26
Characteristics of the Tools	16-26
The Data Validation Tool Prompts	16-27
The Data Validation Tool Help Messages	16-27
The Data Validation Tool Error Messages	16-28
Message Formatting	16-28
The Shell Commands	16-28
The Visual Tools	16-29

Glossary

Volume 2 Index

Illustrations

Figure 1-1. Typical Trouble Report Form	1-14
Figure 2-1. Sectors and Tracks on a 690-Megabyte HSA	2-13
Figure 2-2. Cylinder on a 690-Megabyte HSA	2-13
Figure 2-3. Example of 1.2 GB Fixed Disk Drive Partitions	2-16
Figure 2-4. Format of a File System	2-17
Figure 2-5. Cylinder Groups and Directories	2-18
Figure 2-6. Cylinder Groups	2-19
Figure 2-7. Partial Bit Map	2-23
Figure 2-8. Disk Striping on VP with Two Member Partitions	2-63
Figure 2-9. Mirrored VP with Two Member Partitions	2-63
Figure 3-1. An Operating System File System	3-1
Figure 3-2. Adding the <code>/usr</code> File System	3-2
Figure 3-3. The Operating System View of a <code>ufs</code> File System	3-7
Figure 3-4. The Address Chain in a <code>ufs</code> File System	3-8
Figure 3-5. Multi-Initiator SCSI <code>xfsd</code> Configuration	3-25
Figure 5-1. Outline of Typical Troubleshooting Procedure	5-41
Figure 9-1. The Process Scheduler	9-3
Figure 10-1. Creating a Backup Script	10-11
Figure 10-2. Sample <code>/var/adm/dumpdates</code> File	10-12
Figure 12-1. A Directly Connected Printer	12-3
Figure 12-2. Printer Server Configuration	12-4
Figure 12-3. Network Configuration	12-4
Figure 13-1. How LP processes print request <code>lp -d deskjet500 file</code>	13-44
Figure 14-1. The System Administration Menu Interface Window	14-2

Screens

Screen 2-1. Displaying Device Security Attributes	2-6
Screen 2-2. Hard Link Counts	2-25
Screen 2-3. Directory Links	2-26
Screen 2-4. Rebooting Your System	2-29
Screen 2-5. Entering the Format Program	2-29
Screen 2-6. Initializing a Default Geometry Block	2-30

Screen 2-7. Examining the Geometry Block	2-30
Screen 2-8. Formatting the Disk.	2-31
Screen 2-9. Using the Write and Quit Commands to Exit	2-32
Screen 2-10. Sample user discovered flaw list for an SCSI disk	2-34
Screen 2-11. Directory Listings for a User's Directory and /dev	2-41
Screen 2-12. The Storage Device Management Menu	2-69
Screen 3-1. Main Menu for File System Administration	3-36
Screen 8-1. Kernel Configuration Utility Main Menu	8-9
Screen 8-2. System Tunable Menu	8-9
Screen 8-3. Kernel Module Menu	8-11
Screen 8-4. Hardware Adapters Menu	8-13
Screen 8-5. Scheduling Class Configuration Menu	8-15
Screen 8-6. Kernel Build Menu	8-16
Screen 8-7. Kernel Options Menu	8-17
Screen 10-1. System Installation, Main Menu	10-22
Screen 10-2. File System Restore, Main Menu	10-22
Screen 10-3. Identification of Disk Requiring Restoring, Menu	10-23
Screen 10-4. Create File System Help	10-23
Screen 10-5. Create File System.	10-23
Screen 10-6. Verify File System Creation	10-24
Screen 10-7. Option to Run format (1M) Command	10-24
Screen 10-8. Creating File Systems	10-25
Screen 10-9. Option to Advance Tape	10-25
Screen 10-10. Selecting a Directory, Menu	10-25
Screen 10-11. Selecting Tape Archive Format, Menu	10-26
Screen 10-12. Option to Enter fsrestore Argument String	10-26
Screen 10-13. Verification of Information Entered	10-27
Screen 10-14. Option to Restore Additional File System	10-27
Screen 10-15. Completion of File System Restore.	10-28
Screen 11-1. Sample Display of a Backup History Log	11-40
Screen 12-1. Main Menu for Print Service.	12-49
Screen 13-1. Main Menu for Print Service.	13-54
Screen 14-1. The Command Menu.	14-8
Screen 14-2. Sample Pop-Up CHOICES Menu: Valid Time Zones.	14-9
Screen 14-3. System Administration Main Menu	14-11
Screen 14-4. Step 2: the Menu Selected from the Main Menu.	14-12
Screen 15-1. Item Help File for One Form.	15-11
Screen 15-2. Item Help File for Multiple Forms	15-12
Screen 16-1. Item Help File for One Form.	16-14
Screen 16-2. Item Help File for Multiple Forms	16-15

Tables

Table 2-1. 690 Mega-Byte Disk Capacity	2-14
Table 2-2. Sample Drive Names.	2-15
Table 2-3. Standard Disk Name Formats	2-15
Table 2-4. Contents of the Super Block	2-20
Table 2-5. Contents of Cylinder Group's Super Block.	2-21
Table 2-6. Contents of Cylinder Group's Bookkeeping Information	2-22
Table 2-7. Effect of The Processor "boot" Register on the Boot Process	2-28
Table 2-8. Recommended Attribute and Default Values	2-47
Table 3-1. Number of Bytes Addressable.	3-9
Table 3-2. Menus and Commands For FS Administration.	3-36

Table 8-1. Required Modules	8-19
Table 10-1. tar Command Options and Modifiers.	10-4
Table 10-2. cpio Command Options and Arguments	10-8
Table 10-3. System Files Requiring Regular Backups	10-10
Table 10-4. Additional Options Available	10-17
Table 12-1. Maximum Filename Lengths for Different File System Types	12-8
Table 16-1. The Shell Commands.	16-29
Table 16-2. The Visual Tools	16-30

Introduction to Basic Administration

How to Use This Book	1-1
New Administrators	1-1
Experienced Administrators	1-2
How This Book Is Organized	1-2
What's In Volume 1	1-2
What's In Volume 2	1-4
Related Documents	1-5
Introduction to System Administration	1-7
Administrative Interfaces	1-7
Administration Procedures	1-8
Set Up	1-8
Administrative Privileges	1-8
Booting the System	1-9
System States	1-9
Multi-User State	1-9
Shutdown State	1-9
Users and Groups	1-10
Peripherals (Terminals, printers, networks, media)	1-10
Terminals	1-11
Printers	1-11
Networks	1-11
Media (Hard Disks, Tapes and CD-ROM)	1-11
Communicating with Users	1-12
Message of the Day	1-12
The wall Command	1-12
News	1-12
Electronic Mail	1-13
Collecting Users' Requests	1-13
Notation Conventions Used in This Book	1-14
Related Documentations	1-16

Introduction to Basic Administration

How to Use This Book

System Administration is a book in two volumes that will help you understand the job of a system administrator and tell you how to set up, configure, and maintain the PowerMAX OS^{TM1}. The PowerMAX OS is based on UNIX^{®2} System V Release 4.2 ES/MP. For reasons of brevity, the term “operating system” or “OS” will sometimes be used throughout this manual.

You may be personally responsible for maintaining a single computer. Or you may be an administrator for a large organization in which many users share a network of computers. In either case, this book will help you install and maintain various services on your system, and serve the needs of your users.

PowerMAX OS includes multiprocessing capabilities. With these capabilities come the associated need to manage multiple processors and processes. There are new commands that allow administrators to control the availability of processors, to display information about processors, and to bind processes to processors. This binding can be exclusive, that is, it can exclude all other user processes. In addition, there is a new entity, the light-weight process (LWP). This entity can be bound to a processor and it is the basic entity that is scheduled for the OS. These new features are described, where appropriate, in these two volumes.

This book explains how to do administrative tasks using the UNIX command line interface, often referred to as the “shell.” We assume that you know how to enter commands at the shell prompt, and that you understand such UNIX system fundamentals as the directory structure and the shell. You should also feel comfortable working with the computer hardware itself; you should know how to boot your computer, how to shut it down, and how to install peripherals (such as modems, terminals, and printers). For information on these and other hardware topics, see your computer installation manual and any documents that came with the peripherals.

New Administrators

If you have no experience as a UNIX system administrator, this book adds structure to what can seem a confusing tangle of individual commands. Begin by reading “*Introduction to System Administration*”, which describes the duties of an administrator, suggests how to organize those duties, and tells you where this book offers more information about them.

1. PowerMAX OS, Night Hawk, PowerMAXION and Power Hawk are trademarks of Concurrent Computer Corporation.
2. UNIX is a registered trademark, licensed exclusively by X/Open Company Ltd.

Once you have a general idea of what's involved, you can read individual chapters as necessary, to learn about particular tasks you need to do. Many of the tasks described in this book are required of all administrators. Some of the tasks you may never need to do, depending on your resources and your users.

Experienced Administrators

If you are an experienced administrator you'll probably use this book as a reference to procedures. When you want more information than is covered here, you'll find it in the relevant on line manual pages.

How This Book Is Organized

System Administration is divided into two volumes, each containing parts covering discrete topics. The next sections describe the contents of volume 1 and volume 2.

What's In Volume 1

Volume 1 of *System Administration* contains two parts. Part 1, "*Setting Up the System*", begins where the *Release Notes* ends: it assumes you've finished installing the software on your computer. Now you're ready to power up the computer and set up your system. It includes the following chapters:

- "*Setting Up the Work Environment*" describes some tasks you may need to complete or re-execute following installation of the system. It covers changing basic system parameters if necessary (such as the date and time on your computer), assigning passwords to system logins or administrative commands and local memory administration.
- "*Booting and System States*" explains how start and stop the system and how to do tasks that affect the way in which your computer operates, or that provide information on the current state of your computer.
- "*Creating and Managing User Accounts*" tells you how to set up and control accounts for users and user groups, file and directory access, and command authorization on your computer.
- "*Managing Ports*" tells about the Service Access Facility, and how to administer the **listen** and **tymon** port monitors.
- "*Collecting Data on System Use*" tells how to monitor system use—by time, user account, or specified software—of the resources on your system. Use these programs to bill users and create optimization strategies for resource usage. "Directories and Files" provides a map to the system directories and files that you will need to know about as a system administrator.

- “*Installing Add-on Software*” tells how to install software packages or sets on your system, how to store packages on your system for later installation, and how to remove packages.
- “*Directories and Files*” provides a map to the system directories and files that you will need to know about as a system administrator.

Part 2, “Security Administration” includes the following chapters:

- “*Introduction to Security*” will help you understand the security needs of your system, and the role that you play in assuring system security as an administrator. It is intended to describe the rationale for security, how the various mechanisms are implemented on the system, and the procedures you should follow to keep your system secure.
- “*Installing Software on an Enhanced Security System*” provides administrative users specific guidelines on how to install, configure, and run the Operating System (OS) with the Auditing and Enhanced Security Utilities installed. It illustrates the intended use of the features available to administrators.
- “*Maintaining an Enhanced Security System*” describes procedures and guidelines you should use in maintaining your Enhanced Security system.
- “*User Account and Group Management*” describes items of particular security relevance when creating new accounts and managing existing accounts. This chapter contains guidelines on items involved in managing and creating user accounts of particular security relevance.
- “*Administering Printers, Terminals, and Devices*” covers the particular security aspects of administering devices.
- “*File Protection*” discusses file attributes that have security relevance. Attributes specific to device files are discussed in the previous chapter.
- “*Administering Mandatory Access Control and Multilevel Directories*”. The Mandatory Access Control (MAC) mechanism enforces access restrictions that do not depend on the actions of the user. MAC is based on the comparison of security levels that are assigned to users, processes, files, and other objects. MAC supplements Discretionary Access Control (DAC) to prevent accidental or malicious disclosure of sensitive information.
- “*Administering Privilege*” discusses executable file privileges, process privileges, the privileged user mechanism, the least privilege mechanism, and how to choose and enable a privilege mechanism.
- “*Trusted Facility Management*” gives administrative users specific guidelines on how to install, configure, and run the TFM database, how to administer roles and individual commands, how to use the tfadmin command, and other ways to grant privilege.
- “*Trusted Backup and Restore*” tells you how to perform trusted backups and how to restore backed-up data if the Enhanced Security Utilities are installed on your system. The trusted backup commands save both the data and file security attributes necessary for maintaining security.
- “*Security Procedures*” tells how to set up a security policy, and gives procedures for checking user accounts, devices, and files. It tells how to check

for files with fixed privileges, etc., how to use cron, how to check the TFM database, and how to check the level databases.

What's In Volume 2

Volume 2 of *System Administration* contains five parts.

Part 1, "File System and Storage Device Administration", contains the following chapters:

- "*Managing Storage Devices*" tells how to add, remove and maintain devices such as disks, tape drives, CD-ROM and ethernet/FDDI controllers. This chapter also provides information on Virtual Partition (VP), what it is, and how to configure and de-configure VP in your system.
- "*Managing File System Types*" tells how to create and maintain each of the following types of file systems:
 - **ufs**,
 - **sfs**
 - **xf**s
 - **xfsd**
- "*File System Problems*" tells how to check file systems for consistency.

Part 2, "System Performance Administration" contains the following chapters:

- "*Managing System Performance*" describes ways to monitor and enhance the performance of your system.
- "*Managing Dynamically Loadable Modules*" tells how to add, remove and maintain DLMs on demand. This chapter also explains the concepts underlying the mechanism.
- "*Tunable Parameters*" describes procedures for modifying tunable parameters contained in the Operating System. The chapter also provides guidelines about when and what you should tune, and instructions for reconfiguring the operating system to enable new parameters.
- "*Configuring and Building the Kernel*" explains how to configure and deconfigure the kernel, how to configure kernel modules, how to configure non-required kernel modules, how to update hardware adapter files, and how to build and install the kernel. Also describes the config utility which is a menu-driven centralized utility that provides a front-end interface to the current configuration method.
- "*Process Scheduling*" explains the working of the system scheduler, a program that determines when processes run and thus greatly affects performance. Once you understand how the scheduler works, you'll be able to (a) judge whether you need to change the default tuning (to improve performance), and (b) foresee problems that may arise from the misuse (whether accidental or intentional) of its functions.

Part 3, “Backup and Restore Services” contains the following chapters:

(**Note:** This service only supported by PowerMAX OS release 4.2 and earlier.)

- “*Archiving and Restoring Data*” describes services for archiving and restoring data. Chapter covers: tape archiving and restoring using tar and cpio, dumping and restoring files using fsdump and fsrestore, comparison between the fsdump/fsrestore and backup/restore service utilities and how to restore file systems using the menu driven File System Restore Utility.
- “*Backup and Restore Services*” describe services that allow you to make copies of files at regular intervals, and provide copies to users who lose original files. Backing up your system's files to another medium is an important means of protecting your system from loss of data. A solid backup schedule will ensure that files that are lost or damaged could be restored later from backup copies. The backup and restore services contain a comprehensive set of utilities and files for customizing your system's backup and restore procedures.

Part 4, “Print Service Administration” contains the following chapters:

- “*Basic Print Service*” describes basic Line Printer services. Information includes how to configure a single printer on a single computer, manage everyday aspects of printers for your users, such as manage printer classes, print queue priorities, and troubleshooting.
- “*Advanced Print Service*” describes configuring printers on a network. In addition, information on administering print filters, and pre-printed forms, is given. There are also hints for troubleshooting network printing problems and customizing the print service.

Part 5, “The sysadm Interface” contains the following chapters:

- “*Using the sysadm Interface*” contains detailed instructions for using the **sysadm** command. For the most common administrative procedures, a user-friendly menu interface is provided with this command. It includes a sample walk-through for one menu and definition of all the menu system components.
- “*Customizing the sysadm Interface*” describes how you can modify the sysadm interface to suit your environment.
- “*Modifying the sysadm Interface*” provides background information and details the procedures necessary to design and write your package administration and also how to package it.

Related Documents

The area of network administration is large and complex, and the operating system includes several approaches to it. For that reason, we present networking topics separately, in a book titled *Network Administration*. Because you may be setting up network services using a mix and match of applications and protocols, *Network Administration* is organized as seven parts that address the major topics listed below.

- Part 1, “Network Services Administration”

This part is an overview of networking. It provides information on selecting a network and setting up name-to-address mapping. It discusses the connection server (which establishes connections for network services that communicate over TLI connection-oriented and dialup connections), using authentication schemes (for additional system security), and setting up and administering ID mappings (for users on remote systems). It also covers administration and use of the Basic Networking Utilities (BNU) (for communicating to other systems that support BNU), and interactive remote execution (REXEC) utilities (to allow remote administration of a machine).

- Part 2, “Mail Service Administration”

This part describes administration of the online facility that allows users to exchange messages. Once basic networking is configured, you don't need to do any additional administration to use the mail facility. This part, however, will help you set up some special features, such as establishing a domain name, setting up mail directories to be shared across a networked file system, and setting up a connection to another site that uses the Simple Mail Transfer Protocol (SMTP).

- Part 3, “TCP/IP Network Administration”

This part provides information needed to set up and run TCP/IP on your system. The discussion includes information about configuring Internet addresses and describes how to use TCP/IP commands and files to implement a wide range of TCP/IP features. First, it introduces you to important TCP/IP concepts you should be familiar with as an administrator. Next, it steps you through some basic TCP/IP administrative tasks. Finally, it describes some features in depth, such as domain name service and troubleshooting. Topics include the concepts you need to understand to effectively administer your system, step-by-step procedures for many basic administrative tasks, how to expand and manage growing systems by setting up and using routers and subnets, how to use SNMP to do network monitoring and management functions, concepts and procedures relating to domain name service, how to diagnose problems and tune your system to improve TCP/IP performance, how to obtain and complete IP address registration forms, how to obtain and complete domain name registration forms, and how to synchronize time among the machines on your network.

- Part 4, “Distributed File System Administration”

This part describes the DFS command interface for NFS. For example, the DFS software provides you with the **share** command, which allows you to share a resource on your system using NFS. DFS Administration is covered in the following chapters: “Introduction to DFS Administration”, “Setting Up DFS”, “Using DFS Commands and Files”, and “DFS sysadm Interface”.

- Part 5, “Network File System Administration”

This part tells you how to set up and maintain NFS on your system, including how to share and mount resources, how to mount resources automatically using a feature called the automounter, and how to set up Secure NFS. NFS Administration is covered in the following chapters: “Introduction to NFS Administration”, “Setting Up NFS”, “Sharing and Mounting NFS Resources Explicitly”, “Obtaining NFS Information”, “Troubleshooting and Tuning NFS”, “Setting Up Secure NFS”,

“Using the NFS Automounter”, “The NFS Network Lock Manager”, and “Using the NFS sysadm Interface”.

- Part 6, “Remote Procedure Call Administration”

This part tells you how to administer the files used by RPC, a mechanism for resource sharing between hosts used by NFS and NIS. Information about setting up and establishing secure RPC domains is also provided.

- Part 7, “Network Information Service Administration”

This part explains how to set up, administer, and update NIS, a distributed database service used for password and host file administration.

- Glossary

Contains definitions for terms and abbreviations used throughout this book.

- Index

Contains an alphabetical index of topics covered in the book.

Introduction to System Administration

To help you get started, this section provides an overview of the tasks and features that make up system administration. It then points you to the section of the book where you can find more information. Before you use this book you may find it useful to become familiar with the shell and file editors, as well as system features such as file permissions and logins. To do this, we recommend you read the *User's Guide*.

Chapter 9, “*Introduction to Security*” in volume 1, explains the concept of security embodied in the Enhanced Security Utilities and gives an overview of security concepts and mechanisms. The section “*How the Components of the System Work Together*” in chapter 9, explains in more detail the assignment of security levels and the access checking mechanisms and algorithms as they apply with the Enhanced Security Utilities installed.

If you intend to administer your system according to B2 B1 security criteria, as established by the *Trusted Computer System Evaluation Criteria* (DoD 5200.28-STD, 1985), see the section entitled “*System Startup and Security*” in chapter 9.

Also see the chapter entitled “*Trusted Facility Management*” in volume 1 for a complete description of the installation, configuration, and maintenance of the system, as well as a complete guide to the security relevant information to be found in this and other related documentation.

Administrative Interfaces

Almost all the procedures in this book can be done by issuing shell level commands. The descriptions and procedures in each chapter are presented in terms of shell level

commands. Basic shell administrative commands are in the base operating system when you install it. The OS provides a non-graphical menu interface that helps you do administrative functions without using shell commands. It is accessed through the **sysadm** command. When the functions described in a chapter can be done through the menu interface, brief instructions for invoking the appropriate menu are included in a section near the end of the chapter. Because the menu interface is self-explanatory (it includes on-line help), this book does not explain how to use it to complete particular tasks. However, “Using the sysadm Interface”, defines all the components of the menu system, and walks you through one menu.

Administration Procedures

The administrative facilities available can be intimidating for the first-time system administrator. To help you become comfortable with administration, we have provided an overview of some of the most important administrative procedures.

Set Up

Most aspects of system setup are handled automatically through the installation procedure. However, you may find you need to change some system parameters, or assign administrative logins before you can use it effectively. This information includes setting: the date, time and time zone; the system name; and the communications node name.

Accurate date/time stamps will ensure accurate representations of when files are created, processes are run, and mail is transferred. The system name and communication node name identify your system to the applications and users that access it.

For information on how to set up your system, see Chapter 2 “*Setting Up the Work Environment*” in volume 1.

Administrative Privileges

Administrators of earlier releases (identifiable because they answer to “Root”) had a great deal of power simply by having the UID of 0. The advantage was obvious: the administrator was able to do all necessary work with a minimum of red tape. At least one disadvantage, however, was that an administrator could not delegate any work to others without giving them, too, the same blanket permissions he or she enjoyed. The result was that anyone given permission to do administrative tasks on a system had the power to do many good deeds — or wreak havoc — with the system.

The OS offers a solution to this problem: a database called **tfadmin** (short for “trusted facility administration”) in which individual tasks are associated with isolated sets of permissions. This database is set up automatically when your machine comes up; if you're the first person to set up a login name on the system, your login will be recorded in the **tfadmin** database and all available privileges will be given to you. Later, however, you can add the logins of others to whom you want to assign permissions for specific tasks. By assigning task-specific privileges in this database, you can avoid conferring the amount of authority that makes the term “superuser” (another popular term for the owner of UID 0) meaningful.

This does not mean the UID of 0 is no longer significant. As the “owner” of your system (the first person to log in on a new system, which is usually the administrator), you’ll still be assigned the UID of 0 and you’ll still enjoy special privileges associated with that UID. Now, however, having that UID isn’t the only way to obtain administrative privileges.

Booting the System

Each time you power up your computer, a complex series of steps occur automatically to start the system. By default, `/stand/unix` is booted.

As part of the boot procedure, a set of processes are started and file systems are mounted as defined by the system state set for your system.

System States

The system state defines the levels of activity and accessibility for your system. In a single user state (1, S, or s), only one user can be active on the system and many of the file systems are not accessible. In multi-user state (2), other terminals can access the system and more local file systems are mounted. In Networking state (3), all multi-user processes are started, plus remote directories are mounted.

Shutdown system states are 0 (power down state) and 6 (stop, then reboot). (See the `init(1M)` online manual page for more details on system states.)

Multi-User State

The system boots to system state 2 (multi-user state) by default. The default system state is set by the `initdefault` line in the `/etc/inittab` file. All entries in the `inittab` file that correspond to the `initdefault` system state are run during the start-up process.

When the system boots up to system state 2, commands beginning with S in the `/etc/rc2.d` and `/etc/dinit.d` directories are also started. These commands do things like start print schedulers, clean up old spool files, and start up network daemons. (There are also directories for system states 0, 1, and 3.)

Some of the processes started up at boot time are run as background processes. These background processes run continuously, waiting for something to occur. Examples are the `listen` process, which monitors networking ports for incoming network requests, and `ttymon`, which listens to terminal ports for login requests.

To change your default system state, you would edit the `/etc/inittab` file and change the number 2 in the following line to the state number you want (1 or 3, for example).

```
is:2:initdefault:
```

Shutdown State

Before you turn off your system you must go to system state 0. When you go down to system state 0, commands beginning with K in the `/etc/rc0.d` directory are run. Primarily, these commands are run to kill daemon processes started in higher system states. Once the computer has completed its transition to state 0, you can turn off the computer.

The following is an example of the **shutdown** command:

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

Users and Groups

Definition of users and passwords is the most important means of protecting your system security. User and group assignments are used to define the ownership of files on the system and the accessibility of those files to other users.

User and group management can be broken down into the following types of duties:

- **Default Environment** - Before you add a user, you can set up a series of defaults that will apply to each user you add to your system. The defaults will define such things as the directory under which the user's home directory will be added (**/home**), the group the user will be a part of (**other, 1**), the point at which the login will expire or become inactive after disuse, and the location of default files to be placed in the user's home directory (**/etc/skel**).

The files in the **/etc/skel** directory are automatically copied to the user's home directory when the user is added to the system. A sane **.profile** is the most important default file to have in **/etc/skel**. In that file you can define the user's mail file, terminal type, path, and other information that will make the user's login immediately usable. When users log in, they can then tailor their **.profile** to their own needs.

There is also a system-wide profile (**/etc/profile**) that is executed each time the user logs in. It can be set up to do things that apply to each user, like print the message of the day, check if the user has mail, and list the availability of file system space.

- **Adding Groups** - By defining groups on your system, you can add a layer of accessibility that applies to a group of users, instead of just an individual user. After you create a group and add users to that group, those users can share files and directories that are not accessible to users outside that group.
- **Adding Users** - When you add a user to your system, you can use the defaults described above (or change them), assign that user to a group, and add a password for that user.

Once users are added to your system, you will have to support those users. The system provides methods for communicating with users immediately [**wall (1)** command] when they log in (**/etc/motd** file), or when they read the news [**news (1)** command].

Peripherals (Terminals, printers, networks, media)

As you add peripheral hardware to your computer, it will be up to you to identify each peripheral to the operating system. Peripheral connections are made through outlets on your computer called I/O (input/output) ports. Before you can use a port, you need to

allocate it for use by a particular device. Instructions for doing this are in Chapter 5, “*Managing Ports*”, in volume 1 of the *System Administration* book. Once you have allocated the ports on your system, connect your terminals, printers, and modems. For details about physically installing each peripheral, see the hardware manual that accompanies it.

Terminals

When you connect a terminal to your system, you need to define the device number associated with the port and the line speed. There is also a set of terminal attributes you can modify as needed. Procedures for adding and removing terminals are contained in Chapter 5, “*Managing Ports*” in volume 1 of the *System Administration* book.

Printers

If you want to make printers available to your users, you should install the printers and the LP print service software. See your printer installation manual for hardware installation instructions. The “*Basic Print Service*” and “*Advanced Print Service*” chapters contained in *volume 2*, describe how to add a printer and provide a full description of the LP service.

Networks

When describing computer networks, it is useful to make a distinction between media (the hardware that carries information from one computer to another) and services (the software that lets you use the network for file transfer, remote login, and remote execution).

Documentation that comes with the media (communications boards, modems, etc.) usually describe how to set the media up on your system. Networking services considered to be part of the system include Basic Networking Utilities (BNU), Remote Execution Utilities (REXEC), TCP/IP Network Services, Network File System (NFS), Remote Procedure Call Services (RPC), and Network Information Service (NIS) services.

Setting up and administering your system networking services is described in the *Network Administration* guide

Media (Hard Disks, Tapes and CD-ROM)

One important category of peripheral devices is storage media, such as hard disks, tape and CD-ROM drives. To learn how to install storage devices, see the “*Managing Storage Devices*” chapter in volume 2 of the *System Administration* book. That chapter also describes how to format media, label them, and partition them.

Back up and restore facilities are described in their own chapters later in this book. Those facilities are used to create copies of your stored data, usually on removable media so you can restore the data later in case it is lost or destroyed.

Communicating with Users

As an administrator, you will frequently want to send messages to users. To do so, use any of the following four tools.

Message of the Day

When you want to ensure that every user who logs in to the system will see an announcement or inquiry, simply add your message to the `/etc/motd` file. When a user logs in, all messages in that file are displayed, as in the following example:

```
The system will be down from 1700 to 2300 hours on
Friday, September 30, for upgrades and preventive
maintenance
```

Edit the `/etc/motd` file regularly to remove obsolete notices.

The wall Command

When you really need to get in touch with users fast, use the `wall` command to write to all who are logged-on.

It is good practice to reserve the `wall` command for those times when you need to ask users to log off quickly. For example, you can warn users of unexpected shutdowns:

```
# wall
The system is coming down in ten minutes for unexpected maintenance.
Please log off soon in order to save your files.
<CTRL><d>
#
```

While `wall` is unmatched for getting urgent information out quickly, some users dislike having their work interrupted by an uninvited message. Many users guard against this by including the command `mesg -n` in their `.profile`. This command blocks the output of `wall` sent by ordinary users. But a user with appropriate privileges can execute the `wall` command and override the `mesg -n` command. See the `wall(1)` in the on-line manual page for more information.

News

A somewhat less intrusive way to get information to your users is the `/var/news` command. It displays the contents of message files that have been placed in a designated

directory. As with a bulletin board, anyone can post messages and anyone who “passes by” can read them.

To post a news item:

1. Create a file and type your message in it. A filename that suggests the content can be helpful later in identifying files you may want to edit or remove.
2. Move this message file to the `/var/news` directory:

```
mv filename /var/news
```

News items remain in `/var/news` until they are removed.

Unlike the message of the day, which users cannot turn off, `news` can be totally ignored. To read news items, a user must type `news` at the shell prompt. Only then will the computer display news items posted since the last time the user executed `news`.

Users have a choice of ways to read the news: select only some items, read and delete items, ignore all items, or remove all items. See the `news(1)` online manual page for more information.

If you don't want to leave it entirely to chance, there are a couple of things you can do to make it more likely that your users will read news regularly. You can add the `news` command with no options as the last line of the default user profile (`/etc/skel/.profile`) so that new users will have this command in their `$HOME/.profile` files. Or you can ask users to add `news` as the last line in their `.profile` files themselves. Doing so causes news items to be displayed upon logging in, just before the shell prompt appears.

Electronic Mail

The `mail` and `mailx` commands allow you and your users to communicate with each other more privately. If your system is part of a network, you can also use these commands to communicate with people on other systems.

`mail` is the basic command for sending and receiving messages. `mailx` builds upon `mail` by adding to it additional features that are useful for storing messages in files, adding headers, and so on. If you use `mailx`, you may find it helpful to use a file called `.mailrc` to customize its behavior to suit your needs. For details about using this file, see the `mailx(1)` on-line manual page.

Collecting Users' Requests

From time to time, you'll need to collect forms and feedback from users. You may find it useful to keep your own log of problems reported by users (in addition to the system log described in the “*Security*” chapters under “Part 2” of volume 1). Users' problems fall into patterns, and by keeping a record of how you resolve them, you can avoid reinventing the wheel when a problem recurs.

We also strongly recommend you provide users with a formal mechanism for reporting problems. The form shown in Figure 1-1 is an example of typical trouble report that you can use to keep track of system problems.

TROUBLE REPORT	
Machine	_____
Program running	_____
Production or development	_____
Type	_____
Symptoms	_____
Scope	_____
Error messages	_____

Person reporting	_____
Login	_____
Location	_____
Phone	_____

Figure 1-1. Typical Trouble Report Form

Notation Conventions Used in This Book

This section describes the notation conventions used in this book.

- References to literal computer input and output (such as commands entered by the user or screen messages produced by the system) are shown in a monospace font, as in the following example:

```
$ ls -l report.oct17
-rw-r--r--  1 jim  doc   3239 May 26 11:21 report.oct17
```

- Commands that are too long to fit on one line are separated by a backslash (\). This is not a character to be typed, but indicates that the command line continues on one line.

- Substitutable text elements (that is, text elements that you are expected to replace with specific values) are shown in an *italic* font, as in the following example:

```
$ cat file
```

The *italic* font is a signal that you are expected to replace the word *file* with the name of a file.

- Comments in a screen display—**that** is, asides from the author to the reader, as opposed to text that is not computer output—**are** shown in an *italic* font and are indented, as in the following example:



- Instructions to the reader to type input usually do not include explicit instructions to press the <RETURN> key at the appropriate times (such as after entering a command or a menu choice) because this instruction is implied for all system commands and menus.

In one circumstance, however, an instruction to press the <RETURN> key is explicitly provided: when, during an interactive routine, you are expected to press <RETURN> without having typed any text, an instruction to do so will be provided, as follows:

```
Type any key to continue: <RETURN>
$
```

- Keyboard references are sometimes shown with the key graphic. <Enter> and <Esc> are two examples.
- Control characters are shown by the string <CTRL>-*char* where *char* is a character such as “d” in the control character <CTRL><d>. To enter a control character, hold down the <CTRL> key and press the letter shown. Be sure to type the letter exactly as specified: when a lowercase letter is shown (such as the “d” in the example above), enter that lowercase letter. If a character is shown in upper case (such as <CTRL><D>), you should enter an upper case letter.
- The system prompt signs shown in examples of interactive sessions are the standard default prompt signs.
 - the dollar sign (\$) for an ordinary user
 - the pound sign (#) for the owner of the **root** login.

Related Documentations

The following manuals should be referenced for additional information:

<i>HN6200 Console Reference Manual</i>	0830047
<i>HN6200 Architecture Manual</i>	0830048
<i>HN6800 Console Reference Manual</i>	0830045
<i>HN6800 Architecture Manual</i>	0830046
PowerMAXION Console Reference Manual	0830052
PowerMAXION Architecture Manual	0830053
TurboHawk Console Reference Manual	0830055
TurboHawk Architecture Manual	0830056
<i>Motorola SBC Console Reference Manual</i>	0830050
<i>Power Hawk Series 700 Console Reference Manual</i>	0830059
<i>PowerMAX OS Programming Guide</i>	0890423
<i>Device Driver Programming</i>	0890425
<i>Users Guide</i>	0890428
<i>System Administration Volume 2</i>	0890430
<i>Audit Trail Administration</i>	0890431
<i>Network Administration</i>	0890432
<i>Compilation Systems Volume 1 (Tools)</i>	0890459
<i>Compilation Systems Volume 2 (Concepts)</i>	0890460
<i>Documentation Overview</i>	0890470
<i>VERITAS Volume manager (VxVM) Documentation Set</i>	0890471 & 0890472

(Continued next page)

The following related release notes should be referenced for additional information:

HN6200/HN6800 PowerMAX OS™ Release Notes	0890454-(reln, e.g. 5.1)
Motorola SBC PowerMAX OS™ Release Notes	0891058-(reln, e.g. 5.1)
PowerMAXION PowerMAX OS™ Release Notes	0891061-(reln, e.g. 5.1)
TurboHawk PowerMAX OS™ Release Notes	0891071-(reln, e.g. 5.1)
Power Hawk Series 700 PowerMAX OS™ Release Notes	0891084-(reln, e.g. 5.1)

File System and Storage Device Administration

Replace with Part 1 tab for 0890430

Part 1 - File System and Storage Device Administration

Part 1 File System and Storage Device Administration

Chapter 2	Managing Storage Devices	2-1
Chapter 3	Managing File System Types	3-1
Chapter 4	File System Problems	4-1

Managing Storage Devices

What Are Storage Devices?	2-1
Types of Devices	2-1
Physical Types of Devices	2-1
Logical Types of Devices	2-2
Block Type Devices	2-2
Character Type Devices	2-2
Supporting Software for Devices	2-2
Device Security Mechanism	2-3
Basic Device Access Control	2-4
Device Security Attributes	2-4
Device Driver Flags	2-6
Device Allocation	2-7
Using admalloc on Devices	2-7
Device Allocation During the Login Process	2-9
Devices Requiring Special Processing	2-9
Guidelines for Multiplexing	2-10
Guidelines for Clones	2-10
Logical and Secure Device Aliases	2-10
Secure Device Alias	2-11
Logical Device Alias	2-11
Device Attributes	2-11
Suggestions for Managing Storage Devices	2-11
Managing Disk Drives	2-12
Physical Disk Organization	2-12
Logical Disk Address	2-13
Disk Capacity	2-14
Physical File Systems	2-14
Devices	2-14
Disk Partitions	2-15
Cylinder Groups	2-16
Blocks and Fragments	2-17
Components of the Device	2-18
System Block	2-18
Geometry Block	2-18
Super Block	2-19
Summary Information	2-20
Cylinder Group	2-20
Super Block	2-21
Bookkeeping Information	2-22
Bit Map	2-22
SFS I-nodes	2-23
Data Blocks	2-23
Fragments	2-24
Mounting and Dismounting	2-24
Linking Files and Directories	2-24
Directory Links	2-25
Creating New File Systems	2-26

Classes of Devices	2-26
Preparing the Special File.	2-26
Creating the Empty File System.	2-26
Checking the Integrity of the File Systems	2-27
Formatting and Partitioning SCSI Disks	2-27
Formatting and Partitioning a SCSI Disk	2-28
Formatting HSA Disks with the Console Processor	2-32
Management of Media Flaws for SCSI Disks	2-33
SCSI Disk Flaw Management	2-33
Procedure For Identifying Flaws	2-34
Reformatting SCSI Disk	2-34
The Device Database: Adding and Removing Storage Devices	2-35
Using putdev to Add Storage Devices	2-35
Removing Storage Devices.	2-35
Removing a Non-Critical Device from Service	2-36
Maintaining Storage Devices and Media.	2-39
Identifying Existing Devices	2-39
Conventions Used to Position a Device File	2-40
Obtaining /dev Directory Listings	2-40
Formatting Floppy Diskettes and Hard Disks.	2-41
Displaying Information About Storage Devices.	2-42
Copying Data on Storage Media.	2-42
Copying Files from Disk to Disk.	2-43
Copying the Contents of a Directory from Hard Disk to Tape	2-43
Copying Files from Tape to Hard Disk	2-43
Checking a Corrupt File System	2-43
Erasing Storage Media	2-44
Defining Disk Partitions	2-44
Do You Need to Change the Disk Partitions?	2-44
Redefining Partitions	2-44
Expanding the Swap Area	2-45
Maintaining the Device Database	2-45
Using putdev to Create a Device Entry in the DDB.	2-46
Recommended Attributes and Default Values.	2-46
Standard Attributes	2-47
Using getdev to List Devices	2-51
Creating Customized Device Lists	2-51
Specifying Devices for a Listing	2-52
Specifying Criteria for a Listing	2-52
Sample Requests for Customized Lists	2-52
Listing Attributes	2-53
Using putdev to Change the Values of Attributes.	2-54
Deleting Attributes from a Device Entry.	2-54
Removing a Device Entry.	2-55
Managing Devices in Groups.	2-55
The Device Groups Database	2-55
Using putdgrp to Create a Device Group	2-55
Using putdgrp to Remove a Device Group	2-56
Using getdgrp and listdgrp to Maintain the Device Groups Database	2-56
Using getdgrp to Get a List of Groups	2-56
Requesting a Customized List	2-56
Specifying Device Groups for Your List.	2-57
Specifying Criteria for Your List.	2-57
Examples of Customized Lists	2-57

Using listdgrp to Get a List of Device Group Members	2-58
Managing Device Reservations	2-58
Using devreserv and devfree in Device Reservation	2-59
Using devreserv to Check Device Reservation Status	2-59
Using devreserv to Reserve a Device.	2-60
Using devfree to Free a Reserved Device	2-60
Device Names and Default Partitions	2-61
Disks	2-61
Disk Partitions	2-61
Tapes	2-62
Virtual Partition	2-62
Striped VP	2-62
Mirrored VP	2-63
VP Driver	2-63
Configuration	2-63
Geometry	2-64
Initialization	2-64
Device Naming Convention	2-65
Vpstat Command.	2-65
/etc/vptab	2-66
Example VP Set-Up	2-66
Example Mirrored VP Set-Up	2-67
Initial mirrored VP set-up	2-67
Variable Steps for Mirrored VP Set-Up	2-67
Final Steps for Mirrored VP Set-Up	2-68
Restoring Faulty Member of a Mirrored VP	2-69
Quick Reference Guide to Managing Devices	2-69
Adding a new device to the system:	2-70
Copying files from tape to disk:	2-71
Removing a noncritical device:	2-71
Verifying the usability of a disk:	2-72
Creating a device entry:	2-72
Listing devices:.	2-73
Listing device attributes:	2-73
Modifying a device entry:	2-73
Removing a device entry:	2-73
Deleting an attribute from a device entry:	2-74
Creating a device group:.	2-74
Listing device groups:	2-74
Listing the members of a device group:	2-74
Modifying a device group:	2-74
Removing a device group:	2-75
Reserving a device:.	2-75
Freeing a reserved device:	2-75
Checking device reservation status:	2-75
Example Procedures on Adding Devices	2-76
Adding a Disk Drive.	2-76
Adding a CD-ROM Drive	2-79
Adding a Tape Drive	2-80
Adding Ethernet/FDDI Controller Device(s)	2-82
Adding New Controller Device and Software for First Time.	2-82
Adding New Controller Device - Software Previously Installed	2-83

Managing Storage Devices

What Are Storage Devices?

As an administrator, one of your jobs is maintaining the devices available on your system for storing data. Your system may support any combination of disks, tape drives, and CD-ROM devices.

NOTE

Instructions for the administration of other types of devices are provided elsewhere: terminal administration is described under “Managing Ports”; printer administration in *System Administration* (under “Print Service” and “Advanced Print Service”); and communications devices in *Network Administration*.

Types of Devices

The operating system treats each type of hardware as one of two logical device types: a block device or a character device. These physical and logical classifications are briefly described below.

Physical Types of Devices

disk devices	The operating system keeps system software and user files on disks. Disks are available in a variety of sizes, providing a flexible range of storage capacity that allows you to add more devices as the number of users on your system and their computing needs grow.
tape devices	Because tapes have a high storage capacity and are portable, they provide an efficient way of storing and transferring data. Tape drives are used primarily for doing high-speed file system backups and restoring file systems or individual files. Your computer may be equipped with cartridge, 9-track, DAT, 8-millimeter, or other types of tape drives.
CD-ROM devices	CD-ROM devices are optical-based, read-only media used for permanently storing information. They are useful for accessing high volumes of reference information.

Logical Types of Devices

All logical devices are either block type or character type; the classification of a device as one of these two types depends on how the device is accessed. For most devices, either character or block access can be used, but one type is usually preferred. For example, character access is preferable for terminals. If block access is used, characters do not appear on the screen as the user types them; characters don't appear until the user presses <CR>.

Block Type Devices

When data is accessed in fixed-length blocks (that is, when the device does not permit access until a block of data has been accumulated), the device on which it's stored is classified as a block device. Examples of block devices are disk drives and tape drives.

Character Type Devices

When a device accesses data in chunks, each of which consists of a specific number of characters (usually one), the device is classified as a character device. Terminals and printers are examples of character devices. In the operating system, standard C language subroutines transfer data to these types of devices one character at a time.

Supporting Software for Devices

The operating system supports the use of various types of devices by providing software for attaching them to your system and regulating their use. Thus, your task of maintaining your system's storage devices entails maintenance of the following files and databases.

device drivers Programs that manage the signals between devices and the operating system. You can use the drivers provided with the system or write your own. (See the appropriate manual page and *Device Driver Programming* for instructions on writing your own.)

device special files Files that link devices to the appropriate device drivers. These files define devices to the kernel: how your system interacts with a device is determined by three attributes of the device special file. These attributes are the name of the file (which serves as the name of the device), the location of the file (described later in the chapter, along with naming conventions, under "*Device Names and Default Partitions*"), and the combination of two numbers—a major device number and a minor device number—in the file.

A major device number serves as an index to the appropriate block or character switch table; that is, it identifies—to the kernel—the device driver to be used for the specified device. A minor device number specifies the device connected to the device driver; it's passed to the device driver when a specific device driver function is called. The procedure for determining minor numbers for special files is device dependent.

DDB	The Device Database: a database in which you keep a list of attributes about each device on your system. The entry for each device includes a device alias (a name unique in the database that's mapped to the pathname of the device it represents) and values for a set of attributes.
device group facility	A database that allows you to organize the devices on your system in logical groups so you can execute administrative commands for multiple devices simultaneously.
device reservation facility	A system that allows you to oversee the sharing of devices by users and processes.

Device Security Mechanism

Mandatory Access Control (MAC) and Discretionary Access Control (DAC) restrictions affect access to devices. However, additional access restrictions apply to devices. These restrictions prevent a non-privileged process from opening (or otherwise accessing) a device that has not been allocated for public use and restrict changes of the security level associated with the device.

All storage and I/O devices are protected by a range of security levels, which restrict what data can be stored on the device and the levels at which it can operate. For each device on the system, the security level range is defined by a minimum level and a maximum level that are used to enforce access restrictions. These restrictions ensure that a device receives only data that is appropriate for its location and configuration. For example, you would not want to print highly sensitive information on a printer located in a public area, accessible to everyone on the site. The level range of the device is designed to protect the sensitivity of the data flow on a device.

By default, every device on the system is a system private resource and can be accessed only by processes with the appropriate privileges. Explicit action by a trusted program is needed to grant unprivileged access to a device.

A device can be accessed by multiple pathnames residing on the file system. This allows a device with different characteristics to be accessed in different ways. For example, it is possible to have different pathnames for the same tape drive when it operates at different speeds. A mapping of the device pathnames to physical devices resides in the Device Database (DDB). Information on security characteristics of devices is also stored in the DDB, which is described under “*Maintaining the Device Database*” later in this chapter. The DDB also includes authorization information on device usage.

The security attributes in the DDB define the characteristics the device will have when it is allocated for use by the operating system's kernel. During allocation, the `admalloc` command validates any information passed against the information in the DDB. Therefore, the device must be defined in the DDB before it can be allocated. Once the device is allocated, the security characteristics are maintained in kernel data structures.

Basic Device Access Control

Because a physical device is represented by one or more device special files (DSFs), the device is always protected by the DAC and the MAC mechanisms as they apply to files.

The MAC and DAC information for the device is taken from the device special file used to access the device. The MAC and/or DAC information can be different on each device special file, and changing the DAC or MAC information on one device special file does not affect the access restrictions on other device special files that can be used to access the device.

For example, there can be two different device special files for a tape device, one used when the tape operates at 1600 bits per inch, the other at 800 bits per inch. Each device special file would refer to the same physical device, but could have different access permissions; one set would not affect the other.

```
ls -l /dev/rmt/0m /dev/mt/0m
brw-r--r--  2 root  sys    103,1285 Apr 12 11:26 /dev/mt/0m
crw-rw-rw-  2 root  sys    103,1285 Apr 12 11:26 /dev/rmt/0m
```

To provide appropriate protection for the devices themselves, the system relies on device security attributes stored in the DDB. The device allocation routines consult the Device Database to obtain the list of device special files that map to the same physical device. The allocation routines make sure that none of the device special files are in use before they allocate the device and set the MAC and DAC attributes on one or all of the device special files.

Device Security Attributes

Associated with each device special file are several security-related attributes. These attributes, which are maintained by the kernel, provide an administrator a mechanism to regulate access to devices.

Four security attributes are associated with each device special file. The per-device attributes are:

release flag	This attribute shows how all the device security attributes are associated with a device. The release flag shows whether the device is allocated and the way in which it is allocated. The flag can have one of three values:
persistent	This value shows that the security attributes are set explicitly and remain associated with the device special file while the system is running or until the attributes are explicitly changed.
lastclose	This value shows that the security attributes are set explicitly and remain associated with

	the device until the last reference to the device is closed.
system	This value shows that the security attributes are set by the system. If the driver flag was set to be <code>initpub</code> , then the state of the device is set to <code>public</code> . Otherwise, it is set to <code>private</code> . The mode is set to <code>static</code> , and the <code>hilevel</code> and <code>lolevel</code> of the level range are set to level identifier (LID) values of zero (0) to show that the device does not have any applicable level range.
state	The state attribute must either be <code>private</code> or <code>public</code> . A device state of <code>private</code> shows that the device is a private Trusted Computing Base (TCB) resource and that unprivileged access to the device is denied. A device state of <code>public</code> shows that unprivileged access to the device is allowed. The state is changed from <code>private</code> to <code>public</code> when the device is allocated for unprivileged access and is changed from <code>public</code> to <code>private</code> when the device is deallocated.
level range	The device level range is represented by a <code>hilevel-lolevel</code> pair. The level range constrains the allowed values for the security level of the device and should be based on the physical constraints of the device (such as device location). The high level of the device level range must dominate the low level of the device level range. The device level (as set in the device special files for the device) must be contained in the level range.
mode	<p>The device mode should always be <code>static</code>, which prevents changing the MAC level of the device if the device state is <code>public</code> and there are active I/O connections to the device. For all other cases, MAC level change is allowed.</p> <p>The other possible value for the device mode, <code>dynamic</code>, is provided only for those sites that require dynamic changes in MAC levels while a device has open I/O connections. When the device mode is <code>dynamic</code>, MAC access checks are performed for each I/O operation. Thus, if the level of the device is changed, processes accessing the device must continue to pass MAC access checks. A device mode of <code>dynamic</code>, however, should never be used if you are running your system in compliance with the B1 or B2 security guidelines. It is provided as a means by which multi-window terminals can run several windows at different MAC levels; terminals of this type are outside the configuration of the evaluated system. (See the “Security Administration” part of this manual for more information on the evaluated system.)</p>

The device security attributes and the device use count can be examined with the **devstat** command (available only on secure systems). The use count is a flag (values 0 and 1) that shows if the device is in use (file descriptor open or **mmap** active). The **devstat** command is invoked with either the **-Z** or **-z** flag and an optional argument specifying the device. The **-Z** flag causes **devstat** to print the full level names of the level range values, and the **-z** flag causes **devstat** to print the alias name for the levels.

If no device name is specified to **devstat**, the current security attributes for all devices defined in the DDB are displayed. A device can be specified as either an absolute pathname to a device special file or a device alias. (See “*Maintaining the Device Database*” later in this chapter.) If a device alias is specified, then the current security attributes for all pathnames defined for the specified device are displayed.

Screen 2-1 shows the use of the **devstat** command.

```
# devstat -z /dev/systty
device name:systty
path name:/dev/systty
state:      public
mode:      static
high:      SYS_PRIVATE
low:       SYS_PUBLIC
use count:1
release flag:lastclose
#
```

Screen 2-1. Displaying Device Security Attributes

Device Driver Flags

There are three driver flags added for security to handle complex devices, specifically those that allow multilevel access without violation of the security policy. The kernel uses these flags associated with each device driver to determine MAC access when performing I/O operations. The device driver flags are:

- | | |
|---------------|---|
| NOSPECMACDATA | This flag shows that no MAC access checks will be done by the kernel for data transfers and the access time field in the inode will not be updated. |
| INITPUB | This flag shows that when the device has the release flag set to <code>system</code> , the device is in <code>public</code> state. A device that has the <code>INITPUB</code> flag set in the driver will by default be accessible by non-privileged processes. |
| RDWREQ | This flag shows that all accesses (both read and write) require strict level equality. |

Device Allocation

Device allocation programs modify the device security attributes of one or more device special files to allow unprivileged access to one or more users. The device allocation programs check the DDB to determine or verify the security attributes and user ID for the device. The device allocation programs also check the device to determine what device special files need to be allocated for a specified logical or secure device alias.

The general purpose device allocation program is **admalloc**, which is used to allocate most devices for users. The **login** process allocates login terminals for users automatically during login.

Using admalloc on Devices

The **admalloc** command is used both to allocate and to deallocate devices. In its simplest invocation, **admalloc** allocates the specified device for the invoking user ID and group ID at the level of the invoking process and places the device in public state if the device attribute for *state* in the DDB is *public* or *pubpriv*. The following example shows the use of **admalloc** without any options to allocate a device.

```
admalloc /dev/rsave
```

The device specified may be a secure device alias, a logical device alias, or a device special file.

The **-m** option of **admalloc** restricts the device state to *private*. In this state, only a process with the appropriate privilege has access to the device. After the following command is executed, the physical device represented by the alias **tty21** will be in the *private* state; unprivileged processes will not be able to access it.

```
admalloc -m tty21
```

The **-r** option of **admalloc** sets the device range to the specified MAC security levels. This option takes, as an argument, two levels separated by a minus sign. The first level is the high level of the level range; the second, the low level of the range. The high level must dominate the low level, and the new range must be contained in the range specified in the DDB. The levels must be specified as either security level aliases or fully qualified level names. (For information on security levels, see the “Mandatory Access Control” section of the “Managing Files Securely” chapter in *System Administration*, Volume 1.).

In the following example, a tape device is allocated with a level range of **Confidential** (a level defined by the security administrator) to **SYS_PUBLIC** (a predefined level). This will allow the tape to be used to back up a file system that has **Confidential** as the highest level for data stored on it.

```
admalloc -r Confidential-SYS_PUBLIC tapedrive1
```

NOTE

When you allocate a device for trusted backup and restore, you need to ensure that the level range specified to the **-r** option of **admalloc** will include the levels of all files that will be backed up. If the level range is set incorrectly upon device allocation, the backup will fail. If you are backing up all files included on the distribution tape, you may need to allocate the tape device with a broad security range. If a device is allocated with a broad security level, you should ensure that the DAC settings on the device special file will prevent unauthorized users from accessing the device during any backup operations.

If you set the range of a partition, that range must be within the range set for the entire disk.

The **-w** option of **admalloc** specifies the MAC security level to be set for the allocated device. The level must be within the device level range and is specified as either a level alias or a fully qualified level name. This option must be used with the **-u** option of **admalloc**, which specifies the user ID and group ID to be set for the allocated device. The argument to **-u** consists of a user name and a group name, separated by a comma. The user specified must have allocation permission specified in the DDB. The following example shows a terminal port being allocated to the user **lp** and the group **lp**. The port, which will be used to support a printer, is allocated at the **Secret** level (another level defined by the security administrator).

```
admalloc -w Secret -u lp,lp /dev/term/tty027
```

After a device is allocated at one level for a specific user, you may want to change its level to allow another user to access the device. You can use the **chlvl** command to change the level of a device after it has been allocated. There may be some restrictions on changing a device level, however. For example, if a device is in **public** state, it can be changed only if it is not in use (that is, not open or mapped). The new level must be within the device level range allocated. This operation also requires appropriate privileges. [For a full list of restrictions governing the changing of the level of a device, see **chlvl (1M)** .]

The **chlvl** command takes two arguments. The first argument is a level, which can be either a level alias or a fully qualified level name. The second argument is either a file or a list of files, the security level of which will be set to the level given in the first argument to **chlvl**. The filename can be a block special device or character special device.

When you change the level of a device, none of its files may be opened or mapped, unless the device is in **dynamic** mode or the **private** state. The new level must also be within the device level range. You may need to change the mode of the device after allocation or a change of level.

The **-d** option of **admalloc** is used to deallocate a device. If the device still has open connections, the deallocation will fail unless the **-f** (force) option is also specified. When a device is deallocated, the device security attributes are set to allow only privileged access. The following command could be used to deallocate the terminal port allocated in the previous example. In this case, the **-f** option is not used, so the command will fail if the device has any open connections.

```
admalloc -d /dev/term/tty027
```

If no device is specified with the `-d` option, `adm11oc` attempts to deallocate all devices defined in the DDB.

CAUTION

When the `-d` option is used with `-f` and no device argument is specified, all devices defined in the DDB can be deallocated.

The `adm11oc` command is also used to allocate those devices that are configured to be allocated at system startup. The `-s` option will cause `adm11oc` to allocate all devices with the startup attribute set to `y` in the DDB. This option cannot be combined with any other option. This option should be used only at system startup, in a script invoked by `/etc/rc2`.

Device Allocation During the Login Process

The process of logging in can be invoked only by using the Secure Attention Key (SAK).

The `login` process allocates the controlling terminal for the user in a trusted state and performs the identification and authentication checks, including the checks to ensure that the specified login level is within the device range in the DDB and that the user is authorized to use the device.

The device attributes are set as follows:

device special file level	The level of the device special file is set to the user's login level.
device special file DAC	The Discretionary Access Control settings on the device special file are set as follows: the owner of the file is set to the user, the group is set to the user's default group, and the permissions bits are set to allow read and write for the owner and no access for group and other (<code>crw-----</code>).
level range	The device level range used to allocate the device is set to the range defined for the device in the DDB.
state	The device state is set to <code>public</code> .
mode	The device mode is set according to the attribute in the DDB (which should always be <code>static</code>).
release flag	The release flag is set to <code>lastclose</code> .

When the last open connection to the login terminal is closed, the device is deallocated.

Devices Requiring Special Processing

Some devices require special processing by the kernel or the driver for Mandatory Access Control. These devices include (but are not limited to):

- `null`
- `zero`
- `tty`
- `log devices`

Some multiplexing and cloning devices may also need special processing for support of multilevel security.

Guidelines for Multiplexing

The multiplexing device and the device being multiplexed are treated as separate devices. Applications performing the multiplexing are responsible for setting the correct labels on these devices. It is recommended that applications reserve such devices as `private` devices.

Guidelines for Clones

Each cloning driver must be examined to see if it supports multilevel cloning. The clone minor device created inside the kernel will have the same attributes as the cloneable file being opened. For a driver that supports multilevel cloning, you should follow these steps.

- You should create multiple cloning device special files, each servicing a range of minor devices.
- If the driver installation procedure creates device special files corresponding to minor devices, then these nodes should be created at the same level as the cloneable file that generates the minor devices with the same minor number as the clone files.
- Each cloneable file and the minor device special files corresponding to its clones should be defined in the same device entry in the DDB.
- When you allocate a cloneable file with `admalloc`, all other clone files are allocated at the same level.
- If unprivileged users must use cloneable files, you should allocate the entries at system startup so that unprivileged users can access the files as long as the users pass DAC and MAC access checks.
- It is recommended that administrators allocate these kinds of devices while the system is running.
- Before changing the label of a cloneable file, you should either deallocate the file with `admalloc -d` or use the `devstat` system call with the `DEV_GET` command to ensure that no one has any device open.

Logical and Secure Device Aliases

The DDB defines the attributes of all devices configured on the system. Each device is known to the system by an alias unique in the DDB. This alias is limited to 64 characters (`MAX_ALIAS`) and should contain only alphanumeric characters and the special characters underscore (`_`), period (`.`), dollar sign (`$`), or minus sign (`-`).

The secure and logical types of device aliases can be defined in the DDB.

Secure Device Alias

This is the name of a physical device configured on the system. Each secure device alias defines one independent set of security attributes that governs that physical device. These security attributes are defined in the `DDB_SEC` file. The secure device alias can also define device special files that map to it and non-security attributes.

Logical Device Alias

This is the name of a logical device that defines only non-security attributes and maps to a secure device alias that defines the security attributes. A logical device alias does not define any security attributes of its own. Therefore, many logical aliases may define different sets of non-security attributes, yet share only one set of security attributes defined by the secure device alias to which they map. A logical device alias can also define device special files that map to it.

Almost all aliases defined in the DDB are secure device aliases, that define only one set of security and one set of non-security attributes. However, you may define additional logical device aliases that have their own non-security attributes but share the security attributes defined for the secure device alias to which they map.

Device Attributes

The DDB (described in “*Maintaining the Device Database*”) contains entries for device aliases. Each entry provides values for a set of attributes for the device represented by a particular alias. See the section “*Creating a Device Entry*” for a description of all possible attributes.

NOTE

By reserving a device, you do not allocate it for use; the device must be allocated with `admalloc` before it can be used.

Suggestions for Managing Storage Devices

How much time you spend overseeing the storage devices on your system depends a lot on the number of users on your system and how they use its resources. A large user community and many resources require a lot of care and feeding. Here are a few suggestions for administering storage devices:

- Before you start re-partitioning a disk, make sure you do a complete system backup first.
- If resources are depleted in either the `/tmp` or `/var/tmp` directory (but not both), ask users to distribute temporary files more evenly to both

directories. (To do this, define the **TMPDIR** environment variable in the user's profile as **TMPDIR=/tmp** or **TMPDIR=/var/tmp**.)

- For a more secure computer, mount the **/usr** file system with read-only permission.
- To stop yourself from inadvertently erasing data, use the write-protect tab on the magnetic tape before you start copying valuable files from a magnetic tape to another medium.
- When copying large files, the value of **ulimit** may need to be changed to a number as large as or larger than the character count of the largest file (**ulimit=number** where *number* is larger than the character count of the file as determined with **ls -l filename**).

Managing Disk Drives

This section explains disk organization, file systems and the formatting of disk drives.

Physical Disk Organization

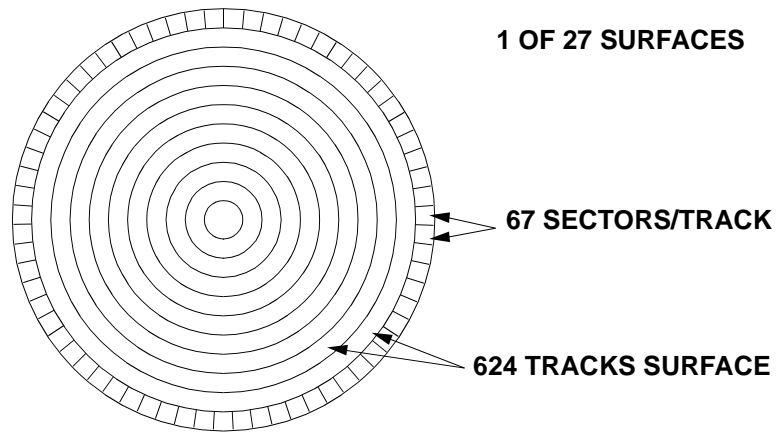
This section presents an example of disk organization (the physical addressing scheme), and disk capacity (in bytes) for the 690 Mb disk on an HSA controller. The disk organization and capacity of disks on an HSA controller may be obtained by using the HSA format program **format (8)** or **format (1m)**. The physical layout of a disk is based on the addressing scheme that the system uses to access data on it. On a 690-megabyte disk, for example, the physical addressing scheme uses information about the:

- pack
- cylinder
- track
- sector.

On a 690-megabyte HSA, a sector (shown in Figure 2-1) is a physical division of a disk surface containing 512 bytes. A track is also a physical division of a disk surface; it contains 67 sectors and forms a circle on the disk's surface. Each of the 27 disk surfaces has 624 tracks.

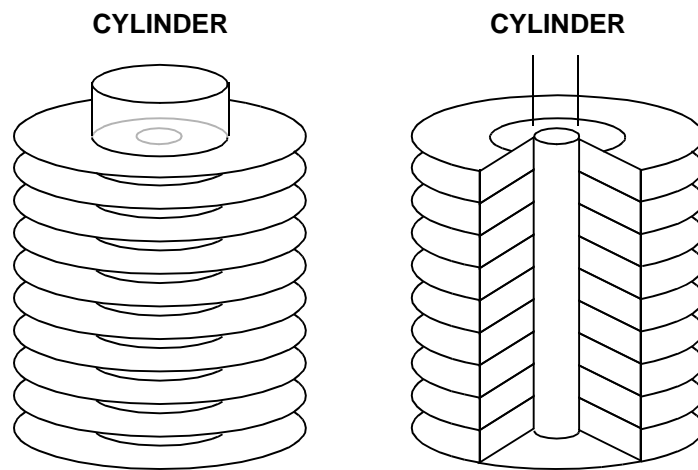
A cylinder consists of a set of tracks (one per disk surface) over which the 27 read/write heads may be simultaneously positioned. For example, a cylinder on a 690-megabyte SCSI disk consists of a set of 27 tracks as shown in Figure 2-2.

For any read/write command, all read/write heads address the same track on each disk surface. There are 624 possible cylinder addresses, one for each of the 624 tracks on each surface.



158290

Figure 2-1. Sectors and Tracks on a 690-Megabyte HSA



143640

Figure 2-2. Cylinder on a 690-Megabyte HSA

Logical Disk Address

To perform a read or write command, the disk drive must be given a logical disk address to locate the desired area. Each logical address specifies the following:

- disk drive to address
- record to begin reading or writing. The SCSI disk converts this logical record number into a physical disk address internally.

Disk Capacity

The total storage capacity of a formatted 690-megabyte pack on an HSA controller is 577,953,792 bytes of information (see Table 2-1 below for details).

Table 2-1. 690 Mega-Byte Disk Capacity

690 megabyte disk		
	624	cylinders/pack
X	27	tracks/cylinder
	16,848	tracks/pack
X	67	sectors/track
	1,128,816	sectors/pack
X	512	bytes/sector
	577,953,792	bytes/pack

For other disk models, the capacity is calculated in a similar manner.

For RAID devices, from the OS:

RAID devices are viewed by the file system as single disk even though they are composed of multiple disk devices.

Physical File Systems

A file system is composed of files and directories that reside on the same logical disk. A logical disk refers to a subset of a disk that is treated logically as if it were a physical disk.

Devices

A typical file system resides on one device. A device is either an entire disk or a large physical partition of a disk. Each device has a unique name which reflects the:

- SCSI Host Adapter to which the drive is attached (0 through 63).
- SCSI ID of the drive (0 through 6)
- SCSI LUN of the drive (0 through 7).
- partition on the drive (one integer from 0 through 7).

Table 2-2 shows some sample device names

Table 2-2. Sample Drive Names

Adapter	SCSI ID	LUN
0	0	0
0	0	0

Standard disk names are used. The standard format (see Table 2-3) places disk devices in subdirectories under `/dev`.

Table 2-3. Standard Disk Name Formats

	<code>/dev/{r}dsk/#s#</code>	
<code>r</code>	Optional.	This character indicates a raw interface to the disk. The default is the normal system buffering.
<code>dsk</code>	Required.	This refers to a disk device.
<code>#</code>	Required.	This is the logical unit number (e.g. 0).
<code>s#</code>	Required.	This is the section (partition) number (e.g. s5).

Disk Partitions

A physical disk is divided into one or more partitions as shown in Figure 2-3. Every file system resides on one partition, and any partition contains at most one file system. Partitions can be placed arbitrarily on the physical disk (within limits imposed by the format programs). If you use the defaults for the format program, your disk will be partitioned into a “01237” layout. The format programs, `format(8)` and `format(1M)`, may be used to examine or modify partitions. The system administrator should adjust the partition sizes to fit the needs of the system's applications.

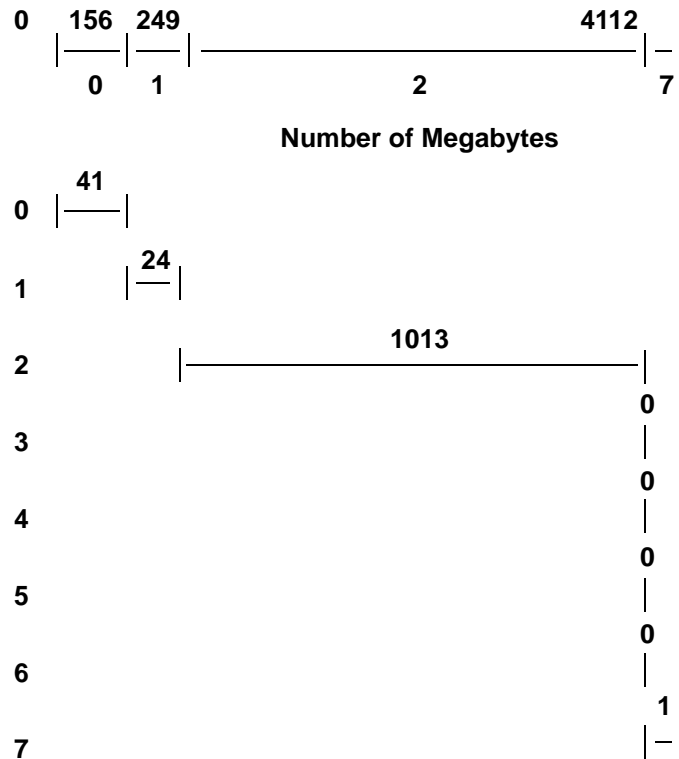
Partitions have certain customary uses. Disk zero (i.e. `/dev/dsk/0s0`) by convention, uses the “01237” partitioning. The contents are partitioned as follows:

- 0 holds the root file system
- 1 is used as the primary swap device
- 2 holds the `/usr` file system
- 3 holds the `/var` file system
- 7 holds the flwmap, diagnostic information, and geometry block.

When a disk is to be used entirely for one purpose, say to hold user files, it is usually partitioned into a single full partition. It may be desirable to place `/tmp` on a partition of its own (perhaps a "0" partition on a disk other than disk 0). It may also be desirable to allocate several partitions for swapping.

Cylinder Groups

Each file system is organized into record groups; each group is a collection of consecutive records on the disk. The operating system divides data into record groups to reduce the number of searches done by the read/write heads.



162850

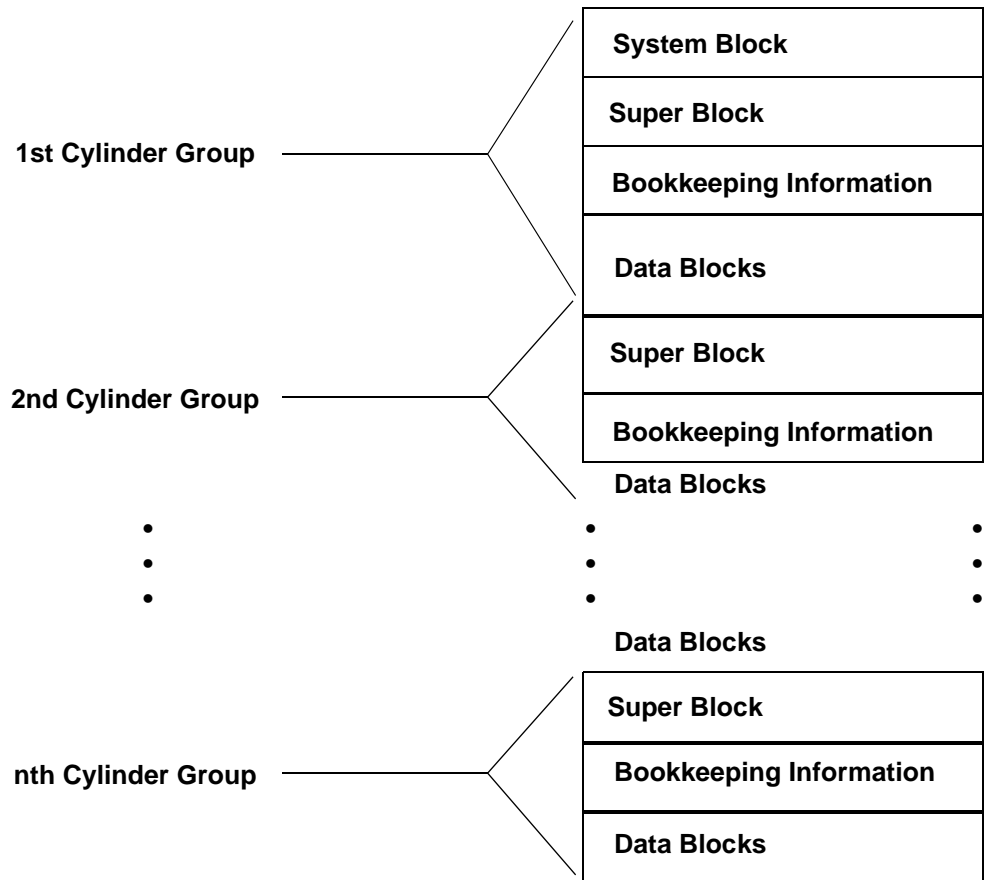
Figure 2-3. Example of 1.2 GB Fixed Disk Drive Partitions

Cylinder group bookkeeping information is located at intervals throughout the pack. This bookkeeping information begins at a floating offset from the beginning of the cylinder group. The offset for each successive cylinder group is calculated to be about one track further from the beginning of the cylinder group than the last cylinder group's offset. In this way, the redundant super blocks (see Figure 2-4 and Figure 2-5) spiral down into the pack. Thus, a single track, cylinder, or platter may be lost without losing all copies of the super block.

Typically, each partition contains one file system, and each file system contains many cylinder groups. As shown in the following figure, directories in a file system are created on different cylinder groups.

Blocks and Fragments

Each device is composed of blocks that may each contain from 2 Kilobytes, 4 Kilobytes and 8 Kilobytes. The block size for all file systems defaults to four kilobytes.



162860

Figure 2-4. Format of a File System

Typically, each block is divided into 512 byte fragments. If you want to change the number of fragments allotted to each block, you must use **newfs** and reconfigure the file system. In general, each fragment must contain at least 512 bytes (the disk's logical sector size). Each block may contain two, four, eight, or 16 fragments, and a block size must be a power of two.

Components of the Device

Figure 2-6 shows the components of a device, emphasizing the structure of the cylinder group. Explanations of all components follow the figure.

System Block

Each file system has an eight-kilobyte system block in the first cylinder group. This block is always the first block of a file system. This first sector is not used by HVME bus disk controllers such as the HSA.

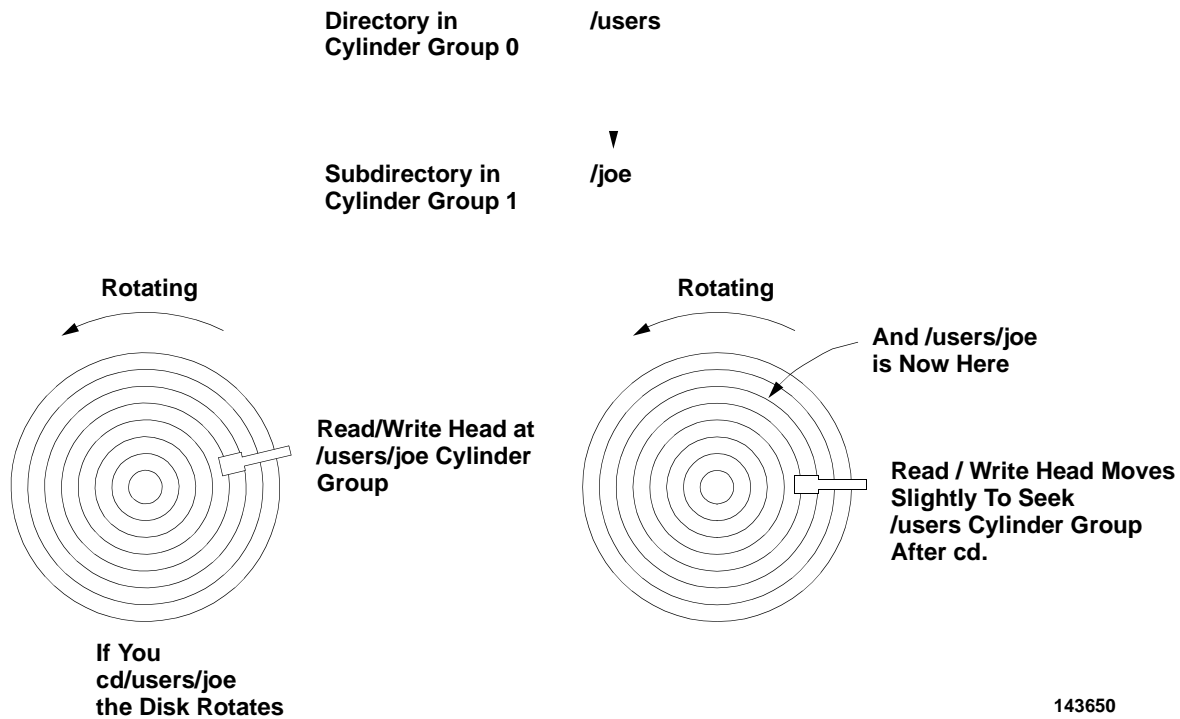
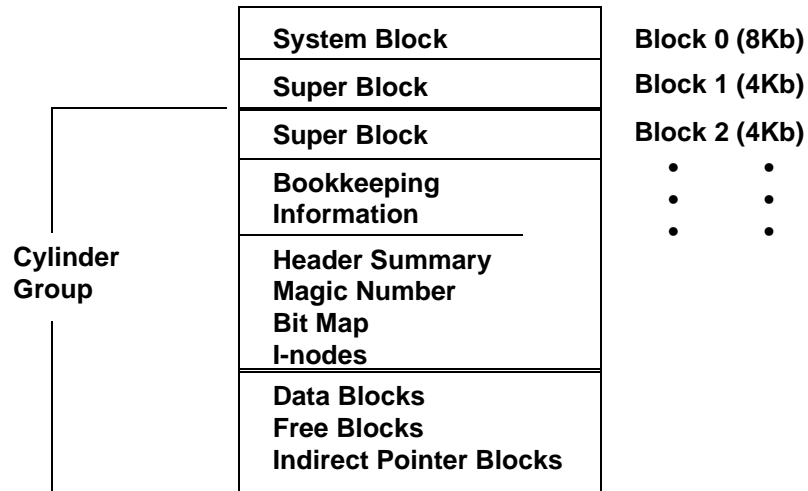


Figure 2-5. Cylinder Groups and Directories.

Geometry Block

Every disk can be defined in terms of several parameters, such as number of cylinders, number of tracks per cylinder, number of sectors per track, starting location of partitions, sizes of partitions, and block and fragment sizes of partitions. A disk's partition information is stored in its geometry block, while model-dependent information is stored in a disk status block. The geometry block resides in the first sector of the disk definition cylinder on SCSI drives. The definition of this block is found in the include file `/usr/include/sys/dsk.h`. The disk status block resides in profiling proms for SCSI drives. SCSI disk drives are self-configuring and require no geometry information from outside sources. The definition of this block is found in the include file `/usr/include/sys/dskio.h`.

The `ioctl` command, `DSKIOCGEOMETRY`, returns geometry block (partition) information for a specified drive. The `ioctl` command, `DSKIOCSTATUS`, returns status block information. The library function, `getdiskbyname`, uses these `ioctl`s to return a disk-tab structure. Geometry blocks are written using the `format` program which uses user-supplied information, or default values.



162870

Figure 2-6. Cylinder Groups

Super Block

Each file system has a super block after its system block. The super block contains all essential information about the file system (all information available to `mkfs`). The super block is built when the file system is created (`newfs (1M)`) and never changes. The super block contains the basic parameters of the file system, such as the number of data blocks it contains and a count of the maximum number of files. `newfs` replicates this critical data to protect against catastrophic loss. The default super block always resides at a fixed offset from the beginning of the disk partition of the file system. Redundant super blocks are not referenced unless a head crash or other hard-disk error causes the default super block to be unusable. The redundant blocks are dispersed throughout the disk partition.

Within the file system are files. Certain files are distinguished as directories and contain collections of pointers to files that may themselves be directories. A descriptor called an `i-node` is associated with each file. The `i-node` contains information describing:

- ownership of the file
- time stamps indicating modification and access times for the file
- an array of indices pointing to the data blocks for the file.

The first 12 blocks of the file are directly referenced by values stored in the i-node. The i-node may also contain references to indirect blocks containing additional data block indices.

In a file system with a 8192-byte block size, a single indirect block contains 2048 additional block addresses. A doubly indirect block contains 2048 addresses of additional single indirect blocks; and a triply indirect block contains 2048 addresses of additional doubly indirect blocks.

To create files with up to 16,875,520 bytes, using only one level of indirection, the minimum block size is 8192 bytes. The block size must be a power of two, currently selected from 4096 or 8192. The default block size is 8192 bytes. The block size of the file system is maintained in the **super block**. File systems of different block sizes can be accessed simultaneously on the same system. The block size must be decided when **newfs** creates the file system and cannot subsequently be changed without rebuilding the file system.

Table 2-4 summarizes the contents of the second block--the super block.

Table 2-4. Contents of the Super Block

File System Name (Mount Name)
File System Size
Time of Last Update
Number of Free sfs I-nodes
Number of Free Blocks
Number of Free Fragments
Location of First Cylinder Group
Location of First sfs I-node Block
Location of First Data Block
Number of Cylinder Groups in the File System
Block Size in File System
Fragment Size in File System
Minimum Number of Free Blocks Allowed
Bit Masks Used to Find Free Contiguous Fragments
Number of Tracks per Cylinder
Number of Sectors per Cylinder
Number of Sectors per Track
Number of Cylinders in File System
Magic Number (used to verify that file system exists before mounting it)

Summary Information

Certain summary information that is not elsewhere duplicated in the system is associated with each super block. The summary information changes when the file system is modified. It contains the number of blocks, fragments, i-nodes and directories in the file system.

Cylinder Group

The file system partitions the disk into areas called cylinder groups. A cylinder group is made up of one or more consecutive cylinders on a disk. Each cylinder group includes:

- i-node slots for files
- a block map describing available blocks in the cylinder group
- summary information describing the usage of data blocks within the cylinder group.

A fixed number of i-nodes is allocated for each cylinder group when the file system is created. By default, one i-node is allocated for each 2048 bytes of disk space.

Bookkeeping information for the cylinder group begins at a floating offset from the beginning of the cylinder. This placement prevents loss of the redundant super blocks in the event the top platter of the disk (beginning of the cylinder group) is damaged because of hardware failure. The offset for the $i + 1$ st cylinder group is about one track further from the beginning of the cylinder group than it was for the i th cylinder group. The redundant information spirals down into the pack so that any single track, cylinder, or platter can be lost without losing all copies of the super blocks. Data is stored in the space between the beginning of the cylinder group and the beginning of the cylinder group book-keeping information.

Super Block

Each cylinder group begins with a copy of the constant values of the super block. Referring to Table 2-5, a cylinder group's super block does not contain information such as the number of free i-nodes in the system and the number of free fragments in the system. lists the contents of a cylinder group's super block.

When you synchronize the super blocks using

```
# sync
```

the system synchronizes only the first super block in the file system.

Table 2-5. Contents of Cylinder Group's Super Block

File System Name
File System Size
Time of Last Update
Location of First Cylinder Group
Location of First I-node Block
Location of First Data Block
Number of Cylinder Groups in the File System
Fragment Size in File System
Minimum Number of Free Blocks Allowed
Bit Masks Used to Find Free Contiguous Fragments
Number of Tracks per Cylinder
Number of Sectors per Cylinder
Number of Cylinders in File System
Magic Number (used to verify that file system exists before mounting it)

Bookkeeping Information

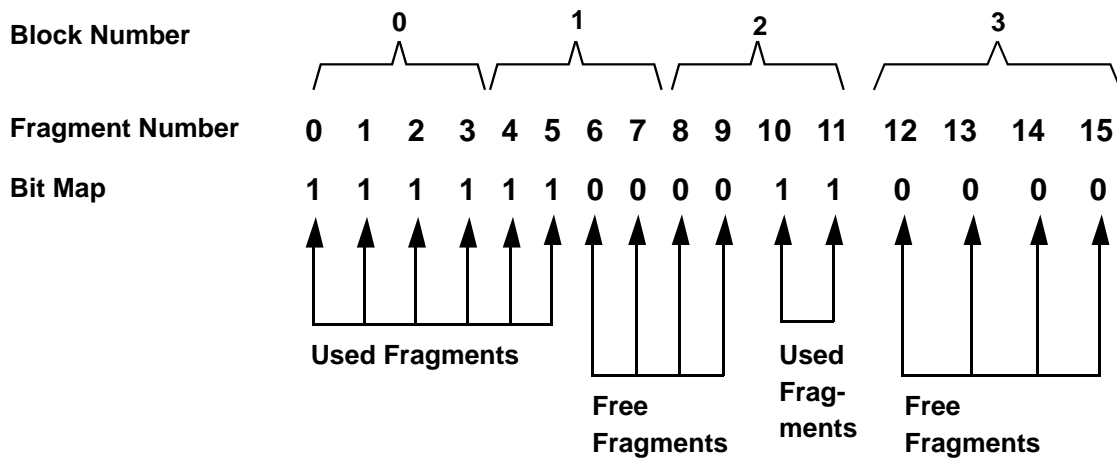
Following the cylinder group's super block is a collection of data about the cylinder group. This collection begins with a linked list of header information (see Table 2-6). Summary information follows the header.

Bit Map

Each cylinder group has a bit map that identifies the free fragments in the group's data blocks. Figure 2-7, for example, shows a partial bit map.

Table 2-6. Contents of Cylinder Group's Bookkeeping Information

-----	Time of Last Update
	Sequential Number of Cylinder Group in
Header	File System
	Number of Cylinders in Group
-----	Number of I-node Blocks in Group
-----	Number of Directories
	Number of Free Blocks
	Number of Free I-nodes
	Number of Fragments
	Position of Last Used Block
	Position of Last used Fragment
Summary	Position of Last Used I-node
	Counts of Available Fragments in Each Block
	Total Blocks Used in Each Cylinder
	Positions of Free Blocks in Each Cylinder
	Used I-node Map for Cylinder Group
	Magic Number
	Free Block Map
-----	sfs I-nodes



143660

Figure 2-7. Partial Bit Map

SFS I-nodes

The system identifies the file by its sfs i-node number. SFS I-nodes reside in an i-list that follows the bit map.

Each SFS i-node contains file information such as the:

- user ID of the owner
- group ID attached to the file
- size
- permission codes
- date of last use
- pointers to data blocks.

Data Blocks

The data blocks may contain pointers to the data of the files, to free blocks in memory, and to other addresses that point to the data.

A direct address points directly to a data block. An indirect address points to another address which, in turn, points to a data block.

A data block pertaining to a cylinder group may reside in another cylinder group's data block space. If a cylinder group does not contain sufficient space for more data, the system checks neighboring cylinder groups for free, contiguous fragments. If the adjacent cylinder groups have insufficient space, the system continues to search other cylinder groups until it locates the necessary fragments. The i-nodes for the file are the physical (or indirect) addresses of the data blocks, and they are stored in the original cylinder group's i-list.

Fragments

The file system space allocator divides a single, file system block into one or more fragments. The fragmentation is specified when the file system is created. Each block can be optionally broken into two, four, or eight addressable fragments. The logical sector size provides the lower bound for the size of these fragments. Usually, 1024 bytes is the lower bound. The block map associated with each cylinder group records the space available at the fragment level. Aligned fragments are examined to determine block availability.

On a file system with 4096-byte blocks and 1024-byte fragments, a file contains zero or more 4096-byte blocks of data and, possibly, a single, fragmented block. If a block must be fragmented to obtain space for a small amount of data, the remainder of the block is available for allocation to other files. For example, consider an 11000-byte file stored on a 4096/1024-byte file system. This file uses two full-size blocks and a 3072-byte fragment. If no fragment with at least 3072 bytes is available when the file is created, a full-size block is split to provide the necessary 3072-byte fragment and an unused 1024-byte fragment. This remaining fragment can be allocated to another file, as needed.

Mounting and Dismounting

Directories in the **root** file system do not point to files in other file systems. To access files in other file systems, you must first mount the file system on its logical disk device. For example,

```
# /sbin/mount /dev/dsk/1s2 /users
```

maps **/users** on **/dev/dsk/1s2**. The system may translate references to files under **/users** to the **/users root** directory.

File systems may be physically moved from one place to another on a drive or from one drive to another. However, you should always mount the file system on the same **root** directory.

To dismount the file system, request

```
# /sbin/umount /dev/dsk/1s2
```

Linking Files and Directories

Files that are hard-linked in a file system share an i-node and point to the same data blocks. If you link files, you save disk space and allow users to share common files.

The i-node maintains a link count for the files which indicates the number of hard links to the file as shown in Screen 2-2.

```

ls -lai /users/molly
total 495
12320 drwxr-xr-x   6 molly users 1024 Jul 18 11:38 .
      2 drwxr-xr-x  14 molly users 1024 Jul 18 11:40 ..
      .
      .
12335 -rwxr-xw-x   2 molly users 4186 Jul 4 15:26 letters
      |
      Has two files hard-linked to it.

```

Screen 2-2. Hard Link Counts

To link two files in the same file system, use the command

```
# ln file_1 file_2
```

where **file_1** and **file_2** are any files other than directories. These two files share an i-node and point to the same data blocks. For example,

```
# ln /users/lo/junk /users/harry/stuff
```

creates a hard link between **/users/lo/junk** and **/users/harry/stuff**. An **ls -lai** for **/users/lo/junk** shows

```
17188 -rw-r--r-- 2 abc users 21 Sep 11 09:53 /users/
lo/junk
```

For **/users/harry/stuff**, the **ls -lai** yields

```
17188 -rw-r--r-- 2 abc users 21 Sep 11 09:53 /users/
harry/stuff
```

Each file has the same i-node number (17188), showing that the two files point to the same data space.

Directory Links

A directory link count, shown in Screen 2-3, reflects the number of directories that are immediately below the directory. The directory link does not refer to shared i-nodes.

When you create a directory, the system assigns a directory link count of 2 (for the current directory and the parent directory). When you request an **ls -lai**, the system defaults to the current directory (**.**) and does not require you to type the full pathname. If you want to see a listing of another user's directory, for example, an

```
# ls -lai/user_name
```

shows a listing of **user_name's** directory.

```
ls -lai
total 74
 2 drwxr-xr-x 14 root sys 1024 Jul 18 11:40 .
 2 drwxr-xr-x 12 root sys 1024 Jul 18 13:36 ..
                |
                | Current directory has 12 directories
                | (plus. and ..) under it.
```

Screen 2-3. Directory Links

Creating New File Systems

Creating a file system requires a great deal of thought and operating system knowledge. This section introduces you to some of the procedures, but you are strongly urged to consult appropriate system man pages before trying to create file systems.

Classes of Devices

Types of devices are recognized: block and character. A block device may be a disk or tape which performs input/output through buffer chunks. A character device, also known as a raw device, performs its I/O directly to and from user space on a character-by-character basis.

Regardless of its class, a device has a major device number and a minor device number. Together, these numbers make up the device's name. The major number pertains to the driver that controls the device, and the minor is a subdevice such as a terminal port.

Preparing the Special File

A file system must reside on a block device. Before you create a new file system, make sure your system has a disk with ample space and a device driver for that disk.

It may be necessary to create the special file for the device using `/sbin/mknod` or `/etc/conf/bin/idmknod` which creates a new file with a specified pathname.

Creating the Empty File System

To create the new file system, use the `/sbin/newfs` command. This command accepts a wide variety of parameters, including the name of the logical disk and the size of the file system. To change the size of an existing file system, you may:

- dump the file system.
- run the format program, selecting the “modify partition sizes” option.
- run `/sbin/newfs`.
- restore the file system.

Checking the Integrity of the File Systems

To ensure the integrity of the file systems, the system automatically runs a file system consistency check (**fsck**) when it is booted to multi-user mode. If the checked file system is consistent (as it should be), the **fsck** program reports the number of files, the number of used blocks, and the number of free blocks.

If the checked file system contains inconsistencies, the **fsck** program attempts to repair the file system. If it cannot repair a particular inconsistency, **fsck** will prompt you for assistance.

You should run **fsck** on a weekly basis or if you suspect damage. You should also use **fsck** to check individual file systems before they are backed-up. Run **fsck** while the file system is unmounted and on the raw device.

If **fsck** encounters a major problem, it displays the following message and aborts.

```
error message
UNEXPECTED INCONSISTENCY; RUN fsck MANUALLY.
```

You should fix the damage using the standard file system repair procedures described in the “File System Problems” chapter. In general, you should:

1. Login as `root` at the system console.
2. Place the system in single-user mode.
3. Run `/sbin/fsck`.
4. Analyze the errors and take the recommended recovery procedures.
5. Boot the system back to multi-user mode.
6. Update the system log.

Formatting and Partitioning SCSI Disks

Before a disk surface can be written to, it must be prepared. Preparing consists of partitioning (creating a disk geometry block) and formatting.

NOTE

You do not have to prepare disks that have already been used.

This section contains the necessary steps to prepare an SCSI disk for use on your system. Only the root disk should be formatted by this procedure. On-line HSA formats can be used for all other drives. The section titled “*Formatting and Partitioning as SCSI Disk*” describes how to run the StandAlone Disk Format Utility with an SCSI.

Formatting and Partitioning a SCSI Disk

This section describes how to run **format (8)** with an SCSI disk.

You prepare your disks using the format program. The general procedure to follow is:

1. Reboot your system
2. Enter the format program
3. Examine/Modify/Create the geometry block
4. Format the disk
5. Exit the program.

You can perform other tasks with the format program; these steps are just the minimum you need to prepare your disk. For more information on the format program, refer to the **format (8)** man page.

- Step 1: Reboot Your System

The contents of the processor (p) “boot” register determine how the system boots. Table 2-7 below lists the possible values for the processor boot register and the effect of those values on the boot process. Note that the values can be added together.

Table 2-7. Effect of The Processor “boot” Register on the Boot Process

Processor Boot Value (Hex)	Effect
0	Boots automatically without option.
1	Requests file name for boot. Asks user to specify the program to load.
2	Boots /OS to single-user mode.
80	Debug option (load symbol table).
100	Load and then halt before enabling VM and after enabling VM.

If you have not rebooted your system, follow the instructions below:

1. Power up the system or, if the system is already powered up, press the CPU Restart switch. When the system powers up, it attempts to start bootstrap procedures from the disk.
2. When prompted, type # within 10 seconds.
3. Load the distribution boot tape and place the tape drive on line.
4. Issue the following commands: (Screen messages appear on the left; comments that explain a message are to the right of the arrow as shown in Screen 2-4).

```

#>h.#>y.#>pboot 3.00000000          -Halt the CPU and disable
                                   automatic boot.

#>fd mt(0,5,1)                    -Define tape as default device.

#>fb                               Boot

Initialize HVME                    -These two lines appear only if
mt(0,0,0,0)/hsawcs                an HSA is in the system.
.
  Boot messages.
.
.
.
Boot
:
```

Screen 2-4. Rebooting Your System

- Step 2: Enter the Format Program

When the colon appears on the screen, the system is ready for the format command. Type **format** and then the device name for the disk you wish to examine and possibly modify. In the example shown in Screen 2-5, we have chosen `dsk(13,0,0,1)`, which is partition 0 of disk drive 0 connected to the SCSI controller in slot 19 (13 hex) of bus 1.

NOTE

Substitute the appropriate device name for your disk.

```

: format                            -Boot using format. For more
                                   information, see the format(8)
                                   man page.

40560+19984+2435976 start 0x1300    -These numbers are the size of
                                   text, data, and bus. Your
                                   numbers may vary.

Standalone disk formatter           -In this example, the disk to be
dsk device: dsk(13,0,0,1)          formatted is in HVME slot
                                   13(hex) on bus 1. The partition
                                   should be 0.
```

Screen 2-5. Entering the Format Program

If you make an error and specify a nonexistent drive, you will get an error message and be asked for the disk device again.

- Step 3: Examine/Modify/Create the Geometry Block

After you have typed the format command and specified a device name, **format** will try to read the geometry block on the disk into memory.

- If **format** returns the message in Screen 2-6, there is no geometry block on the disk. Press the carriage return and **format** will create a default geometry block for your disk. Your screen should then display something similar to Screen 2-7. If the amount of Kilobytes or cylinders used is not zero, there is a geometry block on the disk

```
.  
. .  
There is no geometry block on this disk.  
Initializing a default geometry block.  
Type <CR> to continue <CR>
```

Screen 2-6. Initializing a Default Geometry Block

```
Opening 'dsk(13,0,0,1)'  
Reading the geometry block.  
/dev/rdisk/14s7  
disk model 16  
megabytes: 377 logical, 471 physical  
cylinders: 837 logical, 842 physical  
heads:      20 logical, 20 physical  
sectors:    44 logical, 45 physical  
bytes:      512 logical, 622 physical  
1 drive description cylinders at cylinder 834  
2 diagnostic cylinders at cylinder 835  
3961 revolutions per minute  
geometry block version 1  
flaw map size is 6728  
partition 0   366960 kbytes      0 start kbyte  
partition 1      0 kbytes      0 start kbyte  
partition 2      0 kbytes      0 start kbyte  
partition 3      0 kbytes      0 start kbyte  
partition 4      0 kbytes      0 start kbyte  
partition 5      0 kbytes      0 start kbyte  
partition 6      0 kbytes      0 start kbyte  
partition 7    1320 kbytes    366960 start kbyte  
368280 kbytes used    0 kbytes unused
```

Screen 2-7. Examining the Geometry Block

- Step 4: Format the Disk

NOTE

Make sure the write-protect switch on your drive is off.

When you format a disk, headers and other information are written onto the disk surface a track at a time. A flawmap is necessary to make sure that no flawed spots are written to. After the disk is formatted, the flawmap and the geometry block are written to partition 7 on the disk. Every disk has a flawmap that is unique to that disk and must be used only with that disk.

You must decide if you need to format the disk. If the disk has been used before to store information, it has been formatted and probably does not need to be reformatted. You can then skip this step and go to Step 5, Exiting the Program. The exception is erasing the contents of the entire disk to create more room. In that case, you can reformat the disk using the same procedure shown in Screen 2-8 for formatting the disk the first time.

```
format: format mt(0,0,0,0)

This command overwrites all information on the disk.
Do you want to continue? (y|n) y
Opening the flawmap file
Reading flaw records
Read 163 records
Validating 163 flaws
formatting
cylinder 0 of 842...
cylinder 50 of 842...
.
.
.
cylinder 800 of 842...
Formatting complete
Writing the flawmap to disk
Writing the geometry block to disk
format: verify

check

cylinder 0 of 818...
cylinder 50 of 818...
.
.
.
cylinder 800 of 818...
Verification complete.
```

-In this example, we format the disk using a flawmap on magnetic tape. format writes the current geometry block to the disk when formatting has completed.

-You can use verify to perform a read from each sector as a after formatting.

Screen 2-8. Formatting the Disk

- Step 5: Exit the Program

You are now ready to exit the program.

There are two main ways to exit: **write** followed by **quit** and **q!**. Using **write** causes the program to write the geometry block (and if the flawmap pathname is included, the flawmap) to disk. You must then **quit** to exit.

Because formatting writes the geometry block to disk, however, you only need to use the write command when you modify or create the geometry block and do not format the disk afterwards. In all other cases, you can use the **quit** command shown in Screen 2-9 to exit the program.

If you want to exit the program without writing the geometry block that you created or modified during this format session, there is the command, **q!**, that allows you to do so. Using **q!** means the program terminates without writing the flawmap and geometry block in memory to the disk.

```
format: write
This command overwrites the geometry block and flawmap (optional).
Do you want to continue? (y|n) y
format: quit
boot
.
.
.
```

Screen 2-9. Using the Write and Quit Commands to Exit

Formatting HSA Disks with the Console Processor

To format a disk drive that cannot be formatted using the regular **format** program because the **format** process was interrupted, enter the following console processor commands:

#>i 8000 9000 0.	
#>wl 8000 1.	
#>wl 8008 8080.	
#>wl 8084 \$006f.	\$ is the SCSI device number (in HEX) for the disk drive.
#>wl 8088 8100.	
#>wl 8100 6.	This is the size of the Command Descriptor Block (CDB). The 6 specifies 6 bytes to send across the SCSI bus.
#>wl8104 4000000.	This is the CDB. 4000000 signifies the format command.
#>wl c\$040000 8000.	\$ is the HVME slot number (in HEX) for the HSA. This will actually start the format operation and the HSA green LED light should come on.
#>d1 8010 801c.	This will return one of the following lines of output.

```

00008010 00000000 00000010 00000000 00000000
|
|__10 means the drive is being
|   formatted and should take
|   approximately 15 minutes.

00008010 00000000 00000080 00008080 00000000
|
|__80 means the formatting has
|   completed without any problems.

00008010 00000000 00000081 00008080 00000000
|
|__81 means the formatting has
|   completed with problems.

00008010 00000000 00000082 00008080 00000000
|
|__82 means the formatting has failed.

```

NOTE

This procedure will generally fail immediately the first time. Just type the following command again to retry formatting the disk

```

#>w1 c$040000 8000.      $ is the HVME slot number (in HEX) for
                          the HSA. If it fails this time, then
                          something is wrong.

```

When this has completed without any problems start up the regular **format** program and write a geometry block on the disk followed by a verify operation. After this, the drive is ready for use.

Management of Media Flaws for SCSI Disks

The information contained in the following sections for SCSI disks apply to IS (Integral SCSI) disks and HSAs (HVME SCSI Adapters).

SCSI Disk Flaw Management

SCSI disk drives usually support dynamic relocation of flaws between format operations through the use of the SCSI “Relocate Block” primitive. Most SCSI drives also have their factory flaw information encoded on the disk media. Since this flawmap is already available to the SCSI disk controller, no flawmap needs to be supplied with the format operation.

Procedure For Identifying Flaws

Procedures for identifying new flaws are similar, but dealing with them is much easier with SCSI disks since we can relocate them without first reformatting the disk. The **errprt (1M)** program can be used to peruse the error log to look for both hard and soft (recovered) disk errors that have been logged on all active disks. The error log entry identifies the particular cylinder, head, and sector of each disk error encountered. Disk sectors with multiple errors at different times are probably flawed, and their performance may worsen as time goes on. These sectors represent good candidates for relocation. The **errprt (1M)** program also prints a line for SCSI disks that is the exact command to give the format program to relocate the bad sector and record it in the SCSI's internally maintained flaw list. The sector will be relocated to an unused portion of the disk reserved by the drive embedded SCSI controller for errors, and all future references to this sector will be routed to this new sector.

Reformatting SCSI Disk

When a SCSI disk drive is reformatted, the drive typically forgets all relocations. To deal with this, the **format (1M)** program maintains a record of all relocations that have occurred on this disk on the disk definition cylinders. The system's administrator may list the flaws, read them from a file, save them on the disk's definition cylinders, etc. New flaws can be added using the relocate subcommand of **format (1M)** or by simply adding an entry to a flaw list backup file with a text editor. Screen 2-10 shows a portion of a sample flaw list for an SCSI disk.

```
# Flaw map for disk1 (389 Mbyte SCSI ctrlr 0 drive 1)
cyl 183 head 12 sector 2 length 1
cyl 24 head 12 sector 12 length 1
cyl 560 head 4 sector 11 length 2
```

Screen 2-10. Sample user discovered flaw list for an SCSI disk

The Device Database: Adding and Removing Storage Devices

This section explains how to add storage devices to your system and how to remove them.

Using `putdev` to Add Storage Devices

A storage device is part of your system if it's physically attached to the computer and is represented by an entry in the Device Database (DDB). Each entry in the DDB contains a list of attributes for a device. The `alias` attribute is required; all other attributes are optional, and can vary from device to device. (For a list of possible attributes, see the table under “*Creating a Device Entry.*”)

To add a device to your system, create an entry for it in the DDB by running `putdev` as follows:

```
putdev -a alias [attribute=value [ . . . ]]
```

where *alias* is the alias name of the device to be added to the database and *attribute=value* is a list of attribute values to be associated with the device.

If the list of attributes described with the `putdev` command does not provide enough information for a device definition, you can create new attributes.

Before you can allocate a device, you must add it to the DDB.

Removing Storage Devices

CAUTION

Component failure may occur if you disconnect any storage device from your computer while the power is on.

Performing this procedure will destroy the data (that is, the mounted file systems) on the device you want to remove. Make sure you have backed up the contents of the disk before removing it. For complete instructions, see “The Backup and Restore Services” chapter in *System Administration*.

Do not remove the disk on which the root file system is mounted.

Sometimes, such as when a disk has a defect, you will need to take a device out of service. Critical devices, such as those on which essential file systems are mounted, can never be removed from service. You may, however, remove noncritical devices for servicing later, when the system is scheduled for shutdown. To remove a noncritical device from service, complete the following procedure.

NOTE

Before starting this procedure, be sure you know the path of the device (whether block or character type) you want to remove.

Removing a Non-Critical Device from Service

1. If your system supports multiple users, send users a message (with the `/usr/sbin/wall` command) warning them that the device is to be taken out of service.

Make the announcement as specific as possible: explain which file systems will not be accessible and which services will not be available. Whenever possible, allow enough time for users to finish their work on a device before removing it from service.

2. Run `/usr/sbin/devnm /` to determine the device on which the root file system is mounted. For example:

```
# /usr/sbin/devnm /
/dev/dsk/1s0 /
#
```

3. To determine the major and minor device numbers of the device on which the root file system is mounted, enter:

```
ls -l special
```

(For a description of major and minor device numbers, see the description of device special files in the “*Supporting Software for Devices*” section of the introduction to this chapter.) The value of *special* should be the partition number you received in Step 2. In this example, the partition number reported by `devnm` is `/dev/dsk/1s0`.

```
# ls -l /dev/dsk/1s0
brw----- 2 root    root      47,      0 Feb 23 1991 /dev/dsk/1s0
#
```

The output shows that the major device number is 47 and the minor device number is 0.

4. Execute `devnm /usr` to identify the device on which the `/usr` file system is mounted. For example:

```
# devnm /usr
/dev/dsk/1s0 /usr
#
```

- To determine the major and minor device numbers of the device on which the `/usr` file system is mounted, enter:

```
ls -l special
```

The value of *special* should be the partition number you received in Step 3.

In this example, the partition number reported by `devnm` is `/dev/dsk/1s0`.

```
# ls -l /dev/dsk/1s0
brw----- 2 root  root  4,  3 Feb 23 1991 /dev/dsk/1s0
#
```

The output shows that the major device number is 4 and the minor device number is 3 .

- Execute `ls -l device_path`

where *device_path* is the path to the character or block device you want to remove. Record the major and minor device numbers for this device.

```
# ls -l /dev/dsk/1s0
brw----- 1 root  sys   47, 16 Feb 23 1991 /dev/dsk/1s0
#
```

The output shows that the major device number is 47 and the minor device number is 16 .

- Verify that the major and minor numbers of the device you want to remove are not the same as those of the root or `/usr` file systems (see the output of the commands run in 3 and 5 above).

NOTE

If both the major and minor device numbers of this device match the major and minor device numbers of the devices on which the root and `/usr` file systems are mounted, the device may not be removed.

- List the current directories mounted on this device with the `grep` command, as shown next.

```
# grep ls0 /etc/mnttab  
/dev/dsk/ls0 /home0 ufs suid,rw,noquota 767466409  
#
```

NOTE

Keep a written record of these directories; they must be restored once the device is returned to service. (See “The Restore Service” in *System Administration* for complete instructions on restoring these directories from a backup tape.)

9. Save the current file system table by copying it:

```
# cp /etc/vfstab /etc/Ovfstab
```

10. Unmount the directories from the device with the `/etc/umount` command:

```
# /etc/umount /home4  
#
```

11. If the device contains one or more file systems, remove all entries associated with it from the `/etc/vfstab` file.

Remove the device from the DDB. Some devices, such as disks, have multiple logical aliases. For this type of device, you must remove all logical device aliases before removing the secure device alias. In the following example, `disk1` is the secure device alias of a disk with eight partitions.


```
# getdev secdev=disk1<RETURN>
disk1
dlpart0
dlpart1
dlpart2
dlpart3
dlpart4
dlpart5
dlpart6
dlpart7
# putdev -d dlpart0<RETURN>
# putdev -d dlpart1<RETURN>
# putdev -d dlpart2<RETURN>
# putdev -d dlpart3<RETURN>
# putdev -d dlpart4<RETURN>
# putdev -d dlpart5<RETURN>
# putdev -d dlpart6<RETURN>
# putdev -d dlpart7<RETURN>
# putdev -d disk1<RETURN>
```

12. Identify the groups to which the device belongs. Then remove the name of the device from the group membership list or lists in the device group database. For example, to get a list of groups to which `disk1` belongs and then remove `disk1` from the device group database, enter the following:

```
# getdgrp alias=disk1
device_group1
device_group2
# putdgrp -d device_group1 disk1
# putdgrp -d device_group2 disk1
```

Maintaining Storage Devices and Media

This section describes routine maintenance chores (such as identifying and formatting floppies and partitioning disks) associated with storage devices and storage media.

Identifying Existing Devices

Before any device can be used with your system, it must be made known to the system through two mechanisms:

- an entry in the Device Database.
- a special file that connects the device to the appropriate device driver.

All device special files reside in the `/dev` directory (either directly under `/dev` or in a directory below it). Special files are provided for any devices delivered with your system, so those devices are available as soon as you boot your system. Whenever you add devices, special files for them are created automatically the next time you boot your system.

Conventions Used to Position a Device File

The conventions used in positioning a device file depend on the type of computer and whether the device is internally or externally controlled. Standard file positions are used to identify disk and tape devices. A distinction is made between character (raw) and block devices. (For details, see *“Logical Types of Devices”* in this chapter.)

- Raw devices usually do not hold files or file systems and their names are positioned in the raw device directory (usually a `ttty` assignment in the `/dev` directory that is linked to a file in the `/dev/rdisk` directory). Terminals, line printers, and tape drives are examples of raw devices.
- Block devices usually hold files and file systems. Their names are positioned in the block device directory (usually `/dev/dsk` for disk devices). Disk drives are examples of devices that are best accessed a block at a time.

Obtaining /dev Directory Listings

To identify the devices installed on your system, check the listings for `/dev` and the directories below it. The directory listings for special files differ from those for ordinary files in several respects. Screen 2-11 contrasts the output produced by running the `ls -l` command on a user’s directory and on the directories under `/dev`.

In Screen 2-11, files `core`, `core.786` and `core.822` are ordinary files; all other files shown are device files. Four of the fields in this listing (the first, fifth, sixth, and last) identify a file as a device.

- The first field consists of a 10-character string; the first character in this string shows the type of file. Entries for regular files have a dash (-) in this position; entries for device files have the letters b (for block devices) or c (for character devices) in this position.
- For a regular file, the fifth and sixth fields show the character count (as one field); for a device file, these fields show the major and minor device numbers for the appropriate device. The major number specifies to the kernel which device driver is used for that device.” **** M. Brown <marcus> **** Feb 19 15:45:43 **** Major numbers serve as an index to the appropriate block or character switch table.

The minor number specifies which device or subdevice is connected to the device driver. Minor numbers are passed to the device driver when a specific device driver function is called. The procedure for determining minor numbers for special files is device dependent.

```

# ls -l
total 1084
-rw-r--r-- 1 root sys 2904 Apr 27 11:52 ar -t | pg
-rw-r--r-- 1 root sys 2904 Apr 27 11:55 ar -u
-rw-r--r-- 2 ronh other 91580 Apr 27 15:21 core
-rw-r--r-- 1 ronh other 91580 Apr 27 15:19 core.786
-rw-r--r-- 2 ronh other 91580 Apr 27 15:21 core.822
-rw-r--r-- 1 root sys 53698 Apr 27 14:01 ron
-rw-r--r-- 1 root sys 128364 Apr 27 09:40 temp
-rw-r--r-- 1 root sys 51976 Apr 27 14:00 z 0 479232
429112
0
/dev/dsk:
total 0
brw-r----- 1 root sys 101, 0 Apr 18 13:06 0s0
brw-r----- 1 root sys 101, 1 Apr 18 13:06 0s1
.
.
.
/dev/rdisk:
total 0
crw-r----- 1 root sys 101, 0 Apr 18 13:06 0s0
crw-r----- 1 root sys 101, 1 Apr 18 13:06 0s1
.
.
.

```

Screen 2-11. Directory Listings for a User's Directory and /dev

Block files can have the same major and minor numbers as character files. However, such file pairs either have different file names or are in different directories (for example, `/dev/rdsk/1s0` and `/dev/dsk/1s0`).

- The last field shows (by its location in the `/dev` directory) how the system interacts with the device. For example, devices in the `/dev/dsk` directory are treated as block devices and devices in the `/dev/rdsk` directory are treated as character (or raw) devices.

Formatting Floppy Diskettes and Hard Disks

Before a disk can be used for storing information, it must be formatted (that is, it must be given an addressing scheme). When a disk is formatted, the disk's surface is mapped into tracks and sectors that can be addressed by the disk controller. A portion of the disk is reserved for data about that specific disk, such as the disk geometry block, which shows how the partitions on the disk are allocated. On a disk, another reserved area maps portions of the disk that may not be usable.

When a used disk is formatted, existing data is erased and the tracks are redefined.

NOTE

Unusable portions of a disk are called bad blocks.

To format a disk use the format command. See **Format (1M)** for detailed information about the format command.

Displaying Information About Storage Devices

- To display information about a cartridge tape:

```
cpio -it < rawdevice
```

where *rawdevice* is the path to the raw device file to the cartridge tape device (such as `/dev/rmt/0m`). The `-it` options of the `cpio` command print a table of contents of the cartridge tape device.

- To display information about a CD-ROM disc:

```
          (device) (mount point)
mount -F cdfs /dev/cd/0 /mnt/cdrom
ls /mnt/cdrom
pkginfo -d /mnt/cdrom/pkgs.dstream
```

- To display disk partition information:

```
format rawdevice
```

where *rawdevice* is the path to the raw device file (such as `/dev/rdisk/1s1`).

- To display a device name for a mounted file system:

```
devnm file_system
```

where *file_system* is the name of a mounted file system (such as `/usr`).

- To display the number of free blocks and virtual nodes:

```
df
```

You may also use `df` to display information about the generic superblock for mounted or unmounted file systems, directories, or unmounted resources. [See **df (1M)** .]

Copying Data on Storage Media

There are two approaches to copying data placed on storage media.

- The first is to duplicate the entire contents of one medium on another.
- The second is to copy specific files from one medium to another.

If the file systems on both media are already mounted, you can use the `cp` command to perform either operation.

If the file systems on both media are not mounted, you can use the **dd** command.

If the file systems on both media are not mounted and the device is formatted in **cpio** format, you can also run **cpio**. Recommended methods for copying data are described in the sections below.

To copy entire file systems quickly from disk to tape or to another disk, use the **volcopy** command. This command is normally for large copy operations.

Examples of the **cp**, **cpio**, and **dd** commands appear in “Quick Reference Guide to Managing Devices” later in this chapter. For information about the **volcopy** command, see **volcopy (1M)**.

Copying Files from Disk to Disk

The **cp** command is commonly used when both the source and the destination file systems are already mounted. [See **mount (1M)**.] This command is generally used for copying small files quickly from one location to another.

Copying the Contents of a Directory from Hard Disk to Tape

Enter

```
ls dirname | cpio -ovB -O device
```

where *dirname* is the path of the directory you want to copy (such as **/tmp**) and *device* is the path to the tape (character) device (such as **/dev/rmt/0m**).

Copying Files from Tape to Hard Disk

1. Enter

```
cd dirname
```

where *dirname* is the path to the directory into which you want to copy the files (such as **/var/tmp**).

2. Enter

```
cpio -iudvB -I device
```

where *device* is the path to the tape (character) device (such as **/dev/rmt/0m** or **ctape1**).

Checking a Corrupt File System

When attempting to mount a file system, you may get an error message saying the file system is corrupt. Use the **fsck** command to check the integrity of any mounted file system and, if possible, to repair it. Refer to “File System Problems” for full information on **fsck**.

Erasing Storage Media

When you want to increase your disk space by removing selected files, it's usually appropriate to run the **rm** command. When you want to erase all data, however, which procedure you follow depends on the type of medium you want to erase.

Use the **format** command discussed earlier to erase an entire hard disk. The **'Mt'** command can be used to erase an entire tape.

Defining Disk Partitions

Each disk may contain between 1 and 7 partitions (numbered from 0 through 6). The partitions on your disk(s) are initially assigned as part of the installation process. By default the partitions are allocated in a standard arrangement. (See the *Operating System Release Notes* for details.)

Do You Need to Change the Disk Partitions?

The default disk partitions definition is fundamentally a compromise. After your system has been in operation for a few months, you may want to evaluate whether the default disk partitions definition is the best possible one for your system. The basic question is whether your users would be better served by a system with a larger number of smaller file systems than you currently have. Begin by analyzing how well your system performs with the existing file system arrangement. See the chapter called "Managing System Performance" in this manual for a description of the command that evaluates performance: **sar (1M)**.

Then answer the following questions:

- What group IDs are defined? Are the right number of groups defined? Are users appropriately assigned to these groups? (See "Creating and Managing User Accounts" in *System Administration* for more information on group IDs.)
- What type of processing is done by these groups? Does their work require temporary data storage? Is there a big difference between the type of processing done by one group and that done by other groups?
- Has software that affects the current plan for space requirements been added to the system? Will such software be added in the future?

The **sar** command provides performance information about your existing file system arrangement. It's described in the "Managing System Performance" chapter in this manual.

Redefining Partitions

To reassign partitions on a boot disk, do a full system restore (with the extended **restore** command), as described under "The Backup and Restore Services" in *System Administration*.

If your users need a large region of temporary space, you may want to create a separate partition for **/var/tmp**. If you have two disks, you can balance your disk loads by

positioning the `/var/tmp` partition at the beginning of a disk other than the one on which your `/ (root)` and `/usr` partitions reside.

If you frequently get console messages warning of insufficient memory, the amount of main memory or the swap area configuration may be insufficient to support users' demands. Before adding more main memory, try to expand the swap area or add more swap areas.

Swap space needs should be carefully calculated. Adding too little swap space will result in unnecessary out-of-memory conditions for your applications. Adding too much swap space will result in too much of your system's memory being locked down for swap space management. The total amount of swap space should be at least 1.5 times the size of physical memory. An initial swap partition is provided on the system disk. If this partition is insufficient, it is recommended that additional swap partitions be added, preferably on other disks.

Every Gigabyte of swap space results in 4 MB of physical memory being used for swap space management. This rule demonstrates that it would be impractical to create a 9 GB swap device on a 64 MB system, as this would result in more than half (36 MB) of physical memory being utilized by the kernel for swap space management.

Expanding the Swap Area

Full partitions on mounted disk drives can be used as additional swap partitions. These partitions are specified by adding lines to the `/etc/vfstab` file similar to the example below:

```
/dev/dsk/ls1 - - swap - yes -
```

Maintaining the Device Database

Because some applications (such as the backup and restore service) require device-specific information, the system maintains a list of attributes for each device in a database known as the Device Database (DDB). This section explains how to create new entries in the DDB, as well as how to check existing entries, and, when necessary, change or remove them. The command that lets you do all these tasks, `putdev(1M)`, is described in the following sections.

The DDB resides in three files:

- `/etc/device.tab`
- `/etc/security/ddb/ddb_sec`
- `/etc/security/ddb/ddb_dsmap`

This database is used by device allocation programs, which depend on device-specific information.

The file `/etc/device.tab` has one record per device alias, consisting of a set of non-security attributes that describe the device.

A secure device alias defines the security related attributes of a physical device, which may contain several logical devices. A logical device alias defines the non-security related attributes of logical devices and the secure device alias to which the logical device maps. For example, a single tape drive could operate at both 1600 and 800 bits per inch (bpi). The tape could have a secure device alias of `tapedrive1`, with logical device aliases (`fasttape` and `slowtape`) for the two densities. The device special file names for the two device aliases would be different (for example, `/dev/tape800` and `/dev/tape1600`).

The DDB contains information on the mapping of security attributes to physical devices and on the mapping of device special files to aliases. This information is used to ensure that device security is maintained.

The file `/etc/security/ddb/ddb_sec` contains entries defining the security attributes for the physical devices on the system. There is one entry in the `ddb_sec` file for each physical device.

The file `/etc/security/ddb/ddb_dsmap` has one record per device special file listed in the `/etc/device.tab` file. Each entry lists the name of a device special file, the logical device alias to which the device special file refers, and the secure device alias to which the device special file refers.

NOTE

You can have a device installed that does not have an entry in this database. However, the device will not be allocatable until the appropriate entries are added to the DDB.

Using putdev to Create a Device Entry in the DDB

Before you can allocate a device, you must add it to the DDB. To create an entry for a device in the DDB, run the `putdev` command, as follows:

```
putdev -a alias [attribute=values [ . . . ]]
```

where *alias* is the name (unique in the database) assigned to the device and *attribute=value* is the assignment of a value to a particular attribute. You can make multiple assignments in this format on the `putdev` command line.

Because there's no limit to the number of attributes that can be defined for a device, the set of attributes defined within the DDB can vary from entry to entry, depending on the device. Only one attribute is mandatory: you must assign an `alias` for every device listed in the DDB. Once you've done that, you can stop or define as few or as many attributes as you need.

Recommended Attributes and Default Values

Although only one attribute is required in each entry, we think you'll probably find it useful to include at least four: `alias`, `desc` (description), `medium`, and `type`.

Table 2-8 shows the default values for these attributes when defined for the five most common types of devices.

Table 2-8. Recommended Attribute and Default Values

Alias	Description	Medium	Type
<code>ctapeN</code>	tape drive N	tape	ctape
<code>dMpartN</code>	disk M partition N	n/a	dpart
<code>diskN</code>	integral disk drive N	n/a	disk
<code>cdromN</code>	CD-ROM Drive N	cd-rom	cd-rom

n/a = not applicable

Standard Attributes

If you want to provide more detailed information about a device, you may want to include some of the 28 standard attributes for a logical device alias in the DDB, as described below.

And remember: you're not restricted to this set. You can add any attribute you like to a device entry, simply by specifying it in the format shown above (*attribute=value*) on the `putdev -a` command line.

Attribute	Description
<code>alias</code>	The unique name by which a device is known. No two devices in the database may share the same alias name. The name is limited in length to 14 characters and should contain only alphanumeric characters and the following special characters if they are escaped with a backslash: underscore (<code>_</code>), dollar sign (<code>\\$</code>), hyphen (<code>\-</code>), and period (<code>\.</code>).
<code>bdevice</code>	The pathname to the block special device node associated with the device, if any. The associated major/minor combination should be a unique block device number and should match the number associated with the <code>cdevice</code> field, if any. (You're responsible for ensuring these pathnames are unique in the database.)
<code>bdevlist</code>	A list of additional pathnames of block device special files mapping to the same logical or secure device. The items in the list are separated by commas, and each item must be the absolute pathname of a device special file, with a maximum length of PATH_MAX .
<code>bufsize</code>	The block size to be used by <code>pkgtrans</code> and <code>pkgadd</code> when transferring packages to and from tape.
<code>capacity</code>	The capacity of the device or of the typical volume, if removable.

Attribute	Description
cdevice	The pathname to the character special device node associated with the device, if any. The associated major/minor combination should be a unique character device number and should match the number associated with the <code>bdevice</code> field, if any. (You're responsible for ensuring these pathnames are unique in the database.)
cdevlist	A list of additional pathnames of character device special files mapping to the same logical or secure device. The items in the list are separated by commas, and each item must be the absolute pathname of a device special file, with a maximum length of PATH_MAX .
cyl	Used by the command specified in the <code>mkfscmd</code> attribute.
desc	A description of any instance of a volume associated with this device (such as disk).
dpartlist	The list of disk partitions associated with this device. Used only if <code>type=disk</code> . The list should contain device aliases, each of which must have <code>type=dpart</code> .
dparttype	The type of disk partition represented by this device. Used only if <code>type=dpart</code> . It should be either <code>fs</code> (for file system) or <code>dp</code> (for data partition).
erasescmd	The command string that, when executed, erases the device.
fmtcmd	The command string that, when executed, formats the device.
fsname	The file system name on the file system administered on this partition, as supplied to the <code>/usr/sbin/labelit</code> command. This attribute is specified only if <code>type=dpart</code> and <code>dparttype=fs</code> .
gap	Used by the command specified in the <code>mkfscmd</code> attribute.
mkdtab	If this field is set to true (omitted otherwise), this means the devices entry was added by the command <code>mkdtab</code> and should be removed only by <code>mkdtab</code> .
mkfscmd	The command string that, when executed, places a file system on a previously formatted device.
mountpt	The default mount point to use for the device. Used only if the device is mountable. For disk partitions where <code>type=dpart</code> and <code>dparttype=fs</code> , this attribute should specify the location where the partition is normally mounted.
nblocks	The number of blocks in the file system administered on this partition. Used only if <code>type=dpart</code> and <code>dparttype=fs</code> .
ninodes	The number of inodes in the file system administered on this partition. Used only if <code>type=dpart</code> and <code>dparttype=fs</code> .
norewind	The name of the character special device node that allows access to the serial device without rewinding when the device is closed.
pathname	Defines the pathname to an inode describing the device (used for non-block or character device pathnames, such as directories).

Attribute	Description
secdev	The name of the secure device alias. By default, the value of <code>alias</code> is used. (See above for the definition of <code>alias</code> .) When a name other than the value of <code>alias</code> is specified, the results may vary, depending on whether and how that name has been defined. See <code>putdev (1M)</code> for details.
type	A token that represents inherent qualities of the device. Standard types include: <code>9-track</code> , <code>ctape</code> , <code>disk</code> , <code>directory</code> , and <code>dpart</code> . Set Table 2-8 for a list of recommended attributes for these device types.
volname	The volume name on the file system administered on this partition, as supplied to the <code>/usr/sbin/labelit</code> command. Used only if <code>type=dpart</code> and <code>dparttype=fs</code> .
volume	A text string used to describe any instance of a volume associated with this device. This attribute should not be defined for devices that are not removable. See Table 2-8 for example volume descriptions.

The standard device attributes that can be defined for a secure device alias in the DDB (by invoking the `putdev` command) are listed below. Other attributes (non-security related) can be assigned to a secure device alias when appropriate.

Attribute	Description
mode	A flag that shows whether the device mode has been defined as <code>static</code> or <code>dynamic</code> . If a device is <code>public</code> and its mode is <code>static</code> , strict tranquillity is applied for changing the level of that device. If a device is <code>public</code> and its mode is <code>dynamic</code> , the level of the device can be changed while the device is in use; MAC access checks are performed for each operation. For a full description of mode, see “ <i>Device Security Attributes</i> ” earlier in this chapter.
other	The other authorization, which defines the authorization permissions for <code>other</code> in the set of <code>user</code> , <code>group</code> , <code>other</code> access permissions. It can take one of two values: <code>y[es]</code> or <code>n[o]</code> . If it is not defined, the authorization permission is <code>no</code> . This attribute is optional.
range	This field defines the security level range of the device. It should be a <code>hilevel-lolevel</code> pair, where <code>hilevel</code> and <code>lolevel</code> are both security level aliases or fully qualified level names, with the high level given first. A minus sign(-), is the delimiter between <code>hilevel</code> and <code>lolevel</code> . These levels are stored in the DDB as LIDs (level identifiers) converted to ASCII characters. Before the LIDs are saved in the DDB, the system validates them against the level translation database and ensures that the <code>hilevel</code> dominates the <code>lolevel</code> . If the <code>hilevel</code> does not dominate the <code>lolevel</code> , <code>putdev</code> fails.

Attribute	Description
startup	A flag (y [es]/n [o]) that shows whether the device is to be allocated during system startup. This allocation will fail if pathnames for the device special files do not exist or if they are not on an sfs file system. If startup is defined and the required security attributes are not defined, the allocation will fail. This field is optional. If startup is not defined, the value of this field is assumed to be no.
startup_level	This field contains the MAC level at which the device should be set during startup. The level can be specified as either a level alias or a fully qualified level name. This field is optional.
startup_group	This field contains the access permissions for group , which must be specified as <i>GID</i> >rwX. The <i>GID</i> must be a group ID defined in the /etc/group file. The > character is a delimiter between the group name and the list of access permissions. To deny a permission, replace the appropriate letter with a hyphen (-). This field is optional.
startup_other	This field contains the access permissions for other , which must be specified as >rwX. To deny a permission, replace the appropriate letter with a hyphen (-). This field is optional.
startup_owner	This field contains the access permissions for the user who will own the file at startup time. The value must be specified as <i>UID</i> >rwX. The <i>UID</i> must be a user ID defined in the /etc/passwd file. The > character is a delimiter between the group name and the list of access permissions. To deny a permission, replace the appropriate letter with a hyphen (-). For example, to deny execute permissions, specify startup_owner as <i>UID</i> >rw-. This field is optional.
state	A flag that determines the state at which the device can be allocated. This field can be either private , public , or pub_priv . If the field is pub_priv , the device can be allocated as either private or public . If the device is private , only privileged processes have access to it. Unprivileged processes have access only to public devices. If the startup attribute is enabled, the device is allocated as private at system startup when either private or pub_priv is set. This attribute must be defined.
ual_enable	This flag enables the User Authorization List (UAL) defined by the user and other attributes. It can take one of two values: y [es] or n [o]. The default value is no (no users are authorized to use the device). If the value is yes, the UAL is examined to authorize a user to use the device, and the others attribute must also be defined. This field is optional.
users	The UAL, which defines the allocation permissions for users. Each item is a <i>user_id:allocation_permission</i> pair, separated by a > character. The device control list items are separated by commas. Each <i>user_id</i> must be unique in a device entry and must be defined in /etc/passwd when the attribute is defined. This attribute is optional.

Using `getdev` to List Devices

To get a list of the devices available on your system, run the `getdev` command. (Actually, `getdev` produces a list of the devices defined in your DDB; if you have devices on your system that aren't defined in the DDB, they won't be included in the output of `getdev`.) For example: To list devices, enter

```
getdev [-ae] [criteria [. . .]] [device [. . .]]
```

where *device* is the name of the device or devices you want listed and *criteria* defines selection criteria for the list.

If you run `getdev` with no options or arguments, you get a list of all devices in the DDB.

Specify *criteria* with the four expression types defined in this chapter under “*Listing Devices*”, in the section “*Specifying Criteria for a Listing*”. To be included in the list, a device must match at least one of the criteria given. Use the `-a` option to request that devices match all the given criteria before being included in the list.

All devices named will be included in the list, unless you use the `-e` option, which specifies that the devices named should be excluded from the list.

The following examples show a list of the devices defined in the DDB

```
# getdev
ctape1
disk1
d1part1
d1part2
d1part3
d1part4
d1part5
d1part6
d1part7
d1part8
disk2
d2part1
d2part2
d2part3
d2part4
d2part5
d2part6
d2part7
d2part8
```

Creating Customized Device Lists

You can also request customized lists: lists that include only the information you specify. For example, you may want a list of a subset of the devices on your system, or a list of devices that fulfill criteria you provide. By requesting a customized list, you can use the `getdev` command to get answers to questions such as the following:

- For what devices is a format command defined?

- What devices besides `disk` are set up in the DDB?
- What devices are part of physical `disk1`?

Specifying Devices for a Listing

To request information about particular devices, name those devices on the `getdev` command line, as follows:

```
getdev device [device [ . . . ]]
```

Here *device* is the name of a device you want included in the list. As shown here, you can specify multiple devices.

If you want information about all but a few devices, it may be easier to frame your request in exactly that way: by specifying the devices you want excluded. Use the `-e` option, as follows:

```
getdev -e device [device [ . . . ]]
```

The list you get will include all devices except those you typed on the command line.

Specifying Criteria for a Listing

If you want a list of devices that meet certain criteria, specify the latter on the `getdev` command line, as follows:

```
getdev [-a] criterion [criterion [ . . . ]]
```

You can specify *criterion* in one of the following four formats or “expressions”:

<i>attribute=value</i>	selects all devices for which <i>attribute</i> is defined and is equal to <i>value</i>
<i>attribute!=value</i>	selects all devices for which <i>attribute</i> is defined and does not equal <i>value</i>
<i>attribute:*</i>	selects all devices for which <i>attribute</i> is defined
<i>attribute!:*</i>	selects all devices for which <i>attribute</i> is not defined

You can include one or more expressions (each pair of which should be separated by a space) on a command line. If the `-a` option is used, only those devices matching all criteria will be included. If it isn't, any device that satisfies at least one criterion in the list will be included.

Sample Requests for Customized Lists

The following is a list of questions you might have about the devices on your system. Each question is followed by the `getdev` command line you would execute to obtain an answer. (Each answer will be in the form of a list of devices.)

- Which devices besides `disk1` are set up in the DDB?

```
getdev -e disk1
```

- For which devices is the `fmtcmd` attribute defined?

```
getdev fmtcmd:*
```

- For which devices is the `fmtcmd` attribute not defined?

```
getdev fmtcmd!:*
```

- For which devices is the `type` attribute defined as `disk` or the `part` attribute defined?

```
getdev type=disk part:*
```

- For which devices is the `type` attribute defined as `disk` and the `part` attribute defined?

```
getdev -a type=disk part:*
```

(Note that this example differs from the previous one by requiring that a device adhere to both criteria, not just one.)

- For which devices in the named list (`disk1`, `disk3`, and `disk5`) is the `type` attribute defined as `disk` or the `part` attribute defined?

```
getdev type=disk part:* disk1 disk3 disk5
```

- Which devices are part of physical `disk1`?

```
getdev secdev=disk1
```

Listing Attributes

To obtain a list of the attributes defined (in the DDB) for a device, along with their values, run the `devattr` command. The output will appear in one of two formats: the short format (the default) or the verbose.

- The default format is a list of only the values assigned to the relevant attributes; the names of the attributes with which these values are associated are not output.
- The verbose format (requested with the `-v` option) is a list of attribute-value assignments in the format `attribute=value`.

For a list of device attributes, type

```
devattr [-v] device [attribute [ . . . ]]
```

where *device* is the pathname or alias of the device for which you want a list of attributes and *attribute* is the specific attribute for which you want to know the value. For example, if you want the value for the `mountpt` attribute for a device called `disk1`, enter

```
# devattr /disk1 mountpt
/install
```

The value of the `mountpt` attribute for `disk1` is `/install`.

If you don't name a specific attribute, all attributes associated with the device are shown in alphabetical order. Thus, for example, suppose you want to see a list of all attributes defined for `disk`. Execute `devattr` without specifying any attributes.

```
cdevice=' /dev/rdisk/ls0'
copy='true'
display='true'
mode='static'
mountpt='/dgk1'
range='SYS_RANGE_MAX-SYS_RANGE_MIN'
state='pub_priv'
```

Using `putdev` to Change the Values of Attributes

To change existing values for the attributes of a device, to delete attributes, or to add new attributes to a device entry, run the `putdev` command, as follows:

```
putdev -m device attribute=value [attribute=value [ . . . ]]
```

where *device* is the pathname or alias of the device entry being modified, *attribute* is the name of the attribute being modified, and *value* is the value that should be assigned to the attribute.

If the specified attribute is currently listed for this device in the Device Database, `putdev -m` modifies its value. If the attribute does not exist, it is added and given the specified value (*value*), except in the case of `range`, `state`, and `mode` (the essential security attributes), all three of which must be defined together. If any other security attribute is not defined and you try to add it to an alias for a device that is not secure (by specifying `putdev -m`), the `putdev` command fails.

The `alias` attribute cannot be changed with `putdev -m`. This prevents an accidental modification or deletion of a device alias from the DDB.

Deleting Attributes from a Device Entry

To delete the definition of an attribute from a device entry, run `putdev` with the `-d` option, as follows:

```
putdev -d device attribute
```

Here *device* is the name of the entry from which an attribute definition will be deleted; *attribute* is the name of the attribute. For example, to remove the attribute `volume` from the entry for `disk`, enter


```
putdev -d disk volume
```

To delete the value of an attribute while keeping the attribute in the device entry, use the same format as above with the following exception: assign the attribute the value of `NULL`. For example, to remove the value of the `volume` attribute while retaining `volume` in the device entry, execute

```
putdev -m disk volume=““
```

Removing a Device Entry

To delete an entry for a device from the DDB, run **putdev** as follows:

```
putdev -d device
```

where *device* is the pathname or alias of the device to be deleted.

Managing Devices in Groups

If you're responsible for a large number of devices, you may want to organize them into groups for more efficient administration. For example, you may want to classify several disks as a single "hard-disk" group. Once you've done that, you'll be able to perform most administrative procedures on all your disks simultaneously, simply by issuing a single command and specifying the group.

The Device Groups Database

To facilitate the handling of devices in groups, the operating system provides a database (in `/etc/dgroup.tab`) in which you can list your groups and their component devices. The following section explains how to create groups by creating entries in the this database.

Using **putdgrp** to Create a Device Group

Groups are created when entries for them are added to the device groups database. Although you may peruse this database, you shouldn't edit it directly. Instead, edit it through the **putdgrp (1M)** command.

To add an entry for a new group, enter: **putdgrp**, as follows:

```
putdgrp group_name alias [alias [ . . . ]]
```

where *group_name* is the name of the group you're creating and *alias* is the alias for each device being included.

For example, to create a disk group for two disks (with aliases `disk1` and `disk2`) , run

```
putdgrp disk disk1 disk2
```

Using putdgrp to Remove a Device Group

To delete an entry for a device group from the device group database, run

```
putdgrp -d group_name
```

where *group_name* is the token associated with the group you want to remove.

Using getdgrp and listdgrp to Maintain the Device Groups Database

From time to time, after your database has been set up, you'll want to get information from it. You can extract lists of groups and their component devices by executing the `getdgrp` and `listdgrp` commands, respectively.

Using getdgrp to Get a List of Groups

To find out which device groups are already defined in your database, enter

```
getdgrp [-ae] [criteria [. . .]] [group_name [. . .]]
```

where *group_name* is the name of the device group or groups you want included in the list generated by this command. *criteria* defines selection criteria for the list.

If you run `getdgrp` with no options or arguments, you get a list of all device groups.

Specify *criteria* with the four expression types defined in this chapter in the following section “*Specifying Criteria for Your List*”. To be included in the list, a device group must have at least one member that meets at least one of the given criteria. However, you can require a device group to have at least one member that meets all criteria before being included in the list. To do so, use the `-a` option.

All device groups named will be included in the list unless you use the `-e` option, which specifies that the named device groups should be excluded from the list.

Requesting a Customized List

At times you may want to query the database for answers to questions such as the following:

- To which device groups, besides `disk`, do I have access?
- Which groups have devices for which the `fmtcmd` attribute is defined?

There are three ways of specifying the groups you want included in the list you request. When you run `getdgrp`, include one of the following on the command line:

- a list of device groups to be included

- criteria that must be met for a member of a group to be included
- a list of device groups and criteria

Specifying Device Groups for Your List

Name device groups on the **getdgrp** command line by running

```
getdgrp [-e] group_name [group_name [. . . ]]
```

where *group_name* is the name of the device groups you want included in the list.

If you want a list of all but a few groups, it may be more convenient to name the ones you don't want than to specify the ones you do. To do this, run **getdgrp** with the **-e** option and specify the groups to be excluded.

Specifying Criteria for Your List

Provide criteria for groups to be included by running **getdgrp** as follows:

```
getdgrp [-a] criteria [criteria [. . . ]]
```

where *criteria* are specified with one or more of the following expressions.

<i>attribute=value</i>	selects all device groups with at least one member whose attribute <i>attribute</i> is defined and is equal to <i>value</i>
<i>attribute!=value</i>	selects all device groups with at least one member whose attribute <i>attribute</i> is defined and does not equal <i>value</i>
<i>attribute:*</i>	selects all device groups with at least one member whose attribute <i>attribute</i> is defined
<i>attribute!:*</i>	selects all device groups with at least one member that does not have the attribute <i>attribute</i> defined

You can define a list of criteria simply by supplying multiple expressions, separated by white space. To be included in the list, a device group must have at least one member that meets at least one of the criteria, unless the **-a** option is used. If **-a** is specified, a device group must have at least one member that meets all criteria.

Examples of Customized Lists

The following is a list of questions you might have about the devices on your system. Each question is followed by the **getdgrp** command line you would execute to obtain an answer. (Each answer will be in the form of a list of device groups.)

- To which device groups, besides the `ctape` group, do I have access?

```
getdgrp -e ctape
```

- Which device groups have members for which the `fmtcmd` attribute is defined?

```
getdgrp fmtcmd:*
```

- Which device groups have members for which the `fmtcmd` attribute is not defined?

```
getdgrp fmtcmd!:*
```

- Which device groups have members for which the `type` attribute is defined as `disk` or the `part` attribute is defined?

```
getdgrp type=disk part:*
```

- Which device groups have members for which the `type` attribute is defined as `disk` and the `part` attribute is defined?

```
getdgrp -a type=disk part:*
```

(Note that this example differs from the previous one by requiring that a group include a member that fulfills both criteria, not just one.)

- Which device groups in the named list (`group1`, `group3`, `group5`) have members for which the `type` attribute is defined as `disk` or for which the `part` attribute is defined?

```
getdgrp type=disk part:* group1 group3 group5
```

Using `listdgrp` to Get a List of Device Group Members

To get a list of the devices that make up a group, enter

```
listdgrp group_name
```

For example:

```
# listdgrp disk  
disk1  
disk2  
#
```

The output shows the group `disk` is composed of two members: `disk1` and `disk2`.

Managing Device Reservations

NOTE

If the Enhanced Security Utilities are installed and running on your system, run `admalloc` instead of `devreserv` for device use.

To accommodate administrators trying to manage multiple devices, the operating system provides a mechanism through which they can make “reservations” for devices. The term

“device reservation” does not mean you can request use of a device in the future; instead, it means you can prevent other users or processes from using a device while you use it.

Using `devreserv` and `devfree` in Device Reservation

Device reservations are made through the `devreserv` command. This command places the name of the specified device (and the process ID of the process through which the reservation was made) on the device reservations list. Then, whenever a process requests that device, the device reservations list is checked, the name of the specified device is found, and the request is denied.

When the user or process no longer needs a device, the reservation is canceled (or “freed”) through the `devfree` command: the name of the device is removed from the list, making the device available for a new reservation.

The `devreserv` and `devfree` commands are used on a voluntary basis; they do not actually prevent and allow access to a device. Rather, they provide a centralized bookkeeping point for those who agree to use it. A device that has been reserved cannot be used by processes that use the device reservation function until the reservation has been canceled.

If no reservation has been made and a process requests a particular device, the reservations list is checked but the name of the specified device is not found. Therefore the device is assumed to be available. The device is made available to the requesting process and its name is added to the reservations list, thus ensuring any subsequent requests to reserve it will be denied.

Use of the device reservation system is voluntary; the reservation system does not place any constraints on access to devices that have been reserved. It works only if users and processes respect the system: that is, if users or processes denied access to devices that have been reserved honor those reservations by not using those devices. (Not all processes use the device reservation system. Those that don't never request a device reservation so they don't check the reservations list before trying to use a device.) There is no mechanism to prevent a user or process from using a reserved device.

As an administrator, you can reserve a device for exclusive use, release a reservation once you are finished with it, and check the status of a device.

NOTE

Because it's possible for applications to call devices directly—without invoking `devreserv`—use of the `devreserv` command does not guarantee exclusive use of a device.

Using `devreserv` to Check Device Reservation Status

There are two ways to check the status of device reservations.

- To list all devices currently reserved, enter

devreserv

- To list all devices currently reserved to a particular key, enter

devreserv *key*

Using devreserv to Reserve a Device

To reserve a device for exclusive use, enter

devreserv *key device*

where *key* is a positive integer that will be associated with this reservation and must be used later to free this particular reservation. The key should be unique. A suggested convention is to use the process ID of the calling process (specified by \$\$ in the shell) as the key. *device* may be either

- the alias or pathname of a device that should be reserved
- a list of devices

For example, to reserve a device with the alias `diskette1` to the current shell process, enter

devreserv \$\$ *diskette1*

If *device* is a list, the first available device in the list is reserved.

Using devfree to Free a Reserved Device

1. To free a device reservation, enter

devfree *key* [*device* [. . .]]

where *key* is the “reservation number” to which the device has been reserved and *device* is the alias or pathname of the device(s) that should be freed from reservation.

2. If you want to cancel the reservations for all the devices associated with a particular process ID, enter

devfree *key*

Device Names and Default Partitions

This section describes the device names and default partitioning for the operating system.

Disks

Disk device files use `/dev/dsk` for the block device and `/dev/rdsk` for the raw (character) device. The device name is a character string that identifies the device number and the partition (also called slice). This has the following format:

dnumssnum
where

- *dnum* is the logical disk drive number.
- *snum* is the disk slice number (a number between 0 and 7).

Examples are `1s0`, `3s1`, and `10s7`.

Disk Partitions

Each disk drive accessible through the operating system must be partitioned properly. This partitioning is done during installation of the system (at least for the system disk) or when new disk drives are added to the system.

The utility **format (1M)** is used to examine and/or modify the partitioning of a particular disk. Note that this utility may be called interactively while the system is running or via a standalone fashion. Partition 7 on a disk contains the Volume Table of Contents (VTOC), the information about how this logical disk drive is divided into slices and the physical description of the underlying disk (geometry data). **format (1M)** may be used to specify the sizes of partitions 0 through 6. During system installation, the system disk is generally divided across partitions 0 through 3.

partition 0	root (/) file system (accessed via <code>/dev/root</code>)
partition 1	initial swap partition (accessed via <code>/dev/swap</code>)
partition 2	<code>/usr</code> file system (accessed via <code>/dev/usr</code>)
partition 3	<code>/var</code> file system (accessed via <code>/dev/var</code>)

The slicing of the disk described above applies only to the system disk drive. Additional disk drives will usually be sliced differently.

Tapes

A tape drive (whether for cartridge, 8-millimeter, or DAT tape) is a character device and uses device names in the `/dev/rmt` directory. The device name is a character string that identifies the device number plus additional options. The format is:

`tnumsdens{speed}{n}`

where

- `tnum` is the logical tape drive number.
- `dens` is the density - high (h), medium (m), intermediate (i), or low (l). Note that a particular drive type may or may not support multiple densities.
- `speed` optionally provides the rotation speed: fast (f) or slow (s). Note that a particular drive type may or may not support multiple speeds.
- `n` indicates no rewind on close; the default is rewind.

Virtual Partition

Virtual Partition (VP) is a pseudo disk device driver which combines multiple disk partitions into a single virtual partition. A configured VP appears to the rest of the system as a single partition though it is actually made up of multiple real, member partitions.

This technology, known as RAID (Redundant Arrays of Independent Disks), may contain up to 7 levels (0 - 6). VP implements levels 0 and 1. A VP can be configured either as a striped virtual partition (RAID, level 0) or as a mirrored virtual partition (RAID, level 1)

Striped VP

A striped VP divides the contiguous data of the VP into slices. The slice size is configurable, the default being 32 kilobytes. Up to 16 partitions are combined into a single VP by distributing these slices across the member partitions in an alternating fashion. This is called disk *striping*. For example, if a VP has two member partitions, all the even slices are on the first member partition while all the odd slices are on the second member partition (see Figure 2-8). Advantages and disadvantages of VP are:

- disk striping for some applications can significantly increase performance by parallelizing large I/Os across multiple disks
- a single file system can be distributed across multiple disks, increasing the maximum file system size. Without VP, the file system size is limited to the size of a single partition
- because a single file system is distributed across multiple disks, the reliability of the file system as a whole decreases (the probability of a file system failing increases with the number of disks used by the file system)

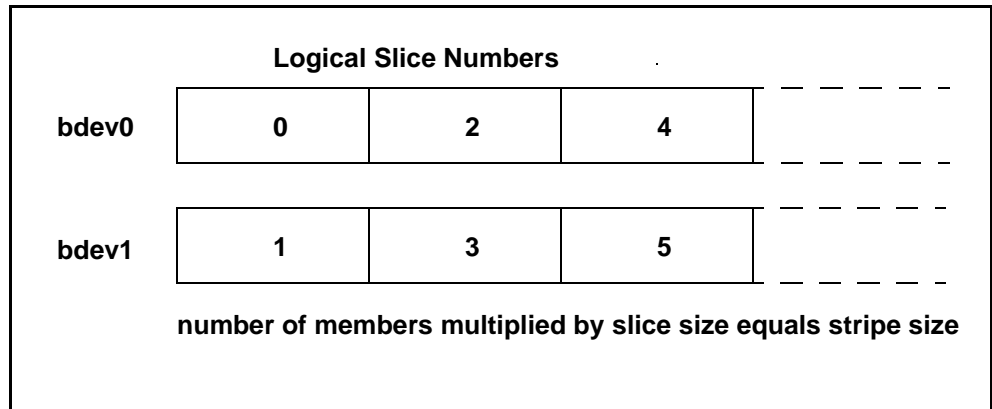


Figure 2-8. Disk Striping on VP with Two Member Partitions

Mirrored VP

A mirrored VP replicates the data of the VP on two or more member partitions. The member partitions are identical copies of each other, except for the last 512-byte block which is used by the VP. If one member fails, the other member can handle requests until the failed member is restored. The primary benefit of mirrored VP is increased availability. A system using mirrored VP can survive a disk failure without unscheduled down time or loss of data. Figure 2-9 illustrates a mirrored partition.

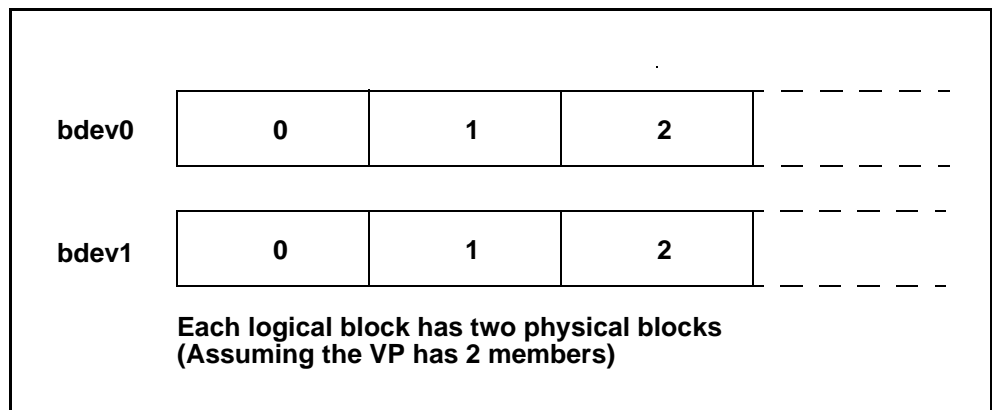


Figure 2-9. Mirrored VP with Two Member Partitions

VP Driver

Configuration

The VP driver is an optional device driver with the name “vp”.

The administrator configures `vp` into the kernel like any other driver, by modifying the `sdevice.d/vp` file. For example, the following `sdevice.d/vp` file would configure a kernel with a max of 13 virtual partitions:

```
vp      Y      13
```

VPs are numbered from 1 to `VP_UNITS`. VP unit 0 is reserved by the system. The major number reserved for VPs is 105 (decimal). The minor number is the VP number. The maximum number of member partitions per VP is a hard limit of 16.

The administrator also has the option of using the `config` utility to enable the `vp` device driver module. See the “Configuring and Building the Kernel” chapter in Volume 2 for details.

Geometry

This section discusses how member partitions are mapped into virtual partitions. The last physical disk block of the member partitions is used to hold VP control data so the sizes of the member partitions are reduced by one disk block before making the following calculations.

There is a special case when stripe size is zero for a striped VP. In this case, the member partitions are simply concatenated together, not striped. This configuration might be used to extend the file system size without improving performance. In this case, the member partitions are utilized without rounding or truncation of the members (save the one disk block for VP control data). The size of a striped VP with stripe size zero is simply the sum of the sizes of the members.

If the stripe size is non-zero, the sizes of striped VP members are rounded down to a multiple of the slice size and then truncated to the size of the smallest member partition. The size of a striped VP is the size of the smallest member partition times the number of member partitions. To utilize the maximum capacity of the member partitions, all the members must be the same size.

The size of a mirrored VP is equal to the size of the smallest member partition (minus 1 block for the VP control data).

Initialization

The `vpinit` command configures or un-configures a VP. Usage is as follows:

```
vpinit [-c] [-s slicesize] vpdev bdev1 [bdev2 ...]
vpinit -m [-c -f] vpdev bdev1 bdev2
vpinit -r vpdev
vpinit -u vpdev
vpinit -d N
```

-c (create) creates a VP. This writes VP control blocks to the member partitions. A VP must be configured the first time with the `-c` option. After that, the `-c` option should not be used to configure the VP.

<i>slicesize</i>	is the size of the slice on each disk in disk blocks (512-bytes). The default is 64 blocks (32 Kb). The slice size can be any number up to the size of the smallest member partition. A value of 0 also implies concatenation only, no striping.
<i>bdev1 ... bdevN</i>	are the block device nodes of the physical partitions that make up the VP.
-m	signifies this is a mirrored VP.
-f	force “synchronized state” for a mirrored VP. Only meaningful when creating a mirrored VP. Without this option, a restore operation is required. This option should only be used if the contents of both partitions are identical or if a subsequent newfs (1M) will be done on the virtual partition before it is used.
-r	restore the failed or uninitialized member of a mirrored VP. This is a required step after a mirror VP is created unless the -f option is specified.
-u	un-configures a VP. This option must be used before reconfiguring a VP.
-d	spawn VP daemons required for mirrored VP. <i>N</i> is the number of daemons to start, 2 is generally sufficient but the number may be increased if there is heavy usage. This daemon does error retries and error reporting and is required to use mirrored VPs. Generally, the command to spawn these daemons should be done within the script /etc/vptab .

See the online manual page **vpinit (1M)** for more details.

Device Naming Convention

The VP driver is accessed through the device nodes **/dev/rdsk/vpN** (character or raw device node) and **/dev/dsk/vpN** (block device node) where *N* is the VP number.

Vpstat Command

The **vpstat** command output status for a given VP. Usage is as follows:

```
vpstat [-v] /dev/rdsk/vpN [...]
```

Example output for a VP:

```
unit 1: size 8287.938 Mb (16973696 sectors), 2 members, 32 Kb slice size
```

The **-v** option (verbose) outputs additional information for each member partition.

Example verbose output for a VP:

```
unit 1: size 8287.938 Mb (16973696 sectors), 2 members, 32 Kb slice size
      member 0: major 101, minor 127488 : /dev/dsk/2s0
            phys size 4144.000 Mb, virt size 4143.969 Mb
      member 1: major 101, minor 127744 : /dev/dsk/3s0
            phys size 4144.000 Mb, virt size 4143.969 Mb
```

Example output for a mirrored VP:

```
unit 1: size 4144.000 Mb (8486911 sectors), mirror state: synchronized
```

Example verbose output for a mirrored VP:

```
unit 1: size 4144.000 Mb (8486911 sectors), mirror state: synchronized
      member 0: major 101, minor 127488 : /dev/dsk/2s0
            phys size 4144.000 Mb, virt size 4144.000 Mb
            up    open    0 read errors
      member 1: major 101, minor 127744 : /dev/dsk/3s0
            phys size 4144.000 Mb, virt size 4144.000 Mb
            up    open    0 read errors
```

For a mirrored VP, the mirror state will be one of:

synchronized - both member partitions are operational.
primary down - the primary member partition is downed.
secondary down - the secondary member partition is downed.

/etc/vptab

The script **/etc/vptab** contains **vpinit** commands used to initialize VPs when the system is booted.

If **/etc/vptab** exists it should contain a list of **vpinit** commands to initialize each virtual partition (but not create). Each **vpinit** command would have the same format as that used when the VP was created, except the **-c** option is omitted.

The **/etc/vptab** script is executed early in the boot process, so that VPs are established before file system checks are done.

NOTE

If mirrored VP's are being used, then the VP daemons must be running before **vpinit** commands are executed. The VP daemons are spawned by using the command: **vpinit -d 2**.

Example VP Set-Up

1. If the new VP will be made over an existing VP, un-configure the existing VP:

```
vpinit -u /dev/rdisk/vp1
```

2. Create a VP with default slice size:

```
vpinit -c /dev/rdisk/vp1 /dev/dsk/2s0 /dev/dsk/3s0
```

3. Make a file system with 32 Kb blocks on the VP:

```
newfs -F <sfs> -b32768 -g 0 [filesystem specific options]
/dev/rdisk/vp1
```

where <sfs> is one of ufs, sfs, xfs or xfsd filesystem type.

4. Mount the file system:

```
mount -F sfs /dev/dsk/vp1 /vp1
```

5. Edit **vptab** file to include **vpinit** command used to initialize the VP:

```
vpinit /dev/rdisk/vp1 /dev/dsk/2s0 /dev/dsk/3s0
```

6. Update the **vfstab** File System Table to include file system made on VP.

Example Mirrored VP Set-Up

Initial mirrored VP set-up

An example initial mirrored VP set-up is shown below:

1. If the new VP will be made over an existing VP, un-configure the existing VP:

```
vpinit -u /dev/rdisk/vp1
```

2. Ensure that the VP daemons are running. The daemons are required and are used for error retries and error reporting. If the daemons are not running, issue the command:

```
vpinit -d 2
```

Generally, the command to spawn these daemons should be done within the script **/etc/vptab**.

Variable Steps for Mirrored VP Set-Up

Some steps for setting up a mirrored VP differ based on the conditions of the member partitions and also whether any data contained on the member partitions will be preserved. There are three alternatives:

- A. Data on the member partitions will not be preserved.

1. Create the mirrored VP and force “synchronized state”:

```
vpinit -m -f -c /dev/rdisk/vp1 /dev/dsk/2s0 /dev/dsk/3s0
```

2. Make a file system with 32 Kb blocks on the mirrored VP:

```
newfs -F <sfs> -b32768 -g 0 [filesystem specific options]  
/dev/rdisk/vp1
```

- B. Data on the member partitions will be preserved and the primary and secondary member partitions have identical contents.

1. Create the mirrored VP and force “synchronized state”:

```
vpinit -m -f -c /dev/rdisk/vp1 /dev/dsk/2s0 /dev/dsk/3s0
```

- C. Data on the member partitions will be preserved and the primary and secondary member partitions differ. In this case, the contents of the primary member partition will be duplicated on the secondary member partition.

1. Create the mirrored VP:

```
vpinit -m -c /dev/rdisk/vp1 /dev/dsk/2s0 /dev/dsk/3s0
```

2. Restore secondary member partition:

```
vpinit -r /dev/rdisk/vp1
```

NOTE

This could take a rather long time on large file systems (> 10 minutes for > 4 GB).

Final Steps for Mirrored VP Set-Up

The subsequent steps for setting up a mirrored VP are the same regardless of the “alternative” set-up used. Example final set-up:

1. Mount the file system:

```
mount -F <sfs> /dev/dsk/vp1 /vp1
```

2. Edit **vpstab** file to include **vpinit** command used to initialize the mirrored VP and also to start **vp** daemons:

```
vpinit -d 2  
vpinit -m /dev/rdisk/vp1 /dev/dsk/2s0 /dev/dsk/3s0
```

3. Update the vfstab File System Table to include file system made on mirrored VP. See online manual page **vfstab(4)**.

Restoring Faulty Member of a Mirrored VP

A member of a mirrored VP may be automatically taken offline if errors are encountered. This will occur on any write error or if there are N read errors (N is configurable and is specified in the file `/etc/conf/pack.d/vp/Space.c`).

Upon replacement of the faulty disk, it can be restored using `vpinit (1M)`. An example:

```
vpinit -r /dev/rdisk/vp1
```

When a mirrored VP is created, if the `-f` option is not specified (force synchronized state) then the secondary member partition will be automatically downed. A restore is required to duplicate the primary and secondary member partitions and bring the secondary partition on-line.

NOTE

This could take a rather long time on large file systems (> 10 minutes for > 4 GB).

Quick Reference Guide to Managing Devices

The administrative commands in this chapter are based on shell commands. If you prefer, however, you can do many tasks through a set of administration menus, instead. (Using the menus may be convenient if you're administering a system for the first time.) To use these menus, enter

```
sysadm storage_devices
```

The main menu for managing data storage devices will appear as shown in Screen 2-8.

```
1 Storage Device Operations and Definitions
add - Add Storage Device
copy - Makes Duplicate Copies of Storage Volumes
devices - Device Alias and Attribute Management
display - Displays Information About Storage Devices
erase - Erases the Contents of Storage Volumes
format - Formats Removable Volumes
remove - Remove Storage Device
```

Screen 2-12. The Storage Device Management Menu

The following table lists common tasks associated with managing file systems, and shows the shell commands available for performing each task.

Task to Be Performed	Shell Command
Adding a device	mknod (1M)
Copying files to/from cartridge tape	tcpio (1)
Displaying contents of a cartridge tape	cpio (-t) (1M)
Displaying contents of a disk	mount (1M) ls (1)
Displaying security attributes of a device	devstat (1M)
Duplicating disks	dd (1M) cpio (1) dcopy (1M) volcopy (1M)
Erasing disks and tapes	rm (1) mt (1), format (1M)
Labeling file systems when copying to tape	labelit (1M)
Partitioning a disk	format (1M)
Removing disks and tapes	devnm (1M), ls (1), grep (1), cp (1), umount (1M), rm (1)
Verifying the usability of media	fsck (1M)
Allocating and de-allocating devices for secure use	admalloc (1M)

The rest of this section is a handy index to the shell commands for managing the storage devices on your system.

Adding a new device to the system:

1. **mknod** *name* *b | c* *major* *minor*

name is the name of the special file, followed by the device type (either *b*—a block device, such as a disk drive—or *c*—a character device, such as a terminal), *major* is the offset into the device driver (or controller) table in the kernel, and *minor* is the identifying number of this specific physical device.

2. **mknod** *name* *p*

where *name* is the name of the special file and *p* shows that *name* is a pipe. (that is, that it functions as a first-in, first-out device).

Copying files from tape to disk:

1. `cd dirname`

where *dirname* is the path to the directory into which you want to copy the files (such as `/var/tmp`)

2. `cpio -iudvB -I device`

where *device* is the path to the tape (character) device (such as `/dev/rmt/0m` or `ctape1`).

Removing a noncritical device:

CAUTION

Removing a noncritical device while the system is still powered up can have potentially disastrous results and should be done only in an emergency. This procedure should be performed only by experienced administrators.

1. `/usr/sbin/wall`
2. `/usr/bin/devnm /`

This produces the path of the partition on which the root file system is mounted.

3. `ls -l special`

where *special* is the file representing the partition. Record the major and minor device numbers for this device (this is the root device).

4. `/usr/bin/devnm /usr`
5. `ls -l special`

Record the major and minor device numbers for this device (`/usr` device).

6. `ls -l device_path`

where *device_path* is the path to the character or block device that you want to remove. Record the major and minor device numbers for this device (removed device). You will use the file name of this device in 13 and 16 below.

7. Ensure that the major and minor numbers of the device that you want to remove are not the same as those of the root or `/usr` device.
8. `grep device_name /etc/mnttab`

where *device_name* is the name of the device that you want to remove (file

name of *device_path* in 6 above). Record the directories that are mounted on this disk.

9. **cp /etc/vfstab /etc/Ovfstab**

10. **/etc/umount** *directory*

where *directory* is the name of the directory mounted on the disk that you want to remove (results of 8 above).

11. Edit the **/etc/vfstab** file and remove references to the device you want to remove.

12. **rm /dev/dsk/filename /dev/rdisk/filename**

where *filename* is the file name found in 10 above.

13. **devattr -v** *device_path* *alias*

where *device_path* is the full pathname of the device to be removed. This command returns the alias name for this device.

14. **putdev -d** *alias*

where *alias* is the alias name found in 13 above.

15. **getdgrp** *alias=device*

where *device* is either the pathname or the alias of the device. The output of this command is the alias of the device, which you will use in the next step.

16. **putdgrp -d** *device_group device_path*

where *device_group* is the name of the group from which the device will be removed and *device_path* is the alias name of the device found in 15 above.

Verifying the usability of a disk:

mount *special directory*

where *directory* is the directory mount point and *special* is the path to the disk (block) device (such as **/dev/SA/disk1**).

If you get an error message saying the data on the disk needs to be checked, there is a problem with the integrity of the file system. Run **fsck -F** *fstype file_system* where *file_system* is the path to the file system you want to check (such as **/usr**), and **fstype** is the filesystem type.

Creating a device entry:

putdev -a *alias* [*attribute=value* [. . .]]

where *alias* is the alias name of the device to be added to the DDB and *attribute=value* is a list of attribute values to be associated with the device. Possible attributes are defined in “*Maintaining the Device Database*”.

Listing devices:

```
getdev [-ae] [criteria [ . . . ]] [device [ . . . ]]
```

where *device* is the name of the device or devices you want listed and *criteria* defines selection criteria for the list.

If you run **getdev** with no options or arguments, you get a list of all devices in the DDB.

Specify *criteria* with the four expression types defined in this chapter under “*Listing Devices*”, in the section “*Specifying Criteria for a Listing*”. To be included in the list, a device must match at least one of the criteria given. Use the **-a** option to request that devices match all the given criteria before being included in the list.

All devices named will be included in the list, unless you use the **-e** option, which specifies that the devices named should be excluded from the list.

Listing device attributes:

```
devattr [-v] device [attribute [ . . . ]]
```

where *device* is the pathname or alias of the device whose attributes should be displayed and *attribute* is this specific attribute whose value should be displayed. Only the attribute value is shown. Use the **-v** option to display attribute values in *attribute='value'* format.

Modifying a device entry:

```
putdev -m device attribute=value [attribute=value [ . . . ]]
```

where *device* is the pathname or alias of the device entry being modified, *attribute* is the name of the attribute being modified, and *value* is the value that should be assigned to the attribute.

Removing a device entry:

```
putdev -d device
```

where *device* is the pathname or alias of the device being deleted from the DDB.

Deleting an attribute from a device entry:

```
putdev -d device [attribute [ . . . ]]
```

where *attribute* is the attribute definition to be removed and *device* is the pathname or alias of the device for which an entry is being modified.

Creating a device group:

```
putdgrp group_name device [device [ . . . ]]
```

where *group_name* is the name of the group you are creating and *device* is the pathname or alias of the member, or members, of the group.

Listing device groups:

```
getdgrp [-ae] [criteria [ . . . ] [group_name [ . . . ]]
```

where *group_name* is the name of the device group or groups you want included in the list generated by this command. *criteria* defines selection criteria for the list.

If you run **getdgrp** with no options or arguments, you get a list of all device groups.

Specify *criteria* with the four expression types defined in this chapter under “*Maintaining the Device Groups Database*”, in the section “*Specifying Criteria for Your List*”. To be included in the list, a device group must have at least one member that meets at least one of the given criteria. However, you can require a device group to have at least one member that meets all criteria before being included in the list. To do so, use the **-a** option.

All device groups named will be included in the list unless you use the **-e** option, which specifies that the named device groups should be excluded from the list.

Listing the members of a device group:

```
listdgrp group_name
```

where *group_name* is the name of a group for which a list of members should be displayed.

Modifying a device group:

```
putdgrp [-d] group_name device [device [ . . . ]]
```

where *group_name* is the name of the group for which the definition is to be modified. *device* is the pathname or alias of the device to be added to the group definition or, if the **-d** option is used, the name of the alias to be deleted from the group definition.

Removing a device group:

```
putdgrp -d group_name
```

where *group_name* is the name of the device group definition to be removed.

Reserving a device:

```
devreserv pid device
```

where *pid* is the process ID to which the device should be reserved and *device* is the path-name or alias of the device that should be reserved.

Freeing a reserved device:

```
devfree pid [device]
```

where *pid* is the process ID to which the device has been reserved and *device* is the path-name or alias of the device, or devices, that should be freed from reservation.

Checking device reservation status:

```
devreserv
```

lists all devices that are currently reserved.

Example Procedures on Adding Devices

This section provides guidelines on how to add the following devices to your Concurrent system. Only standard devices supported by Concurrent are covered.

- Disk Drive
- CD-ROM Drive
- Tape Drive
- Ethernet Controller Device
- FDDI Controller Device

Adding a Disk Drive

Follow the steps outlined below to add a disk drive to your system.

1. Perform an orderly shutdown and then power down the system.
2. Install the disk drive and connect all applicable cables.
3. Bring up the system to single-user mode.
4. Verify a disk adapter is present using the **hwstat** command.

```
# hwstat
```

```
I/O Configuration: Bus0: HVME Bus1: (none) Bus2: (none) Bus3: (none).
5 Devices Configured:
```

Device	Major/Minor	Bus	Bus I/O Addr	Std I/O Addr
IE	(7, 91)	Bus0	0x97100000	0x00000000
RTC	(38, 0)	Bus0	0x9C000000	0x00000000
HSA	(101, 0)	Bus0	0xCA040000	0x00000000
IS	(101, 28672)	Bus0	0x97200000	0x00000000
IS	(103, 29952)	Bus0	0x97200000	0x00000000

5. Verify the proper entry in the **/etc/conf/node.d/gd** file reflects the drive you are configuring:

```
# cat /etc/conf/node.d/gd
```

#Driver	Node	Node	Adapter	Adapter	SCSI	SCSI	Owner	Group	Modes	Level
#Name	Name	Type	Type	Number	ID	LUN				
#	---	---	---	---	---	---	---	---	---	---
#										
gd	dsk/0	D	hsa	0	0	0	0	3	0640	2

```
gd      dsk/1  D   is      0      0      0      0      3    0640    2
      ↑
      | Number used
      | must be unique
      | among gd
      | devices
      ↓
```

6. You now need to edit this file and add the disk drive to the disk drive adapters by adding the following line

```
gd      dsk/2  D   is      0      1      0      0      3    0640    2
```

NOTE: The Adapter Type, Adapter Number and SCSI ID must match the one shown in the `hwstat` output.

7. Create the device special files for the partitions being configured using the `/etc/conf/bin/idmknod` command:

```
# /etc/conf/bin/idmknod -M gd
```

The `idmknod` command creates the special device files in the `/dev/rdsk` and `/dev/dsk` directories of the form: *NsP* [where *N* = node number as shown in Node Name of file - `node.d/gd` and where *P* = partition 0 thru 7]

8. The new drive must have partitions formatted onto the drive using the `/etc/format` utility program:

```
# /etc/format N [where N is node number shown in 'Node Name'
in file - node.d/gd]
```

Opening '/dev/rdsk/2s7'.
Reading the geometry block.

```
/dev/rdsk/2s7
Integral SCSI revision 0, SCSI disk model 12
megabytes: 496 physical
cylinders per drive: 1983 physical
heads per cylinder : 16 physical
sectors per head   : 32 physical
bytes per sector   : 512 physical
512 drive description blocks at block 1013760
1024 diagnostic blocks at block 1014272
3600 revolutions per minute
geometry block version 2
flaw map size is 0 bytes
partition 0    307200 kbytes          0 start kbyte
partition 1    199680 kbytes        307200 start kbyte
partition 2     0 kbytes            0 start kbyte
partition 3     0 kbytes            0 start kbyte
partition 4     0 kbytes            0 start kbyte
partition 5     0 kbytes            0 start kbyte
partition 6     0 kbytes            0 start kbyte
partition 7     768 kbytes          506880 start kbyte
507648 kbytes used    0 kbytes unused
```

format:

Use the **format "p"** command to set partition sizes and the **"w"** command to update the geometry block for the disk with new partitions and then **"quit"** to exit **format**.

Note: You do not need to perform a full format of this disk drive. Refer to manpage **format (1m)** if this should become necessary.

- For each partition the disk needs to be formatted with file system structures:

```
# /sbin/newfs -F <Fstype> /dev/rdisk/2s0
where <Fstype> is one of the following: ufs, sfs or xfs
```

- An absolute mount point needs to be made to mount the partition, for example:

```
# /usr/bin/mkdir /<mountpt>
```

- The **/etc/vfstab** file needs updating to enable the file system to be mounted when the system boots:

```
#special      fsckdev      mountp      fstype      fsckpass    automnt    mntopts
/dev/root     /dev/rroot   /           sfs         0           yes       -
/dev/swap     -           -           swap        -           yes       -
/dev/usr      /dev/rusr    /usr        sfs         1           yes       -
/dev/var      /dev/rvar    /var        sfs         0           yes       -
/dev/proc     -           /proc       proc        -           no        -
/dev/fd       -           /dev/fd     fdfs        -           no        -
/system/processor - /system/processor profs      -           no        -
#
/dev/dsk/1s0 - -           swap        -           yes       -
/dev/dsk/1s1 /dev/rdisk/1s1 /usr1      ufs         1           yes       -
#
#New Disk Partitions:
#
/dev/dsk/2s0 /dev/rdisk/2s0 /usr2      ufs         1           yes       -
/dev/dsk/2s1 /dev/rdisk/2s1 /usr3      ufs         1           yes       -
#
```

- At this point you should run the file system checker program **fsck** on the new partition to verify it was created accurately. (Note: This step not needed for **xf**s filesystems.)

```
# /sbin/fsck -F <Fstype> /dev/rdisk/2s0
where <Fstype> is one of the following: ufs or sfs
```

- Now the file system may be mounted using the standard **mount** command:

```
# /etc/mount /dev/rdisk/2s0 # which is seldom used, or
# /etc/mount /dev/dsk/2s0 # or, more commonly
# /etc/mount /<mountpt>
```

The disk partition is now accessible for file access.

Adding a CD-ROM Drive

Follow the steps outlined below to add a CD-ROM Drive to your system.

1. Perform an orderly shutdown and then power down the system.
1. Install the CD-ROM Drive.
2. Bring up the system to single-user mode.

Note

The 'add' and 'make' steps shown below are only required if a CD-ROM Drive is added to an existing system. These steps are also required if the drive controller number or drive select changes.

3. Edit the `/etc/conf/node.d/gr` file and using the examples provided in the file, copy an applicable line and add it to the bottom of the file. Be sure to uncomment the line (delete #). Select the line according to the adapter used.

```
#
#Name Node      Type Adapter Adapter SCSI ID LUN uid gid mode level
#      Name      Type      Number  ID
#-----
#gr   cd/0       0   ncr    0      2      0  0  3  0644  1
#gr   cd/1       0   ncr    0      3      0  0  3  0644  1
```

Notes:

Node Name = ID you assign, usually in numerical order. Must be unique among CD-ROMs.

Adapter Type = ncr, via, is, etc

Adapter Number

SCSI ID number

For example, you would yank a given line, add it to the bottom of the file and make any necessary changes. Again, don't forget to uncomment the line.

```
gr   cd/0       0   ncr    0      4      0  0  3  0644  1
```

4. Make the device nodes for the CD-ROM drive.

```
/etc/conf/bin/idmknod -M gr
```

5. Create the CD-ROM mount point directory.

```
mkdir /<mountpt>
```

6. Add the CD-ROM to the `/etc/vfstab` file. This will simplify mounting the CD-ROM. For example:

```
/dev/cd/0 - /<mountpt> cdfs - no ro
```

7. To mount the CD-ROM drive, insert a CD disc in the CD-ROM drive and execute the mount command:

```
mount /cdrom
```

8. To unmount the CD-ROM drive, execute the unmount command:

```
umount /cdrom
```

Adding a Tape Drive

Follow the steps outlined below to add a tape drive.

1. Perform an orderly shutdown and then power down the system.
2. Install the tape drive and connect all applicable cables.
3. Bring up the system to single-user mode.
4. Verify a tape drive adapter is present using the **hwstat** command:

```
# hwstat
```

```
I/O Configuration: Bus0: HVME Bus1: (none) Bus2: (none) Bus3: (none).
5 Devices Configured:
Device                Major/Minor      Bus  Bus I/O Addr  Std I/O Addr
=====
IE      0 Ethernet      ( 7, 91 ) Bus0  0x97100000    0x00000000
RTC     0 RT Clocks     ( 38, 0 ) Bus0  0x9C000000    0x00000000
HSA     0 Disk 0/0      ( 101, 0 ) Bus0  0xCA040000    0x00000000
IS      0 Disk 0/0      ( 101, 28672) Bus0  0x97200000    0x00000000
IS      0 Tape 5/0       ( 103, 29952) Bus0  0x97200000    0x00000000
```

5. Verify the proper entry in the **/etc/conf/node.d/gt** file reflects the tape drive you are configuring:

```
# cat /etc/conf/node.d/gt
```

```
#ident "@(#)kern-nh:io/scsi/gt/gt.cf/Node1.1"
#
#Name Node      Type Adapter Adapter# Ctlr Dev  Usr  Grp  Mode  Level
#----
#example:
#gt    mt/0      T   hsa     0       5    0    0    3    0666  1
gt     mt/0      T   is      0       5    0    0    3    0666  2
```

6. You now need to edit this file and add tape drive 1 to the adapter by adding the following line:

```
gt     mt/1      T   is      0       4    0    0    3    0666  2
```

NOTE: The Adapter Number must match the one shown in the **hwstat** output.

7. Create the device special files of the tape drive being configured using the **/etc/conf/bin/idmknod** command:

```
# /etc/conf/bin/idmknod -M gt
```

The special device files are created in the **/dev/rmt** and **/dev/mt** directories and the tape device should now be accessible.

8. Once the tape drive is configured, the output of the **hwstat** command should show the new device:

```
# hwstat
```

```
I/O Configuration: Bus0: HVME Bus1: (none) Bus2: (none) Bus3: (none).
6 Devices Configured:
Device                               Major/Minor      Bus  Bus I/O Addr  Std I/O Addr
=====
IE      0 Ethernet      ( 7, 91 ) Bus0  0x97100000    0x00000000
RTC     0 RT Clocks     ( 38, 0 ) Bus0  0x9C000000    0x00000000
HSA     0 Disk 0/0      ( 101, 0 ) Bus0  0xCA040000    0x00000000
IS      0 Disk 0/0      ( 101, 28672) Bus0  0x97200000    0x00000000
IS      0 Tape 4/0      ( 103, 29953) Bus0  0x97200000    0x00000000
IS      0 Tape 5/0      ( 103, 29952) Bus0  0x97200000    0x00000000
```

Note: The system may require a reboot for this information to become active.

9. The file **/etc/default/tape** may need to be updated to reflect the new tape drive configuration. It contains an entry similar to:

```
device = /dev/rmt/0ms
```

10. The file **/etc/default/tar** may also need updating to reflect the new tape drive configuration:

```
#ident"@(#)tar:common/cmd/tar/tar.dfl1.1.3.5"
#ident "$Header: /sms/sinixV5.4es/rcs/s19-full/usr/src/cmd/tar/tar.dfl,v 1.1 91/02/28
20:11:52 ccs Exp $"
#      device          block   volume size(Kbytes)
archive0=/dev/rmt/0m    20      0
archive1=/dev/rmt/0mn  20      0
archive7=./tarfile     1       0
#
# The default device in the absence of a numeric or "-f device" argument
archive=/dev/rmt/0m    20      0
```

11. Before you can allocate a device, you must add it to the Device Database (DDB). To create an entry for a device in the DDB, run the **putdev (1m)** command, as follows:

```
putdev -a alias [attribute=values [ . . . ]]
```

where *alias* is the name (unique in the database) assigned to the device and *attribute=value* is the assignment of a value to a particular attribute. You can make multiple assignments in this format on the **putdev (1m)** command line.

A more specific example is shown below.

```
putdev -a tapel volume="4mm Dat" cdevice=/dev/rmt/0hf \  
desc="4mm DAT Tape" removable="true" type="qtape"
```

Note. The “removable” and “type” attributes will make the drive useable with the **backup (1)** command.

Adding Ethernet/FDDI Controller Device(s)

Adding New Controller Device and Software for First Time

Follow the steps outlined below to install a new controller device and its applicable software package to your system .

1. Perform an orderly shutdown and then power down the system.
2. Install the controller device and connect all applicable cables.
3. Bring the system up to single-user mode.
4. Install the applicable software package and answer questions as needed.
5. Rebuild the kernel.
6. Reboot the system.
7. Relink the new kernel.
8. To configure the system to recognize the controller you just installed, run the **/usr/bin/netsetup** script. In the example shown below an Eagle Ethernet Controller board (**eg1**) is being configured on system ‘**amber2**’ with a device unit number of ‘**5**’. This same script will allow you to configure other controller boards supported by Concurrent. Note that the interface network subnetmask (**xxx.xxx.xxx.xxx**) will depend on your network (usually 255.255.255.0)

```
# /usr/bin/netsetup
```

```
*****  
*  
*           Eagle Ethernet Board Configuration           *  
*  
*  
*****
```

```

Configure an Eagle ethernet controller ? (default: y) y

Enter the interface device unit number [0-5,?] 0
Enter the interface Internet address or hostname (<cr> for amber2):<cr>
Enter the interface network subnetmask in decimal dot notation
(format xxx.xxx.xxx.xxx): 255.255.255.0    <-- default shown

The following line will be added to /etc/confnet.d/inet/interface file:
egl:0:amber2:/dev/egl0:netmask 255.255.255.0 arp::

Is this ok ? (default: y) y

#

```

9. Reboot the system.
10. Verify the controller device is present using the **hwstat** command:

Adding New Controller Device - Software Previously Installed

Follow the steps outlined below to install a new controller device to a system in which the applicable software package was previously installed.

1. Perform an orderly shutdown and then power down the system.
2. Install the controller device and connect all applicable cables.
3. Bring the system up to single-user mode.
4. Run the **/usr/bin/netsetup** script after the kernel knows about the new device. Refer to step 7 above for example procedure

Edit the **/etc/conf/sadapters/kernel** file and using the examples provided in the file, copy an applicable line and add it to the bottom of the file. Be sure to uncomment the line. Examples of the Eagle Ethernet Controller entries are shown below

```

# %%Adapter%%egl
# %%Abstract%%Interphase 4027 Eagle Ethernet Controller
# %%Examples%%
# Adapter Logical Bus Intr Slot I/O I/O
# Name Adptr # Type Type Number Address 1 Address 2
# -----
# egl 0 vme0 intr - ffff0800 f2000000
# egl 1 vme0 intr - ffff1000 f2080000
# egl 2 vme0 intr - ffff1800 f2100000
# egl 3 vme0 intr - ffff2000 f2180000
# egl 4 vme0 intr - ffff2800 f2200000
# egl 5 vme0 intr - ffff3000 f2280000
# %%Adapter_End%%

```

For example, if you just added Eagle Ethernet Controller device unit number 0, you would yank a given line, add it to the bottom of the file and make any necessary changes. Again, don't forget to uncomment the line.

5. Rebuild the kernel.
6. Reboot the system.

Managing File System Types

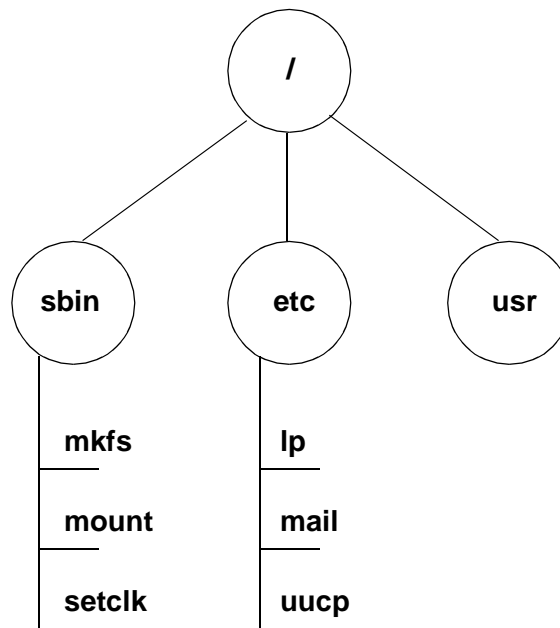
What Is a UNIX File System?	3-1
The Relationship between the File System and the Storage Device	3-3
Formatting the Storage Device	3-3
Partitions on the Storage Device	3-3
Size Limitations on a File System	3-3
Avoiding File System Damage	3-4
Administering File Systems and Data Integrity	3-4
Do You Suspect File System Damage?	3-5
The ufs File System Type	3-5
The ufs Cylinder Group Map	3-5
The ufs Super-Block	3-6
ufs Inodes	3-6
What Does a ufs Inode Contain?	3-6
The ufs Inode's Disk Block Addresses	3-6
ufs Storage Blocks	3-9
ufs Free Blocks	3-9
Quotas on the ufs File System	3-10
Using Quotas	3-10
What Effect Do Quotas Have on the User?	3-11
Time and Space Optimization	3-12
The xfs File System Type	3-13
xfs File System Features	3-13
Fast File System Recovery	3-13
Block Size and Fragment Support	3-14
Space Allocation for Large Files	3-14
Contiguous Files	3-14
Large File Systems	3-15
Prioritized I/O	3-15
xfs File System Structure	3-15
Block 0	3-16
The xfs SuperBlock	3-16
xfs Free Space List	3-16
xfs Inode List	3-16
xfs Root Directory	3-16
xfs Cyclic Log	3-16
xfs Data Blocks	3-17
xfs Space Allocation	3-17
xfs File System Administration	3-18
Creating an xfs File System	3-19
The Distributed xfs File System Type	3-20
xfsd Overview	3-20
Required Network Connection	3-21
Mounting an xfsd File System	3-21
Limitations on Client Access to the xfsd File System	3-22
System Failures	3-23
Example on How to Create an xfsd File System	3-24
xfsd Setup	3-24

Hardware Connections	3-25
Software Setup	3-25
The sfs File System Type	3-26
The sfs Inodes	3-26
The memfs File System Type	3-27
Administering a File System	3-28
The Generic Administrative Commands	3-28
How the Administrative Commands are Invoked	3-28
The vfstab File System Table	3-29
What Does the vfstab File System Table Do?	3-29
Commands for sfs File Systems	3-30
Listing Installed File System Types	3-31
Identifying the Type of an Unmounted File System	3-31
Creating a File System	3-31
Using mkfs to Create a File System	3-31
Choosing a Logical Block Size	3-32
What to Consider When Choosing a Logical Block Size	3-33
Using mkfs to Create a ufs File System	3-33
Using mkfs to Create an sfs File System	3-33
Moving and Copying File Systems	3-34
Step 1: Copying the Source File System	3-35
Copying a File System with cpio	3-35
Copying a File System with volcopy	3-35
Steps 2 and 3: Removing the Source File System and Editing vfstab	3-36
Mounting and Unmounting File Systems	3-36
Example of Mounting File Systems	3-36
Using the mount Command	3-36
Unmounting a File System	3-37
Mounting ufs File System under sfs	3-37
Mounted File System Level Range	3-38
Device Level Range	3-38
Maintaining a File System	3-39
Monitoring the Percentage of Disk Space Used	3-39
Monitoring Files and Directories That Grow	3-39
Identifying and Removing Inactive Files	3-40
Identifying Large Space Users	3-41
The du Command	3-41
The find Command	3-41
Deleting /tmp Directories	3-41
Quotas	3-41
Quick Reference Guide to Managing File System Types	3-42

Managing File System Types

What Is a UNIX File System?

PowerMAX OS (OS) supports multiple file system types (FSTypes) of varying characteristics. In the operating system, a file is a string of bytes (with no other structure implied) that resides in a directory. The directory (merely another type of file) is part of a hierarchy of directories which, together with the files it supports, makes up an OS file system. Figure 3-1 illustrates this structure: the circles represent directories; the command names listed below them, files.



162880

Figure 3-1. An Operating System File System

The starting point of any OS file system is a directory that serves as the root of that file system. Somewhat confusingly, in the operating system there is always one file system that has the name “root.” Traditionally, the root directory of the root file system is represented by a single slash (/). The file system shown in Figure 3-1 is a root file system. If we mount another file system onto the root file system at a directory called `/usr`, the result will look like the diagram shown in Figure 3-2.

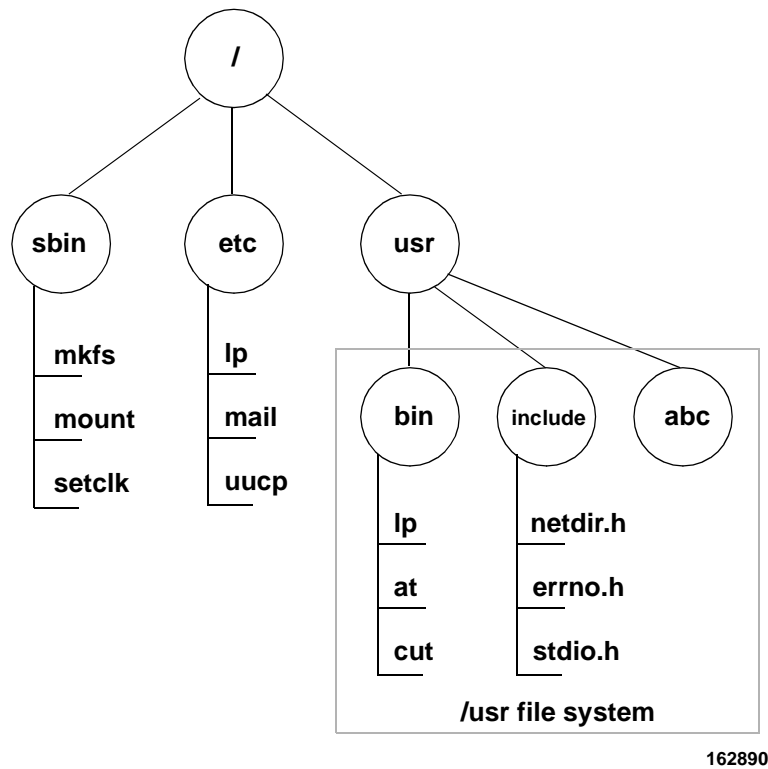


Figure 3-2. Adding the /usr File System

A directory such as /usr that is used to form the connection between the root file system and another mountable file system is sometimes called a “leaf” or “mount point.” Regardless of the term used, such a directory is the root of the file system that descends from it. The name of that file system is the name of the directory. In our example the name of the file system is /usr.

The diagrams in Figure 3-1 and Figure 3-2 may be convenient representations of the file and directory structure of file systems, but they do not provide a particularly accurate or helpful way of illustrating how the operating system views a file system. The sections that follow describe FSTypes as they appear to the operating system.

The great range of FSTypes makes it impossible to document the administration of all of them in one place; this chapter addresses the following common types:

- **ufs** (the UNIX file system)
- **xfs** (the resilient file system)
- **xfsd** (the distributed xfs file system)
- **sfs** (the secure file system)
- **memfs** (the memory file system)

Information on other file system types is available elsewhere: /proc is described on **proc (4)**; the **nfs** file system type in *Network Administration*.

NOTE

If the Enhanced Security Utilities are installed on your system, you should use the **sfs** file system types to take advantage of the additional security features offered. The **ufs**, **xf**s and **xfsd** file systems do not support the additional file security features included in the Enhanced Security Utilities. See the “Security Administration” part of “*System Administration*”, Volume 1, for more information.

The Relationship between the File System and the Storage Device

In the operating system, file systems are kept on random-access disk devices. (A file system can be put on tape, but that is normally done only to create a backup copy in case the file system must be restored.) Before you can install a file system on a storage device, you must format the device. The material in this part of the chapter is a summary of material described in more detail in Chapter 2, “*Managing Storage Devices*”.

Formatting the Storage Device

Before a disk can be used by the operating system it must be formatted into addressable sectors. A disk sector is a 512-byte portion of the storage medium that can be addressed by the disk controller. The number of sectors is a function of the size and number of surfaces of the disk device. Sectors are numbered from zero on up.

NOTE

Hard disk units on many computers are formatted when installed. The only time they might have to be reformatted would be after a catastrophic hardware failure. We recommend you contact your service representative if such an event occurs.

Partitions on the Storage Device

A partition consists of one or more sectors. Up to 8 partitions can be defined on a hard disk.

Size Limitations on a File System

The maximum number of physical blocks that can be allocated to a file system is close to the total number of sectors on the disk device. This maximum may be reduced by space set aside for swapping or paging. Also, recall that under **ufs** a two-gigabyte upper limit is

imposed by the size field in the inode. For **xfs** and **xfsd** file systems this limit is increased to 1TB (1 Terabyte = 1000 Gigabyte).

The **ufs** FSType does not have a rigid upper limit; roughly one inode is created automatically for each 4K of data space, although this default can be overridden at the time the file system is created. The size of a block on a disk is 512 bytes, the same as a disk sector. However, internal file I/O works with logical blocks rather than with 512-byte physical blocks. The size of a logical block is set with the **mkfs (1M)** command. **ufs** uses logical block sizes of 4096 and 8192 bytes; the default is 8192-byte blocks. Because of the large block size used by **ufs**, small files could waste a lot of space. To deal with this, **ufs** has a subdivision of a block called a fragment. When a **ufs** file system is created using **mkfs**, the fragment size must be a power of two currently selected from the range 512 to 8192 bytes. The default is 1024. When a block must be fragmented, the remaining fragments are made available to other files to use for storage. The information on which fragments are in use and which are available is kept in the cylinder group summary information block.

Avoiding File System Damage

To be specific, a file system (and the disk as well) can be severely damaged if one of the following events happens while data is being written to the disk:

- The computer is turned off with the power switch.
- The computer's reset button is pressed.
- The computer experiences a power failure.

To help avoid file system damage, here are some suggestions. These suggestions apply whether or not you have a caching disk controller.

- Never shut off the computer while the disk is active, or while data sent to the disk by the operating system is still in the cache. If you aren't sure when the disk is active, or when data is in the cache but not on the disk, use the **shutdown (1M)** command to shut the computer down.
- If loss of data due to a power failure is a concern, install an uninterruptible power supply for your computer.

Administering File Systems and Data Integrity

Normally, file system damage is automatically repaired for you when you bring the computer back up. If the ordering is changed by a disk controller, and then interrupted before all of the cache is written to disk, then the file system may be fooled into thinking that all is well when damage is present.

The only way this can happen for **ufs** or **sfs** is if the superblock is stamped clean before other writes have been done. A superblock is stamped clean only when it has been unmounted or when **fsck** has been run on it. When the computer comes back up, **fsck** will not be run in this case, because the file system appears clean. In all other cases, the file system will not appear to be clean, and **fsck** will check the entire file system.

Do You Suspect File System Damage?

What do you do if you suspect file system damage that the computer is overlooking?

- Unmount the file system and run **fsck**.
- If the file system is root, things get stickier since you cannot unmount the root file system to do the **fsck**. The root file system is always “fscked” while the system is booting. In this case, simply reboot the machine.

The ufs File System Type

NOTE

If the Enhanced Security Utilities are installed on your system, use only user file systems of type **sfs** to ensure the security of your system; **ufs** file system type will not support Mandatory Access Control restrictions.

The **ufs** FSType is considerably complex in its design. For example, it has a mount-time option to specify. The **soft** option to **-o** specifies that writes need not be written out to disk until the file system is about to be unmounted. This volatility can improve system throughput as much as eight percent when the temporary directories (for example, **/tmp** and **/var/tmp**) are mounted this way. (See also “*The memfs File System Type*” section in this chapter for more information).

There is also a radically different method of allocating and managing blocks associated with the file system’s information management disk areas. Of primary interest is the fact that multiple super-blocks are made during the **mkfs** procedure. One of the replicas is stored in each cylinder group, offset by a certain amount. For multiple platter disk drives, the offsets are calculated so that a super-block appears on each platter of the drive. So if the first platter is lost, an alternate super-block can be retrieved. For platters other than the top one in a pack, the leading blocks created by the offsets are reclaimed for data storage.

Kept with the super-block is a summary information block. This block is not replicated, but is grouped together with the first super-block, normally in cylinder group 0. This summary block is used to record changes that take place as the file system is used, and lists the number of inodes, directories, fragments, and blocks within the file system.

The ufs Cylinder Group Map

Another feature of **ufs** is the “cylinder group map.” This is a block of data found in each cylinder group that records the block usage within the cylinder. This information is kept directly following the super-block copy for that cylinder group.

To give an idea of the appearance of a typical **ufs** file system, Figure 3-3 shows a series of cylinder groups in a generic **ufs** file system.

The ufs Super-Block

Much of the information about the file system is stored in the super-block. A few of the more important things it contains are the

- size and status of the file system
- label (file system name)
- size of the file system in logical blocks
- date and time of the last update
- cylinder group size
- number of data blocks in a cylinder group
- summary data block.

ufs Inodes

The inode information is kept in the cylinder information block. An inode contains all the information about a file except its name, which is kept in a directory. An inode is 128 bytes long. One inode is created for every 2K of storage available in the file system. This parameter can be changed when **mkfs** is used to create the file system, but it is fixed thereafter.

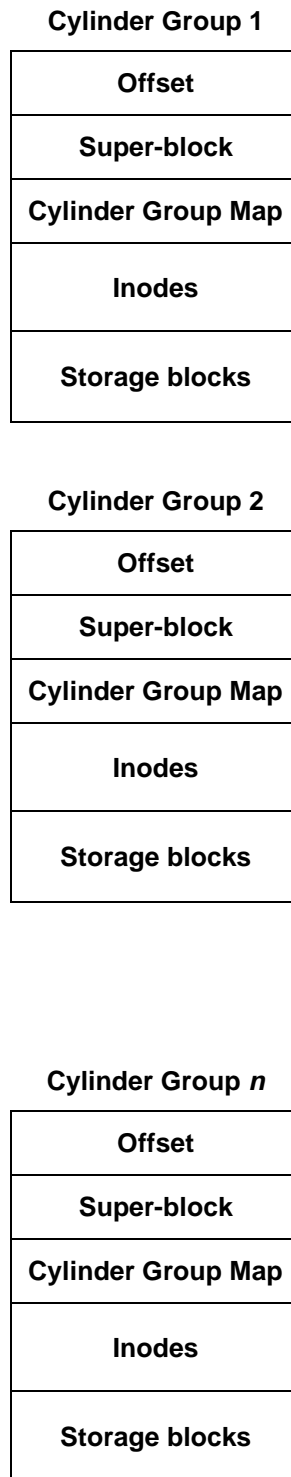
What Does a ufs Inode Contain?

A **ufs** inode contains the

- type and mode of the file-**the** type can be regular, directory, block, character, symbolic link, or FIFO, also known as named pipe; the mode is the set of read-write-execute permissions
- number of hard links to the file
- user ID of the owner of the file
- group ID to which the file belongs
- number of bytes in the file
- two arrays comprising a total of 15 disk block addresses
- date and time the file was last accessed
- date and time the file was last modified
- date and time the file was created.

The ufs Inode's Disk Block Addresses

The heart of the inode is two arrays that, together, comprise 15 disk block addresses.



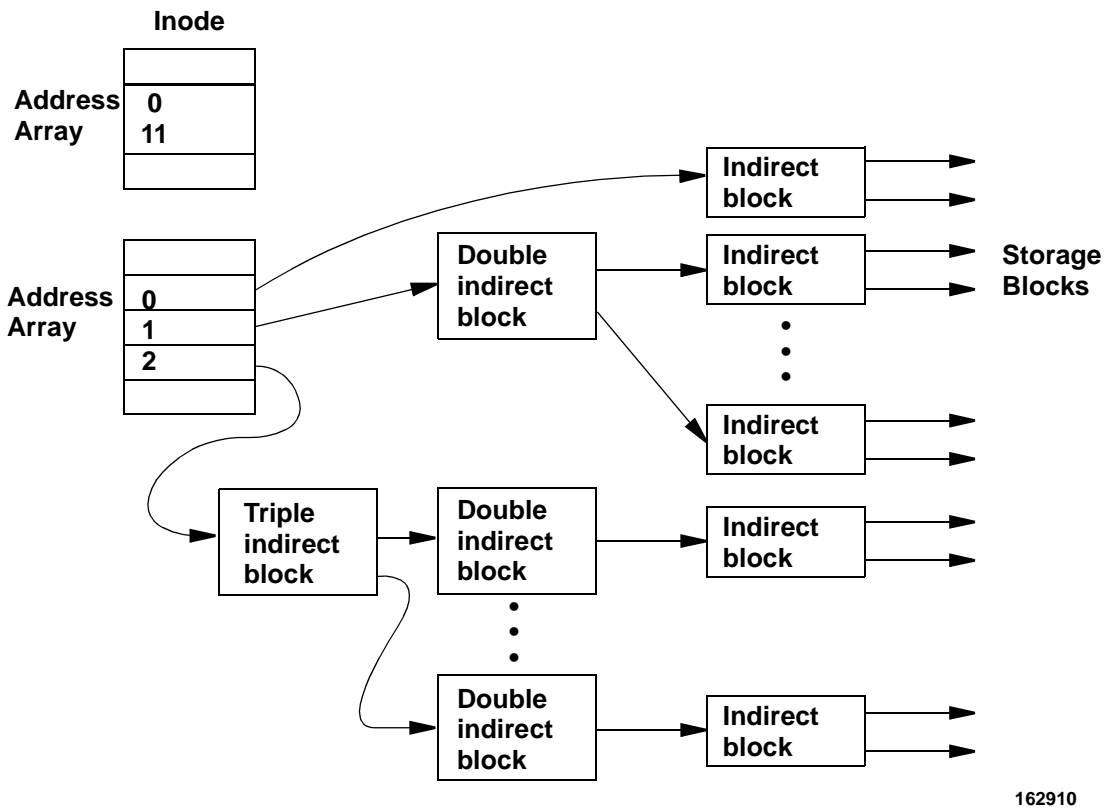
162900

Figure 3-3. The Operating System View of a ufs File System

- The first array contains 12 direct addresses, that is, addresses that point directly to the first 12 logical storage blocks of the contents of the file (addresses 0-11). If the file is larger than 12 logical blocks, the first address of the second array (address 0) points to an indirect block, which contains direct addresses instead of file contents.
- The second address (1) points to a double indirect block, which contains addresses of indirect blocks.
- The third address (2) points to a triple indirect block, which contains addresses of double indirect blocks.

Figure 3-4 illustrates this chaining of address blocks stemming from the inode.

Table 3-1 shows the number of bytes addressable by the different levels of indirection in the inode address array for **ufs** file systems. These numbers are calculated using the logical block size of the file system and the number of bytes used to hold an address.



162910

Figure 3-4. The Address Chain in a ufs File System

Table 3-1. Number of Bytes Addressable

Logical Block Size	Maximum number of bytes addressable by		
	Direct Blocks	Single Indirect Blocks	Double Indirect Blocks
2048 bytes	24K	1M	512M
4096 bytes	48K	4M	4G
8192 bytes	96K	16M	32G

The table shows the number of bytes addressable using the level of indirection in the column header plus all lower levels of addressing. For example, the table values for single indirect blocks also include bytes addressable by direct blocks; and the table values for indirect blocks include bytes addressable by direct blocks and single indirect blocks. In a file system with a 2048-byte block size, files larger than 512M use triple indirect blocks.

The theoretical maximum size of a **ufs** file system is the same as the size of a file addressable with triple indirection. In practice, however, file size is limited by the size field in the inode. This is a signed 32-bit field, so file sizes are limited to two gigabytes. Because of the large size of **ufs** logical blocks, double indirect blocks rarely appear in **ufs** file systems. The result is that data retrieval in large files is much quicker than it would otherwise be.

ufs Storage Blocks

The rest of the space allocated to the file system is occupied by storage blocks, also called data blocks. The size of these storage blocks is determined at the time a file system is created, and can be 2048, 4096, or 8192 bytes. Because of these large block sizes, and the potential for waste with small files, **ufs** also has a subdivision of a block called a fragment. When a file system is created, the fragment size must match or be smaller than the block size: you may set it to 512, 1024, 2048, 4096 or 8192 bytes. Fragments of 1024 bytes are the most commonly employed. For a regular file, the storage blocks contain the contents of the file. For a directory, the storage blocks contain entries that give the inode number and the filename. **ufs** filenames may be up to 255 bytes long. Each entry represents a file or subdirectory that is a member of the directory.

ufs Free Blocks

Blocks not currently being used as inodes, as indirect address blocks, or as storage blocks are marked as free in the block map kept in the cylinder group summary information block. This block also keeps track of fragments in order to prevent fragmentation from degrading disk performance excessively.

Quotas on the ufs File System

The quota system (available only on **ufs** file systems) is built around limits on the two principal resources of a file system

- inodes and
- data blocks.

For each of these resources, users may be assigned quotas. A quota in this case consists of two limits, known as the soft and hard limits.

The hard limit represents an absolute limit on the resource, blocks or inodes, that the user may never exceed under any circumstances.

Associated with the soft limit is a time limit set by the administrator. Users may exceed the soft limits assigned to them, but only for a limited amount of time—the time limit set by the administrator. This allows users to temporarily exceed limits if needed, as long as they are back under those limits before the time limit expires. An example of such a situation might be the generation of a large file that is printed and then deleted.

In summary, for each user, you can assign quotas (soft and hard limits) for both blocks and inodes. You can also define a time limit that applies to all users on a file system indicating how long they can exceed the soft limits. There are actually two time limits: one for blocks, and one for inodes. You may define different time limits for different file systems. Also, users may have different quotas set on different file systems.

Using Quotas

Before turning quotas on for a file system for the first time, do the following:

1. Mount the file system and change directory to the mount point.
2. Create a file called **quotas**. This file should be owned by **root**, and should not be writable by others.
3. Enter

```
edquota -t
```

to change the time limits for exceeding the soft limits for blocks owned, and/or inodes owned.

These limits are initially set to the values defined for **DQ_FTIMELIMIT** and **DQ_BTIMELIMIT** in **/usr/include/sys/fs/sfs_quota.h-604800** (the number of seconds in one week). If you leave either time limit (the one for exceeding the block limit or the one for exceeding the inode limit) at 0, or if you set either limit to 0, the default values will apply.

You can, of course, change them to something else.

4. Enter

```
edquota
```

(with or without the `-p` option) to set user quotas.

Once you have set the quotas for a user, you can use the `-p` option to set the same quotas for another user. Note that because you are not limited to UIDs that are already being used, you may set quotas for future users.

NOTE

Before turning on quotas on a file system, always run **quotacheck** on the file system. This will sync up the quotas with the actual state of the file system, so that if the file system has been used since the last time the quotas were turned on, all of the quotas will be updated to reflect the current state. This also provides a sanity check on the **quotas** file.

You can also do other quota-related functions:

- To turn quotas on, enter
quotaon
- To turn quotas off, enter
quotaoff
- To report on quotas, you can use **repquota** or **quot** to get information on all users on a file system
- Use **quota** to get information on a single user.

NOTE

Regular users can use **quota** to get information on their own quotas; they cannot get information on anyone else's quotas.

What Effect Do Quotas Have on the User?

The following are the major effects of the use of quotas on users:

- If a user exceeds his/her soft limit for blocks or inodes, the timer is started. If the user then reduces usage to a level under the soft limit, the timer is turned off and all is well. But if the user still has not reduced usage to an appropriate level when the timer expires, any further attempts by the user to acquire more file system resources will fail and the user will receive error messages saying that the file system is full. These messages will persist until the user has reduced usage to a level below the soft limit.
- If a user tries to exceed the hard limit at any time, the attempt will fail and the utility will indicate it has run out of space.
- Because no warning is given when the user has exceeded the soft limit, users should be advised to run **quota** frequently. Users should be

encouraged to include `quota` in their `.profile` so it runs when they log in.

Time and Space Optimization

If you are using a `ufs` file system, the operating system attempts to optimize the time it takes to perform operations on the file system. In certain circumstances, the operating system will attempt to optimize the space on the storage media being used by the file system.

Space optimization will occur when file fragmentation exceeds: $(free - 2) \%$ of free disk space. When file fragmentation exceeds this value, the message

```
Optimization changed from TIME to SPACE
```

will be displayed, and the operating system will begin to optimize the space used by the file system.

The value of `free` is, by default, 10 percent, but it can be set to any value using the `mkfs` command. The main use of `free` is to specify the minimum percentage of free disk space allowed. It is also used to limit the fragmentation of the file system.

When a file grows past an 8 Kbyte boundary under time optimization, the operating system allocates a fragment (which is 1 Kbyte in size) to contain the data. When the boundary of the fragment is exceeded, the operating system allocates a new 8 kbyte block, and the fragment is copied to it. This means that there could be nearly 7 Kbytes of wasted space at the end of some files.

With space optimization, however, whenever a file grows past a 1 Kbyte boundary, all the fragments since the last 8 Kbyte boundary are copied into a new, larger fragment.

If your system has swapped to space optimization, and you want to revert to time optimization, you will have to increase the value of `free`. Use the `tunefs` command to do this. You can increase `free` to any value you want; however, keep in mind the two functions of the `free` parameter if you want to set realistic values. `free` is used to:

- Set the minimum percentage of free disk space allowed. Therefore, if you set the value of `free` too high, ordinary (non-privileged) users will not be able to write files (as described in the section “*Using mkfs to Create a ufs File System*”).
- Set the maximum percentage of free disk space that can be fragments. Therefore, if you set the value of `free` too low, the optimization will change from time to space and file writes will be slower.

Set the value of `free` carefully, taking into account the typical file size, and the number of directories on your system.

The xfs File System Type

The **xfs** file system type - the resilient file system – is an extent-based high integrity file system that provides enhanced performance and reliability. The main topics covered in this section are:

- **xfs** file system features
- **xfs** file system structure
- **xfs** file system administration

NOTE

If the Enhanced Security Utilities are installed on your system, use only user file systems of type **sfs** to ensure the security of your system; **xfs** file system type will not support Mandatory Access Control restrictions.

xfs File System Features

The main features of the **xfs** file system include:

- fast file system recovery
- configurable extent and block size, and fragment support
- improved space allocation policy for large files
- support for contiguous files
- large file systems
- support for prioritized I/O

Fast File System Recovery

Conventional UNIX file systems rely on the full structural verification of the `fsck` utility as the only means of recovery from a system failure. The `fsck` utility for `ufs` and `sfs` checks the entire structure to verify that the file system is intact, and corrects any inconsistencies that are found. For large disk configurations, this process can be very time consuming.

xfs provides recovery only seconds after a system failure by utilizing a tracking feature called **intent logging** – a circular logging scheme that records pending changes to the file system structure in an ‘intent log’. Actions involved in file operations (creation, deletion, renaming, etc.) are unified, such that either all or none are applied. Once the intent has been recorded, the actual disk updates can occur asynchronously, the program that initiated them being unconcerned as to when the structures are physically updated on disk.

Upon recovery from a system failure, the `mount` process scans the intent log, nullifying or completing file system operations that were active when the system failed; there is no need for a structural check of the entire file system. Except for the fact that `xfs` file system recovery is completed in a few seconds, the intent log recovery feature is not readily apparent to either the user or the system administrator.

Block Size and Fragment Support

An `xfs` file system can be configured with a block size of between 512 bytes and 32Kbytes, in powers of 2. Using the largest possible block size, 32Kbytes, the potential number of seeks required for a file can be radically reduced.

To avoid potential space wastage for very small files when using a larger block size, `xfs` sub-divides its space into fragments. Files of less than the configured block size are accommodated in fragments. A fragment can be between 512 bytes and 4Kbytes, in powers of 2. The configured fragment size must be less than or equal to the block size. A fragment size of 4Kbytes is only allowed for file system extent sizes greater than 64Kbytes.

For example, for a file system with 8Kbytes blocks and 512 byte fragments, up to 16 small files (< 512 bytes) can be accommodated in a single block. A file of between 512 bytes and 1024 bytes occupies two consecutive fragments.

Note

Because of space optimization, it will not be possible to completely fill a Resilient File System. While the file system may report a full condition, `df` will report that there is still a small percentage free.

Space Allocation for Large Files

In `xfs` the blocks allocated to a file are tracked in an inode. An inode can reference ten extents directly before disk I/O is required to obtain the location of further blocks indirectly. Extents are larger allocation units (up to 4Mbytes) than blocks, that allow far more data to be referenced without incurring the overhead of obtaining block locations indirectly. A file of up to 504Kbytes can be referenced without indirection with the default block size (8Kbytes) and extent size (64Kbytes).

Contiguous Files

`xfs`'s space allocation algorithms try to ensure that a file's data blocks are reasonably contiguous while minimizing the amount of space wastage. If it is important that the blocks are truly contiguous, they can be pre-allocated using `fallocate(3x)`, which attempts to allocate the specified amount of space in a single extent. Pre-allocation improves performance while a file is being written by pre-paying the cost of allocation at create/open time.

Large File Systems

The size of a **ufs** file system is limited to 2GB; for an **xfs** file system this limit is increased to 1TB (1 Terabyte = 1000 Gigabyte).

Note

Even though an **xfs** file system can exceed 2GB, only the first 2GB is addressable if the file system is accessed using a block device. If the **xfsdump** command is invoked using a block device, the appropriate character device is opened and used instead.

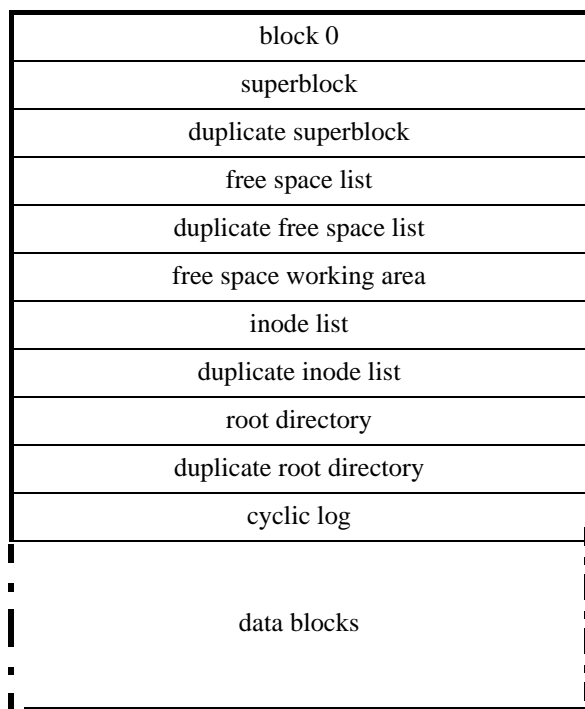
Prioritized I/O

A mount option is provided for the benefit of real-time applications using direct I/O which results in queued I/O's to or from the file system being sorted in order of the priority of the thread that issued the I/O request.

xfs File System Structure

To provide the requisite structural integrity, redundant copies of all disk resident structures, except the cyclic log, are required.

Apart from the superblock, which occupies blocks 1 and 2, all other structures are locatable anywhere in the file system so as to avoid any bad blocks.



Block 0

Block 0 is unused in **xfs**.

The xfs SuperBlock

The **xfs** superblock contains details about the disk partition. However, this is restricted to static information only.

Block 1 is always the first copy of the superblock, and a duplicate of the superblock is written to block 2.

When the superblock is written (at file system mount and dismount time), both block 1 and block 2 are written (ensuring that one has completed before the second starts).

xfs Free Space List

Free space is handled by accessing a bit map where each fragment in the file system is represented by a single bit whose state indicates whether the fragment is free or not. A duplicate copy of the bit map is maintained. Bit map blocks are written alternately, so that if the system should fail during an I/O operation and the list corrupted, it can be reconstructed from the duplicate copy and the intent log. A free space working area, representing changes to the file system in progress or recently completed but not yet written to disk, is also maintained.

xfs Inode List

A small number of inodes are allocated when the file system is created; further inodes are created on demand, and are written in duplexed 'inode pages' in the data area of the partition; this locality of reference reduces access time.

The inode list consists of inode control pages (also duplexed) that contain pointers to all the inode pages. To reduce the overhead of the indirection required to locate inode pages, the inode list is maintained in memory.

xfs Root Directory

The root directory is inode number 2 and there is a single block (plus shadow copy) containing the entries "." and ".." when the file system is created.

xfs Cyclic Log

The cyclic log is a group of contiguous sectors into which structural changes are recorded before they are made in the file system. As its name implies, once the end is reached, logging continues from the beginning.

The cyclic log is subdivided into sections, each section being given an event number. Changes to system structure pages (inodes, directories, free space etc) are associated with

event numbers; changes to data blocks are not recorded in the log. Before a cyclic log section can be reused, all page changes relating to that event must be flushed to disk.

When recovering from a system failure, the cyclic log is read and those structural changes recorded in the log but not yet applied are applied.

Since the log contains only a snapshot of the changes that have been applied, a page containing structure information lost during a system failure would prevent the file system being recovered without accessing the whole disk. To remove such single points of failure, all of the structure pages are replicated and written to alternately. If a system failure corrupts a page, then it can be rebuilt from the other copy, in addition to the application of the changes in the cyclic log.

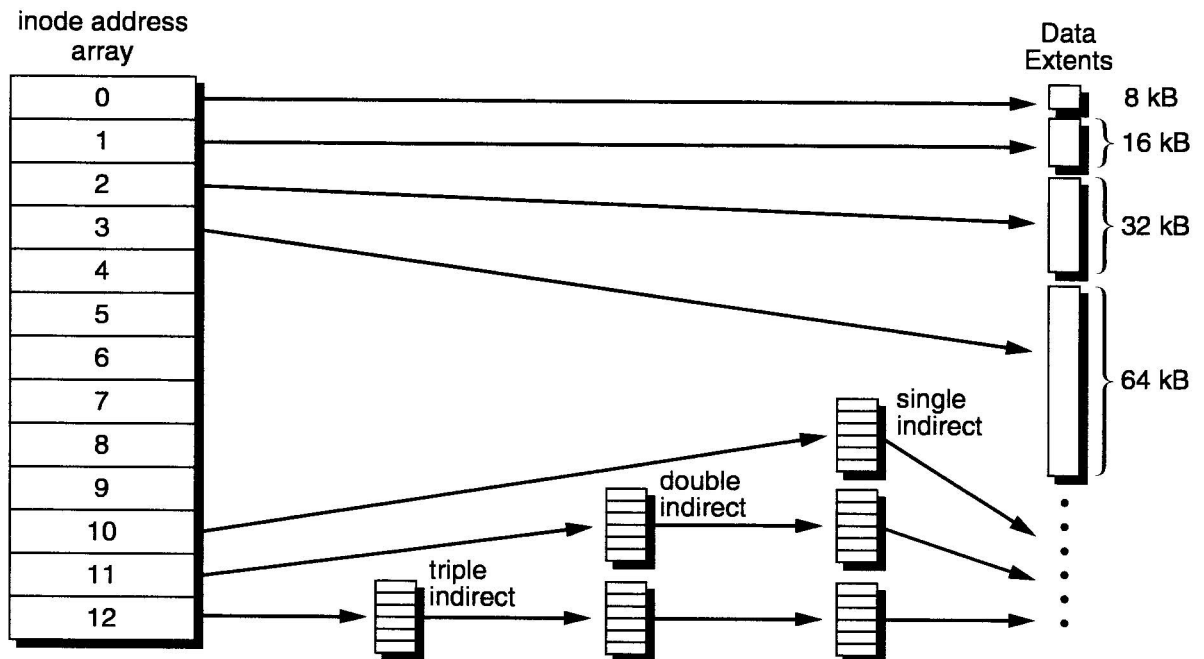
xfs Data Blocks

Data blocks in an **xfs** file system are handled in exactly the same way as in the other file systems.

xfs Space Allocation

In an **xfs** file system space is divided into blocks and fragments. A block can be one of 512 bytes, 1Kbytes, 2Kbytes, 4Kbytes, 8Kbytes, 16Kbytes or 32Kbytes. A fragment can be 512 bytes, 1Kbytes, 2Kbytes or 4Kbytes. The fragment size must be less than or equal to the block size.

The space allocated to a file is tracked in an **xfs** inode, which contains pointers to all the blocks in use. Since an inode is a 64 byte structure it could not contain direct references to every block used in a large file. Instead an inode contains 13 block pointers: ten of these reference blocks containing real data directly; the remaining three pointers reference data with increasing levels of indirection.



The first direct pointer of an inode can reference fragments, allowing the efficient use of space for small (less than a block in size) files.

The “single indirect” pointer references an indirect block that contains pointers to blocks that contain real data. To access data via the indirect block, the operating system must read the indirect block, find the appropriate direct block pointer, and then read that direct block to find the data. “Double indirect” pointers reference indirect blocks that contain references to further indirect blocks. “Triple indirect” pointers reference indirect blocks that contain pointers to double indirect blocks.

To reduce the need for indirection for large files an extent-based allocation policy is used. The file system can be configured with an extent size between 64Kbytes (the default) and 1024Kbytes, in powers of 2.

For ordinary (non-contiguous) files, the initial allocation is of fragments up to the file system’s configured block size, from which point space is allocated in blocks. If the first write to the file is larger than the block size, the first direct pointer references an extent equal to the size of the write rounded up to a power of 2 in the range 8Kbytes to 4Mbytes. The maximum extent size is evaluated by 128 times the configured block size. Subsequent extents have sizes which are twice the previous extent size up to a limit equal to the greater of the first extent size and the configured extent size.

The following examples illustrate the above rules.

The default file system configuration is extent size 64Kbytes, block size 8Kbytes, and fragment size 512 byte. Assuming the file is written to sequentially using 256 byte records, up to 16 fragments can be referenced by the first direct pointer. Further space is allocated in extents of 16Kbytes, 32Kbytes and 64Kbytes. Thus, the direct pointers can reference approximately $(8 + 16 + 32 + 7*64) = 504$ Kbytes.

If the configured block size is 32Kbytes and the first write to a file is 4Mbytes, then this size will be used for the first extent. All subsequent extents will be this size, regardless of the configured extent size. Therefore, for this file $(10*4) = 40$ Mbytes of data can be referenced by the direct pointers.

For contiguous files - specified using **fallocate (3x)** - the first direct pointer is to the contiguous extent. Further space may be added to the file by writing beyond this extent, in which case the second extent is the smaller of twice the size of the first extent and the maximum allowable as detailed above, and so on for the next and subsequent extents.

Excessive space wastage within non-contiguous files is avoided by the reclamation of any unused blocks within an extent after a file has been closed. For instance, with an 8Kbytes block size, a file of 9Kbytes is initially allocated an 8Kbytes extent and a 16Kbytes extent. However, since one block in the second extent is unused it is reclaimed after the file has been closed, reducing wasted space from 15Kbytes to only 7Kbytes.

For contiguous files, unwanted space must be explicitly returned using **free (3x)**.

xfs File System Administration

The following file system administration commands have **xfs**-specific variants:

- **df** – report file system space usage;

- **ff** – list filenames and statistics for file system;
- **labelit** – provide labels for file system;
- **mkfs** – construct file system (see “Creating an xfs File System” below);
- **mount** – mount file system;
- **ncheck** – list pathnames versus i-numbers;
- **volcopy** – make literal copy of file system.

The following generic commands apply to **xfs** file systems:

- **fstyp** – determine file system type;
- **mountall** – mount multiple file systems;
- **umount** – unmount file system;
- **umountall** – unmount multiple file systems.

Most of these commands may be invoked this way:

```
command [-F xfs] [-V] [-o specific_options] operands
```

-F xfs

can be omitted if the FSType is determinable from `/etc/vfstab` by matching an entry in that file with one of the *operands* specified.

-V

causes the command to echo the completed command line, including additional information derived from the `vfstab` file – this can be used to verify and validate the command line. It does not cause the command to execute.

-o specific_options

are specified in a comma-separated list of keywords and/or keyword-attribute pairs for interpretation by the FSType-specific module of the command.

operands

are FSType-specific.

Creating an xfs File System

To create a new file system or convert an old one to a new logical block size, use the following procedure:

1. If the new file system is to be created on a disk partition that contains an old file system, back up the old file system. For information, see Chapter 10, “Archiving and Restoring Data”. To back up systems with one or more hard disks use **cpio**.
2. If the new file system is to be created from an old file system, run **labelit**, which reports the mounted file system name and the physical

volume name of the old file system. These labels are destroyed when you make the new file system, so you must save them to restore later.

3. If the new file system is to be created from an old file system and the new file system will have a larger logical block size, then, because of fragmentation, the new file system will allocate more disk blocks for data storage than the old. For example, to find out how full the old file system is use the **df** command as shown below:

```
df -F xfs -k /dev/dsk/0s6
```

and if the old file system is nearly full, use **format (1M)** to find out the size of your current disk partitions. If the new file system requires a disk repartition, see section “*Types of Devices*” in Chapter 2.

4. Use **mkfs** to make the new file system with the appropriate logical block size.
5. Run **labelit** to restore the file system and volume names.
6. Populate the new file system – for example, do a restore from a file system backup or, if your system has two hard disks, do a **cpio** from a mounted file system. (The **volcopy** and **dd** commands copy a file system image; they cannot convert logical block size.)

The Distributed xfs File System Type

The distributed **xfs** file system (**xfsd**) is designed to provide shared access to a single file system from multiple host systems on a multi-initiator disk bus. Unlimited read access to files in the shared file system is supported. Only limited write access to files in the shared file system is supported. The main topics covered in this section are:

- **xfsd** Overview
- Required Network Connection
- Mounting an **xfsd** File System
- Limitations on Client Access to the **xfsd** File System
- System Failures (in **xfsd** file system)
- Example on How to Create an **xfsd** File System

xfsd Overview

The **xfsd** file system is non-symmetrical in the way that the file system is accessed. One host system first mounts the file system via a normal **xfs** mount. This system is thereafter considered the master system. Other client systems must mount the shared file system with an extra parameter, which indicates the hostname of the master system. Certain file system operations can only be performed on the master system. The master system must

mount the file system before any client systems are permitted to do a client mount of the file system.

System partitions such as / (root), /usr and /var should not be shared. It is also recommended that these system partitions be kept on a disk that is accessed through a private disk bus (that is, accessible only from one host processor). Any partitions that are located on a disk that is accessed from multiple systems, could accidentally be directly mounted (that is, mounted as a standard **xfs** file system) by more than one host. Directly mounting a file system by more than one host can easily cause that file system to be corrupted, because file system data is being modified and cached by more than one host system.

Required Network Connection

A network connection is required between the master system and the client systems of a distributed **xfsd** file system. The network connection is used to pass information about file system structures in support of operations that can only be performed on the master system and to maintain file system consistency. When mounting an **xfsd** file system on a client system, an extra parameter must be specified, which indicates the hostname of the master system. The hostname used on the mount command will determine the network path used for communication to the master, on systems that have multiple networking interfaces.

In many ways, the **xfsd** file system is similar to the **NFS** file system. The difference is that **NFS** passes file data across its network connection while the **xfsd** file system reads and writes data directly to a shared disk drive and only file system structure and synchronization information is passed across the network connection. The network connection media for an **xfsd** configuration could be ethernet, FDDI or VMENet.

Mounting an xfsd File System

The master system in an **xfsd** configuration mounts the **xfsd** file system using a standard **mount** command. Once the master system has mounted the **xfsd** file system, the client systems can do their remote mounts of the file system. The client mounts specify the master's hostname, which is used to identify the network connection to the master system.

Only one host can be a master for a given file system. The **XFS_SYSID** tunable can be used to prevent multiple host systems from attempting to directly mount an **xfs** file system. This tunable should be assigned a unique value for all systems attached to the multi-initiator bus. This tunable defines a unique system ID. When a mount of an **xfs** file system is performed, and the **XFS_SYSID** tunable is set, the system ID is written to a special location in the file system. If there is already a system ID stored in the file system, it must match the system ID of the mounting system for the mount to succeed. It is possible to mount a file system that has an outdated system ID by use of the “**-o force**” option on the **mount** command. This option will ignore the existing master system ID in the file system and overwrite the file system's master system ID with the mounting system's ID. This should not be done unless it is known that no other system currently has the file system actively mounted as the file system master.

Once the master system has mounted the **xfsd** file system in the normal manner, a client system would specify its mount by using an extra option on the **mount** command that indicates the hostname of the master host. The format of this option would be:

```
-o host=<hostname>:<host_mount_path>
```

It is strongly recommended that all client systems set their respective **XFS_SYSID** tunable so that they do not inadvertently do a direct mount of the file system by omitting the client specific option on their **mount** commands.

There are no file system imposed limitations on the number of client systems that can mount an **xfsd** file system. The multi-initiator bus used to access the shared disk drive is the limiting factor on the number of clients that can be configured to share the same file system.

Limitations on Client Access to the xfsd File System

Only the master system in an **xfsd** configuration is allowed to directly modify file system structures on the disk. This is because the operating system will cache file system modifications in memory. If multiple systems were allowed to modify file system structures, this memory caching would prevent remote systems from operating on the most recent file system data and would cause file system corruption. To prevent corruption of file system structures, only the master may make modification to these structures. This means that client systems cannot directly create files, truncate files, grow files, or change a file's high water mark. These operations are supported on a client system, but will result in remote procedure calls to the master system, where the operation is performed on behalf of the client system. Other operations which reference file system structures in a read-only mode, for example filename lookup, file open, directory read, or **statvfs(2)**, are also passed to the master system via remote procedure call over the network link for performing the actual operation. This is because the master system might have file system structures in its memory cache.

Writing to a file from a client system is allowed, but only when the write overwrites an existing portion of the file being written. If a file has disk space allocated and the high water mark is set beyond the data being written, then that portion of the file can be written from any connected system that has the file open. Use the **fallocate(3x)** function to pre-allocate disk space. Use the **ftruncate(3c)** function to set a file's high water mark.

Updates to the file are serialized by access to the disk bus. Having multiple writers to a file which have the file opened in buffered mode will cause unpredictable updates to the file. Buffered data resides in memory and is therefore not seen on other connected systems until it has been written to disk by an operation such as **sync(2)**, **fsync(2)**, **close(2)**, or one of the internal operating system primitives that causes data to be flushed from the memory cache to the disk. Processes that are reading a file on other connected systems cannot see the buffered data until it has been truly written to disk. If files that are being written are to be accessed simultaneously from multiple systems, they should be opened for synchronous I/O (**O_SYNC** or **O_DSYNC**), which forces data to be written to the disk before the I/O operation is considered complete. The use of synchronous I/O allows the user to avoid synchronization problems introduced because of buffer cache behavior. Direct I/O could also be used to write data directly to disk, however, it should be noted that direct I/O has restrictions on the alignment and length of I/O trans-

fers. If these restrictions are not met, the I/O request is silently turned into a buffered I/O request.

Operations which modify the size of a file or which set the high water mark of a file (**write(2)**, **fallocate(3x)** or **ftruncate(3c)**) are only allowed on a client system when that system is the only system which currently has the file open. When a file is opened on more than one system, that file's structure is essentially locked in place by the rules that the master system enforces. If a client system meets the restriction of having the only opened access to the file and it issues an operation that modifies the file's structure, the modifications to the file structure are performed on the master system. The new file structure is returned to the client system so that it may update its local cached version of the file's structure. It should be noted that file expansion can only be performed on client systems via a combination of the **fallocate(3)** call (which pre-allocates disk space to the file) and the **ftruncate(3)** call (which sets the high water mark of the file).

Because of the limitations on writing files from a client system, standard utilities such as **ftp**, **tar**, **cp**, **cc**, etc, cannot be used to write to a file in an **xfsd** file system from a client system. This is because these utilities open files in truncate to zero mode, making those files non-writable on a client system.

The **xfsd** file system supports client operations on files, directories and symbolic links as file system objects. Block device files, character device files and named pipes may exist in an **xfsd** file system, but they can only be created, deleted or accessed from the master system. Processes on client systems will be returned an error if they attempt to access these file system objects.

No file locking is supported between systems in an **xfsd** configuration.

System Failures

The **xfsd** file system was designed for high availability. Restart of both the master system and client systems is supported. If a master system fails and cannot be restarted, one of the client systems can be designated as the new master system. Failure of one of the systems in an **xfsd** configuration does not impact I/O operations in progress on running client systems. That is, client systems may continue performing read operations or write operations to file space that is already allocated while the master or another client system is down.

The master and client systems monitor the health of one another using keep-alive synchronization messages that are transmitted once per second. If this synchronization times out, the other system is considered down. When a master system fails and is subsequently brought up again, the client systems must remount their disks. This remount re-establishes any file update locks on the master that the client system may hold. The remount of client systems is accomplished with an option to the **mount(1M)** command:

```
-o remount,host=<hostname>
```

It is not necessary to unmount the file system before doing a remount of a client file system. File I/O operations on the client can proceed while the remount is taking place. Operations that modify file system structures should not be attempted until all client systems have remounted the shared file system. This is important since the client systems may be actively doing I/O to a file. If the master were to modify the structure of a file (for

example if the file were removed or truncated) without re-mounting a client system that is actively writing to the file, then file system corruption could occur.

If a master system cannot be restarted, one of the client systems can mount the shared file system as the new master system. If the `XFS_SYSID` tunable is being used, then a special option on the `mount (1M)` command is required so that the system will overwrite the master system ID in the file system with the system ID of the new master system. The format of the option on the `mount (1M)` command for rewriting the master system ID is:

`-o force`

Example on How to Create an xfsd File System

This section provides an example on how to create an `xfsd` file system. Topics covered are:

- `xfsd` Setup
- Hardware Connections
- Software Setup

`xfsd` Setup

The following procedure describes an example set up for a multi-initiator shared disk. See Figure 3-5 for connection details. SCSI ID numbers shown are for example purposes only and may differ in your system configuration. This example is for sharing a SCSI disk between two single board computers in a closely-coupled system (CCS) configuration. The NCR1 SCSI controller is used as the multi-initiator bus for sharing an `xfs` file system. See the *Diskless Systems Administrator's Guide* for more information on CCS.

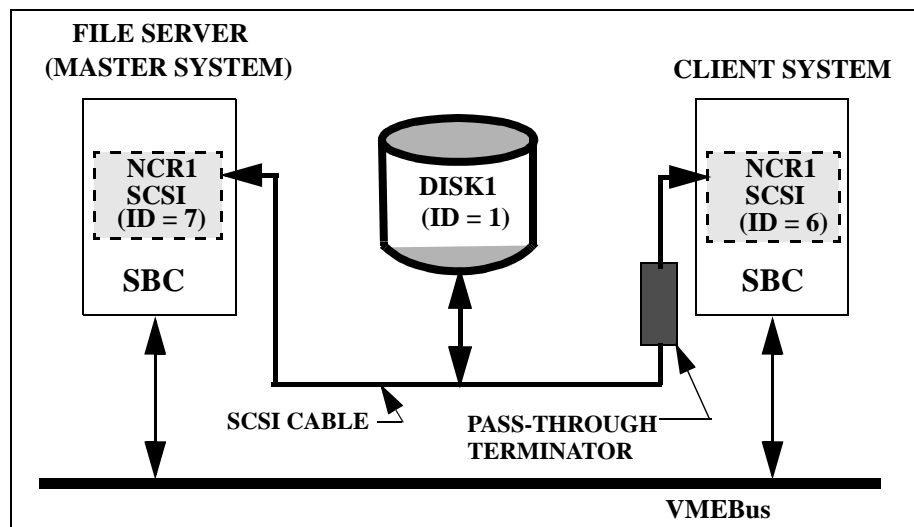


Figure 3-5. Multi-Initiator SCSI `xfsd` Configuration

Hardware Connections

1. Connect one end of the SCSI cable to NCR1 of the master SBC.
2. Connect the other end of the cable via a Pass-Through terminator to NCR1 of the client SBC.
3. Connect your shared disk to any of the other nodes on the SCSI cable.

Software Setup

The software setup involves modifying several files and then rebuilding the kernel statically. This discussion assumes that a shared disk has already been configured on the local client as a local disk. See the section on configuring a local disk in Chapter 6 of the *Diskless System Administrator's Guide*.

1. Verify that the **xsfd** software package has been installed on the master and client systems. To verify, run **pkginfo -l xsfd**. If the **xsfd** package has not been installed, refer to Chapter 7, “Installing Add-on Software” in Volume I of the *System Administration* manual, for installation instructions.
2. Modify the tunable **XFS_SYSID** on each SBC to assign them a unique number as their system ID. For example, the file server's tunable **XFS_SYSID** could be set to 1 while the clients's tunable **XFS_SYSID** could be set to 2. Run the **config** utility on each SBC system and modify the tunable **XFS_SYSID**. For the client system run **config -r /home/vroots/<clientname>** on the file server.
3. Modify the **NCR_1_ADAPTER_ID** on the client system and assign a unique ID. The NCR controller is normally ID 7, but you cannot have two NCR controllers on the same SCSI bus with the same SCSI ID. Run **config -r /home/vroots/<clientname>** and modify the tunable **NCR_1_ADAPTER_ID** to be 6 (or any other SCSI ID number that does not conflict with any other SCSI ID number on this bus).
4. Ensure that there is a node entry for the shared disk device for each system sharing the disk. The file server's node file is located at **/etc/conf/node.d/gd** while the client's node file is located at **/home/vroots/<client>/etc/conf/node.d/gd**.
5. Run **config -r /home/vroots/<clientname>** on the file server and enable the following modules if not already enabled.

```
scsi  ncr  gd  xfs  xfsth  xsfd
```

6. Build the master kernel as a static kernel. Refer to Chapter 8, “Configuring and Building the Kernel” for information on how to build a static kernel.
7. On the file server, remove any share entry lines in the file **/etc/dfs/dfstab.diskless** for disk partitions that are a part of the **xsfd** file system. **xsfd** partitions cannot be shared via NFS.
8. On the file server, place entries in the **/etc/vfstab** file for shared disk partitions. These entries should be set up as standard mount point entries.

9. On the client system place entries similar to the following in the `/home/vroots/<clientname>/etc/vfstab` file. Note that these entries include an additional parameter of the form “host=<hostname:mount point>,” for specifying the network hostname of the master system.

```
/dev/dsk/1s0 /dev/rdisk/1s0 /1s0 xfs - yes host=vmenet:/1s0
/dev/dsk/1s1 /dev/rdisk/1s1 /1s1 xfs - yes host=vmenet:/1s1
```

10. To ensure that the your mount points, `/1s0` and `/1s1` in this example, stay intact on the client system across reboots, make entries for them in the file `/etc/diskless.d/custom.conf/client.private/<client-name>/memfs.files.add`. Alternatively, you could place your mount points in the directory `/home/vroots/<clientname>/var` on the client system where they will remain intact. If the latter method is used, you will need to adjust the preceding example accordingly (i.e., pathname would be “`/var/1s0`” and “`/var/1s1`” in each instance).
11. Execute the command `mkvmebstrap -r -all <clientname>` in order to rebuild the `bstrap` image on the client system.
12. Reboot the system. The shared file system will initially mount up as a normal `xfs` file system on the file server. Next, the shared file system will mount up as a distributed `xfs` file system on the client system as the client boots up.

The sfs File System Type

The `sfs` FSType is a variation of the `ufs` FSType. It includes changes to support security features, including extended discretionary access control (DAC) and mandatory access control (MAC) data structures that are stored for each file. (These access control features are available only if the Enhanced Security Utilities are running on your system.) The boot block, super-block, storage blocks, and free blocks for the `sfs` file system type are, at the administrative level, identical to those for `ufs`. See the discussion of these items in “*The ufs File System Type*.” The inodes, which differ from `ufs` inodes, are described below.

The sfs Inodes

An alternate inode scheme is used for `sfs`, in which two inodes are allocated for each file.

- The first inode, which is always an even-numbered inode, contains common file information and is essentially identical to a `ufs` inode.
- The second (alternate) inode, which is always an odd-numbered inode, contains security information.

Programs that report inode usage will show that only even-numbered inodes are in use, because directory entries contain only even inode numbers.

All alternate inodes on the file system are preallocated by the **mkfs** command.

The alternate inode contains the following information:

- the level identifier (LID) for the file's Mandatory Access Control (MAC) level
- the flags for multilevel directory
- Discretionary Access Control (DAC) information, which includes
 - the count of Access Control List (ACL) entries for the file
 - the count of default ACL entries for the file
 - a disk block pointer for extended storage of ACL entries
 - an array of ACL entries

The count of ACL entries in the alternate inode is defined by `NACLI`.

The memfs File System Type

The **memfs** file system type is a high-performance, volatile memory file system type. The nature of this file system type is such that users are able to create directories and files, but, because the file system is only in memory, when it is un-mounted, the directories and files go away. As a result, there is no file system for a system administrator to administer.

When mounting **memfs** (using `mount_memfs`), you can use the `-o` option to specify file system-specific options. These file system-specific options are

- `swapmax`
- `rootmode`

NOTE

Refer to `mount_memfs` for complete information on this file system-specific `mount` command.

`swapmax` sets the maximum amount of memory allowed in bytes. `rootmode` specifies the mode of the root directory of mounted **memfs**. `rootmode` allows system administrators to set a sticky bit for `/tmp` and `/var/tmp`.

A **memfs** file system can be configured for each of `/tmp` or `/var/tmp` by using the `mount_memfs` command. Files in `/tmp` or `/var/tmp` do not need to survive system crashes or reboots, so they can take advantage of a high performance, volatile file system.

For more information on `/tmp` and `/var/tmp`, refer to the “Managing System Performance” chapter in “*System Administration*”.

Administering a File System

The Virtual File System architecture allows multiple file system types (FSTypes) to coexist in the operating system kernel. Each FSType has certain characteristic features that it does not share with any other FSTypes. However, the file system administrative commands provide a common interface that allows the administrator to maintain file systems of differing types.

NOTE

If the Enhanced Security Utilities are installed on your system, use only user file systems of type **sfs** to ensure the security of your system; **ufs** and **xfs** file system types will not support Mandatory Access Control restrictions.

The Generic Administrative Commands

The following commands for file system administration can be used on multiple FSTypes (exceptions noted):

- **dcopy (1M)**
- **df (1M)**
- **ff (1M)**
- **fsck (1M)** (not applicable for **xfs** file system)
- **fsdb (1M)**
- **fstyp (1M)**
- **labelit (1M)**
- **mkfs (1M)**
- **mount (1M)**
- **mountall (1M)**
- **ncheck (1M)**
- **umount (1M)**
- **umountall (1M)**
- **volcopy (1M)**

How the Administrative Commands are Invoked

Most of these commands may be invoked this way:

```
command [-F FSType] [-V] [current_options] [-o specific_options]
operands
```

- The **-F** is used to specify the FSType on which the command must act. The FSType must be specified on the command line or must be determinable from **/etc/vfstab** by matching an entry in that file with one of the *operands* specified. (See the section below for information on the **vfstab** file.)
- The **-V** option causes the command to echo the completed command line. The echoed line will include additional information derived from the **vfstab** file. This option can be used to verify and validate the command line. It does not cause the command to execute.
- The **-o** option is used to specify FSType-specific options. *specific_options* are options specified in a comma-separated list of keywords and/or keyword-attribute pairs for interpretation by the FSType-specific module of the command.
- *operands* are FSType-specific; the FSType-specific manual page of the command should be consulted for a detailed description.

The vfstab File System Table

Because the generic commands work on multiple FSTypes—for example, **mount** can mount a **sfs** or **ufs** file system among other types—they require FSType-specific information which may be provided explicitly on the command line or implicitly through the file system table **/etc/vfstab**.

What Does the vfstab File System Table Do?

The file system table is an ASCII file with two functions:

- to describe the file systems that will be mounted automatically
- to provide missing default values for the **mount** command for file systems it (**vfstab**) lists.

For each file system type, **vfstab** contains a record consisting of the following fields (separated by spaces):

```
special fsckdev mountp fstype fsckpass automnt mntopts ceiling
```

The meaning of each field is as follows:

<i>special</i>	the block special device for local devices or the resource name for remote file systems (for example, nfs). (For more information on the nfs file system types see <i>Network Administration</i> .)
<i>fsckdev</i>	the character special device that corresponds to the <i>special</i> . The block special device is used if the character special device is not available. Use a “-” where there is no applicable device. (For example, a memfs file system would have a “-” for this field.)

<i>mountp</i>	the default mount directory (mount point).
<i>fstype</i>	the type of the file system on the special device.
<i>fsckpass</i>	the pass number to be used by ff , fsck , and ncheck to decide whether to check the file system automatically. Use “-” to inhibit automatic checking of the file system.
<i>automnt</i>	yes or no for whether the file system should be automatically mounted by mountall when the system is booted. If this field is yes and the file system is not clean, the file system will be checked with the -y option.
<i>mntopts</i>	a list of comma-separated options that will be used in mounting the file system. Use “-” to show no options. See mount (1M) for a list of the available options. Lines beginning with the # character are comments.
<i>ceiling</i>	the maximum of the mounted file system Mandatory Access Control (MAC) level range. The minimum is taken from the level of the directory that is the mount point. You can use the -l option of the mount command to explicitly set the maximum of the file system level range. See the mount (1M) manual page and the “Administering Mandatory Access Control and Multilevel Directories” chapter of System Administration, Volume 1, for information on MAC levels and level ranges. This field is used only when the Enhanced Security Utilities are installed.

NOTE

Do not store information in the **vfstab** file other than the fields described above; fields may be added to this file in future releases and are reserved for future use.

Commands for sfs File Systems

Because **sfs** file systems use alternate inodes, it is important to use the **sfs** variants of all commands that either report or manipulate **inode** information. In most cases, the commands for **sfs** file systems use the generic command interface with the **-F sfs** option. This option makes the generic command invoke the correct version for **sfs** file systems. The following commands all have **sfs** variants:

df (1M)	This command is used to report free disk space on file systems. When the command reports the number of free and used inodes, it will report only on even-numbered inodes.
fsck (1M)	This command is used to check and repair file systems.
fsdb (1M)	This command is a file system debugger.
labelit (1M)	This command is used to provide labels for sfs file systems. (Note that these labels are not sensitivity labels.)

mkfs (1M)	This command is used to construct file systems.
mount (1M)	This command is used to mount a file system that has been created by mkfs .
ncheck (1M)	This command prints a list of pathnames, based on inodes. The inodes numbers given to the command must be even.
volcopy (1M)	This command is used to make a literal copy of a file system.

For details, see the manual pages for **sfs-specific** commands in section 1M.

Listing Installed File System Types

Use the **crash (1M)** command to display a list of FSTypes installed in the kernel. The following will produce such a list:

```
crash <<!
vfsw
!
```

In addition to the familiar FSTypes, **crash** will also list certain internal FSTypes, such as **specfs**, that have no user interface.

Identifying the Type of an Unmounted File System

Most commands that are used in file system administration require that the FSType of a file system be provided on the command line or in the file system table. Most of these commands also attempt to distinguish the type of a file system by themselves, so if the administrator provides the wrong type the command may fail. However, it is important to specify the correct type because file systems may be damaged if a command fails to detect an administrator's error and an operation applicable only to one type of file system is applied to another.

Sometimes, the administrator will have to try to determine the type of an unmounted file system type either because the **vfstab** file contains outdated information or because it contains no information at all. The command **fstyp (1M)** uses heuristics to determine the type of an unmounted file system. **fstyp** determines and displays the file system type on standard output. If the type cannot be determined, the message **unknown_fstyp (no matches)** appears as standard error output.

Creating a File System

Using mkfs to Create a File System

Once a disk is formatted the next step is to define the file system. The **mkfs (1M)** command is used for this purpose. This is the generic format of **mkfs (1M)**:

```
mkfs [-F FSType] [-V] [-m] [current_options] [-o specific_options] \  
    special [operands]
```

(The above command line is shown on two lines for readability; the backslash (\) is used to show that the second line is a continuation of the first line.)

- **mkfs** constructs a file system by writing on the *special* file, which must be the first argument. The file system is created based on the *FSType* specified with the **-F** option, the *specific_options*, and *operands* specified on the command line.
- Use the **-F** option to specify the *FSType* on which the command must act. You must either specify *FSType* on the command line or let **mkfs** determine it from `/etc/vfstab` by matching an entry in that file with one of the *operands* specified. (See the section above for information on the `vfstab` file.)
- The **-V** option causes the command to echo the completed command line. The echoed line will include additional information derived from `/etc/vfstab`. This option can be used to verify and validate the command line. It does not cause the command to execute.
- The **-m** is used to return the command line that was used to create the file system. The file system must already exist and this option provides a means of determining the attributes used in constructing the file system. Note that file systems cannot be constructed for all *FSTypes*. Care must be taken to specify valid *FSTypes*.
- The **-o** option is used to specify *FSType*-specific options if any. *specific_options* are options specified in a comma-separated list of keywords and/or keyword-attribute pairs for interpretation by the *FSType*-specific module of the command.
- *operands* are *FSType*-specific; the *FSType*-specific manual page of the command should be consulted for a detailed description.

NOTE

For complete information on all filesystem-specific **mkfs (1M)** commands, please refer to the appropriate manual page. All **mkfs** options are described.

Choosing a Logical Block Size

Logical block size is the size of the blocks that the operating system kernel uses to read or write files. The logical block size is usually different from the physical block size, which is the size of the smallest block that the disk controller can read or write, usually 512 bytes.

The **ufs** file systems support logical block sizes of 2048 (2K), 4096 (4K), and 8192 (8K) bytes. An administrator can choose a logical block size during installation or with the **mkfs (1M)** command when making a new file system after the system is installed. For the **sfs** and **ufs** file systems, a default block size equal to 8192 bytes is used. **mkfs (1M)**

uses a fragment size equal to 1/8, 1/4, or 1/2 of the block size; otherwise the default fragment size will be 1024 bytes.

What to Consider When Choosing a Logical Block Size

To choose a reasonable logical block size for your system, you must consider both the performance desired and the available space. For information on disk performance, see section entitled “*Improving and Controlling System Performance*” in Chapter 5.

Using mkfs to Create a ufs File System

To create a new file system or convert an old one to a new logical block size, use the following procedure:

If the new file system is to be created on a disk partition that contains an old file system, back up the old file system. For information, see the “*The Backup and Restore Services*” chapter in this manual; To back up systems with one or more hard disks use **cpio(1)**.

1. If the new file system is to be created from an old file system, run the **labelit** command, which reports the mounted file system name and the physical volume name of the old file system [see **volcopy(1M)**]. These labels are destroyed when you make the new file system, so you must save them to restore later.
2. Use the **mkfs(1M)** command to make the new file system with the appropriate logical block size. The **mkfs(1M)** command is described in the previous section, “*Using mkfs to Create a File System.*”
3. Run the **labelit** command to restore the file system and volume names.
4. Populate the new file system—for example, do a restore from a file system backup or, if your system has two hard disks, do a **cpio(1M)** from a mounted file system. [The **volcopy(1M)** and **dd(1M)** commands copy a file system image; they cannot convert logical block size.]

Using mkfs to Create an sfs File System

The steps for creating an **sfs** file system are identical to those for creating a **ufs** file system. To create an **sfs** file system, follow the procedure outlined in “*Using mkfs to Create a ufs File System,*” using the **sfs-specific** version of **mkfs**. However, you must consider that because of discretionary access control (DAC) and mandatory access control (MAC), significantly more information is being kept by the system for each file. (These access control features are available only if the Enhanced Security Utilities are running on your system.) To create a new **sfs** file system or to convert an old file system to **sfs**, use the following procedure:

1. If the new file system is to be created on a disk partition that contains an old file system, use the **tcpio** command to back up the old file system, maintaining complete security information on each file. For more information on **tcpio**, see the “*Trusted Backup and Restore*” chapter in “*System Administration*”, Volume 1.

If the old file system is a **ufs** or **sfs** file system, use **cpio** to back up the file system. You will read it back in with **cpio -i**. When the files are restored, they are given your current security level (that is, your login level when you execute **cpio -i**). If this level is not appropriate for the data in the files, use the **chlvl** command to change the security levels.

2. If the new file system is to be created from an old file system, run the **labelit** command, which reports the mounted file system name and the physical volume name of the old file system [see **volcopy(1M)** .] These labels are destroyed when you make the new file system, so you must restore them.
3. Use the **mkfs(1M)** command to make the new file system with the appropriate logical block size. The **mkfs(1M)** command is described above under “Using *mkfs* to Create a File System.”
4. Run the **labelit** command to restore the file system and volume names.
5. Populate the new file system—for example do a restore from a file system backup, or, if your system has two hard disks, do a **tcpio(1M)** from a mounted file system. [The **volcopy(1M)** and **dd(1M)** commands copy a file system image; they cannot convert logical block size.]

Moving and Copying File Systems

At times you may want to move file systems on your system. You may want to do this, for example, because you have more space available in slice A than in slice B, and you want to add data to the data already in slice B. Similarly, you may decide that by rearranging some of the file systems within a single disk, you can make more efficient use of the total amount of space on that disk. Most of the time the source and destination slices are both file systems, and you have to swap them.

To swap the contents of slices A and B, you must

1. copy the contents of slice A on tape
2. remove the contents of slice A (which you've just copied on tape)
3. copy the contents of slice B on slice A
4. remove the contents of slice B (which you've just copied on slice A)
5. move the contents of the tape (created in step 1) to slice B.

NOTE

Moving the **/usr** file system can be difficult because of dependencies on **/usr/bin**, **/usr/lib**, and any shared libraries.

Before you start moving the contents of disks and/or slices, be sure to back up both the source and destination file systems. (For instructions, see the “*The Backup and Restore Services*” chapter in this manual.)

Moving a file system is a three-step procedure:

1. Copy the source file system to a new location (or “destination”).
2. Remove the source file system from its original location.
3. Edit the `/etc/vfstab` file.

Step 1: Copying the Source File System

To make a copy, use either the `cpio` or `volcopy` command. These two commands accomplish the work of moving file systems in very different ways. `cpio` copies the files in one file system to another, whereas `volcopy` makes a literal copy of the file system as a whole.

The `cpio` and `volcopy` commands are not equivalent. You should be aware of the consequences of using either one and choose whichever makes more sense for your situation.

Once the destination file system is complete, you can remove the source. Follow the instructions under “*Steps 2 and 3: Removing the Source File System and Editing vfstab*” below.

Copying a File System with `cpio`

Using `cpio` presumes that a destination file system already exists, whether newly created or not. The files of the source file system are installed into this existing destination file system.

`cpio` does not move a file system as a single unit; it moves a file system by transferring its constituent files one at a time. Thus, you can use this command when transferring a file system between disks of different sizes. As long as the destination disk has enough free space to accommodate the files it’s receiving, you can transfer that file system to it with `cpio`.

To move a file system with `cpio`, enter

```
find . -mount -print | cpio -pdm dest_directory
```

Copying a File System with `volcopy`

If you’re moving a file system between two disks of the same size, we recommend running `volcopy`, because it’s the faster of the two commands. `volcopy` owes its speed, however, to the fact that it moves a file system as a single unit, overwriting the destination disk completely with the contents of the source disk. As mentioned above, any pre-existing contents on the destination disk are destroyed when you move a file system to that disk with `volcopy`.

The `volcopy` options are not identical for the different types of file systems that may be on your system. For details on the options, see the appropriate `volcopy (1M)` man page. (Separate versions exist for the `ufs`, `xfs` and `sfs` file system types.)

Steps 2 and 3: Removing the Source File System and Editing `vfstab`

Once you've copied the file system to its new location, you can remove it from its original location. The safe way to do this is as follows:

1. Remove the files by entering

```
find src_path -mount ! -type d -print | xargs rm -f
```

2. Remove the directories by entering

```
find src_path -mount -depth -type d -print | xargs rmdir
```

An alternative is to remove both your files and directories simultaneously.

1. Make sure that none of the subdirectories below `src_path` is a mounted file system (since `rm` crosses mount points and might therefore remove more than you intended).

2. Then enter

```
rm -rf src_path
```

3. For either of the above methods, you can complete the procedure by editing the `/etc/vfstab` file (with an editor such as `vi`) so the moved file system refers to the new location (that is, its new slice).

Mounting and Unmounting File Systems

For a file system to be available to users, it must be mounted. The `/usr` and `/var` file systems are always mounted as part of the boot procedure. The `/usr` file system may be on the same disk device as the root file system. The `mount` command that makes these two file systems available is contained in start-up shell procedures.

The `mount` command causes the mounted disk device and the mounted-on directory (the "mount point") to be associated with certain other information in the `/etc/mnttab` file. This information includes data such as the `FSType`, the mount options used during the mount, and the time the mount was performed. [See `mnttab(4)`.]

Example of Mounting File Systems

- The command

```
mount -F ufs /dev/dsk/1s0/usr
```

tells the system to mount `/dev/dsk/1s0` as a `ufs` file system that begins at the directory `/usr`.

Using the `mount` Command

If you try, before the `mount` command is issued, to change directories [with the `cd(1)` command] to a directory in the `/usr` file system, the `cd` command will fail. Until the

mount command completes, the system does not know about any of the directories in the **/usr** file system. True, there is a directory called **/usr**. (It must exist when the **mount** command is issued.) The file system below **/usr**, however, remains inaccessible until the **mount** command completes.

Unmounting a File System

The command for unmounting a file system requires only the name of the special device or the mount point.

- When unmounting a file system being used by other processes (including your own shell process, if your current directory is on the file system you wish to unmount), **umount** fails and displays the following message:

```
umount : file_system busy.
```

- To kill the processes using the file system, enter

```
fuser -cuk file_system
```

- Unmounting is frequently a first step before using other commands that operate on file systems. For example, **fsck**, which checks and repairs a file system, works on unmounted file systems. Unmounting is also an important part of the process of shutting the system down.

NOTE

If you need to unmount a file system containing device special files whose security attributes have been set, you will have to de-allocate the device special files that have the release flag set to either `DEV_PERSISTENT` or `DEV_LASTCLOSE`. Otherwise, the file system will be reported as busy and you will not be able to unmount the file system.

Mounting ufs File System under sfs

On systems with the Enhanced Security Utilities installed, all device special files must be on **sfs** file systems. In addition, certain file systems (for example, the root file system) must be **sfs** file systems. However, it is possible to mount a **ufs** file system on secure systems.

When mounting the **ufs** system, all associated files and directories are treated as having the security level of the mount point. With this type of file system (called single-level file system) the only security level is that of the mount point. Any new files on the single-level file systems can be created only by processes whose security level is equal to the security level of the mount point. Processes can read files in these directories only if their security level dominates that of the mount point.

It is not possible to add Access Control Lists (ACLs) to files in **ufs** file systems. If you attempt to use the **setacl** command on files in a **ufs** file system, it will return an error message. The **getacl** command will report the base entries that correspond to the mode bits set on the files.

Mounted File System Level Range

All **sfs** file systems have the range of security levels assigned to them when they are mounted for use. The mounted file system level range restricts the creation of file system objects in a mounted file system. You must assign each file system a level range. The minimum of the range is taken from the level of the directory that is the mount point, and the maximum is either taken from the `/etc/vfstab` file or is specified by the `-l` option of the `mount` command. The maximum level of the level range is also called the `security level ceiling`. See the “*The vfstab File System Table.*” section of this chapter for information on the security level ceiling. (For information on security levels and the other terms used throughout this section, see the chapter entitled “*Administering Mandatory Access Control and Multilevel Directories*” in “*System Administration*”, Volume 1.)

The argument to the `-l` option of `mount` may be either a security level alias or a fully qualified level name. The maximum level of the mounted file system level range must dominate the minimum level.

Device Level Range

The mounted file system level range must be within the device level range specified in the device database. The `devattr` command prints information about device attributes, including the device level range as level identifiers (LIDs) and other security attributes. (You can use the `lvlname` command to display the level names associated with numeric LIDs.) The `putdev` command can be used to set or modify the security level range for a device, if necessary. See Chapter 2, “Managing Storage Devices”, for information on the device level range and how to administer it.

For example, to mount a file system with the `TopSecret` level as the upper level of the level range, with `/dev/dsk/1s0` as the special file and `/projects/proj14` as the mount point, you would use the following command:

```
mount -l TopSecret -F sfs /dev/dsk/1s0 \
    /projects/proj14
```

- `/projects/proj14` is the mount point for the file system, so its security level is the minimum level for the file system level range.
- The `-z` and `-Z` options of `mount` display information about the security level ceilings for mounted file systems. The `-z` option prints the ceiling as a security level alias, while `-Z` prints the fully qualified level name. For example, `mount -z` would print the following information about the file system mounted in the previous example:

```
/projects/proj14 on /dev/dsk/1s0 setuid/read/write on \
    day date time year TopSecret
```

If an unprivileged process attempts to create an object with a level outside the file system level range, permission is denied. Read access to file system objects is allowed, provided that MAC and DAC access requirements are met.

Maintaining a File System

Once a file system has been created and made available, it must be maintained regularly so that its performance remains satisfactory and so that it does not develop inconsistencies. The maintenance to ensure satisfactory performance will be dealt with in the rest of this section, that to ensure consistency in the next.

There are four tasks that should be part of routine maintenance if the administrator wants to be sure that the performance of the file system will be satisfactory. All of them are aimed at ensuring that disk space does not become so scarce that system performance is degraded. The tasks are:

- monitoring the percentage of disk space used
- monitoring files and directories that grow
- identifying and removing inactive files
- identifying large space users.

Monitoring the Percentage of Disk Space Used

Monitoring disk space may be done at any time to see how close to capacity your system is running. Until a pattern has emerged, it is advisable to check every day.

1. To check disk space, enter

```
df -g
```

The **-g** option causes the total allocated blocks and files to be displayed, as well as free blocks and files.

2. If you name no file systems, information about all mounted file systems is displayed. If you want information on unmounted file systems, you have to specify the file system name. See **df (1M)** for available options.

Monitoring Files and Directories That Grow

Almost any system that is used daily has several files and directories that grow through normal use. Some examples are:

File	Use
/var/adm/wtmp	history of system logins
/var/adm/sulog	history of su commands
/var/cron/log	history of actions of /usr/sbin/cron
/var/adm/spellhist	words that spell (1) fails to match

The frequency with which you should check growing files depends on how active your system is and how critical the disk space problem is. A good way to limit the size of such files is by entering

1. `tail -50 /var/adm/sulog > /var/tmp/sulog`
2. `mv /var/tmp/sulog /var/adm/sulog`

This sequence puts the last 50 lines of `/var/adm/sulog` into a temporary file, and then moves the temporary file to `/var/adm/sulog`, thus truncating the file to the 50 most recent entries. These files should be examined for errors before truncating them.

NOTE

When the Enhanced Security Utilities are installed, you need to ensure that the temporary file (`/var/tmp/sulog`) has the same security level and access control permissions as the original log file.

Identifying and Removing Inactive Files

Part of the job of cleaning up heavily loaded file systems involves locating and removing files that have not been used recently. The `find(1)` command can locate files that have not been modified recently. `find` searches a directory tree beginning at a point named on the command line. It looks for filenames that match a given set of expressions, and when a match is found, performs a specified action on the file. To identify inactive files, enter a command line similar to the example below.

```
find /home -type f -mtime +60 -print > \  
/var/tmp/deadfiles &
```

(The command line is shown on two lines for readability.)

where

- | | |
|--|---|
| <code>/home</code> | specifies the pathname where <code>find</code> is to start. Presumably, your machine is organized in such a way that inactive user files will not often be found in the root file system. |
| <code>-type f</code> | tells <code>find</code> to look only for regular files, and to ignore special files, directories, and pipes. |
| <code>-mtime +60</code> | says you are interested only in files that have not been modified in 60 days. -- |
| <code>-print</code> | means that when a file is found that matches the <code>-type</code> and <code>-mtime</code> expressions, you want the pathname to be printed. |
| <code>> /var/tmp/deadfiles &</code> | directs the output to a temporary file and indicates that the process |

is to run in the background. This is a sensible precaution if your experience tells you to expect a substantial amount of output.

Identifying Large Space Users

The **du** (1M) and **find** (1) commands provide useful information when identifying large space users.

The du Command

du produces a summary of the block counts for files or directories named in the command line. For example:

```
du /home
```

displays the block count for all directories in the **/home** file system.

Optional arguments allow you to refine the output somewhat. For example, **du -s** may be run against each user's login to monitor individual users.

The find Command

The **find** command can be used to locate specific files that exceed a given size limit.

```
find /home -size +10 -print
```

This example produces a display of the pathnames of all files (and directories) in the **/home** file system that are larger than ten (512-byte) blocks.

Deleting /tmp Directories

Files in **/tmp** and **/var/tmp** are deleted by the **SO5RMTMPFILES** script in **/etc/rc2.d** when the system comes up to state 2. However, if subdirectories of **/tmp** or **/var/tmp** are mount points, they are not cleared by the system. If you set up directories under **/tmp** as separate file systems, you can either edit **SO5RMTMPFILES** so that they are cleaned up, or add another **/etc/rc2.d/S*** script to do it [see **rc2 (1M)**].

Quotas

If you're the administrator of a **ufs** file system, you have the option of assigning "quotas" to your users. Quotas are "soft" and "hard" limits for both blocks and inodes; they're available only on **ufs** file systems. Quotas allow you to define a time limit (applicable to all users on a file system) indicating how long users can exceed the soft limits. For a full description of quotas, see "*The ufs File System Type*" above.

Quick Reference Guide to Managing File System Types

The administrative procedures in this chapter are based on shell commands. If you prefer, however, you can do many tasks through a set of administration menus, instead. (Using the menus may be convenient if you're administering a system for the first time.) To use these menus, enter **sysadm file_systems**. The main menu for file system administration will appear on your screen as follows:

```

1          Manage File Systems

check- Check a File System
defaults- Manage Defaults
diskuse- Display Disk Usage
display- Display Installed Types
fileage- List Files by Age
filesize- List Files by Size
identify- Identify File System Type
list - List Mounted File Systems
make - Create A File System
mount- Mount a File System
unmount- Unmount a File System
    
```

Screen 3-1. Main Menu for File System Administration

Table 3-2 lists common tasks associated with managing file systems (FS), and shows both the shell commands and the **sysadm** menu items available for performing each task.

Table 3-2. Menus and Commands For FS Administration

Task Description	Menu Item	Shell Command
Check a file system	check	fsck (1M) (except xf s)
Manage vfstab defaults	defaults	any editor
Display disk usage	diskuse	df (1M)
Display installed types	display	crash (1M)
List files by age	fileage	ls -t
List files by size	filesize	du (1M)
Identify a file system type	identify	fstyp (1M)
List file systems	list	mount (1M)
Create a file system	make	mkfs (1M)
Mount a file system	mount	mount (1M)
Unmount a file system	umount	umount (1M)

File System Problems

What Is File System Checking?	4-1
Using the fsck Utility to Check File Systems	4-2
Generic fsck and Its Options	4-2
Parallel File System Checking with the -P Option.	4-2
Checking ufs File Systems	4-3
ufs File System Components Checked by fsck	4-4
Using fsck_ufs to Check Super-Blocks	4-4
Using fsck_ufs to Check File System Size and Inode List Size	4-4
Using fsck_ufs to Check Free Block List	4-4
Using fsck_ufs to Check Free Block Count	4-4
Using fsck_ufs to Check Free Inode Count	4-5
Using fsck_ufs to Check Inodes.	4-5
Using fsck_ufs to Check Format and Type.	4-5
Using fsck_ufs to Check Link Count	4-6
Using fsck_ufs to Check Duplicate Blocks.	4-6
Using fsck_ufs to Check Bad Block Numbers	4-6
Using fsck_ufs to Check Inode Size	4-7
Data Associated with an Inode.	4-7
Using fsck_ufs to Check Directory Data Blocks	4-7
Using fsck_ufs to Check Directory Unallocated.	4-7
Using fsck_ufs to Check Bad Inode Number	4-7
Using fsck_ufs to Check Incorrect “.” and “..” Entries	4-8
Using fsck_ufs to Check Disconnected Directories	4-8
Running fsck on a ufs File System	4-8
fsck_ufs Initialization Phase.	4-8
fsck_ufs Phase 1: Check Blocks and Sizes.	4-12
fsck_ufs Phase 1 Error Messages	4-13
fsck_ufs Phase 1B: Rescan for More DUP.	4-15
fsck_ufs Phase 2: Check Pathnames.	4-16
fsck_ufs Phase 2 Error Messages	4-16
fsck_ufs Phase 3: Check Connectivity	4-22
fsck_ufs Phase 3 Error Messages	4-22
fsck_ufs Phase 4: Check Reference Counts	4-24
fsck_ufs Phase 4 Error Messages	4-25
fsck_ufs Phase 5: Check Cylinder Groups.	4-27
fsck_ufs Phase 5 Error Messages	4-28
fsck_ufs Cleanup Phase	4-28
Checking sfs File Systems.	4-29
fsck_sfs Phase 1: Check Blocks and Sizes.	4-30
fsck_sfs Phase 1B: Rescan for More DUP.	4-30
fsck_sfs Phase 2: Check Pathnames.	4-30
fsck_sfs Phase 4: Check Reference Counts	4-31

What Is File System Checking?

NOTE

If the Enhanced Security Utilities are installed on your system, use only user file systems of type **sfs** to ensure the security of your system; the **ufs**, **xf**s and **xfsd** file systems do not support Mandatory Access Control restrictions.

When the operating system (OS) is brought up, a consistency check of the file systems should always be done. Often this check is done automatically as part of the power-up process. Included as part of that process is a sanity check of each file system on the hard disk. The sanity check returns a code for each file system indicating whether the consistency checking and repair program, **fsck (1M)**, should be run. Note that **xf**s and **xfsd** file systems does not use **fsck**. **xf**s and **xfsd** provides recovery only seconds after a system failure by using a tracking feature called intent logging. Refer to Chapter 3 “Managing File System Types” for additional information.

fsck should be used to check file systems not mounted routinely as part of the power-up process. If inconsistencies are discovered, corrective action must be taken before the file systems are mounted. The rest of this section is designed to acquaint you with

- the command line options of the **fsck** utility
- the type of checking it does in each of its phases
- the repairs it suggests.

NOTE

File system corruption, while serious, is not all that common. Most of the time a check of the file systems finds that everything is all right. The reason we put so much emphasis on file system checking is that if errors are allowed to go undetected, the loss can be substantial.

Using the fsck Utility to Check File Systems

The file system check (**fsck**) utility is an interactive file system check and repair program. **fsck** uses the information contained in the file system to perform consistency checks. If an inconsistency is detected, a message describing the inconsistency is displayed. At that point you may decide whether to have **fsck** ignore the inconsistency or attempt to fix it. Reasons you might choose to have **fsck** ignore an inconsistency are that you think the problem is so severe that you want to fix it yourself, or that you plan to go back to an earlier version of the file system. Whatever your decision, you should not ignore the inconsistencies reported by **fsck**. File system inconsistencies do not repair themselves. If they are ignored, they get worse. Normally, the root file system is **fscked** automatically when the system boots.

With the exception of the root file system, a file system should be unmounted while it's being checked.

Generic fsck and Its Options

The generic format of the **fsck** command follows:

```
fsck [-F FSType] [-V] [-m] [special . . . ]
fsck [-F FSType] [-V] [current_options] [-o specific_options] [special . . . ]
fsck [-F FSType] [-V] [-PLbyw] [special . . . ]
```

The **-F** is used to specify the *FSType* on which the command must act. The *FSType* must be specified on the command line or must be determinable from **/etc/vfstab** by matching an entry in that file with the *special* specified.

The **-V** option causes the command to echo the completed command line. The echoed line will include additional information derived from **/etc/vfstab**. This option can be used to verify and validate the command line. It does not cause the command to execute.

The **-m** is used to perform a sanity check only. This option is usually used before mounting file systems because it lets the administrator know whether the file system needs to be checked.

The **-o** option is used to specify *FSType*-specific options if any. *specific_options* are options specified in a comma-separated list of keywords and/or keyword-attribute pairs for interpretation by the *FSType*-specific module of the command.

Parallel File System Checking with the -P Option

A fast **fsck** is becoming increasingly desirable because of the growing need for online transaction processing and for quick file system recovery. Implementing a parallel **fsck** causes concern about maximizing the speed and efficiency of the command while avoiding collision and synchronization problems. An example of a bad collision would be the simultaneous checking of several file systems that lie on the same disk. Parallelism is implemented via a new **-P** option. The gain in speed from **fsck -P** depends on the number and size of the file systems and disks being checked, as well as on the distribution of these different file systems and file system types on the available disks.

The **-P** option automatically audits and interactively repairs inconsistent conditions for all file systems in **/etc/vfstab** in parallel. It uses the **fsckpass** field of **/etc/vfstab** to create a list of file systems that need checking on separate disks. This field indicates which separate sequential fsck pass should be used for the filesystem. Filesystems residing on the same physical volume should have different *fsckpass* values (i.e., the first filesystem on each disk should have a '1', the second a '2', etc.). The file systems are then scheduled for checking such that each disk has one file system being checked on that disk at a given time. A check on one file system does not depend on a check of another file system executing on another disk to complete.

“-” in the file system's **fsckpass** field indicates that it is not checked. All other pass numbers are at the system administrator's discretion. The **fsckpass** field is initially filled in at installation time by an architecture-dependent routine, *setvfspass*. The system administrator can tune the algorithm that *setvfspass* employs by altering the **fsckpass** if necessary.

After all the files in **/etc/vfstab** have been checked, the parent **fsck** process spawns single-threaded **fscks** interactively on each individual file system that could not be fixed through the standard procedures.

The **-P** option supports these suboptions:

- L** requests that the output be arranged in order by file system.
- b** prints the file system output in a brief terse format. When successfully completed, the command prints one line, per file system, which is a summary of **fsck** information.
- y** assumes a “yes” response to all questions, and the interactive mode is circumvented.
- w** averts confusion stemming from several different file systems requesting interactive responses. It runs as expected upon successful completion (either verbose or brief). If it fails, the parent **fsck** process collects the failure information, but defers any corrections until the rest of the file systems have been checked.

If the file system is inconsistent the user is prompted for concurrence before each correction is attempted. It should be noted that some corrective actions will result in some loss of data. The amount and severity of data loss may be determined from the diagnostic output. The default action for each correction is to wait for the user to respond yes or no. If the user does not have write permission **fsck** defaults to a no action.

In the rest of this chapter we'll see how to use **fsck** to check **ufs** and **sfs** file systems. The information is presented separately for each FSType. Although this arrangement results in a certain amount of repetition, we hope it will prevent confusion.

Checking ufs File Systems

This section describes the use of **fsck** with **ufs** file systems. It assumes you're running **fsck** interactively, and that all possible errors can be encountered. When an inconsistency

is discovered in this mode, **fsck** reports the inconsistency for you to choose a corrective action. (You can also run **fsck** automatically using the **-y**, **-n**, or **-o p** options.)

ufs File System Components Checked by fsck

Before describing the phases of **fsck** and the messages that may appear in each, we'll discuss the kinds of consistency checks applied to each component of a **ufs** file system.

Using fsck_ufs to Check Super-Blocks

The most commonly corrupted item in a file system is the summary information associated with the super-block, because it is modified with every change to the blocks or inodes of the file system, and is usually corrupted after an unclean halt. The super-block is checked for inconsistencies involving:

- file system size
- inode list size
- free block list
- free block count
- free inode count.

Using fsck_ufs to Check File System Size and Inode List Size

The file system size must be larger than the number of blocks used by the super-block and the number of blocks used by the list of inodes. While there is no way to check these sizes precisely, **fsck** can check that they are within reasonable bounds. All other file system checks require that these sizes be correct. If **fsck** detects corruption in the static parameters of the default super-block, **fsck** requests the operator to specify the location of an alternate super-block.

Using fsck_ufs to Check Free Block List

fsck checks that all the blocks marked as free in the cylinder group block maps are not claimed by any files. When all the blocks have been initially accounted for, **fsck** checks that the number of free blocks plus the number of blocks claimed by the inodes equals the total number of blocks in the file system. If anything is wrong with the block allocation maps, **fsck** will rebuild them, based on the list it has computed of allocated blocks.

Using fsck_ufs to Check Free Block Count

The summary information associated with the super-block contains a count of the total number of free blocks within the file system. **fsck** compares this count to the number of

free blocks it finds within the file system. If the two counts do not agree, then **fsck** replaces the incorrect count in the summary information by the actual free block count.

Using **fsck_ufs** to Check Free Inode Count

The summary information contains a count of the total number of free inodes within the file system. **fsck** compares this count to the number of free inodes it finds within the file system. If the two counts do not agree, then **fsck** replaces the incorrect count in the summary information by the actual free inode count.

Using **fsck_ufs** to Check Inodes

The list of inodes in the file system is checked sequentially starting with inode 2 (inode 0 marks unused inodes; inode 1 is saved for future generations) and progressing through the last inode in the file system. Each inode is checked for inconsistencies involving:

- format and type
- link count
- duplicate blocks
- bad block numbers
- inode size.

Using **fsck_ufs** to Check Format and Type

Each inode contains a mode word. This mode word describes the type and state of the inode. Inodes may be one of six types:

- regular
- directory
- block special
- character special
- FIFO (or named pipe)
- symbolic link.

Inodes may be in one of three allocation states:

- unallocated
- allocated
- neither allocated nor unallocated.

This last state means that the inode is incorrectly formatted. An inode can get into this state if bad data are written into the inode list. The only possible corrective action **fsck** can take is to clear the inode.

Using fsck_ufs to Check Link Count

Each inode counts the total number of directory entries linked to the inode. **fsck** verifies the link count of each inode by starting at the root of the file system, and descending through the directory structure. The actual link count for each inode is calculated during the descent.

Discrepancies between the link count stored in the inode and the actual link count as determined by **fsck** may be of two types:

- the stored count is not 0, the actual count is 0

This can occur if no directory entry appears for the inode. In this case, **fsck** can link the disconnected file to the **lost+found** directory.

- the stored count is not 0, the actual count is not 0, but the counts are unequal

This can occur if a directory entry has been removed but the inode has not been updated. In this case, **fsck** can replace the stored count by the actual link count.

Each inode contains a list, or pointers to lists (indirect blocks), of all the blocks claimed by the inode. Because indirect blocks are owned by an inode, inconsistencies in an indirect block directly affect the inode that owns it.

Using fsck_ufs to Check Duplicate Blocks

fsck compares each block number claimed by an inode against a list of already allocated blocks. If another inode already claims a block number, then the block number is added to a list of duplicate blocks. Otherwise, the list of allocated blocks is updated to include the block number.

If there are any duplicate blocks, **fsck** performs a partial second pass over the inode list to find the inode of the duplicated block. The second pass is needed, since without examining the files associated with these inodes for correct content, not enough information is available to determine which inode is corrupted and should be cleared. If this condition does arise, then the inode with the earliest modify time is usually incorrect, and should be cleared. If this happens, **fsck** prompts the operator to clear both inodes. The operator must decide which one should be kept and which one should be cleared.

Using fsck_ufs to Check Bad Block Numbers

fsck checks the range of each block number claimed by an inode. If the block number is lower than the first data block in the file system, or greater than the last data block, then the block number is a bad block number. Many bad blocks in an inode are usually caused by an indirect block that was not written to the file system, a condition that can occur only if there has been a hardware failure. If an inode contains bad block numbers, **fsck** prompts the operator to clear it.

Using `fsck_ufs` to Check Inode Size

Each inode contains a count of the number of data blocks that it contains. The number of actual data blocks is the sum of the allocated data blocks and the indirect blocks. `fsck` computes the actual number of data blocks and compares that block count against the actual number of blocks the inode claims. If an inode contains an incorrect count, `fsck` prompts the operator to fix it.

Each inode contains a 32-bit size field. The size is the number of data bytes in the file associated with the inode. The consistency of the byte size field is roughly checked by computing, from the size field, the maximum number of blocks that should be associated with the inode, and comparing that expected block count against the actual number of blocks the inode claims.

Data Associated with an Inode

An inode can directly or indirectly reference three kinds of data blocks. All referenced blocks must be the same kind. The three types of data blocks are: plain data blocks, symbolic link data blocks, and directory data blocks. Plain data blocks contain the information stored in a file; symbolic link data blocks contain the pathname stored in a link. Directory data blocks contain directory entries. `fsck` can only check the validity of directory data blocks.

Using `fsck_ufs` to Check Directory Data Blocks

Directory data blocks are checked for inconsistencies involving:

- directory inode numbers pointing to unallocated inodes
- directory inode numbers that are greater than the number of inodes in the file system
- incorrect directory inode numbers for “.” and “..”
- directories that are not attached to the file system.

Using `fsck_ufs` to Check Directory Unallocated

If the inode number in a directory data block references an unallocated inode, then `fsck` will remove that directory entry.

Using `fsck_ufs` to Check Bad Inode Number

If a directory entry inode number references outside the inode list, then `fsck` will remove that directory entry. This condition occurs if bad data is written into a directory data block.

Using fsck_ufs to Check Incorrect “.” and “..” Entries

The directory inode number entry for “.” must be the first entry in the directory data block. The inode number for “.” must reference itself; for example, it must equal the inode number for the directory data block. The directory inode number entry for “..” must be the second entry in the directory data block. Its value must equal the inode number for the parent of the directory entry (or the inode number of the directory data block if the directory is the root directory). If the directory inode numbers are incorrect, **fsck** will replace them with the correct values. If there are multiple hard links to a directory, the first one encountered is considered the real parent to which “..” should point; **fsck** recommends deletion for the subsequently discovered names.

Using fsck_ufs to Check Disconnected Directories

fsck checks the general connectivity of the file system. If directories are not linked into the file system, then **fsck** links the directory back into the file system in the **lost+found** directory.

Running fsck on a ufs File System

fsck is a multi-pass file system check program. Each file system pass invokes a different phase of the **fsck** program. After initialization, **fsck** performs successive passes over each file system, checking blocks and sizes, pathnames, connectivity, reference counts, and the map of free blocks (possibly rebuilding it), and performs some cleanup.

At boot time **fsck** is normally run with the **-y** option, non-interactively. (**fsck** can also be run interactively by the administrator at any time.) **fsck** can also be run non-interactively to “preen” the file systems after an unclean halt. While preening a file system, it will only fix corruptions that are expected to result from an unclean halt. These actions are a subset of the actions that **fsck** takes when it is running interactively. When an inconsistency is detected, **fsck** generates an error message. If a response is required, **fsck** prints a prompt and waits for a response. When preening most errors are fatal. For those that are expected, the response taken is noted. This section explains the meaning of each error message, the possible responses, and the related error conditions.

The error conditions are organized by the phase of the **fsck** program in which they can occur. The error conditions that may occur in more than one phase are discussed below under “*Initialization Phase.*”

fsck_ufs Initialization Phase

Before a file system check can be performed, certain tables have to be set up and certain files opened. The messages in this section relate to error conditions resulting from command line options, memory requests, the opening of files, the status of files, file system size checks, and the creation of the scratch file.

Message

```
cannot alloc NNN bytes for blockmap
cannot alloc NNN bytes for freemap
cannot alloc NNN bytes for statemap
cannot alloc NNN bytes for lncntp
```

fsck's request for memory for its virtual memory tables failed. This should never happen. When it does, **fsck** terminates. This is a serious system failure and should be handled immediately. Contact your service representative or another qualified person.

Message

```
Can't open checklist file: F
```

The file system checklist or default file *F* (usually `/etc/vfstab`) cannot be opened for reading. When this occurs, **fsck** terminates. Check the access modes of *F*.

Message

```
Can't stat root
```

fsck's request for statistics about the root directory failed. This should never happen. When it does, **fsck** terminates. Contact your service representative or another qualified person.

Message

```
Can't stat F
Can't make sense out of name F
```

fsck's request for statistics about the file system *F* failed. When **fsck** is running interactively, it ignores this file system and continues checking the next file system given. Check the access modes of *F*.

Message

```
Can't open F
```

fsck's attempt to open the file system *F* failed. When running interactively, it ignores this file system and continues checking the next file system given. Check the access modes of *F*.

Message

```
F: (NO WRITE)
```

Either the `-n` flag was specified or **fsck's** attempt to open the file system *F* for writing failed. When **fsck** is running interactively, all the diagnostics are printed out, but **fsck** does not attempt to fix anything.

Message

```
file is not a block or character device; OK
```

The user has given **fsck** the name of a regular file by mistake. Check the type of the file specified.

Possible responses to the OK prompt are:

- YES Ignore this error condition.
- NO Ignore this file system and continue checking the next file system given.

Message

```
UNDEFINED OPTIMIZATION IN SUPERBLOCK (SET TO DEFAULT)
```

The super-block optimization parameter is neither OPT_TIME nor OPT_SPACE.

Possible responses to the SET TO DEFAULT prompt are:

- YES Set the super-block to request optimization to minimize running time of the system. [If optimization to minimize disk space use is desired, it can be set using **tunefs (1M)** .]
- NO Ignore this error condition.

Message

```
IMPOSSIBLE MINFREE=D IN SUPERBLOCK (SET TO DEFAULT)
```

The super-block minimum space percentage is greater than 99 percent or less than 0 percent.

Possible responses to the SET TO DEFAULT prompt are:

- YES Set the minfree parameter to ten percent. [If some other percentage is desired, it can be set using **tunefs (1M)** .]
- NO Ignore this error condition.

Message

```
MAGIC NUMBER WRONG  
NCG OUT OF RANGE  
CPG OUT OF RANGE  
NCYL DOES NOT JIVE WITH NCG*CPG  
SIZE PREPOSTEROUSLY LARGE  
TRASHED VALUES IN SUPER BLOCK
```

followed by the message:

```
F: BAD SUPER BLOCK: B  
USE -b OPTION TO FSCK TO SPECIFY LOCATION OF AN ALTERNATE  
SUPER-BLOCK TO SUPPLY NEEDED INFORMATION; SEE fsck(1M) .
```

The super-block has been corrupted. An alternative super-block must be selected from among the available copies. Choose an alternative super-block by calculating its offset or call your service representative or another qualified person. Specifying block 32 is a good first choice.

Message

```
INTERNAL INCONSISTENCY: M
```

fsck has had an internal panic, whose message is *M*. This should never happen. If it does, contact your service representative or another qualified person.

Message

```
CAN NOT SEEK: BLK B (CONTINUE)
```

fsck's request to move to a specified block number *B* in the file system failed. This should never happen. If it does, contact your service representative or another qualified person.

Possible responses to the **CONTINUE** prompt are:

YES Attempt to continue to run the file system check. (Note that the problem will often persist.) This error condition prevents a complete check of the file system. A second run of **fsck** should be made to recheck the file system. If the block was part of the virtual memory buffer cache, **fsck** will terminate with the message:

```
Fatal I/O error
```

NO Terminate the program.

Message

```
CAN NOT READ: BLK B (CONTINUE)
```

fsck's request to read a specified block number *B* in the file system failed. This should never happen. If it does, contact your service representative or another qualified person.

Possible responses to the **CONTINUE** prompt are:

YES Attempt to continue to run the file system check. **fsck** will retry the read and print out the message:

```
THE FOLLOWING SECTORS COULD NOT BE READ: N
```

where *N* indicates the sectors that could not be read. If **fsck** ever tries to write back one of the blocks on which the read failed it will print the message:

```
WRITING ZERO'ED BLOCK N TO DISK
```

where *N* indicates the sector that was written with zero's. If the disk is experiencing hardware problems, the problem will persist. This error condition prevents a complete check of the file system. A second run of **fsck** should be made to recheck the file system.

If the block was part of the virtual memory buffer cache, **fsck** will terminate with the message:

```
Fatal I/O error
```

NO Terminate the program.

Message

```
CAN NOT WRITE: BLK B (CONTINUE)
```

fsck's request to write a specified block number *B* in the file system failed. The disk is write-protected; check the write-protect lock on the drive. If that is not the problem, contact your service representative or another qualified person.

Possible responses to the **CONTINUE** prompt are:

YES Attempt to continue to run the file system check. The write operation will be retried. Sectors that could not be written will be indicated by the message:

```
THE FOLLOWING SECTORS COULD NOT BE WRITTEN: N
```

where *N* indicates the sectors that could not be written. If the disk is experiencing hardware problems, the problem will persist. This error condition prevents a complete check of the file system. A second run of **fsck** should be made to recheck this file system. If the block was part of the virtual memory buffer cache, **fsck** will terminate with the message:

```
Fatal I/O error
```

NO Terminate the program.

Message

```
bad inode number DDD to ginode
```

An internal error was caused by an attempt to read non-existent inode *DDD*. This error causes **fsck** to exit. If this occurs, contact your service representative or another qualified person.

fsck_ufs Phase 1: Check Blocks and Sizes

This phase checks the inode list. It reports error conditions encountered while:

- checking inode types
- setting up the zero-link-count table
- examining inode block numbers for bad or duplicate blocks
- checking inode size
- checking inode format.

All the errors in this phase except **INCORRECT BLOCK COUNT** and **PARTIALLY TRUNCATED INODE** are fatal if the file system is being preened.

fsck_ufs Phase 1 Error Messages

Message

UNKNOWN FILE TYPE I=*I* (CLEAR)

The mode word of the inode *I* indicates that the inode is not a special block inode, special character inode, regular inode, symbolic link, FIFO file, or directory inode.

Possible responses to the CLEAR prompt are:

YES De-allocate inode *I* by zeroing out its contents. This will always generate the UNALLOCATED error message in Phase 2 for each directory entry pointing to this inode.

NO Ignore this error condition.

Message

PARTIALLY TRUNCATED INODE I=*I* (SALVAGE)

fsck has found inode *I* whose size is shorter than the number of blocks allocated to it. This condition should only occur if the system crashes while truncating a file. When preening the file system, **fsck** completes the truncation to the specified size.

Possible responses to the SALVAGE prompt are:

YES Complete the truncation to the size specified in the inode.

NO Ignore this error condition.

Message

LINK COUNT TABLE OVERFLOW (CONTINUE)

An internal table for **fsck** containing allocated inodes with a link count of zero has no more room.

Possible responses to the CONTINUE prompt are:

YES Continue with the program. This error condition prevents a complete check of the file system. A second run of **fsck** should be made to recheck the file system. If another allocated inode with a zero link count is found, the error message is repeated.

NO Terminate the program.

Message

B BAD I=*I*

Inode *I* contains block number *B* with a number lower than the number of the first data block in the file system or greater than the number of the last block in the file system. This error condition may generate the EXCESSIVE BAD BLKS error message in Phase 1 if inode *I* has too many block numbers outside the file system range. This error condition generates the BAD/DUP error messages in Phases 2 and 4.

Message

```
EXCESSIVE BAD BLKS I=I (CONTINUE)
```

There are too many (usually more than 10) blocks with a number lower than the number of the first data block in the file system or greater than the number of the last block in the file system associated with inode *I*.

Possible responses to the **CONTINUE** prompt are:

YES Ignore the rest of the blocks in this inode and continue checking with the next inode in the file system. This error condition prevents a complete check of the file system. A second run of **fsck** should be made to recheck this file system.

NO Terminate the program.

Message

```
BAD STATE DDD TO BLKERR
```

An internal error has scrambled **fsck's** state map to have the impossible value *DDD*. **fsck** exits immediately. If this occurs, contact your service representative or another qualified person.

Message

```
B DUP I=I
```

Inode *I* contains block number *B* that is already claimed by another inode. This error condition may generate the **EXCESSIVE DUP BLKS** error message in Phase 1 if inode *I* has too many block numbers claimed by other inodes. This error condition invokes Phase 1B and generates the **BAD/DUP** error message in Phases 2 and 4.

Message

```
BAD MODE: MAKE IT A FILE?
```

This message is generated when the status of a given inode is set to all ones, indicating file system damage. This message does not indicate disk damage, unless it appears repeatedly after **fsck -y** has been run. A response of *y* causes **fsck** to reinitialize the inode to a reasonable value.

Message

```
EXCESSIVE DUP BLKS I=I (CONTINUE)
```

There are too many (usually more than 10) blocks claimed by other inodes.

Possible responses to the **CONTINUE** prompt are:

YES Ignore the rest of the blocks in this inode and continue checking with the next inode in the file system. This error condition prevents a complete check of the file system. A second run of **fsck** should be made to recheck the file system.

NO Terminate the program.

Message

DUP TABLE OVERFLOW (CONTINUE)

An internal table in **fsck** containing duplicate block numbers has no more room.

Possible responses to the **CONTINUE** prompt are:

- YES** Continue with the program. This error condition prevents a complete check of the file system. A second run of **fsck** should be made to recheck the file system. If another duplicate block is found, this error message will repeat.
- NO** Terminate the program.

Message

PARTIALLY ALLOCATED INODE I=*I* (CLEAR)

Inode *I* is neither allocated nor unallocated.

Possible responses to the **CLEAR** prompt are:

- YES** De-allocate inode *I* by zeroing out its contents.
- NO** Ignore this error condition.

Message

INCORRECT BLOCK COUNT I=*I* (*X* should be *Y*) (CORRECT)

The block count for inode *I* is *X* blocks, but should be *Y* blocks. When preening, the count is corrected.

Possible responses to the **CORRECT** prompt are:

- YES** Replace the block count of inode *I* by *Y*.
- NO** Ignore this error condition.

fsck_ufs Phase 1B: Rescan for More DUP

When a duplicate block is found in the file system, the file system is rescanned to find the inode that previously claimed that block. When the duplicate block is found, the following informational message appears:

Message

B DUP I=*I*

Inode *I* contains block number *B* that is already claimed by another inode. This error condition generates the **BAD/DUP** error message in Phase 2. You can determine which inodes have overlapping blocks by examining this error condition and the **DUP** error condition in Phase 1.

fsck_ufs Phase 2: Check Pathnames

This phase removes directory entries pointing to bad inodes found in Phases 1 and 1B. It reports error conditions resulting from:

- incorrect root inode mode and status
- directory inode pointers out of range
- directory entries pointing to bad inodes
- directory integrity checks.

All errors in this phase are fatal if the file system is being preened, except for directories not being a multiple of the block size and extraneous hard links.

fsck_ufs Phase 2 Error Messages

Message

```
ROOT INODE UNALLOCATED (ALLOCATE)
```

The root inode (usually inode number 2) has no allocate mode bits. This should never happen.

Possible responses to the ALLOCATE prompt are:

YES Allocate inode 2 as the root inode. The files and directories usually found in the root will be recovered in Phase 3 and put into the **lost+found** directory. If the attempt to allocate the root fails, **fsck** will exit with the message:

```
CANNOT ALLOCATE ROOT INODE
```

NO Terminate the program.

Message

```
ROOT INODE NOT DIRECTORY (REALLOCATE)
```

The root inode (usually inode number 2) of the file system is not a directory inode.

Possible responses to the REALLOCATE prompt are:

YES Clear the existing contents of the root inode and reallocate it. The files and directories usually found in the root will be recovered in Phase 3 and put into the **lost+found** directory. If the attempt to allocate the root fails, **fsck** will exit with the message:

```
CANNOT ALLOCATE ROOT INODE
```

NO **fsck** will then prompt with **FIX**.

Possible responses to the **FIX** prompt are:

YES Change the type of the root inode to directory. If the root inode's data blocks are not directory blocks, many error messages will be generated.

NO Terminate the program.

Message

DUPS/BAD IN ROOT INODE (REALLOCATE)

Phase 1 or Phase 1B has found duplicate blocks or bad blocks in the root inode (usually inode number 2) of the file system.

Possible responses to the **REALLOCATE** prompt are:

YES Clear the existing contents of the root inode and reallocate it. The files and directories usually found in the root will be recovered in Phase 3 and put into the **lost+found** directory. If the attempt to allocate the root fails, **fsck** will exit with the message:

CANNOT ALLOCATE ROOT INODE

NO **fsck** will then prompt with **CONTINUE**.

Possible responses to the **CONTINUE** prompt are:

YES Ignore the **DUPS/BAD** error condition in the root inode and try to continue running the file system check. If the root inode is not correct, this may generate many other error messages.

NO Terminate the program.

Message

NAME TOO LONG F

An excessively long pathname has been found. This usually indicates loops in the file system name space. This can occur if a privileged user has made circular links to directories. These links must be removed.

Message

I OUT OF RANGE I=*I* NAME=*F* (REMOVE)

A directory entry *F* has an inode number *I* that is greater than the end of the inode list.

Possible responses to the **REMOVE** prompt are:

YES Remove the directory entry *F*.

NO Ignore this error condition.

Message

UNALLOCATED I=*I* OWNER=*O* MODE=*M* SIZE=*S* MTIME=*T* TYPE=*F*
(REMOVE)

A directory or file entry *F* points to an unallocated inode *I*. The owner *O*, mode *M*, size *S*, modify time *T*, and name *F* are printed.

Possible responses to the REMOVE prompt are:

YES Remove the directory entry *F*.

NO Ignore this error condition.

Message

```
DUP/BAD I=I OWNER=O MODE=M SIZE=S MTIME=T TYPE=F (REMOVE)
```

Phase 1 or Phase 1B has found duplicate blocks or bad blocks associated with directory or file entry *F*, inode *I*. The owner *O*, mode *M*, size *S*, modify time *T*, and directory name *F* are printed.

Possible responses to the REMOVE prompt are:

YES Remove the directory entry *F*.

NO Ignore this error condition.

Message

```
ZERO LENGTH DIRECTORY I=I OWNER=O MODE=M SIZE=S MTIME=T  
DIR=F (REMOVE)
```

A directory entry *F* has a size *S* that is zero. The owner *O*, mode *M*, size *S*, modify time *T*, and directory name *F* are printed.

Possible responses to the REMOVE prompt are:

YES Remove the directory entry *F*; this will generate the BAD/DUP error message in Phase 4.

NO Ignore this error condition.

Message

```
DIRECTORY TOO SHORT I=I OWNER=O MODE=M SIZE=S MTIME=T  
DIR=F (FIX)
```

A directory *F* has been found whose size *S* is less than the minimum size directory. The owner *O*, mode *M*, size *S*, modify time *T*, and directory name *F* are printed.

Possible responses to the FIX prompt are:

YES Increase the size of the directory to the minimum directory size.

NO Ignore this directory.

Message

```
DIRECTORY F LENGTH S NOT MULTIPLE OF B (ADJUST)
```

A directory F has been found with size S that is not a multiple of the directory block size B .

Possible responses to the ADJUST prompt are:

- YES Round up the length to the appropriate block size. When the file system is being preened, only a warning is printed and the directory is adjusted.
- NO Ignore the error condition.

Message

```
DIRECTORY CORRUPTED I= $I$  OWNER= $O$  MODE= $M$  SIZE= $S$  MTIME= $T$ 
DIR= $F$ 
(SALVAGE)
```

A directory with an inconsistent internal state has been found.

Possible responses to the SALVAGE prompt are:

- YES Throw away all entries up to the next directory boundary (usually a 512-byte boundary). This drastic action can throw away up to 42 entries, and should be taken only after other recovery efforts have failed.
- NO Skip to the next directory boundary and resume reading, but do not modify the directory.

Message

```
BAD INODE NUMBER FOR `.' I= $I$  OWNER= $O$  MODE= $M$  SIZE= $S$  DIR= $F$ 
MTIME= $T$ 
(FIX)
```

A directory I has been found whose inode number for “.” does not equal I .

Possible responses to the FIX prompt are:

- YES Change the inode number for “.” to be equal to I .
- NO Leave the inode number for “.” unchanged.

Message

```
MISSING `.' I= $I$  OWNER= $O$  MODE= $M$  SIZE= $S$  MTIME= $T$  DIR= $F$ 
(FIX)
```

A directory I has been found whose first entry is unallocated.

Possible responses to the FIX prompt are:

- YES Build an entry for “.” with inode number equal to I .
- NO Leave the directory unchanged.

Message

```
MISSING `.` I=I OWNER=O MODE=M SIZE=S MTIME=T DIR=F
CANNOT FIX, FIRST ENTRY IN DIRECTORY CONTAINS F
```

A directory *I* has been found whose first entry is *F*. **fsck** cannot resolve this problem. The file system should be mounted and entry *F* moved elsewhere. The file system should then be unmounted and **fsck** should be run again.

Message

```
MISSING `.` I=I OWNER=O MODE=M SIZE=S MTIME=T DIR=F
CANNOT FIX, INSUFFICIENT SPACE TO ADD `..`
```

A directory *I* has been found whose first entry is not “.”. This should never happen. **fsck** cannot resolve the problem. If this occurs, contact your service representative or another qualified person.

Message

```
EXTRA `.` ENTRY I=I OWNER=O MODE=M SIZE=S MTIME=T DIR=F
(FIX)
```

A directory *I* has been found that has more than one entry for “.”.

Possible responses to the FIX prompt are:

- YES Remove the extra entry for “.”.
- NO Leave the directory unchanged.

Message

```
BAD INODE NUMBER FOR `..` I=I OWNER=O MODE=M SIZE=S
MTIME=T DIR=F
(FIX)
```

A directory *I* has been found whose inode number for “.” does not equal the parent of *I*.

Possible responses to the FIX prompt are:

- YES Change the inode number for “.” to be equal to the parent of *I*.
(Note that “.” in the root inode points to itself.)
- NO Leave the inode number for “.” unchanged.

Message

```
MISSING `..` I=I OWNER=O MODE=M SIZE=S MTIME=T DIR=F
(FIX)
```

A directory *I* has been found whose second entry is unallocated.

Possible responses to the FIX prompt are:

- YES Build an entry for “.” with inode number equal to the parent of *I*.
(Note that “.” in the root inode points to itself.)

NO Leave the directory unchanged.

Message

```
MISSING '..' I=I OWNER=O MODE=M SIZE=S MTIME=T DIR=F
CANNOT FIX, SECOND ENTRY IN DIRECTORY CONTAINS F
```

A directory *I* has been found whose second entry is *F*. **fsck** cannot resolve this problem. The file system should be mounted and entry *F* moved elsewhere. The file system should then be unmounted and **fsck** should be run again.

Message

```
MISSING '..' I=I OWNER=O MODE=M SIZE=S MTIME=T DIR=F
CANNOT FIX, INSUFFICIENT SPACE TO ADD '..'
```

A directory *I* has been found whose second entry is not “..” (the parent directory). **fsck** cannot resolve this problem. The file system should be mounted and the second entry in the directory moved elsewhere. The file system should then be unmounted and **fsck** should be run again.

Message

```
EXTRA '..' ENTRY I=I OWNER=O MODE=M SIZE=S MTIME=T DIR=F
(FIX)
```

A directory *I* has been found that has more than one entry for “..” (the parent directory).

Possible responses to the **FIX** prompt are:

YES Remove the extra entry for “..” (the parent directory).

NO Leave the directory unchanged.

Message

```
N IS AN EXTRANEIOUS HARD LINK TO A DIRECTORY D (REMOVE)
```

fsck has found a hard link *N* to a directory *D*. When preening, the extraneous links are ignored.

Possible responses to the **REMOVE** prompt are:

YES Delete the extraneous entry *N*.

NO Ignore the error condition.

Message

```
BAD INODE S TO DESCEND
```

An internal error has caused an impossible state *S* to be passed to the routine that descends the file system directory structure. **fsck** exits. If this occurs, contact your service representative or another qualified person.

Message

```
BAD RETURN STATE S FROM DESCEND
```

An internal error has caused an impossible state *S* to be returned from the routine that descends the file system directory structure. **fsck** exits. If you encounter this error, contact your service representative or another qualified person.

Message

```
BAD STATE S FOR ROOT INODE
```

An internal error has caused an impossible state *S* to be assigned to the root inode. **fsck** exits. If this occurs, contact your service representative or another qualified person.

fsck_ufs Phase 3: Check Connectivity

This phase checks the directories examined in Phase 2. It reports error conditions resulting from:

- unreferenced directories and
- missing or full **lost+found** directories.

fsck_ufs Phase 3 Error Messages

Message

```
UNREF DIR I=I OWNER=O MODE=M SIZE=S MTIME=T (RECONNECT)
```

The directory inode *I* was not connected to a directory entry when the file system was traversed. The owner *O*, mode *M*, size *S*, and modify time *T* of directory inode *I* are printed. When preening, the directory is reconnected if its size is non-zero; otherwise it is cleared.

Possible responses to the **RECONNECT** prompt are:

YES Reconnect directory inode *I* to the file system in the directory for lost files (usually the **lost+found** directory). This may generate the **lost+found** error messages in Phase 3 if there are problems connecting directory inode *I* to the **lost+found** directory. It may also generate the **CONNECTED** error message in Phase 3 if the link was successful.

NO Ignore this error condition. This generates the **UNREF** error message in Phase 4.

Message

```
NO lost+found DIRECTORY (CREATE)
```

There is no **lost+found** directory in the root directory of the file system; When preening **fsck** tries to create a **lost+found** directory.

Possible responses to the **CREATE** prompt are:

YES Create a **lost+found** directory in the root of the file system. This may produce the message:

```
NO SPACE LEFT IN / (EXPAND)
```

See below for the possible responses. Inability to create a **lost+found** directory generates the message:

```
SORRY. CANNOT CREATE lost+found DIRECTORY
```

and aborts the attempt to link up the lost inode. This in turn generates the UNREF error message in Phase 4.

NO Abort the attempt to link up the lost inode. This generates the UNREF error message in Phase 4.

Message

```
lost+found IS NOT A DIRECTORY (REALLOCATE)
```

The entry for **lost+found** is not a directory.

Possible responses to the REALLOCATE prompt are:

YES Allocate a directory inode, and change **lost+found** to reference it. The previous inode referenced by the **lost+found** directory is not cleared. Thus it will either be reclaimed as an UNREF'ed inode or have its link count ADJUST'ed later in this phase. Inability to create a **lost+found** directory generates the message:

```
SORRY. CANNOT CREATE lost+found DIRECTORY
```

and aborts the attempt to link up the lost inode. This in turn generates the UNREF error message in Phase 4.

NO Abort the attempt to link up the lost inode. This generates the UNREF error message in Phase 4.

Message

```
NO SPACE LEFT IN /lost+found (EXPAND)
```

There is no space to add another entry to the **lost+found** directory in the root directory of the file system. When preening the **lost+found** directory is expanded.

Possible responses to the EXPAND prompt are:

YES Expand the **lost+found** directory to make room for the new entry. If the attempted expansion fails **fsck** prints the message:

```
SORRY. NO SPACE IN lost+found DIRECTORY
```

and aborts the attempt to link up the lost inode. This in turn generates the UNREF error message in Phase 4. Clear out unnecessary entries in the **lost+found** directory. This error is fatal if the file system is being preened.

NO Abort the attempt to link up the lost inode. This generates the **UNREF** error message in Phase 4.

Message

```
DIR I=I1 CONNECTED. PARENT WAS I=I2
```

This is an advisory message indicating that a directory inode *I1* was successfully connected to the **lost+found** directory. The parent inode *I2* of the directory inode *I1* is replaced by the inode number of the **lost+found** directory.

Message

```
DIRECTORY F LENGTH S NOT MULTIPLE OF B (ADJUST)
```

A directory *F* has been found with size *S* that is not a multiple of the directory block size *B*. (Note that this may reoccur in Phase 3 if the error condition is not corrected in Phase 2).

Possible responses to the **ADJUST** prompt are:

YES Round up the length to the appropriate block size. When preening the file system only a warning is printed and the directory is adjusted.

NO Ignore the error condition.

Message

```
BAD INODE S TO DESCEND
```

An internal error has caused an impossible state *S* to be passed to the routine that descends the file system directory structure. **fsck** exits. If this occurs, contact your service representative or another qualified person.

fsck_ufs Phase 4: Check Reference Counts

This phase checks the link count information obtained in Phases 2 and 3. It reports error conditions resulting from:

- unreferenced files
- missing or full **lost+found** directory
- incorrect link counts for files, directories, symbolic links, or special files
- unreferenced files, symbolic links, and directories
- bad or duplicate blocks in files, symbolic links, and directories.

All errors in this phase (except running out of space in the **lost+found** directory) are correctable if the file system is being preened.

fsck_ufs Phase 4 Error Messages

Message

```
UNREF FILE I=I OWNER=O MODE=M SIZE=S MTIME=T (RECONNECT)
```

Inode *I* was not connected to a directory entry when the file system was traversed. The owner *O*, mode *M*, size *S*, and modify time *T* of inode *I* are printed. When preening the file is cleared if either its size or its link count is zero; otherwise it is reconnected.

Possible responses to the RECONNECT prompt are:

- YES Reconnect inode *I* to the file system in the directory for lost files (usually the **lost+found** directory). This may generate the **lost+found** error message in Phase 4 if there are problems connecting inode *I* to the **lost+found** directory.
- NO Ignore this error condition. This will always invoke the CLEAR error condition in Phase 4.

Message

```
(CLEAR)
```

The inode mentioned in the error message immediately preceding cannot be reconnected. This message cannot appear if the file system is being preened, because lack of space to reconnect files is a fatal error.

Possible responses to the CLEAR prompt are:

- YES De-allocate the inode by zeroing out its contents.
- NO Ignore this error condition.

Message

```
NO lost+found DIRECTORY (CREATE)
```

There is no **lost+found** directory in the root directory of the file system; when preening **fsck** tries to create a **lost+found** directory.

Possible responses to the CREATE prompt are:

- YES Create a **lost+found** directory in the root of the file system. This may generate the message:

```
NO SPACE LEFT IN / (EXPAND)
```

See below for the possible responses. Inability to create a **lost+found** directory generates the message:

```
SORRY. CANNOT CREATE lost+found DIRECTORY
```

and aborts the attempt to link up the lost inode. This in turn generates the UNREF error message in Phase 4.

NO Abort the attempt to link up the lost inode. This generates the **UNREF** error message in Phase 4.

Message

```
lost+found IS NOT A DIRECTORY (REALLOCATE)
```

The entry for **lost+found** is not a directory.

Possible responses to the **REALLOCATE** prompt are:

YES Allocate a directory inode and change the **lost+found** directory to reference it. The previous inode reference by the **lost+found** directory is not cleared. Thus it will either be reclaimed as an **UNREF**'ed inode or have its link count **ADJUST**'ed later in this phase.

Inability to create a **lost+found** directory generates the message:

```
SORRY. CANNOT CREATE lost+found DIRECTORY
```

and aborts the attempt to link up the lost inode. This generates the **UNREF** error message in Phase 4.

NO Abort the attempt to link up the lost inode. This generates the **UNREF** error message in Phase 4.

Message

```
NO SPACE LEFT IN /lost+found (EXPAND)
```

There is no space to add another entry to the **lost+found** directory in the root directory of the file system. When preening the **lost+found** directory is expanded.

Possible responses to the **EXPAND** prompt are:

YES Expand the **lost+found** directory to make room for the new entry. If the attempted expansion fails **fsck** prints the message:

```
SORRY. NO SPACE IN lost+found DIRECTORY
```

and aborts the attempt to link up the lost inode. This generates the **UNREF** error message in Phase 4. Clear out unnecessary entries in the **lost+found** directory. This error is fatal if the file system is being preened.

NO Abort the attempt to link up the lost inode. This generates the **UNREF** error message in Phase 4.

Message

```
LINK COUNT TYPE I=I OWNER=O MODE=M SIZE=S MTIME=T COUNT=X  
SHOULD BE Y (ADJUST)
```

The link count for inode *I* is *X* but should be *Y*. The owner *O*, mode *M*, size *S*, and modify time *T* are printed. When preening the link count is adjusted unless the number of references is increasing, a condition that should never occur unless precipitated by a

hardware failure. When the number of references is increasing during preening, **fsck** exits with the message:

```
LINK COUNT INCREASING
```

Possible responses to the ADJUST prompt are:

YES Replace the link count of file inode *I* by *Y*.

NO Ignore this error condition.

Message

```
UNREF TYPE I=I OWNER=O MODE=M SIZE=S MTIME=T (CLEAR)
```

Inode *I* was not connected to a directory entry when the file system was traversed. The owner *O*, mode *M*, size *S*, and modify time *T* of inode *I* are printed. Since this is a file that was not connected because its size or link count was zero, it is cleared during preening.

Possible responses to the CLEAR prompt are:

YES De-allocate inode *I* by zeroing out its contents.

NO Ignore this error condition.

Message

```
BAD/DUP TYPE I=I OWNER=O MODE=M SIZE=S MTIME=T (CLEAR)
```

Phase 1 or Phase 1B has found duplicate blocks or bad blocks associated with inode *I*. The owner *O*, mode *M*, size *S*, and modify time *T* of inode *I* are printed. This message cannot appear when the file system is being preened, because it would have caused a fatal error earlier.

Possible responses to the CLEAR prompt are:

YES De-allocate inode *I* by zeroing out its contents.

NO Ignore this error condition.

fsck_ufs Phase 5: Check Cylinder Groups

This phase checks the free block and used inode maps. It reports error conditions resulting from:

- allocated blocks in the free block maps
- free blocks missing from free block maps
- incorrect total free block count
- free inodes in the used inode maps
- allocated inodes missing from used inode maps
- incorrect total used inode count

fsck_ufs Phase 5 Error Messages

Message

```
CG C: BAD MAGIC NUMBER
```

The magic number of cylinder group *C* is wrong. This usually indicates that the cylinder group maps have been destroyed. When running interactively, the cylinder group is marked as needing reconstruction. This error is fatal if the file system is being preened.

Message

```
BLK(S) MISSING IN BIT MAPS (SALVAGE)
```

A cylinder group block map is missing some free blocks. During preening the maps are reconstructed.

Possible responses to the **SALVAGE** prompt are:

YES Reconstruct the free block map.

NO Ignore this error condition.

Message

```
SUMMARY INFORMATION BAD (SALVAGE)
```

The summary information was found to be incorrect. When preening, the summary information is recomputed.

Possible responses to the **SALVAGE** prompt are:

YES Reconstruct the summary information.

NO Ignore this error condition.

Message

```
FREE BLK COUNT(S) WRONG IN SUPERBLOCK (SALVAGE)
```

The super-block free block information was found to be incorrect. When preening, the super-block free block information is recomputed.

Possible responses to the **SALVAGE** prompt are:

YES Reconstruct the super-block free block information.

NO Ignore this error condition.

fsck_ufs Cleanup Phase

After a file system has been checked, a few cleanup functions are performed. The cleanup phase displays advisory messages about the status of the file system.

Message

```
V files, W used, X free (Y frags, Z blocks)
```

This is an advisory message indicating that the file system checked contains *V* files using *W* fragment-sized blocks, and that there are *X* fragment sized blocks free in the file system. The numbers in parentheses break the free count down into *Y* free fragments and *Z* free full sized blocks.

Message

```
***** FILE SYSTEM WAS MODIFIED *****
```

This is an advisory message indicating that the file system was modified by **fsck**. If this file system is mounted or is the current root file system, you should reboot. If the file system is mounted you may need to unmount it and run **fsck** again; otherwise the work done by **fsck** may be undone by the in-core copies of tables.

Checking sfs File Systems

The **fsck** command for **sfs** file systems is a modified version of the **fsck** command for **ufs** file systems. From the operator's point of view, the two commands are virtually identical. Both have the same number of passes, and all error messages printed by the **ufs** version of **fsck** are also produced by the **sfs** command. The changes to **fsck** are in these major areas:

- Modifications to inode checking routines.
- As mentioned in the section “*The sfs File System Type*,” in the chapter “*Managing File System Types*” in this manual, two inodes are allocated for each file on an **sfs** file system. The even-numbered inodes contain standard file system information and are basically identical to **ufs** inodes. The odd-numbered inodes, or alternate inodes, contain security information. These modifications ensure that only even-numbered inodes are checked for file system consistency.
- Checks for Access Control List (ACL) information.

ACL information is stored in the alternate (odd-numbered) inodes. If a bad ACL entry is encountered, **fsck** prints a message asking you what action you wish to take. You may either do nothing or delete the file containing the bad ACL.

The **fsck** command for **sfs** file systems prints additional messages in Phase 1, Phase 1b, and Phase 2. The messages in Phase 1 and Phase 1b are informational; if errors are found in these phases, you will be prompted to take action in Phase 2.

The command silently performs additional processing in phase 4.

fsck_sfs Phase 1: Check Blocks and Sizes

In addition to the checks performed on **ufs** file systems, Phase 1 of **sfs fsck** checks the ACL consistency checks for each valid file inode. The **fsck** command does not prompt you to take action until Phase 2, when filenames are known. The following informational messages may be printed during this phase.

Message

```
B ACL BAD I=I
```

Inode *I* contains block number *B* with inconsistent ACL information. **fsck** will not prompt for any action until Phase 2.

Message

```
B ACL DUP I=I
```

Inode *I* contains block number *B*, which is a duplicate block (an ACL and file point to the same data block.) **fsck** will not prompt for any action until Phase 2.

fsck_sfs Phase 1B: Rescan for More DUP

In addition to the checks performed on **ufs** file systems, Phase 1b of **sfs fsck** reports ACL references that are the first of any duplicate block references. The following informational message may be printed.

Message

```
B ACL DUP I=I
```

Inode *I* contains block number *B* that is an ACL reference that is the first of any duplicate block references. This error condition generates the DUP/BAD/INVALID ACL error message in Phase 2. You can determine which inodes have overlapping blocks by examining this error condition and the ACL DUP error condition in Phase 1. **fsck** will not prompt for any action until Phase 2.

fsck_sfs Phase 2: Check Pathnames

In addition to the checks performed on **ufs** file systems, Phase 2 of **sfs** prompts you for the action to take for bad ACLs found during Phase 1.

Message

```
DUP/BAD/INVALID ACL I=I OWNER=O MODE=M SIZE=S MTIME=T  
TYPE=F (REMOVE)
```

Phase 1 or Phase 1B has found duplicate ACL blocks or bad ACLs associated with directory or file entry *F*, inode *I*. The owner *O*, mode *M*, size *S*, modify time *T*, and directory name *F* are printed.

Possible responses to the REMOVE prompt are:

- | | |
|-----|---|
| YES | Remove the file <i>F</i> containing the bad ACL. We recommend you remove the file and restore it from backup media, if possible. This ensures the security information for the file can be trusted. |
| NO | Ignore this error condition. (If you ignore the error condition, you need to verify that the file's security information can be trusted. One way to do this is to examine the version of the file on the backup tapes.) |

fsck_sfs Phase 4: Check Reference Counts

In addition to the actions performed for **ufs** file systems, **sfs fsck** updates the inode block map and duplicate block list when a file inode that consists of ACL blocks is cleared. There are no additional messages in this phase.

System Performance Administration

Replace with Part 2 tab for 0890430

Part 2 - System Performance Administration

Part 2 System Performance Administration

Chapter 5	Managing System Performance	5-1
Chapter 6	Managing Dynamically Loadable Modules	6-1
Chapter 7	Tunable Parameters	7-1
Chapter 8	Configuring and Building the Kernell	8-1
Chapter 9	Process Scheduling	9-1

Managing System Performance



- Introduction 5-1
- Improving and Controlling System Performance 5-1
 - Modifying the Tunable Configuration Parameters. 5-2
 - Improving and Controlling File System Usage 5-2
 - Balancing File System Space 5-2
 - Procedure. 5-2
 - When You Run out of Space. 5-3
 - Selecting a ufs or sfs Logical Block Size 5-4
 - Selecting a xfs Logical Block Size 5-4
 - Controlling Directory Size 5-5
 - Locating and Deleting Inactive Files 5-5
 - Procedure 5-5
 - Reorganizing a Single Directory 5-6
 - Before You Begin. 5-6
 - Procedure 5-6
 - Reorganizing a File System 5-7
 - Reorganizing Any File System 5-7
 - Procedure 5-7
 - Changing the Maximum File Size. 5-8
 - Procedure. 5-8
 - Controlling System Work Loads 5-8
 - Controlling User PATH Variables 5-8
 - Controlling Runaway Processes 5-9
 - Memory-Based File System 5-9
 - Monitoring System Performance. 5-10
 - df and du Usage Reports. 5-10
 - System Performance Analysis Tools 5-10
 - Kernel Profiling. 5-12
 - Loading the System Profiler. 5-13
 - Enabling and Disabling the Sampling Mechanism 5-13
 - Collecting Profiling Data 5-14
 - Formatting the Collected Data 5-14
 - System Activity Reporting 5-15
 - Automatically Collecting System Activity Data 5-16
 - Collecting System Activity Data on Demand 5-17
 - Checking File Access with sar -a 5-18
 - Checking Buffer Activity with sar -b 5-19
 - Checking System Calls with sar -c 5-20
 - Checking Disk Activity with sar -d 5-21
 - Checking Page-Out and Memory Freeing Activity with sar -g 5-22
 - Checking Kernel Memory Allocation Activity with sar -k 5-23
 - Checking Interprocess Communication with sar -m 5-24
 - Checking Page-In Activity with sar -p 5-25
 - Checking Queue Activity with sar -q 5-26
 - Checking Unused Memory with sar -r 5-27
 - Checking File System Usage with sar -t 5-28
 - Checking CPU Utilization with sar -u. 5-29

Checking System Table Status with sar -v	5-30
Checking Swapping and Switching Volume with sar -w.	5-31
Checking Terminal Activity with sar -y	5-31
Checking Overall System Performance with sar -A	5-32
Reporting Application Turnaround with timex.	5-39
Samples of Performance Management Procedures	5-40
Investigating Performance Problems	5-41
Checking for Excess Swapping	5-42
Checking for Disk Slowdowns	5-42
Checking for Modem Interrupts	5-43
Checking for Table Overflows.	5-43
Shifting the Workload to Off-Peak Hours	5-43
Dynamic vs. Static Kernels	5-43
Quick Reference Guide to Managing Performance	5-44

Managing System Performance

Introduction

The performance of a computer system is characterized by the percentage of time the system is available for use, the response time of the system, and the capacity of the system to do work.

This chapter describes ways to monitor and enhance the performance of your operating system by telling you how to find and fix performance problems, by providing examples of how to improve performance, and by listing tools that monitor system performance. These activities make up the task of performance management.

Managing a system's performance is a continual process. It can be done routinely, on a regular schedule, but there will be times when your immediate attention will be required. You may want to do some performance management when you set up your operating system.

When you set up your system for the first time, it is automatically set to a basic configuration that is usually satisfactory for most applications. This default configuration controls the chief characteristics of your operating system. This configuration may not be appropriate for the traffic on your system nor the behavior of certain applications on your system; you may need to reconfigure your system to provide the service required by your users and their applications. For a full explanation of your system's default configuration, refer to your computer installation manual; re-configuration of your system will be described later in this chapter.

While the default configuration may meet the needs, your system will perform best when it is properly tuned for your environment. Tuning not only corrects performance problems, but maximizes customer satisfaction.

Many tools are available to monitor performance and help pinpoint performance problems. These tools help you determine whether a problem is user-related or application-related, and are listed in the section "*Monitoring System Performance*."

The "*Samples of Performance Management Procedures*" section provides suggestions for actions you can take to correct a problem once you have identified it.

Improving and Controlling System Performance

The performance of your system can be degraded for many reasons that are related to user practices or applications. You can run the tools described in this chapter to characterize the

system work load. Use this information to determine the bottlenecks that are slowing your system depending on the workload. The bottlenecks can be any of the following resources: disk, CPU, memory, or networking. If you are experiencing performance problems, see the “*Monitoring System Performance*” section for information about tools that can help you determine which bottleneck is limiting the performance of your system.

Remember that while information about the workload should be gathered during prime time, most system tuning should be done during off-hours.

Modifying the Tunable Configuration Parameters

The parameters that define your system’s configuration can be altered. This procedure is referred to as tuning the system. Use the `/usr/sbin/sysdef` command to get the current parameter values in the present configuration of your system. These parameters, their values, and how to tune them are described in detail in your computer installation manual. An example of this procedure is at the end of this chapter.

Improving and Controlling File System Usage

Making files is easy under the operating system, and therefore, users tend to create numerous files using large amounts of space. On a small system (such as a desktop computer), the file systems containing the following directories should maintain, at least, the following start-of-day counts. You can use the `df` command to get this information.

<code>/tmp</code>	2000 to 4000	512-byte blocks
<code>/usr</code>	500 to 1000	512-byte blocks
<code>/home</code>	3000 to 6000	512-byte blocks
<code>/var</code>	4000 to 8000	512-byte blocks

In general, all file systems should have 10 to 15 percent of their capacity available. Below 10 percent availability, the file system fragmentation will increase and performance will be degraded. Note that when you are using the `sfs`, `ufs`, `xfs` or `xfsd` file systems, if the available file space falls below 10 percent of capacity, then you can’t use the system.

The default system configuration is set up so the file system blocks are allocated in an optimum way for most environments. See the chapter called “Managing File System Types” in this manual for more information on file system allocation.

Balancing File System Space

You can also control file system space by balancing the load between file systems. To do this, user directories often need to be moved. It is best to group users with common interests in the same file system.

Procedure

Use the following procedure to balance file system space:

1. Determine which user directories you want to move and notify the users.

NOTE

Be sure to notify users of moves well enough in advance so they can program around the expected change. Make sure they are not logged in to the system when you move their directories.

2. Use the **find** and **cpio** commands to move directories and manipulate the file system tree. Move groups of users with a single **cpio** command to avoid unlinking and duplicating linked files.

Example: Move directory trees **userx** and **usery** from file system **fs1** to **fs2** where there is more space available.

```
cd /fs1
find userx usery -print -depth | cpio -pdm /fs2
```

3. Verify that the copy was made.
4. Create new default login directories for **userx** and **usery** by running

```
/usr/sbin/usermod -d /fs2/userx userx
/usr/sbin/usermod -d /fs2/usery usery
```

5. Remove the old default login directories by running

```
rm -rf /fs1/userx /fs1/usery
```

6. Send electronic mail to **userx** and **usery** to notify them that their login directories have been moved and their pathname dependencies may need to be changed.

When You Run out of Space

The optimal solution when you run out of space is to buy an extra disk. Don't reject this possibility until you've compared the cost of a new disk with the cost of paying an administrator to police users and "create" space through some of the methods described in this section.

Also, regardless of whether you buy a new disk, you can save space on your machine through two mechanisms:

- The **compress** command shrinks files without losing data. Encourage your users to run this command on files that are rarely accessed but need to be kept on-line.
- Quotas limit the number of files that can be created by any single user. For instructions on setting quotas, see the chapter called "Managing File System Types" in this manual.

Selecting a ufs or sfs Logical Block Size

Generally, you will get the best possible performance (system throughput) from **ufs** and **sfs** file systems if the logical block size is the same as the (machine) page size. The operating system (OS) kernel uses the logical block size when reading and writing files. For example, if the logical block size of the file system is 2K, whenever I/O is done between a file and memory, 2K chunks of the file are read into or out of memory.

Large logical block size improves disk I/O performance by reducing seek time and also decreases CPU I/O overhead. On the other hand, if the logical block size is too large and you don't have fragments, then disk space is wasted. The extra space is lost because even if only a portion of a block is needed the entire block is allocated. For example, if files are stored in 1K (1024 bytes) logical blocks, then a 24-byte file wastes 1000 bytes. If the same 24-byte file is stored on a file system with a 2K (2048 bytes) logical block size, then 2024 bytes are wasted. However, if most files on the file system are very large this waste is reduced considerably.

For a file system with mostly small files, small logical block sizes (512 byte and 1K) or (preferably) small fragment sizes have the advantage of less wasted space on disk. However, CPU overhead may be increased for files larger than the block size.

The best combination is usually a block size that matches the (machine) page size and a 1K or 512-byte fragment size. The block and fragment sizes created by default should be appropriate for most applications.

The **sar -u** command, described later in the chapter, can help determine if large I/O transfers are slowing the system down.

For an **sfs** or **ufs** file system, select a 4K, or 8K block size (default). The 8K block size provides better file performance.

For an **sfs** or **ufs** file system, you can also choose a fragment size. This size can be any power of two between 512 and the block size. The number of fragments per logical block must not be larger than 8. Using fragments is not worthwhile on an **sfs** or **ufs** 2K (block size) file system because the amount of space saved is less than the 10 percent that would be reserved to prevent excessive fragmentation.

Selecting a xfs Logical Block Size

An **xfs** file system can be configured with a block size of between 512 bytes and 32Kbytes, in powers of 2. Using the largest possible block size, 32Kbytes, the potential number of seeks required for a file can be radically reduced.

To avoid potential space wastage for very small files when using a larger block size, **xfs** sub-divides its space into fragments. Files of less than the configured block size are accommodated in fragments. A fragment can be between 512 bytes and 4Kbytes, in powers of 2. The configured fragment size must be less than or equal to the block size. A fragment size of 4Kbytes is only allowed for file system extent sizes greater than 64Kbytes.

For example, for a file system with 8Kbytes blocks and 512 byte fragments, up to 16 small files (< 512 bytes) can be accommodated in a single block. A file of between 512 bytes and 1024 bytes occupies two consecutive fragments.

Note

Because of space optimization, it will not be possible to completely fill an **xf**s file system. While the file system may report a full condition, **df** will report that there is still a small percentage free.

Controlling Directory Size

Very large directories are inefficient and can therefore affect performance. If a directory becomes bigger than 10K (twenty 512-byte blocks or about 600 entries of average name length), then directory searches may be causing performance problems. For larger block sizes, bigger directories will be less of a problem, but they

should be watched carefully. The **find** command, as shown below, can ferret out such problem directories.

```
find / -type d -size +20 -print
```

NOTE

The *size* argument to the **find** command is in 512-byte blocks.

Another important thing to remember is that removing files from a directory does not make that directory smaller. (It's possible to create a file system type for which this statement is not true, but it's true for all the file system types described in this chapter: **ufs**, **sfs**, **xf**s, and **xf**sd) When a file is removed from a directory, the space vacated is made available for adding new files into the directory. This is why a directory doesn't shrink when files are deleted from it.

Note that some file system types, such as **ufs**, **sfs**, **xf**s and **xf**sd support dynamic shrinking of a directory when new files are created in it. However, as free directory data blocks are not coalesced and a directory shrinks up to the block containing the last useful file entry, the same problem (the retention of the largest-ever size) exists if the last useful file entry happens to be in one of the latter data blocks.

Locating and Deleting Inactive Files

You can reduce directory size by locating inactive files, backing them up, and then deleting them.

Procedure

Use the following procedure to locate and delete files:

1. Use the **find** command to locate inactive files.

Example:

```
find / -mtime +90 -atime +90 -print >files
```

where *files* contains the names of files neither written to nor accessed within a specified time period, here 90 days (+90).

2. Notify the user that the files will be deleted.
3. Delete the inactive files.

Reorganizing a Single Directory

Before You Begin

Before you reorganize a directory, use the “Locating and Deleting Inactive Files” procedure to remove files that are no longer useful.

Procedure

Use the following procedure to reorganize a single directory:

1. Move the current directory to another temporary directory.

Example:

```
mv /home/bob /home/obob
```

2. Create or make the new directory.

Example:

```
mkdir /home/bob
```

3. In the old directory, use the **find** and **cpio** commands to copy the files into the new directory.

Example:

```
cd /home/obob
```

```
find . -print | cpio -plm../bob
```

4. Remove the temporary directory.

Example:

```
cd ..
```

```
rm -rf obob
```


Reorganizing a File System

NOTE

If you have only one disk and you've accepted all the default values during installation, ignore this section. If you have more than one disk and you're running a heavily used file system, this section may be useful.

A file consists of multiple disk blocks, which may or may not be contiguous. Files that consist of contiguous disk blocks can be accessed more efficiently than those that aren't. A heavily used file system composed of non-contiguous disk blocks may produce performance problems. You can make your file system more efficient by rearranging the files to make the constituent blocks contiguous, which also has the effect of shrinking your directories. You can't reorganize the root file system. This is a good reason to keep your root file system small and to prevent users from creating files in it. This means you must create individual file systems for directory trees such as `/home`, `/tmp`, `/usr`, and `/var`.

Reorganizing Any File System

NOTE

The following procedure is provided as a way to clean up a severely fragmented and disorganized file system. Because this procedure is cumbersome, we don't recommend following it unless your file system is causing severe performance problems.

Procedure

Use the following procedure to reorganize any type of file system:

1. Make sure the file system you want to clean up is mounted and is not being used by any users or processes.
2. Back up the file system to a cartridge tape or any other available medium. Use the `cpio` or `tar` command.
3. Unmount the file system.
4. To create a file system identical to the original, you need to do two things: get the syntax of the `mkfs` command line used to create the original file system, and run it again. You can do both with a single command if you use the `eval` routine, as follows:

```
eval `mkfs -F file_sys_type -m device`
```

The output of the `mkfs` command with the `-m` option is the original command syntax. By "evaluating" it (with `eval`), you redirect this output such that the command is executed again. For example:

```
eval `mkfs -F ufs -m/dev/dsk/ls0`
```

5. Mount the new (empty) file system.
6. Restore the contents from the backup copy.

Changing the Maximum File Size

If you install an application such as a database program that creates very large files, you need to increase the maximum file size that the system can handle.

The maximum file size for the system is determined by the parameters `SFSZLIM` and `HFSZLIM` in the file:

- `/etc/conf/mtune.d/svc`

Information on these parameters (and others) may be found in Chapter 8 “Configuring and Building the Kernel” in this manual.

Procedure

Use the following procedure to increase the maximum file size:

1. Edit the values of the tunable variables `SFSZLIM` and `HFSZLIM`.

Make the two values identical, unless you have good reason to do so. `HFSZLIM` must not be less than `SFSZLIM`.

Example: To change the maximum file size to 1000000 bytes (10 Mb) change the values of `HFSZLIM` and `SFSZLIM` to `0xA00000` (the `0x` denotes hexadecimal).

2. Rebuild the operating system.

Controlling System Work Loads

Another step that can be performed to improve system performance is to check whether prime-time load can be reduced. You should control:

- less important jobs interfering with more important jobs
- scheduling of large jobs when the system is busy
- the efficiency of user-defined features, such as `PATH` variables.

Controlling User `PATH` Variables

User `PATH` variables are the most difficult items to control. Regular mail should be sent to users on this subject informing them of how these items can cause system problems. You should provide a default `PATH` variable for the system, made up of system directories accessible by all users, in the `/etc/profile` file.

\$PATH is a command line in the user's **.profile** file that is searched for each command execution. Before outputting the **not found** error message, the system must search every directory in **\$PATH**. These searches require both processor and disk time. If there is a disk or processor slowdown, changes here can help performance.

If you suspect performance problems are being caused by inefficient user **PATH** variables, check the following:

- Because **\$PATH** is read from left to right, the first directories specified should be those in which the most commonly used commands are most likely to be found. Make sure a directory is not searched more than once for a command.
- For security reasons, system directories should be listed first in both user **\$PATH** variables and in the default **\$PATH** in **/etc/profile**. If the current directory is included in the path, it should be listed last. This reduces the chance of executing a malicious program in place of a system program.
- In general, **\$PATH** should have the least number of required entries.
- Searches of large directories should be avoided if possible. Put any large directories at the end of **\$PATH**.
- A directory that is actually a symbolic link to another directory should not appear in **\$PATH**. For example, **/bin** should not be in **\$PATH**, but **/usr/bin** should be included.

Controlling Runaway Processes

The **ps** command is used to obtain information about active processes and light-weight processes (LWPs.). This command gives a “snapshot” of what is going on, which is useful when you are trying to identify processes and LWPs that are loading the system. The entries in which you should be interested are **TIME** (the minutes and seconds of CPU time used by processes and LWPs) and **STIME** (the time when the process or LWP first started).

When you spot a runaway process or LWP (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 or LWP should be terminated immediately via the **kill -9 shell** command. When you have a real runaway, it continues to eat up system resources until everything grinds to a halt.

When you spot processes or LWPs that take a very long time to execute you should suggest that users use the **cron** or **at** commands to execute the job during off-hours.

Memory-Based File System

The OS supports a memory-based file system (**memfs**) that may benefit certain sites. **Memfs** is a high performance full-feature file system that does no disk I/O and, therefore, is faster than disk-based file systems like **ufs**, **sfs**, **xfs** or **xfsd**. **Memfs** however, is completely volatile, that is, everything in memory is lost on a system reboot or crash.

Because of its volatility, **memfs** is best suited for temporary files. The OS utilities and tools use **/tmp** and **/var/tmp** for temporary files. Some sites may use other directories for temporary files.

To mount a **memfs** file system on **/tmp**, execute the following:

```
mount -f memfs /dev/null /tmp
```

The **memfs** file system may be added to **/etc/vfstab** as follows:

```
/dev/null      -   /tmp      -   memfs      -   yes
||             ||             ||             ||
[special device] [mount point] [as the file system type] [auto mount - yes or no]
```

[special device] [mount point] [as the file system type] [auto mount - yes or no]

For more *information on memfs*, please refer to the “*The memfs File System Type*” section in Chapter 3. Refer to the section entitled “The vfstab File System Table”, also in Chapter 3, for more details on **/etc/vfstab**.

Monitoring System Performance

The need to improve and control system usage may not be evident unless you monitor your system regularly. For example, it may not be obvious that a system is degraded. Just as a driver might not notice the difference between 48 and 50 miles per hour without the aid of a speedometer, you might not notice a 4 percent degradation without performance monitoring. This section will show you many ways to monitor your system's performance.

df and du Usage Reports

You can monitor your file system usage by executing the **df** command regularly during the day. The **df** command prints out the number of free file system blocks and inodes.

The **du** command can be executed daily after hours. The **du** command summarizes file system usage with a total for each directory being printed out. Note that if there are links between files in different directories, where the directories are on separate branches of the file system hierarchy, **du** will count the excess files more than once.

The output from these commands can be kept for later comparison. In this way, directories that rapidly increase their space usage can be spotted. However, these reports may not provide the amount of detail you need to pinpoint exact performance problems. The following section describes monitoring tools that do provide a lot of detail.

System Performance Analysis Tools

Several tools are available for collecting data about system usage and organizing it in reports so you can quickly analyze patterns of usage on your system. With this type of

analysis you can evaluate your computer's performance and develop strategies for load-balancing and system-tuning—two activities that can improve that performance.

The sections “*Kernel Profiling*” and “*System Activity Reporting*” describe each performance analysis command, its purpose, the use of its options, and examples of command usage. The output is based on processor type, disk size and other variables. Therefore, the output you receive may be different. However, the output you receive should help you identify traffic problems and determine whether they are user- or application-related.

NOTE

If you are running your system in compliance with the security criteria described in the “Security Administration” part of this manual, the commands **prfdc**, **prfld**, **prfpr**, **prfsnap**, **prfstat**, and **timex** will be available only in single-user mode. The **sar** command will be available only to an appropriate administrative login. See the “Security Administration” part of this manual for more information on administrative logins.

The performance analysis tools include the following commands. (Note that the **sadc**, **sa1**, and **sa2** commands are run by other commands or through **cron**; they're not designed to be run from a shell command line.)

prfdc	Performs the data collection function of the profiler by copying the current values of all the text address counters to a file where the data can be analyzed. It runs in the background over a period of time.
prfld	Used to initialize the recording mechanism in the system.
prfpr	Formats the data collected by prfdc or prfsnap .
prfsnap	Collects data (like prfdc) at the time of invocation only.
prfstat	Used to enable, disable, or check the status of the sampling mechanism.
sadc	Used to sample and save the system activity data. Results are saved in binary format.
sar	Calls sadc or uses files created by sadc to sample cumulative activity counters internal to the operating system and provides reports on various system-wide activities.
sa1	Shell script used to collect and store data in binary file /var/adm/sa/sadd where <i>dd</i> is the current day.
sa2	Shell script that writes a daily report in file /var/adm/sa/sardd where <i>dd</i> is the current day.
timex	When timex is used to execute another command, the elapsed time, user time, and system time spent in execution of the command are reported in seconds.

The next two sections describe the performance analysis tools in detail.

Kernel Profiling

Kernel profiling is a mechanism that allows you to determine where the operating system is spending its time during operation. It consists of commands that control the profiling process and generate reports. [See **profiler (1M)** for a description of these commands.] The system profiler samples the program counter on every clock interrupt and increments the counter corresponding to the kernel function shown by that value of the program counter.

When enabled, the system profiler initializes the sampling mechanism with the addresses of kernel functions. During initialization, any dynamically loadable kernel modules that are present in memory will be locked in place. Likewise, any modules that are subsequently loaded will be locked in place if profiling is enabled. Therefore it's important to disable profiling after your measurements are complete; if you don't, the dynamically loaded modules remain resident in memory, decreasing the amount of memory available for other purposes and making your system run more slowly than usual.

Here's a brief description of the profiling commands:

- **prfld** initializes (loads) the profiler with kernel function addresses. It is no longer required, but is still available for compatibility.
- **prfstat** turns the sampling mechanism on or off. If no arguments are specified, it reports whether profiling is loaded, enabled, or disabled. If profiling has not been initialized, you can initialize it by entering **prfstat on**. When profiling is disabled (via **prfstat off**), the profiler is de-initialized and dynamically loadable kernel modules are no longer locked in memory.
- The **prfdc** and **prfsnap** commands collect profiling data and save them in a file. If profiling isn't initialized, **prfdc** and **prfsnap** initialize it. **prfsnap** collects one set of the profiling counters and then exits. **prfdc** collects multiple sets of the profiling counters over a predetermined time interval. Because the amount of time spent in each function is determined by the difference between two sets of samples, at least two samples of the counters must be taken to print the profiling results. Multiple sets of samples are printed as the difference between each subsequent set. Hence **prfsnap** can be used repeatedly to collect samples in the same file.
- The **prfpr** command prints the profiling results from the data collected by either **prfdc** or **prfsnap**.

To operate the system profiler, run either **prfdc** or **prfsnap** to collect samples of the profiling counters, and run **prfpr** to print the profiling results. When profiling is complete, run **prfstat off** to disable the profiler and unlock the dynamically loadable modules.

For example:

```

/usr/sbin/prfsnap temp
.
<some time passes, with or without system activity>
.
prfsnap temp
prfpr temp
prfstat off

```

If you want the profiler to begin automatically when you boot the system to multi-user state, you can add the following lines to the `/etc/init.d/perf` file:

```
/usr/sbin/prfstat on
```

The `prf` shell script is executed during system initialization and the following messages are displayed:

```

profiling enabled
xxx kernel text addresses loaded

```

where `xxx` states how many kernel text addresses are in the current operating system kernel.

The following sections describe kernel profiling commands in detail.

Loading the System Profiler

The `prfld` command initializes, or loads, the system profiler mechanism. One side effect of this process (which isn't necessary but is sometimes useful) is the locking of dynamically loadable modules in place without profiling being enabled. The command has the following format:

```
/usr/sbin/prfld
```

This command generates a table, in memory, containing the starting address of each subroutine as extracted from the kernel. Note that the pathnames and starting addresses of the loaded modules are returned from the kernel and the symbol list is built by the `prfld` command. Typical modules to have loaded are:

```

/stand/unix
/etc/conf/mod.d/*

```

Enabling and Disabling the Sampling Mechanism

The `prfstat` command enables or disables the sampling mechanism. The `prfstat` command has the following format:

```
/usr/sbin/prfstat [off | on [system_namelist]]
```

Profiler overhead is less than 3 percent as calculated for 2000 text addresses. If neither of the optional parameters is supplied, the status of the profiler is displayed. If the `on` param-

eter is supplied, the sampling mechanism is turned on. The opposite happens if `off` is indicated. If the profiler has not been initialized (loaded) with `prfld`, `prfstat` will do so when given the `on` option. With the `off` option, `prfstat` de-initializes the sampling mechanism and unlocks the loadable modules.

Collecting Profiling Data

The `prfdc` command performs the data collection function of the profiler by copying the current value of all the text address counters to a file where the data can be analyzed. The `prfdc` command has the following format:

```
/usr/sbin/prfdc file [period[off_hour]]/usr/sbin/
prfdc file [period [off_hour [system_namelist]]]
```

This command stores the contents of the counters in a *file* every *period* minutes and turns off at *off_hour*. Valid values for *off_hour* are 0 through 24. Note that if you use an *off_hour* of 24, profiling never stops.

For example, the following command line copies the current value of all the text address counters into a file called `temp` every five minutes and turns off at 4:00 P.M.:

```
/usr/sbin/prfdc temp 5 16
```

The `prfsnap` command also performs data collection, but takes a snapshot of the system at the time it is called. The format of this command is as follows:

```
/usr/sbin/prfsnap [file] [system_namelist]
```

where the command appends the counter values to *file*. Both commands initialize profiling if it has not already been initialized by either `prfld` or `prfstat`.

Formatting the Collected Data

The `prfpr` command formats the contents of *file* (data that was collected by `prfdc` or `prfsnap`). The `prfpr` command has the following format:

```
/usr/sbin/prfpr [-P processor_id[,] | ALL] [-t] file
[cutoff]
```

Each text address is converted to a system function name, and the percentage of time used by that function is printed if the activity percentage is greater than the *cutoff* number you specify. The range of *cutoff* is 0 percent to 99 percent; that is the value of *cutoff* can be any decimal value greater than or equal to zero and less than, or equal to, 100. When the value is 0, all non-zero contents print. The default *cutoff* is 1 percent.

The following screen display illustrates the output of the `prfpr` command.


```

# /usr/sbin/prfpr temp 0
02/29/96 09:27:30
02/29/96 10:33:20

kgetnetname          0.01
fifo_close           0.01
fifo_getattr         0.01
fifo_rwlock          0.01
fifo_seek            0.01
nfs_access            0.01
buf_search_hashlist  0.01
bdflush              0.03
dnlc_lookup          0.01
xdr_timeval          0.01
xdr_saargs           0.01
dnlc_purge_vp        0.32
dnlc_search          0.15
nc_inslru            0.01
dow_flush_daemon     0.01
dow_flush_remove     0.01
dow_process_leaf_flush 0.01
dow_order            0.01
.
.
.
(partial screen shown)

```

These are the function calls in the kernel. For detailed information on function calls, refer to the source code, or see an experienced user.

In normal usage, the profiler is used to characterize a workload by its kernel resource usage, in which case percentage time per function is a useful statistic. Under certain circumstances it's useful to know the absolute amount of time spent executing each function. For this reason, the **prfpr** command also takes the **-t** option to report the accumulated counter difference in clock ticks instead of percentages.

On multiprocessing systems, **prfpr** accepts the **-P** option to report per-processor statistics, or **ALL** can be specified to report for all processors.

System Activity Reporting

Another performance analysis tool is system activity reporting. As various system actions occur, counters in the operating system are incremented to keep track of them. The following system activities are tracked:

- central processing unit (CPU) utilization
- buffer usage
- disk and tape input/output activity
- terminal device activity
- system call activity
- switching
- file access

- queue activity
- kernel tables
- interprocess communication
- paging
- free memory and swap space
- Kernel Memory Allocation (KMA).

System activity data can be accessed on a special request basis using the **sar** command or it can be saved automatically on a routine basis using the **sadc** command and the shell scripts **sa1** and **sa2** [see **sar (1M)**]. Generally, the demand system activity reports are used to pinpoint specific performance problems, and the automatic reports are generated as a measure to monitor system performance.

The following sections describe both methods of activity reporting in detail.

Automatically Collecting System Activity Data

The **sadc** command can automatically sample system data. The format of this command is as follows:

```
/usr/lib/sa/sadc [t n] [ofile]
```

The command samples *n* times with an interval of *t* seconds (*t* should be greater than 5 seconds) between samples. It then writes, in binary format, to the file *ofile*, or to standard output. If *t* and *n* are omitted, a default interval is used.

When the Advanced Commands Package (from the Utilities Set) is installed, a number of files should be automatically created and/or appended that will cause system activity commands to be run automatically.

The file **/etc/init.d/perf**, which is linked to **/etc/rc2.d/S21perf**, causes the **sadc** command to be invoked to mark usage from when the counters are reset to zero. The output of **sadc** is put in the file **sadd** which acts as the daily system activity record. The command entry in the **/etc/init.d/perf** file that does this is as follows:

```
su sys -c "/usr/lib/sa/sadc /var/adm/sa/sa`date +%d`"
```

Once the performance package is installed, the **cron** file **/var/spool/cron/crontabs/sys** will contain commands to cause the automatic collection of system activity data. The commands in the **cron** file are **sa1** and **sa2**. The shell script **sa1** has the following format:

```
/usr/lib/sa/sa1 [t n]
```

The arguments *t* and *n* cause records to be written *n* times at an interval of *t* seconds. If these arguments are omitted, the records are written only once. The records are written to the binary file **/var/adm/sa/sadd**, where *dd* is the current date. The **sa1** command is performed automatically by **cron** using the following two entries found in **/var/spool/cron/crontab/sys**:

```
0 * * * 0-6 /usr/lib/sa/sa1
20,40 8-17 * * 1-5 /usr/lib/sa/sa1
```

The first causes a record to be written to `/var/adm/sa/sadd` on the hour, every hour, seven days a week. The second entry causes a record to be written to `/var/adm/sa/sadd` 20 minutes and 40 minutes after each hour from 8:00 A.M. to 5:00 P.M., Monday through Friday (typically considered to be peak working hours). Thus, these two **crontab** entries cause a record to be written to `/var/adm/sa/sadd` every 20 minutes from 8:00 A.M. to 5:00 P.M., Monday through Friday, and every hour on the hour otherwise. These defaults can easily be changed to meet your daily needs.

The shell script **sa2** has the following format:

```
/usr/lib/sa/sa2 [-ubdycwaqvtmogrAR] [-s time] [-e
time] [-i sec]
```

The **sa2** command invokes the **sar** command with the arguments given and writes the ASCII output to the file `/var/adm/sa/sar dd` where dd is the current date. The report starts at `-s time`, ends at `-e time`, and is taken as close to `-i sec` intervals as possible. See the description of the **sar** command in the following section for an explanation of the remaining options.

When installed, the performance package includes the following entry in the `/var/spool/cron/crontabs/sys` file:

```
5 18 * * 1-5 /usr/lib/sa/sa2 -s 8:00 -e 18:01 -i 1200
-A
```

This causes a **sar -A** report to be generated from `/var/adm/sa/sadd`. The report covers twenty-minute intervals in the time period from 8:00 A.M. to 6:01 P.M., Monday through Friday. Note that because `/var/adm/sa/sadd` does not have data for 5:20 and 5:40 if the above **sa1** **cron** entries are used, the **sar** report will not have data for those times either.

Collecting System Activity Data on Demand

The **sar** command can be used either to gather system activity data itself or to extract what has been collected in the daily activity files created by **sa1** and **sa2**.

The **sar** command has the following formats:

```
sar [-P processor_id[, . . .] | ALL] [-ubdycwaqvtmogrAR]
[-o file] t [n]
```

```
sar [-P processor_id[, . . .] | ALL] [-ubdycwaqvtmogrAR]
[-s time][-e time] [-i sec] [-f file]
```

In the first format, **sar** samples cumulative activity counters in the operating system at intervals specified by n for a time (in seconds) specified by t (t should be 5 seconds or greater). The default value of n is 1. If the `-o` option is specified, samples are saved in *file* in binary format.

In the second format, with no sampling interval specified, **sar** extracts data from a previously recorded *file*, either the one specified by the `-f` option or, by default, the standard

daily activity file, `/var/adm/sa/sar`. The `-s` and `-e` options define the starting and ending times for the report. The starting and ending times are of the form `hh[:mm[:ss]]`. The `-i` option specifies, in seconds, the intervals to select records. If the `-i` option is not included, all intervals found in the daily activity file are reported.

For multi-processor systems you can use the `-P` option to specify collection of data for particular processors rather than for system wide activity.

The following options are available with `sar`.

- `-a` checks file access operations
- `-b` checks buffer activity
- `-c` checks system calls
- `-d` checks disk activity
- `-g` checks page-out and memory freeing
- `-k` checks kernel memory allocation
- `-m` checks interprocess communication
- `-p` checks page-in and fault activity
- `-q` checks queue activity
- `-r` checks unused memory
- `-t` checks inode activity by file system type
- `-u` checks CPU utilization
- `-v` checks system table status
- `-w` checks swapping and switching volume
- `-y` checks terminal activity
- `-A` reports overall system performance; same as entering all options
- `-P` reports system activity for specified processors only
- `-R` reports raw data values

Checking File Access with `sar -a`

The `sar -a` option reports on the use of file access operations. The following UNIX operating system routines are reported:

- `iget/s` Number of `ufs`, `sfs`, `xf`s and `xfsd` files located by inode entry per second.
- `namei/s` Number of file system path searches per second. If the kernel does not find a directory name in the directory name lookup cache, it

will perform an `iget` operation to get the directory. Hence, most `igets` are the result of directory name lookup cache misses.

dirbk/s Number of `ufs`, `sfs`, `xf`s and `xfsd` directory block reads issued per second.

%dnlc Hit rate percentage of the directory name lookup cache.

If `-R` is specified then `%dnlc` is replaced by `dnlc-hits` and `dnlc-misses`, the counts of cache hits and misses.

The following is an example of `sar -a` output. It illustrates a one-minute sampling interval.

```

PowerMAX OS domino 1.0 PowerMAX OS NightHawk 04/27/94
00:00:00      iget/s  namei/s  dirbk/s   %dnlc
01:00:00         0         0         0       98
10:00:02        18         49         19       89
10:20:01        11         43         14       24
10:40:02        40         81         43       12
Average         17         43         19   5656565656
#
    
```

The larger the values reported, the more time the UNIX kernel is spending to access files. The amount of time reflects how heavily programs and applications are using the file system(s). The `-a` option is helpful for understanding how disk-dependent an application system is. It may be useful in tuning your directory name lookup cache (`dnlc`) size.

Checking Buffer Activity with `sar -b`

The `-b` option reports on the following buffer activities.

bread/s Average number of physical block (512 bytes each) reads into the system buffers from the disk per second. (Note that the system buffers are only used for inodes, indirect blocks, and other file system internal information. File system data blocks go through the paging system.)

lread/s Average number of logical reads from system buffers per second.

%rcache Fraction of logical reads found in the system buffers (100% minus the ratio of `bread/s` to `lread/s`) .

bwrit/s Average number of physical writes from the system buffers to disk per second.

lwrit/s Average number of logical writes to system buffers per second.

%wcache Fraction of logical writes found in the system buffers (100% minus the ratio of `bwrit/s` to `lwrit/s`).

pread/s Average number of physical read requests per second.

pwrite/s Average number of physical write requests per second.

The most important entries are the cache hit ratios **%rcache** and **%wcache**, which measure the effectiveness of system buffering. If **%rcache** falls below 90, or if **%wcache** falls below 60, it may be possible to improve performance by increasing the buffer space by adjusting the tunable **BUFHWM** in **/etc/conf/mtune.d/kernel**. If the **-R** option has been specified, the **%rcache** and **%wcache** columns are not displayed.

The following is an example of **sar -b** output:

```
PowerMAX OS domino 1.0 PowerMAX OS NightHawk 04/27/94
00:00:00 bread/s lread/s %rcache bwrit/s lwrit/s %wcache pread/s pwrite/s
01:00:00 0 0 100 0 0 63 0 0
02:00:00 0 0 100 0 0 63 0 0
03:00:00 0 0 100 0 0 63 0 0
Average 0 0 100 0 0 63 0 0
#
```

This example shows the buffers are not causing any slowdowns, because all the data is within acceptable limits. *Network Administration*.

Checking System Calls with sar -c

The **-c** option reports on system calls in the following categories:

scall/s All types of system calls per second (generally about 30 per second on a busy four- to six-user system).

sread/s read system calls per second.

swrit/s write system calls per second.

fork/s fork system calls per second (about 0.5 per second on a four- to six-user system). This number increases if shell scripts are running.

lwpcr/s lwpcr system calls made per second.

exec/s exec system calls per second. (If **(exec/s) / (fork/s)** is greater than 3, look for inefficient PATH variables.)

rchar/s Characters (bytes) transferred by read system calls per second.

wchar/s Characters (bytes) transferred by write system calls per second.

Typically, reads plus writes account for about half the total system calls, although the percentage varies greatly with the activities being performed by the system.

This is an example of **sar -c** output:

```
PowerMAX OS domino 1.0 PowerMAX OS NightHawk 04/27/94
00:00:00 scall/s sread/s swrit/s fork/s lwpcr/s exec/s rchar/s wchar/s
01:00:00 4 1 0 0.00 0.00 0.00 37 2
02:00:00 4 1 0 0.00 0.00 0.00 37 2
03:00:00 4 1 0 0.00 0.00 0.00 37 2
Average 4 1 0 0.00 0.00 0.00 37 2
```

Checking Disk Activity with **sar -d**

The **sar -d** option reports the activities of disk devices.

device	Name of the disk device(s) monitored.
%busy	Percentage of time the device spent servicing a transfer request.
avque	The average number of requests outstanding during the monitored period (measured only when the queue was occupied).
r+w/s	Number of read and write transfers to the device per second.
blks/s	Number of 512-byte blocks transferred to the device per second.
await	Average time, in milliseconds, that transfer requests wait idly in the queue (measured only when the queue is occupied).
avserv	Average time, in milliseconds, for a transfer request to be completed by the device. (For disks, this time includes seek, rotational latency, and data transfer times.)

When the **-R** option has been specified, the columns **avque**, **await**, and **avserv** are not displayed. An additional column, **busy**, the total time the disk was active, is displayed. The following is an example of **sar -d** output.

PowerMAX OS domino 1.0 PowerMAX OS NightHawk 04/27/94							
Time	device	%busy	avque	r+w/s	blks/s	avwait	avserv
16:33:49	dsk-0	43	0.7	19	122	16.7	22.3
	dsk-1	9	1.0	6	41	16.3	16.8
16:36:00	dsk-0	9	0.6	4	23	14.1	24.6
	dsk-1	2	0.2	1	6	3.2	20.5
16:37:00	dsk-0	16	0.9	8	33	18.4	20.8
	dsk-1	1	1.1	1	5	27.2	24.2
Average	dsk-0	24	0.8	11	62	16.8	22.3
	dsk-1	4	0.9	2	18	15.6	17.8

Note that queue lengths and wait and service times are measured when there is something in the queue. If **%busy** is small, large queues and service times probably represent the periodic **sync** efforts by the system to ensure altered blocks are written to the disk in a timely fashion.

Checking Page-Out and Memory Freeing Activity with sar -g

The **sar -g** option reports page-out and memory freeing activities as follows:

- pgout/s** The number of times per second that the system performed page-out operations.
- ppgout/s** The number of pages that are paged out per second. (A single page-out operation may involve the paging out of multiple pages.)
- vfree/s** The number of virtual pages per second that are placed on the freelist.
- pfree/s** The number of physical pages per second that are placed on the freelist by recycling policies of the kernel. If this value is greater than 5, it may be an indication that more memory is needed. (This is the same as **rclm/s**, previously reported by option **-p**.)
- vscan/s** The number of virtual pages per second scanned by recycling policies of the kernel. scanned by the page-stealing daemon. If this value is greater than 5, the page-out daemon is spending a lot of time checking for free memory. This implies more memory may be needed.

The following is an example of **sar -g** output:

PowerMAX OS domino 1.0	PowerMAX OS NightHawk	04/27/94			
00:00:00	pgout/s	ppgout/s	vfree/s	pfree/s	vscan/s
01:00:00	0.00	0.00	0.00	0.00	0.00
02:00:00	0.00	0.00	0.00	0.00	0.00
03:00:00	0.00	0.00	0.00	0.00	0.00
Average	0.00	0.00	0.00	0.00	0.00

sar -g is a good indicator of whether more memory may be needed. The amount of CPU time used by the recycling policies of the kernel can be found using **ps -elf**. If the recycling policies have used a lot of time, and the values of **vfree/s**, **pfree/s** and **vscan/s** are high, then you probably have a memory shortage. **sar -p**, **sar -u**, **sar -r**, and **sar -w** are also good memory shortage indicators.

Checking Kernel Memory Allocation Activity with **sar -k**

The **-k** option reports on the following activities of the Kernel Memory Allocator (KMA).

The following information is displayed for each memory pool:

size	The size of buffers in the memory pool, or Ovsz for the oversize pool.
mem	The amount of memory in bytes KMA has for the pool.
alloc	The number of bytes allocated from this pool.
succ	The amount of memory requested by KMA customers and successfully allocated. This may be less than the alloc column since the buffers are predetermined sizes.
fail	The number of requests that were not satisfied (failed).

The KMA allows a kernel subsystem to allocate and free memory as needed. Rather than statically allocating the maximum amount of memory it is expected to require under peak load, the KMA divides requests for memory into many fixed sized pools and one oversize pool. The pools are sized from 16 bytes to 8K bytes in powers of 2. There are also pools with special sizes for system tables that have large numbers of entries whose sizes do not fit well in powers of 2. The oversized requests are satisfied by allocating memory from the system page allocator.

If your system is being used to write drivers or STREAMS that use KMA resources, then **sar -k** will likely prove useful. Otherwise, you will probably not need the information it provides. However, if the total KMA represents a large portion of total memory, it may be an indication that the system is not properly tuned, or that a locally written driver or STREAMS module has a memory leak. Any driver or module that uses KMA resources but does not specifically return the resources before it exits can create a memory leak. A memory leak will cause the amount of memory allocated by KMA to increase over time. Thus, if the **alloc** fields of **sar -k** increase steadily over time, then there may be a memory leak. Another indication of a memory leak is failed requests. If many requests

fail, then it is likely that a memory leak has caused KMA to be unable to reserve and allocate memory.

If it appears that a memory leak has occurred, you should check any locally written drivers or STREAMS that may have requested memory from KMA and have not returned it.

The following is an example of **sar -k** output:

```
PowerMAX OS domino 1.0 PowerMAX OS NightHawk 04/27/94
08:22:53 size      mem      alloc    succ     fail
08:40:00   16      49152    44928    28166     0
           32      81920    53888    46928     0
           64     32768    18368    15572     0
           .         .         .         .         .
           .         .         .         .         .
           .         .         .         .         .
           Ovsz      0      409600    379480     0
           Total 3641344 3822720 3494321     0
           .         .         .         .         .
           .         .         .         .         .
           .         .         .         .         .
Average   16      63195    46613    29281     0
           32      81920    56677    49350     0
           64     46811    19483    16437     0
           .         .         .         .         .
           .         .         .         .         .
           .         .         .         .         .
           Ovsz      0      490350    446530     0
           Total 4163292 4296540 3942273     0
```

Checking Interprocess Communication with **sar -m**

The **sar -m** option reports interprocess communication activities. Message and semaphore calls are reported as follows:

msg/s Number of message operations (sends and receives) per second.

sema/s Number of semaphore operations per second.

An example of **sar -m** output follows:

```

PowerMAX OS domino 1.0 PowerMAX OS NightHawk 04/27/94

14:12:04      msg      sema
14:13:04      0.00    0.00
14:14:04      0.00    0.00
14:15:04      0.00    0.00

Average      0.00    0.00

```

These figures will usually be zero (0.00) unless you are running applications that use messages or semaphores.

Checking Page-In Activity with `sar -p`

The `sar -p` option reports paging-in activity, which includes protection and validity faults.

NOTE

This option has changed significantly from past releases because of the adoption of Virtual Memory.

atch/s	The number of page faults per second that are satisfied by reclaiming a page currently in memory (attaches per second). Instances of this include reclaiming an invalid page from the freelist and sharing a page of text currently being used by another process (for example, two or more processes accessing the code for data).
atfree/s	The number of page faults per second that are satisfied by a page on the free list.
atmiss/s	The number of page faults per second not fulfilled by a page in memory.
pgin/s	The number of times per second that file systems receive page-in requests. (This encompasses and replaces the old <code>sar -p</code> report of pgfil/s , which previously reported the number of validity faults per second satisfied by a page-in from the file system.)
ppgin/s	This field reports the number of pages paged in per second. (A single page-in request, such as a software lock request described below under slock/s or a large block size, may involve the paging in of multiple pages.)
pflt/s	The number of page faults from protection errors per second. Instances of protection faults are illegal access to a page and "copy-on-writes." Generally, this number consists primarily of copy-on-writes. (This field is carried over from the old <code>-p</code> option.)

vflt/s	The number of address translation page faults per second. These are known as validity faults and occur when a valid page is not present in memory. (This field is carried over from the old -p option.)
slock/s	This field reports the number of faults per second caused by software lock (or softlock) requests requiring physical I/O. An example of the occurrence of a softlock request is the transfer of data from a disk to memory. To ensure that the page which is to receive the data is not claimed and used by another process, it is locked by the system software.

The following is an example of **sar -p** output:

```

PowerMAX OS domino 1.0 PowerMAX OS NightHawk    04/27/94
08:22:53  atch/s atfree/s atmiss/s  pgin/s  ppgin/s  pflt/s vflt/s slock/s
08:40:00   11.32   2.34   3.50   3.17     3.17   5.83  10.14   0.00
09:00:01   25.10  12.92   2.89   8.13    14.01   3.38  12.95   1.71
09:20:00   22.24  21.13  14.54  13.45    24.17   0.75  19.89   0.00
Average    10.16    6.45    3.17    4.22     7.14   1.83   7.19   0.22
    
```

If **vflt/s** becomes much higher than 50, then you should look at **sar -g** to determine if there is a memory shortage or if the inode freelist is page bound. (See **sar -t** for more details). In addition, **sar -u**, **sar -w**, and **sar -r** can help verify whether memory is a bottleneck.

Checking Queue Activity with **sar -q**

The **sar -q** option reports the average queue length while the queue is occupied, and the percentage of time the queue is occupied.

prunq	The size of processor private queue of process in memory and run-able.
%prunocc	The percentage of time processor private run queue is occupied.
runq	The number of processes waiting, in memory, to run. Typically, this should be less than 3 for a uniprocessor system. It may be somewhat higher on a multiprocessing system. Consistently higher values mean you are CPU-bound.
%runocc	The percentage of time the run queue is occupied.
swpq	The average number of processes in the swap queue when there were processes in the queue. If there were no processes in the swap queue, this field is blank.
%swpocc	The percentage of time during the sample that there were processes in the swap queue. If there were no processes in the swap queue, this field is blank.

If the **-P** option has not been specified, then the **prunq** and **%prunocc** columns will be blank.

The following is an example of **sar -q** output:

PowerMAX OS domino 1.0		PowerMAX OS NightHawk		04/27/94		
Time	prunq	%prunocc	runq	%runocc	swpq	%swpocc
08:22:53						
08:40:00	1.0	0	1.5	100		
09:00:01	1.0	0	1.5	100		
09:20:00	1.0	0	1.5	100		
11:00:00	1.0	0	1.6	100		
Average	1.0	0	1.5	100		

If **%runocc** is greater than 90 percent and **runq-sz** is greater than 3, the CPU may be heavily loaded and response may be degraded. (See **sar -u**.) In this case, additional CPU capacity may be required to obtain acceptable system response. If **sar -p** shows a large number of validity faults and **sar -g** shows high page-out activity, then more memory may be required.

swpq-sz and **%swpocc** are blank unless there are processes swapped out that are now ready to run. A **swpq-sz** greater than 2 usually indicates a memory shortage. **sar -w** reports the number of processes swapped in and out. The **swpq-sz** reports how many of those were ready to run as opposed to those waiting for an event.

Checking Unused Memory with **sar -r**

The **-r** option records the number of memory pages and swap file disk blocks that are currently unused.

freemem Average number of pages of memory available to user processes over the intervals sampled by the command.

freeswap Number of 512-byte disk blocks available for page swapping.

The following is an example of **sar -r** output:

```
PowerMAX OS domino 1.0 PowerMAX OS NightHawk 04/27/94
14:12:04 freemem freeswp
14:13:04      726    5214
14:14:04      734    5247
14:15:04      734    5020
Average      731    5160
```

Checking File System Usage with `sar -t`

The `-t` option reports the inode usage by file system type. This report indicates when inode limits need to be changed. The `-t` is a companion to the `-v` option, which reports the sum of all inodes in the system, as well as other system table usage.

fstype File system type (**ufs**, **sfs**, **xfs** and **xfsd** combined).

inodes inuse

The current number of inode “table” entries being used by processes.

alloc The current number of existing inode “table” entries. (This number is the sum of the number of inodes in use and the number of free inodes.)

limit The maximum limit of inodes that can be allocated. (This number is a soft upper limit and varies by filesystem. For the **ufs**, **sfs** and **xfs** file systems, the number is dynamically allocated and the maximum may not be exceeded. **alloc** may slightly exceed **limit**.)

fail The number of failures that occur when you're trying to give an inode to a process between sampling points. (Failures can occur when the limit is exceeded or when memory for inodes is unavailable.)

%ipf The percentage of inodes with which reusable pages were associated that were taken off the freelist by **iget**. These pages are flushed and cannot be reclaimed by processes. Thus, this is the percentage of **igets** with page flushes. If this value is greater than 20 percent, then the freelist of inodes is considered page-bound and the number of inodes may need to be increased.

If the `-R` option been specified, the **%ipf** column is not displayed. Instead, the columns **ipage** and **inopage**, equal to the counts of inodes with and without reusable pages respectively, are displayed.

The following is an example of `sar -t` output for a system using the **sfs/ufs** file system:

```

PowerMAX OS domino 1.0 PowerMAX OS NightHawk 04/27/94
00:00:00 file system inodes inuse alloc limit fail %ipf
01:00:00 sfs/ufs 703 706 4000 0 100
other 0 0 0 0 100

02:00:00 sfs/ufs 703 706 4000 0 100
other 0 0 0 0 100

03:00:00 sfs/ufs 703 706 4000 0 100
other 0 0 0 0 100

04:00:00 sfs/ufs 705 706 4000 0 100
other 0 0 0 0 100

Average sfs/ufs 703 706 4000 0 100
other 0 0 0 0 100

```

Because many file systems allocate inodes dynamically, the number of inodes **allocated** will often be less than the **limit**. If the **alloc** number is consistently at the **limit** and the value of **%ipf** is higher than 20, then raising the limit may enhance system performance. As long as memory is available (see the **-p**, **-g**, **-w**, **-q**, and **-u** options), increasing inode limits will help keep inodes and pages in the system so they can be used again.

Checking CPU Utilization with **sar -u**

The CPU utilization is listed by **sar -u**. At any given moment the processor is either busy or idle. When busy, the processor is in either user or system mode. When idle, the processor is either waiting for input/output completion or “sitting still” with no work to do. **sar -u** lists the percentage of time that the processor is in system mode (**%sys**), in user mode (**%user**), waiting for input/output completion (**%wio**), and idle (**%idle**).

In typical timesharing use, **%sys** is usually a little higher than **%usr**. If **%sys** is more than double **%usr**, this may indicate tuning problems or that the system is being asked to do an unusually large amount of work. **ps** and kernel profiling can help identify the problems. In special applications, either of these may be larger than the other without anything being abnormal. A high **%wio** generally means a disk slowdown has occurred. A high **%idle**, with degraded response time, may mean memory constraints are present; time spent waiting for memory is attributed to **%idle**.

The following is an example of **sar -u** output:

```
PowerMAX OS domino 1.0 PowerMAX OS NightHawk 04/27/9
14:12:04 %usr %sys %wio %idle
14:13:04 62 38 0 0
14:14:04 50 41 4 4
14:15:04 60 20 5 15
Average 58 33 3 6
```

Checking System Table Status with sar -v

The **-v** option reports the status of the process, lightweight processes, inode, file, and shared memory record table. From this report you know when the system tables need to be modified.

- proc-sz** Number of process table entries currently being used/allocated in the kernel.
- inod-sz** Number of inode table entries currently being used/allocated in the kernel. (See the **sar -t inuse** and **limit** fields for a breakdown by file system type.)
- file-sz** Number of file table entries currently being used in the kernel. The **sz** is given as 0 because space is allocated dynamically for the file table.
- fail** Number of times a table has overflowed or memory for a resource was unavailable (reported for the three tables listed above). See the **sar -t fail** field for a breakdown by file system type.
- lock-sz** Number of shared memory record table entries currently being used/allocated in the kernel. The **sz** is given as 0 because space is allocated dynamically for the shared memory record table.
- lwp-sz** Number of table entries for lightweight processes.

The following is an example of **sar -v** output:

```
PowerMAX OS domino 1.0 PowerMAX OS NightHawk 04/27/94
08:22:53 proc-sz fail lpw-sz fail inod-sz fail file-sz fail lock-sz
08:40:00 38/400 0 0/54 0 960/4000 0 184/ 0 10/
09:00:01 50/400 0 0/66 0 349/4000 0 201/ 0 10/
09:20:00 46/400 0 0/62 0 381/4000 0 195/ 0 9/
Average 44/400 0 0/60 0 428/4000 0 196/ 0 9/
```

This example shows that all tables are large enough to have no overflows. If the values in these tables never exceed those shown here, you can reduce the size of the tables as a way

of saving space in main memory. The *inod-sz* values are the total of **ufs**, **sfs**, **xf**s and **xfsd** inodes and their maximums. **sar -t** breaks down the inode usage by file system type.

Checking Swapping and Switching Volume with **sar -w**

The **-w** option reports swapping and switching activity. The following are some target values and observations.

swpin/s	Number of transfers into memory (swap ins) per second.
pswin/s	Number of pages transferred for swap-ins (including initial loading of some programs) per second.
swpot/s	Number of transfers from memory to the disk swap area (swap outs) per second. If greater than 1, you may need to increase memory or decrease the amount of buffer space.
pswot/s	Number of pages transferred for swap-outs per second.
vpswout/s	Number of virtual pages transferred for swap-outs per second.
pswch/s	Process switches per second. This should be 60 to 90 on a busy four- to six-user system.

The following is an example of **sar -w** output:

```

PowerMAX OS domino 1.0 PowerMAX OS NightHawk    04/27/94
08:22:53      swpin/s   pswin/s   swpot/s   pswot/s   vpswout/s   pswch/s
08:40:00         0.00      0.0      0.00      0.0      0.0         13
09:00:01         0.00      0.0      0.00      0.0      0.0         46
09:20:00         0.00      0.0      0.00      0.0      0.0         97
Average         0.00      0.0      0.00      0.0      0.0         33
    
```

This example shows that because no swapping is occurring, there is no severe memory shortage for the currently active users.

Checking Terminal Activity with **sar -y**

The **-y** option monitors asynchronous terminal device activities. If you have a lot of terminal I/O, you can use this report to determine if there are any bad lines. The activities recorded are defined as follows:

rawch/s	Input characters (raw queue) per second.
canch/s	Input characters processed by canon (canonical queue) per second.
outch/s	Output characters (output queue) per second.

- rcvin/s** Receiver hardware interrupts per second.
- xmtin/s** Transmitter hardware interrupts per second.
- mdmin/s** Modem interrupts per second.

The number of modem interrupts per second (**mdmin/s**) should be close to 0, and the receive and transmit interrupts per second (**xmtin/s** and **rcvin/s**) should be less than or equal to the number of incoming or outgoing characters, respectively. If this is not the case, check for bad lines.

An example of **sar -y** output follows:

```
PowerMAX OS domino 1.0 PowerMAX OS NightHawk 04/27/94
14:12:04 rawch/s canch/s outch/s rcvin/s xmtin/s mdmin/s
14:13:04 3 2 57 0 0 0
14:14:04 2 2 66 0 0 0
14:15:04 2 2 89 0 0 0
Average 2 2 71 0 0 0
```

Checking Overall System Performance with **sar -A**

The **-A** option provides a view of overall system performance. Use it to get a more global perspective. If data from more than one time slice is shown, the report includes averages.

An example of **sar -A** output follows:

```

PowerMAX OS domino 1.0 PowerMAX OS NightHawk 04/27/94

00:00:00 %usr %sys %wio %idle
01:00:00 0 1 0 98
02:00:00 0 2 0 97
03:00:00 0 1 0 98
. . . .
. . . .
13:00:00 2 8 0 88
Average 0 2 1 95

13:04:02 Unix restarts
13:20:00 2 7 2 87
13:40:01 1 6 0 92
14:00:01 0 3 2 93
Average 1 5 1 91

11:42:32 Unix restarts

00:00:00 bread/s lread/s %rcache bwrit/s lwrit/s %wcache pread/s pwrite/s
01:00:00 0 0 100 0 0 63 0 0
02:00:00 0 0 100 0 0 63 0 0
03:00:00 0 0 100 0 0 63 0 0
. . . .
. . . .
13:00:00 2 8 0 88
13:00:00 0 1 98 0 1 78 0 0
Average 0 2 1 95
Average 0 4 98 0 1 87 0 0

13:04:02 Unix restarts
13:20:00 0 3 94 0 2 77 0 0
13:40:01 0 0 100 0 0 82 0 0
14:00:01 0 2 100 0 1 77 0 0
Average 0 2 97 0 1 78 0 0

11:42:32 Unix restarts

00:00:00 device %busy avque r+w/s blks/s await avserv

13:00:00 2 8 0 88
Average 0 2 1 95

13:04:02 Unix restarts

11:42:32 Unix restarts

```

00:00:00	rawch/s	canch/s	outch/s	rcvin/s	xmtin/s	mdmin/s			
01:00:00	0	0	0	0	0	0			
02:00:00	0	0	0	0	0	0			
03:00:00	0	0	0	0	0	0			
.			
13:00:00	2	8	0	88					
13:00:00	1	0	0	0	0	0			1
Average	0	2	1	95					
Average	1	0	0	0	0	0			4
13:04:02	Unix restarts								
13:20:00	2	0	0	0	0	0			3
13:40:01	0	0	0	0	0	0			0
14:00:01	1	0	0	0	0	0			2
Average	1	0	0	0	0	0			2
11:42:32	Unix restarts								
00:00:00	scall/s	sread/s	swrit/s	fork/s	lwpcr/s	exec/s	rchar/s	wchar/s	
01:00:00	4	1	0	0.00	0.00	0.00	37	2	
02:00:00	4	1	0	0.00	0.00	0.00	37	2	
03:00:00	4	1	0	0.00	0.00	0.00	37	2	
.	
13:00:00	2	8	0	88					
13:00:00	95	22	6	0.17	0.00	0.16	72912	34398	
Average	0	2	1	95					
Average	34	6	4	0.04	0.00	0.04	12279	7802	
13:04:02	Unix restarts								
13:20:00	133	21	5	0.37	0.00	0.33	48220	15602	
13:40:01	82	13	10	0.05	0.00	0.05	28858	16328	
14:00:01	169	71	8	0.08	0.00	0.09	17299	3897	
Average	128	37	8	0.14	0.00	0.14	29267	11464	
11:42:32	Unix restarts								
00:00:00	swpin/s	pswin/s	swpot/s	pswot/s	vpswout/s	pswch/s			
01:00:00	0.00	0.0	0.00	0.0	0.0	6			
02:00:00	0.00	0.0	0.00	0.0	0.0	6			
03:00:00	0.00	0.0	0.00	0.0	0.0	6			
.			
13:00:00	2	8	0	88					
13:00:00	0.00	0.0	0.00	0.0	0.0	128			
13:00:00	21.60	15.99	16.86	13.92	21.81	3.53	23.62	0.00	
13:00:00	0.00	0.00	0.00	0.00	0.00	0.00			
Average	0	2	1	95					
Average	0.00	0.0	0.00	0.0	0.0	21			
Average	8.44	2.91	1.87	2.46	3.22	0.87	4.32	0.80	
Average	0.02	0.02	0.20	0.07	0.39				

```

13:04:02 Unix restarts
13:20:00      0.00      0.0      0.00      0.0      0.0      101
13:40:01      0.00      0.0      0.00      0.0      0.0      98
14:00:01      0.00      0.0      0.01      0.5      1.0      22
Average      0.00      0.0      0.00      0.2      0.4      70

11:42:32 Unix restarts
00:00:00      iget/s  namei/s  dirbk/s  %dnlc
01:00:00          0          0          0         98
02:00:00          0          0          0         98
03:00:00          0          0          0         98
.                .          .          .          .
.                .          .          .          .
13:00:00          2          8          0         88
13:00:00          0          26         0         94
Average          0          2          1         95
Average          1          7          2         77

13:04:02 Unix restarts
13:20:00          1          29          2         93
13:40:01          0          17          0         92
14:00:01          0          3           0         92
Average          0          14         0         92

11:42:32 Unix restarts
00:00:00      prunq %prunocc      runq %runocc      swpq %swpocc
01:00:00          .                2.1      100
02:00:00          .                2.1      100
03:00:00          .                2.2      100
.                .                .
.                .                .
13:00:00          1.0          0          2.2      100
13:00:00          2          8          0          88
Average          1.1          0          2.1      100
Average          0          2          1          95

13:04:02 Unix restarts
13:20:00          1.1          0          1.4      100
13:40:01          1.0          0          1.4      100
14:00:01          1.0          0          1.8      100      1.5      2
Average          1.0          0          1.6      100      1.5      0

11:42:32 Unix restarts
00:00:00 proc-sz  fail lpw-sz  fail  inod-sz  fail  file-sz  fail  lock-sz
01:00:00  44/400  0  0/61  0  703/4000  0  197/  0  10/
02:00:00  44/400  0  0/61  0  703/4000  0  197/  0  10/
03:00:00  44/400  0  0/61  0  703/4000  0  197/  0  10/
.                .  .  .  .  .  .  .  .  .

```

13:00:00	2	8	0	88				
13:00:00	52/400	0	0/73	0	1036/4000	0	195/	0 10/
Average	0	2	1	95				
Average	45/400	0	0/63	0	1140/4000	0	198/	0 10/
13:04:02	Unix restarts							
13:20:00	40/400	0	0/61	0	533/4000	0	187/	0 9/
13:40:01	43/400	0	0/60	0	533/4000	0	196/	0 10/
14:00:01	45/400	0	0/62	0	343/4000	0	197/	0 10/
Average	42/400	0	0/61	0	469/4000	0	193/	0 9/
11:42:32	Unix restarts							
00:00:00	file system		inodes	inuse	alloc	limit	fail	%ipf
01:00:00	sfs/ufs		703		706	4000	0	100
	other		0		0	0	0	100
02:00:00	sfs/ufs		703		706	4000	0	100
	other		0		0	0	0	100
03:00:00	sfs/ufs		703		706	4000	0	100
	other		0		0	0	0	100
.
.
13:00:00	2	8	0	88				
13:00:00	sfs/ufs		1036		1036	4000	0	72
	other		0		0	0	0	100
Average	0	2	1	95				
Average	sfs/ufs		1140		1179	4000	0	3
	other		0		0	0	0	100
13:04:02	Unix restarts							
13:20:00	sfs/ufs		533		533	4000	0	1
	other		0		0	0	0	100
13:40:01	sfs/ufs		533		533	4000	0	0
	other		0		0	0	0	100
14:00:01	sfs/ufs		343		527	4000	0	38
	other		0		0	0	0	100
Average	sfs/ufs		469		531	4000	0	10
	other		0		0	0	0	100
11:42:32	Unix restarts							
00:00:00	msg	sema						

```

01:00:00 0.00 0.00
02:00:00 0.00 0.00
03:00:00 0.00 0.00
.
.
.
13:00:00 2 8 0 88
13:00:00 0.00 0.00
Average 0 2 1 95
Average 0.00 0.00

13:04:02 Unix restarts
13:20:00 0.00 0.00
13:40:01 0.00 0.00
14:00:01 0.00 0.00
Average 0.00 0.00

11:42:32 Unix restarts

00:00:00 atch/s atfree/s atmiss/s pgin/s ppgin/s pflt/s vflt/s slock/s
01:00:00 0.35 0.06 0.02 0.00 0.00 0.15 0.26 0.00
02:00:00 0.35 0.07 0.02 0.00 0.00 0.15 0.26 0.00
03:00:00 0.35 0.07 0.02 0.00 0.00 0.15 0.25 0.00
.
.
.
13:00:00 2 8 0 88
13:00:00 21.60 15.99 16.86 13.92 21.81 3.53 23.62 0.00
13:00:00 0.00 0.00 0.00 0.00 0.00 0.00
Average 0 2 1 95
Average 8.44 2.91 1.87 2.46 3.22 0.87 4.32 0.80
Average 0.02 0.02 0.20 0.07 0.39

13:04:02 Unix restarts
13:20:00 22.01 9.66 13.66 11.49 16.37 7.62 21.93 0.00
13:40:01 8.02 6.10 9.81 7.43 9.22 1.31 12.81 0.00
14:00:01 8.23 4.75 2.92 2.59 3.20 2.03 35.78 0.00
Average 11.54 6.47 8.16 6.60 8.71 3.14 23.71 0.00

11:42:32 Unix restarts

00:00:00 pgout/s ppgout/s vfree/s pfree/s vscan/s
01:00:00 0.00 0.00 0.00 0.00 0.00
02:00:00 0.00 0.00 0.00 0.00 0.00
03:00:00 0.00 0.00 0.00 0.00 0.00
.
.
.
13:00:00 2 8 0 88
13:00:00 21.60 15.99 16.86 13.92 21.81 3.53 23.62 0.00
13:00:00 0.00 0.00 0.00 0.00 0.00
Average 0 2 1 95
Average 8.44 2.91 1.87 2.46 3.22 0.87 4.32 0.80
Average 0.02 0.02 0.20 0.07 0.39

```


	Total	4222976	4174944	3720358	0
Average	16	87381	73301	49149	0
	32	87381	54325	47636	0
	64	65536	46571	35412	0
	128	354987	286635	268393	0
	256	147456	126464	97482	0
	512	32768	4096	3652	0
	1024	65536	34816	25222	0
	2048	136533	83968	69484	0
	4096	1540096	1520981	1516613	0
	8192	682667	617131	367990	0
	176	107861	91931	85237	0
	336	561152	543424	534701	0
	80	92843	80080	68399	0
	2560	136533	111787	105066	0
	Ovsz	0	479232	429112	0
	Total	4098731	4154741	3703551	0
11:42:32 Unix restarts					
	13:00:00	2	8	0	88
	Average	0	2	1	95

Reporting Application Turnaround with `timex`

The `timex` command records the amount of time taken by a command to execute, and reports the system activities that occurred during the time the command was executing. If no other programs are running, then `timex` can give you a good idea of which resources a specific command uses during its execution. A record of system consumption can be collected for each application program and then used for tuning heavily loaded resources. In the following example, the `date` command is used.

While `date`, for its simplicity, was used for the preceding demonstration, it's not representative of many commands because most commands use more system resources.

`timex` can be used in the following way:

```
timex -s application_program
```

Your application program will operate normally. When you finish running your application and exit, the `timex` result will be printed on your screen. This can be extremely interesting; you get a precise record of the system resources used while your program was executing.

If accounting is installed and running, then `timex` can present the information it collects as well. The following options can be used with the existence of accounting:

```
timex [-p [-fhkmrt]] [-o] application_program
```

See `timex(1)` for more information.

```

$ timex -s date
PowerMAX OS domino 1.0 PowerMAX OS NightHawk 04/27/94

15:21:17 %usr %sys %wio %idle
15:21:17 15 46 0 38

15:21:17 bread/s lread/s %rcache bwrit/s lwrit/s %wcache pread/s pwrite/s
15:21:17 0 19 100 0 19 100 0 0

15:21:17 device %busy avque r+w/s blks/s avwait avserv

15:21:17 rawch/s canch/s outch/s rcvin/s xmtin/s madmin/s
15:21:17 19 0 0 0 0 19

15:21:17 scall/s sread/s swrit/s fork/s lwpcr/s exec/s rchar/s wchar/s
15:21:17 2227 310 27 25.71 0.00 25.71 26377 28696

15:21:17 swpin/s pswin/s swpot/s pswot/s vpswout/s pswch/s
15:21:17 0.00 0.0 0.00 0.0 0.0 89

15:21:17 iget/s namei/s dirbk/s %dnlc
15:21:17 17 129 17 100

15:21:17 prunq %prunocc runq %runocc swpq %swpocc
15:21:17

15:21:17 proc-sz fail lpw-sz fail inod-sz fail file-sz fail lock-sz
15:21:17 50/400 0 0/67 0 363/4000 0 203/ 0 11/

15:21:17 file system inodes inuse alloc limit fail %ipf
15:21:17 sfs/ufs 363 412 4000 0 100
other 0 0 0 0 100

15:21:17 msg sema
15:21:17 0.00 0.00

15:21:17 atch/s atfree/s atmiss/s pgin/s ppgin/s pflt/s vflt/s slock/s
15:21:17 428.57 68.57 95.71 0.00 0.00 274.29 362.86 0.00

15:21:17 pgout/s ppgout/s vfree/s pfree/s vscan/s
15:21:17 0.00 0.00 0.00 0.00 0.00

15:21:17 freemem freeswap
Arithmetic Exception - core dumped
Wed Apr 27 15:21:17 EDT 1994

$

```

Samples of Performance Management Procedures

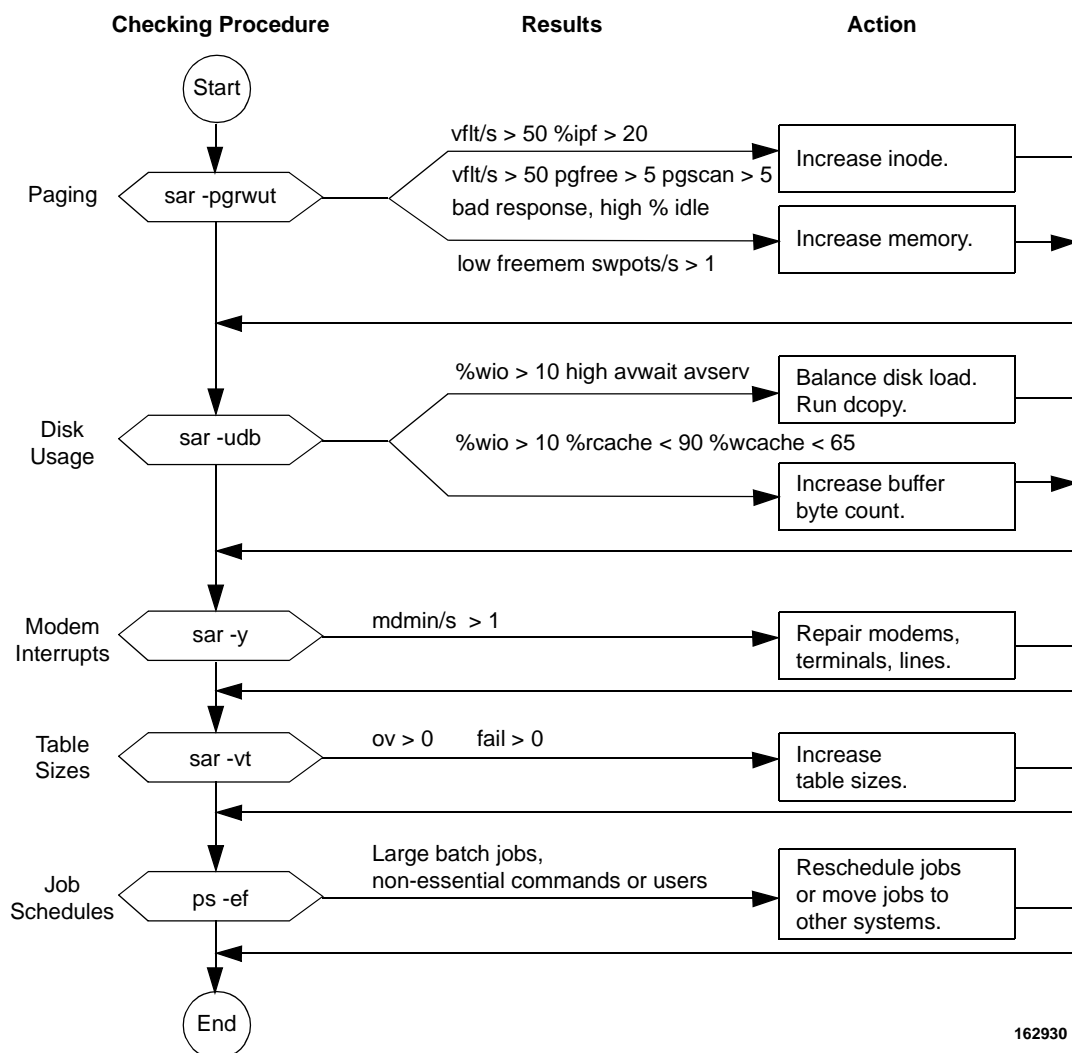
This section describes typical approaches to performance management. First, it describes a general procedure for troubleshooting performance problems. Then it provides a procedure for reconfiguring the system and shows a sample of a typical system reconfiguration. Finally, it provides a procedure for recovering after an unsuccessful attempt at reconfiguring the system.

In this section, references are made to tunable configuration parameters. For the default values of these parameters and complete instructions for altering them, see Chapter 8 “Configuring and Building the Kernel” in this manual.

Investigating Performance Problems

Locating the source of a problem can require some careful detective work. Hence, the following is not a canned procedure, but a sample approach. It covers basic areas where problems usually surface, and suggests some actions that can alleviate problems. The most common indication that a problem exists is consistently poor response time. If you have identified a familiar problem area and you need to make changes to your system parameters, see the “Tunable Parameters” and “Configuring and Building the Kernel” chapters in this manual.

The outline in Figure 5-1 provides a general approach to troubleshooting:



162930

Figure 5-1. Outline of Typical Troubleshooting Procedure

Checking for Excess Swapping

The first thing to look at is the paging activity, because the paging in and paging out of pages is costly in terms of both disk and CPU overhead. Get the `sar -pgrwut` report. With this information, you can reasonably determine whether more memory is needed or if the excess paging activity results from too few inodes, a situation that indirectly causes reusable pages to be discarded.

If the `vflt/s` value shown by `sar -p` is greater than 50, then look at `sar -g`. High values (greater than 5) for `pgscan/s` and `pgfree/s` imply that the page-stealing daemon is working overtime to find free pages because of a memory shortage. This can be verified by looking at the `ps -elf` report, which gives the number of cycles used by the page-stealing daemon. The value of `%ipf`, as reported by `sar -t`, should also be considered. If it is greater than 20 percent, then the freelist of inodes for that file system type is page-bound, which causes reusable pages to be discarded whenever `iget` takes an inode off the freelist. This can be remedied by increasing the tunable `SNINODE` found in `/etc/conf/mtune.d/sfs`. Other indicators of a memory shortage are the `freemem` value of `sar -r` and the `swpot/s` value of `sar -w`. A value of `swpot/s` that's greater than 1.0 is also an indicator of a memory shortage.

If memory shortages occur frequently, then memory should be increased in some way. This can be done in two ways. Un-installing optional kernel utilities that are not needed by your applications frees the memory used by the utilities so it can be used by user applications. If this is not possible, you probably need to add extra memory.

Checking for Disk Slowdowns

If the value of `%wio` (from the `sar -u` report above) is greater than 10 percent, or if the `%busy` for a disk drive (obtained by `sar -d`) is greater than 50 percent, then the system has a disk slowdown. Some ways to alleviate a disk slowdown are:

- Increase the amount of buffer space if `sar -b` indicates low hit rates (`%rcache < 90`, `%wcache < 65`).
- Organize the file system to minimize disk activity. If you have two disks, distribute the file systems for a more balanced load.
- Consider adding more memory if the situation persists. Additional memory reduces swapping/paging traffic and allows pages to remain in memory (reducing the number of user-level reads and writes that need to go out to disk).
- Consider adding an additional disk and balancing the most active file systems across the two disks.
- Consider changing the logical block size of file systems or even changing the file system type. (See “*Selecting a ufs or sfs Logical Block Size*,” or see “*Selecting a xfs Logical Block Size*” earlier in this chapter.)

Checking for Modem Interrupts

Run `sar -y` to get a report describing activity on terminal devices. If the number of modem interrupts per second, `madmin/s`, is much greater than 0, your system may have faulty communications hardware.

Checking for Table Overflows

To check for potential table overflows, get the `sar -vt` report. This report will let you know if overflows have occurred in the process or inode tables. Overflows can occur for either of two reasons: if a memory shortage occurs or if the maximum table size is reached. If memory is the problem, the table limits have not been reached. If the table limits are reached, the tunable should be increased. To avoid overflows in these tables, increase `NPROC` in the `/etc/conf/mtune.d/kernel` file and `SNINODE` in the `/etc/conf/mtune.d/sfs` file.

Shifting the Workload to Off-Peak Hours

Examine the files in `/var/spool/cron/crontabs` 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 or the process accounting commands `accton` or `acctoff`, if the Advanced Commands Package of the Software Development Set or of the Application Server Set is installed, to determine what processes are heavily loading the system. Encourage users to run large, noninteractive commands (such as `nroff` or `troff`) at off-peak hours. You may also want to run such commands with a low priority by using the `nice` or `batch` shell commands.

Dynamic vs. Static Kernels

See the chapter “Configuring and Building the Kernel” for detailed information on how to configure and build the kernel. Information on the types of kernels available and the performance advantages and disadvantages of each are provided below.

The OS supports two types of kernels; dynamic and static. A dynamic kernel consists of a base kernel and a set of dynamic loadable modules (DLMs). Dynamic loadable modules are, in general, only loaded on demand and will automatically unload after a period of inactivity. A static kernel contains no dynamic loadable modules. The entire static kernel is loaded at boot time.

The system administrator can select the type of kernel desired during the kernel build phase. By default, a dynamic kernel is built.

In general, a dynamic kernel would be preferred. Dynamic kernels tend to waste less real memory. A static kernel may be a better choice if:

- the performance overhead of dynamic loading is too severe.
- site is using small embedded systems where all modules configured are used.

- site is using small embedded system where physical memory is tight, the static kernel can be made even smaller by stripping out the symbol tables.
- certain system debugging situations are present.

Quick Reference Guide to Managing Performance

- To initialize the kernel-recording mechanism:
prfld
- To collect kernel profiling data:
prfdc
- To collect kernel profiling data at the time of invocation only:
prfsnap
- To collect system activity data automatically:
sadc
- To collect system activity data on demand:
sar

The following options are available with **sar**:

- a check file access operations
- b check buffer activity
- c check system calls
- d check disk activity
- g check page-out and memory freeing activity
- k check kernel memory allocation
- m check interprocess communication
- p check page-in and fault rates
- q check queue activity
- r check unused memory
- t check inode activity by file system type
- u check CPU utilization
- v check system table status
- w check swapping and switching volume

- y check terminal activity
- A report overall system performance
- P reports system activity for specified processors only
- R reports raw data values

- To compress an entire file system:

```
/usr/sbin/dcopy fs1 fs2
```

- To compress a single directory:

```
mv /home/bob /home/obob
mkdir /home/bob
cd /home/obob
find . -print | cpio -plm../bob
cd ..
rm -rf obob
```

- To determine the elapsed time, user time, and system time spent in execution of a command:

```
timex
```

- To enable, disable, or check the status of the sampling mechanism:

```
prfstat
```

- To find large, inefficient directories:

```
find / -type d -size +20 -print
```

NOTE

The *size* argument to the **find** command is in 512-byte blocks.

- To find inactive files:

```
find / -mtime +90 -atime +90 -print >file
```

- To format the data collected by **prfdc** or **prfsnap**:

```
prfpr
```

- To move user directories:

```
cd /fs1
find userx usery -print | cpio -pdm/fs2
rm -rf /fs1/userx /fs1/usery
```

- To print out the number of free file blocks and inodes:

```
df
```

- To terminate a runaway process:

```
kill -9 pid
```

- To summarize file system usage:

```
du
```

- To use a shell script to collect and store data in the binary file **/var/adm/sa/sadd**:

```
sa1
```

- To use a shell script to collect and store data in the ASCII file **/var/adm/sa/sar**:

```
sa2
```


Managing Dynamically Loadable Modules

What Are Dynamically Loadable Modules?	6-1
The Difference between Static Modules and Loadable Modules	6-1
Types of Dynamically Loadable Modules	6-2
Determining If a Module Is a DLM	6-2
Procedure	6-3
Determining If a DLM Needs to be Configured	6-3
Procedure	6-3
Determining Which DLMs Are Currently Loaded	6-4
Procedure	6-4
Obtaining More Information about a Specific DLM	6-4
Procedure	6-4
Configuring a Dynamically Loadable Module	6-5
Procedure	6-5
Reconfiguring All DLMs and the Entire Kernel	6-5
Procedure	6-5
Loading Modules	6-6
Overview of the Load Process	6-6
Automatic Load	6-6
Demand Loading a DLM	6-7
Procedure	6-7
Procedure	6-7
Resetting the Loadable Modules Search Path	6-7
Procedure	6-7
Unloading Modules	6-8
Overview of the Unload Process	6-8
Automatic Unload	6-8
Demand Unloading a DLM	6-9
Procedure	6-9

Managing Dynamically Loadable Modules

What Are Dynamically Loadable Modules?

Dynamically loadable modules (DLMs) are kernel modules (such as device drivers, file systems, and streams modules) that can be loaded during runtime. By using DLMs, you can avoid the work that was previously necessary (doing an `idbuid`, shutting down your system, bringing it back up, and so on) every time you install a new module. Thus, the first obvious advantage of using DLMs is a much easier installation procedure. Here are some others:

- DLMs improve system availability because drivers can be configured into the kernel while the system is running
- DLMs conserve system resources because infrequently used drivers are unloaded when they are not needed (when needed in the system, drivers are loaded from disk)
- DLMs give the kernel the ability to load and unload drivers automatically
- DLMs give you the ability to load and unload drivers on demand.

These are not the only advantages of using DLMs, however. Using DLMs also allows you to keep the kernel smaller. This is the advantage of dynamic loading over static linking. For modules to be linked statically, they must be linked in the kernel, even if the modules aren't being used. Dynamic loading makes it possible, however, to load modules only when they are to be used (after which you, or the system, can unload them).

The Difference between Static Modules and Loadable Modules

In previous releases, all kernel modules were maintained in individual object files (`.o` files) so they could be included or excluded from the kernel, depending on whether the features they supported were required in the system. The conditional nature of this arrangement meant that when you wanted to add a new module or remove an existing module, you had to relink the entire kernel and reboot the system to cause your new kernel configuration to take effect.

Kernel modules that are configured this way are called static modules. A static module is, by definition, non-loadable. That is, the module always remains linked into the kernel because either it is always required in the system (like the boot hard disk driver), or it is used so frequently or consumes so few resources (like the user terminal pseudo-device driver) that it makes sense to keep the module continuously configured. With DLM, some modules continue to be linked to the kernel in this traditional manner.

Other modules—modules that are not required, are used infrequently, or consume large amounts of resources—can be configured so they can be included or excluded from the kernel dynamically, without a system shutdown and reboot. These modules are called dynamically loadable modules (DLMs).

Like traditional static modules, dynamically loadable modules are maintained as individual object files, but they are not statically linked to the kernel. Instead, they are linked into the kernel when they are needed and unlinked when they are no longer in use. Floppy disk drivers and mouse drivers are two examples of kernel modules that are typically configured as dynamically loadable modules.

Dynamically loadable modules can be auto-loaded/unloaded by the kernel, or demand-loaded/unloaded by you. However, the decision to demand-load or demand-unload DLMs should be carefully considered, especially if you are not an experienced system administrator. Disturbing the complex interrelationship of system files and parameters can, if not carefully handled, result in an unusable system from which recovery will not be simple or quick. In most cases, allowing the automatic process to prevail will provide the performance improvements you may be seeking.

Types of Dynamically Loadable Modules

DLM supports loading and unloading of a variety of kernel module types. Some types of modules that can be loaded are these:

- device drivers (block, character, STREAMS and pseudo)
- High level (HDRV) device drivers
- STREAMS modules
- file systems
- exec modules
- kernel debugger
- miscellaneous modules—for example, modules containing support routines shared among several dynamically loadable modules, which are not needed in the statically configured kernel

Determining If a Module Is a DLM

A system administrator cannot make a kernel module dynamically loadable—that is done by the programmer who wrote the module. But before you can demand-load or demand-unload a module you need to know if it has been configured as a DLM.

Information about each installed module is stored in files named “module-name” in the `/etc/conf/sdevice.d` directory. These files typically start out in the following

format:

```
$version version-number
$loadable module-name
. . .
```

Procedure

Use the following procedure to determine if a file is a DLM:

1. Change directory by executing

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

2. Display the contents of the file.
3. Does the second line of the file begin with `$loadable`?

If yes, the *module-name* is a DLM.

If no, the module is not a DLM.

Determining If a DLM Needs to be Configured

Procedure

Use the following procedure to determine if a DLM is configured:

1. Change directory by executing

```
cd /etc/conf/mod.d
```

2. List the contents of the directory.
3. Is the *module-name* listed?

If no, the DLM module has not been configured.

If yes, the DLM module probably has been configured.

NOTE

Checking for a file named *module-name* in `/etc/conf/mod.d` may not be a foolproof method of determining if a module has been configured. Someone may have copied a module to that directory by hand rather than registering it with the kernel via the `idbuild -M module-name` command. If a module was copied there by hand, the DLM mechanism will not know about it and a later load may fail. See the `idbuild(1M)` and `idmodreg(1M)` manual pages for complete details.

Configuration is a necessary first step toward using a dynamically loadable module, but it still needs to be loaded before you can use it.

Determining Which DLMS Are Currently Loaded

Procedure

Use the following procedure to determine which DLMS are currently loaded:

- To obtain a list of module names (*module-name*) and module identifiers (*modid*) for currently loaded modules by running, execute

```
modadmin -s
```

- To obtain the full status of currently loaded modules, execute

```
modadmin -S
```

Obtaining More Information about a Specific DLM

Procedure

Use the following procedure to obtain more information about a specific DLM:

1. Determine the name, *module-name* . . . , or identifier, *modid* . . . , of the module.
2. List the information for a specific module identifier; execute

```
modadmin -q modid...
```

3. List the information for specific module names; execute

```
modadmin -Q module-name...
```

The command

```
modadmin -Q egl
```

causes the following information to be displayed about an eagle (egl) driver module.

```
# modadmin -Q egl
Module ID: 3,           Module: /etc/conf/mod.d/egl
Size: 125680,          Base Address: 0x42b64000
Hold Count: 2,        Dependent Count: 0
Unload Delay: 600 seconds
Eagle Ethernet Driver
Type: Device Driver
Character Majors:      46 - 51
```

Configuring a Dynamically Loadable Module

To configure a dynamically loadable module create the binary image of the loadable module and make it known to the kernel via the ID/TP kernel configuration tool, **idbuild(1M)**. **idbuild**, builds an operating system base kernel and/or loadable kernel module using the current system configuration in **/etc/conf**.

Procedure

Use the following procedure to configure a specific DLM:

- Execute the following command to configure only the loadable kernel module *module-name*, and put it into the **/etc/conf/mod.d** directory.

```
idbuild -M module-name
```

The loadable kernel modules configuration specified by using the **-M** option also includes making the necessary nodes in the **/dev** directory, adding and activating **/etc/inittab** entries if any **Init** file is associated with the modules, and registering the modules to the running kernel (see **idmodreg(1M)**), making them available for dynamically loading without requiring a system reboot.

Reconfiguring All DLMS and the Entire Kernel

Procedure

Use the following procedure to reconfigure the operating system base kernel:

- Execute

```
idbuild
```

without using the **-M *module-name*** option.

All loadable modules are reconfigured into the `/etc/conf/modnew.d` directory. This directory will be changed to `/etc/conf/mod.d` at the next system reboot if `$ROOT` is null or `/` (slash). This method requires a system reboot before the modules are made available for dynamic loading.

Loading Modules

Overview of the Load Process

When a loadable module needs to be added to the system, the DLM mechanism reads the module's loadable image on disk and copies the module into dynamically allocated kernel memory.

Once the module is in memory, DLM relocates the module's symbols and resolves any references the module makes to external symbols. DLM then executes special code in the module that enables the module to initialize itself dynamically.

When module initialization is complete, DLM executes code specific to the loadable module type. This code logically connects the module to the rest of the kernel.

DLMs can be loaded in either of two modes: demand or automatic (auto-loading). In either demand or automatic mode, the loading process includes the loading of any other modules on which the requested module depends.

Automatic Load

Automatic loading is the procedure implemented by the kernel when a user tries to access a module that's not loaded. When the kernel can't find the requested module, it loads it, does the link-editing and any other internal "administrative" work that's needed, and then makes the module available to the user requesting it. For example, the kernel would call DLM to auto-load a dynamically loadable device driver on the first open of any of the driver's configured devices. A loadable STREAMS module would be auto-loaded on the first `I_PUSH` of the module.

NOTE

High Level (HDRV) drivers cannot be auto-loaded. HDRV drivers can, however, be demand loaded using the `modadmin` command, or demand loaded by `init(1M)` via the `idmodload` command during a system reboot.

Note also that, once loaded, an HDRV driver remains loaded until the next system reboot (no DLM unload mechanism exists for HDRV drivers).

Demand Loading a DLM

Procedure

Use the following procedure to demand load a DLM:

- To request a specific module to be loaded, execute

```
modadmin -l module-name...
```

This command line will search for and load the module *module-name* and any other named module. It also completes all tasks associated with link editing the module to the kernel and making it accessible to the system. See the **modadmin (1M)** manual page for further details.

In some situations, such as a network environment, you may want to load modules from a remote mounted directory or from a location other than the default loading path.

Procedure

Use the following procedure to demand load DLMs from a non-default directory:

- Execute

```
modadmin -d dirname
```

This command prefixes the specified directory path to the current loading path (which, by default, is `/etc/conf/mod.d`).

Resetting the Loadable Modules Search Path

Procedure

Use the following procedure to reset the loadable modules search path to the default value:

- Execute

```
modadmin -D
```

See the **modadmin (1M)** manual page for more details.

Unloading Modules

Overview of the Unload Process

The unload process undoes what was done during the load process.

First, the DLM mechanism executes code specific to the loadable module type that logically disconnects the module from the rest of the kernel. Once the module is disconnected, DLM then executes the module-supplied code that enables the module to clean up for termination. When clean-up is complete, DLM releases the memory allocated for the module.

Auto-loaded modules can be unloaded through either of the two mechanisms: auto-unload or demand-unload. Demand-loaded modules, however, can usually be unloaded only via a demand-unload. There's one exception to this, however: if an administrator tries to execute a demand-unload when the module is being used, the demand-unload will fail. Instead of being unloaded immediately, the module will be unloaded via auto-unload the next time auto-unloading is done on the system.

Automatic Unload

When the amount of available memory is lower than a predetermined low-water mark, the kernel calls DLM, periodically waking up the auto-unload daemon, and attempts to unload any modules that have become candidates for unloading. Modules become candidates for auto-unloading when they are inactive, they have not been accessed for some predetermined amount of time, and no other loadable modules depend on them.

If the attempt to auto-unload a module is successful, the memory allocated for the module is reclaimed. (These unloaded modules remain on the disk, ready to be re-loaded whenever necessary.) Unloading continues until all the unloadable modules are unloaded.

NOTE

Modules that are demand loaded cannot be auto-unloaded. If a demand-loaded module is no longer needed in the system, it must be demand unloaded.

For example, a loadable device driver would become a candidate for auto-unloading on the last `close` of all its configured devices, and a loadable STREAMS module would become a candidate for auto-unloading on its last `I_POP`. The amount of time that must elapse before inactive modules are considered candidates for auto-unloading is controlled by the value of the global tunable parameter `DEF_UNLOAD_DELAY` in `/etc/conf/mtune.d/kernel`. Individual modules can override the value of the global auto-unload delay by specifying their own value for the auto-unload delay parameter, `PFX_UNLOAD_DELAY`, in their **Mtune (4)** files.

NOTE

On a demand unload request, the auto-unload delay parameter value is ignored.

Demand Unloading a DLM

Demand unloading is a manual procedure for administrators who need to unload modules that aren't being used as a way of creating more space in the kernel.

Procedure

- If you know the module identifier, execute

```
modadmin -u modid ...
```

to unload the module *modid*. The argument *modid* is an integer that **modadmin** assigned to the module as an identifier when it was loaded.

- If you know the module name, execute

```
modadmin -U module-name ...
```

More than one *modid* or *module-name* can be supplied on the command line.

If the module is not being used when the request is made, and if no other loaded module depends on the module, DLM will unload it. If the module is being used, or if another loaded module references symbols defined in the module, DLM does not unload the module. The module will be available for the next auto-unload. See the **modadmin (1M)** manual page for further details.

Tunable Parameters



Introduction	7-1
Tunable System Parameters	7-1
Reconfiguring the Operating System	7-2
Autotuning Parameters	7-3

Introduction

This chapter describes procedures for modifying tunable parameters contained in the Operating System (OS). The OS uses an Installable Drivers/Tunable Parameter (ID/TP) scheme. Tunable parameters are variables used to set the sizes and threshold of the various control structures of the OS. Adjusting these parameters can have a significant impact on system performance, both positive and negative. Think carefully about your computer's uses, analyze its current performance, and consider other performance factors (such as file system organization, sticky bits, `$PATH` efficiency, and file system block sizes) before tuning the kernel.

Tunable parameters may be modified using the step-by-step procedure outlined in this chapter or by using the kernel configuration utility, `config(1M)`. The `config` utility allows the administrator to determine what tunables are available, the function of a specific tunable and examine/modify the current value assigned to a specific tunable. Instructions on how to use the `config` utility can be found in the chapter “*Configuring and Building the Kernel*”. You may also refer to the manual page `config(1M)` for additional information.

Tunable System Parameters

Tunable system parameters are used to set various table sizes. The initial 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. To modify kernel parameters, the operating system kernel will have to be re-configured, and the system re-booted.

The ID/TP scheme places all system tunable parameters in files in the kernel configuration directory `/etc/conf/mtune.d`. These files are used to verify the tunable parameters and to specify the default, minimum, and maximum values for them. The format of these files is defined on the `Mtune(4)` manual page.

Each system tunable parameter is assigned a default value along with a minimum and maximum value. You can examine the `Mtune` files to determine the tunable parameter settings for your computer but you should never modify these files. The `idtune` command is provided to tune those tunables specified in `Mtune` files.

Since the appropriate values for some tunables are dependent on memory size, `idbuild` and `idautotune` will adjust the min, max and default values for these tunables. Therefore, you should re-run `idbuild` after adding memory to your system.

Reconfiguring the Operating System

To change the value of a parameter, execute **idtune (1M)**, specifying the parameter to be changed and the desired value for that parameter.

```
/etc/conf/bin/idtune [-f | -m] [-c] parm value
```

Here *parm* is the parameter to be changed and *value* is the desired value for it. (As shown above, **-f**, **-m**, and **-c** are optional. See **idtune (1M)** for details.)

NOTE

You'll be able to change the value for a new parameter only if that parameter has been installed with the **idinstall (1M)** command and an **Mtune** or **Autotune** file.

If a value has already been assigned to the parameter you specify, you are asked to confirm the change with the following message:

```
Tunable Parameter parm is currently set to old_value in  
/etc/conf/cf.d/stune  
Is it OK to change it to value? (y/n)
```

If you answer **y**, the change is made. If you don't, the parameter is not changed and the following message is displayed:

```
parm left at old_value.
```

You can override this step by invoking the **-f** (force) option, which ensures the change is made without any messages.

Whenever you change the value of a parameter, you must rebuild the operating system kernel [with **idbuild (1M)**] and reboot your system. Only after you do this will the new values take effect. For details about rebuilding the kernel and rebooting, see "Booting and System States" in *System Administration Volume 1*.

You can display the current value of a tunable parameter and its default, minimum, and maximum values by specifying:

```
/etc/conf/bin/idtune -g [-c] parm
```

The values from **stune** are displayed, except when the **-c** option is used with the **-g** option, and then the current values of **stune.current** are displayed.

You can change the parameter values in both **stune** and **stune.current**.

```
idtune -c parm value
```

Tune for the current and subsequent loading modules.

Autotuning Parameters

Certain tunable parameters are highly sensitive to the amount of memory present. For these, a table of information relating memory size to appropriate default, min, and max values is provided from which **mtune** entries will be created, and which will also be passed to the kernel to allow autotuning of these parameters on boot. The mtune lines are re-calculated whenever **idtune** or **idbuild** is run. **idtune -g tunable-name** will provide the expected current, default, minimum, and maximum information for the tunable based on the amount of memory currently available.

The files specifying this are in **/etc/conf/autotune.d**. The format for a line in such a file is

```
tunable-name DEF/MIN/MAX LINEAR/STEP memsize (M3) tunable value
```

These files should not be edited; **idtune** should be used instead. If you add memory after idtuning one of these tunables, consider doing an **idtune -d tunable name** to allow it to go back to autotuning.

For example, if you know your application is running on an SFS filesystem and tends to hit inode table overflow messages, you can increase the value on the one default line corresponding to your system's memory size, or you can increase the value on all the default lines, or you can **idtune SFSNINODE** with the value you want. If you run **idtune** and later add memory and rebuild your kernel, the number of inodes will not automatically increase.

The parameters that are autotuned are: FS Filesystem (BUFHWM, DNLCSIZE) and SFS Filesystem (SFSNINODE) .

Configuring and Building the Kernel

Introduction	8-1
System Tunable Parameters	8-1
Configuring Kernel Modules	8-2
Hardware Adapters	8-2
Building and Installing the Kernel	8-3
Dynamic and Static Kernel Support	8-4
Kernel Configuration Utility	8-4
General Information	8-4
Kernel Config Utility	8-5
Terminal Attributes	8-5
Object Directory	8-6
How to Invoke Config	8-6
Current Configuration	8-6
Access Requirements	8-7
Menu Operations	8-7
Output Navigation	8-8
Kernel Config Utility Operations	8-9
Main Menu Functions	8-9
System Tunable Menu	8-9
Kernel Module Menu	8-11
Kernel Hardware Adapters Menu	8-13
Scheduling Class Dispatcher Parameter Tables	8-15
Kernel Rebuild Menu	8-16
Kernel Options Menu	8-17
Other Menu Functions	8-18
Minimizing Kernel Size	8-18
Configuring the Kernel	8-18
Required Modules	8-18
Optional Modules	8-19
Device Drivers	8-20
File Systems	8-20
Memory Management	8-21
Networking	8-21
Miscellaneous	8-21
Scheduling Classes	8-22
Security	8-22
Streams	8-22
VxVM Volume Manager	8-23
Tunable System Parameters	8-23
Minimum Bootable Configuration	8-26
Single User Mode (init S)	8-26
Multi User Mode (init 2)	8-27
Kernel Symbols	8-27
Loading Symbols	8-28
Removing Symbols from the Kernel Object File	8-28

Configuring and Building the Kernel

Introduction

The kernel is held in a file called `/stand/unix` (or `/unix`) which is loaded into memory each time the system is booted. It is a nucleus of essential code that carries out the basic functions of the system. The kernel remains in memory during the entire time that the system is running (it is not swapped in and out like most user programs).

The exact configuration of the kernel depends on a large number of variable parameters, optional components and addresses:

- There are a large number of tunable parameters that define things such as buffer sizes, number of buffers, maximum numbers of files, links and modules.
- The kernel can contain a number of optional device drivers and streams modules.
- The kernel contains a number of definitions for the locations of hardware devices.

Kernel configuration, or reconfiguration, is the process of redefining one or more of these kernel variables and then creating a new kernel according to the new definition.

In general, the supplied kernel is created with tunable parameters and device drivers that are suitable for most systems. However, you will have to reconfigure the kernel if you want to alter any of the tunable parameters to optimize kernel performance.

After you change a tunable parameter, install/de-install kernel modules, or modify the hardware configuration, the kernel will need to be rebuilt.

Kernel configuration, or reconfiguration, may be performed using the step-by-step manual procedural method described below or by using the kernel configuration utility, `config(1M)`. `Config` is a menu-driven centralized utility that provides a front-end interface to the current configuration method. Information on how to use `config` is provided later in this chapter.

System Tunable Parameters

The tunable parameters are used to define limits, enable features, declare initial sizes of tables, etc. The values for tunables can be displayed and/or altered using `idtune(1M)`. Tunables can only be modified within a pre-defined range (i.e. between minimum and

maximum). For more information on tunable parameters, refer to the chapter “*Tunable Parameters*” or the **config** utility described later in this chapter.

Configuring Kernel Modules

A set of object modules are linked together when building the kernel. Some of the modules are required in every system. Others are largely isolated and may be de-configured according to the application environment.

Configuring or de-configuring a module is done using the following procedure or by using the **config** utility described later in this chapter.

1. **cd /etc/conf/sdevice.d**
2. Locate the module. Each module has a separate file in this directory with the name of the module. The list of all non-required modules along with brief descriptions, appears below.
3. In the file associated with the module, change the configure field to “Y” to configure the module or “N” to de-configure it.
4. Re-build the kernel. Refer to “Building and Installing the Kernel” later in this chapter for instructions on how to build the kernel.

Hardware Adapters

The hardware adapters configuration file is: **/etc/conf/sadapters.d/kernel**.

During system software installation, the hardware adapters configuration file is automatically updated based on the hardware present and the configuration information provided by the installer.

Whenever new hardware is installed following initial installation, the adapters file must be edited and an entry for each new adapter added. An exception to this is self-configuring drivers which are not required to be in this file. An example of a self-configuring driver is the Integral SCSI interface (is). Example entries for each of the supported adapters are documented at the beginning of the hardware adapters configuration file.

Note

The kernel must be rebuilt and the system rebooted for changes made to the adapters configuration file to take effect.

An alternative method to the above procedure is the use of the **config** utility which is described later in this chapter.

Building and Installing the Kernel

Base kernel rebuilds are needed after any one the following is accomplished:

1. Kernel module is installed, removed, configured, or de-configured.
2. Any system tunable parameter is modified.
3. Hardware adapters file is updated.

The **idbuild (1M)** command builds a UNIX system base kernel and/or configures loadable kernel modules using the current system configuration in **/etc/conf**. Invoked with the **-B** option, **idbuild** builds the kernel immediately. Invoked without any options, the kernel rebuild is deferred to the next system reboot. See the on-line man page for more details. (An alternative method to the procedure described below is the use of the **config** utility which is described later in this chapter.)

Building a UNIX system kernel consists of three steps:

1. Configuration tables and symbols, and module lists are generated from the configuration data files.
2. Configuration-dependent files are compiled and then are linked together with all of the configured kernel and device driver object modules.
3. Kernel symbol table is attached to the kernel.

By default, the kernel is placed in - **/etc/conf/cf.d/unix**.

If the kernel build is successful, **idbuild** sets a flag to instruct the system shutdown/reboot sequence to replace the standard kernel in **/stand/unix** with the new kernel. Another flag is set to cause the environment (device special files, **/etc/inittab**, etc.) to be reconfigured when the new kernel is booted.

If one or more loadable kernel modules are specified with the **-M** option, **idbuild** will configure only the specified loadable kernel modules and put them into the **/etc/conf/mod.d** directory. Otherwise, a UNIX system base kernel is rebuilt with all the loadable modules reconfigured into the **/etc/conf/modnew.d** directory. This directory is then moved to **/etc/conf/mod.d** at the next system reboot.

When loadable kernel modules are configured with the **-M** option, **idbuild** also creates the necessary nodes in the **/dev/** directory, adding and activating **/etc/inittab** entries if any **Init** file is associated with the modules, and registering the modules to the running kernel. This makes them available for dynamic loading without requiring a system reboot.

For more details on actually booting with a new kernel, refer to the chapter “Booting and System States” in Volume 1, *System Administration* manual.

Dynamic and Static Kernel Support

With the OS both dynamic and static kernels are supported. A dynamic kernel consists of a base kernel and a set of dynamic loadable modules (DLMs). Dynamic loadable modules are generally loaded only on demand and will automatically unload after a period of inactivity. A static kernel contains no dynamic loadable modules. The entire static kernel is loaded at boot time.

The system administrator can select the type of kernel during the kernel build phase. By default, a dynamic kernel is built. The `-S` option to `idbuild(1M)` specifies that the resultant kernel should be static. The kernel configuration utility `config(1M)` (described below) can also be used to build either a dynamic or static kernel by selecting the appropriate menu item.

A summary of the important features of each kernel follows:

Dynamic kernel:

- consists of base kernel and set of dynamic loadable modules (DLMs).
- DLMs loaded on demand and automatically unloaded after a period of inactivity.
- consumes less real memory.
- built by default (may also be selected using configuration utility, `config`).

Static kernel:

- entire static kernel loaded at boot time.
- less overhead than with dynamic loading
- better in certain system debugging situations.
- to select, specify `-S` option to `idbuild`, or select using configuration utility `config`
- better in small, embedded systems.

Kernel Configuration Utility

General Information

The kernel configuration utility (`config(1M)`) is primarily a tool to enable system administrators (or developers) to configure and build the OS kernel.

The `config` utility may be used instead of the kernel configuration practices and methods described earlier in this chapter. The `config` utility has not replaced them. The existing utilities and administrative practices may still be used. For clarification, the following configuration operations may still be done:

system tunables	modification using id tune (1M)
kernel modules	enabling/disabling by manually modifying etc/conf/sdevice.d files
hardware adapters	additions/modifications/deletions by manually modifying the etc/conf/sadapters.d/kernel file
kernel rebuilds	rebuilds using id build (1M)

Kernel Config Utility

Config provides a centralized kernel configuration utility. Kernel configuration consists of the following:

system tunables	System tunables are configuration parameters used to specify limits, enable features and control space vs. performance operations. config allows the administrator to determine what tunables are available, to determine the function of a specific tunable and examine/modify the current value assigned to a specific tunable.
kernel modules	A full set of kernel modules is provided with the OS. Some of the modules are required but most can be optionally enabled or disabled. A disabled module will not be built into a new kernel (which reduces the kernel size). config displays all of the available modules and can be used to enable or disable a specific module. An administrator can also determine what functionality a module provides and information such as size of object files associated with a particular module.
hardware adapters	The set of hardware adapters (or controllers) will vary from system to system and will also vary over time at a specific site. In most cases, the kernel must be notified of an adapters change before it will take place. Config displays the current set of adapters and can be used to add a new adapter or modify/delete an existing one.

Scheduling Class Dispatcher Parameter Tables (DPT)

Each scheduling class has a range of priorities assigned to it and each priority level in that range has characteristics associated with it. All of that information is kept in a table which the user can manipulate to customize the behavior of the scheduling class to fit the user's system requirements.

Config can also be used to rebuild the kernel with the current tunables, modules and adapter configuration.

Terminal Attributes

For correct operation, the **TERM** variable must be set appropriately prior to invocation. Output will be adjusted according to the **LINES** and **COLUMNS** variables if specified.

By default, **config** will use the highlight (standout) feature on the terminal. However, on some terminals, highlight mode behaves atypically and the menus and output will become improperly aligned.

The **-h n** option is used to set the terminal highlight mode (0 = off, non-zero = on). By default, highlight mode is on, except on wyse50 terminals.

Object Directory

The kernel configuration files are located in the directory **<OBJ DIR>/etc/conf**. By default, **<OBJ DIR>** is **"/**. However, in some cases, a system administrator may want to use a different kernel object directory. There are three methods used to specify an object directory. The ordering of these operations is important and is outlined below:

default	By default, the object directory is / .
\$OBJ variable	Upon program invocation, if the \$OBJ environment variable is set then the object directory is set to \$OBJ .
-r option	The -r option is used to specify an object directory. This overrides what may be set in the \$OBJ environment variable.
utility operation	config has an operation to modify the object directory. This overrides any previous value.

How to Invoke Config

You may invoke the **config** utility as follows:

/sbin/config

Available options are:

-r OBJ	See the Object Directory section above for detail.
-h n	See the Terminal Attributes section above for details.

Current Configuration

Config examines and modifies the kernel configuration files located in the object directory. The configuration files may not necessarily agree with the configuration within the currently running kernel. However, it would reflect the configuration if a kernel were rebuilt with these files in their present state.

When a modification is made to a kernel configuration file, this does not immediately alter the running kernel. The kernel must be rebuilt and rebooted before the modification takes effect.

Access Requirements

Kernel configuration files are not readily accessible to all users. In general, the kernel configuration files may be read by all users, but can only be modified by the root user. Similarly, kernel building is done by using the `idbuild(1M)` utility which can only be invoked by the root user. For this reason, `config` should only be invoked by the root user if configuration modification or kernel building will be done.

On a B2-secure system, it is required that kernel builds be done while running a non-secure kernel at single-user mode.

Menu Operations

`Config` uses the `curses(3curses)` and `menu(4)` subsystems. Menus and sub-menus are used within the utility. Menus are used to select the operations to perform. Sub-menus are used to select input values (such as a specific tunable or module name) within an operation. Menus have only one column while sub-menus are multi-column.

There are various mechanisms for navigating through menus and sub-menus. The cursor (highlighted item) can be moved up, down, left or right using the keyboard “arrow” keys and also “control” sequences. The cursor can also be directed to a specific item by providing the first few characters of the item itself. Enough characters must be provided to uniquely specify the item. After the proper item is selected, the `<Enter>` key is used to perform the operation with the selected item.

The following summarizes the menu navigation keys. Some of these keys may not be available on certain terminals. Also, depending on the menu, not all keys may be applicable:

<code><down arrow></code>	next menu item
<code><space></code>	next menu item
<code><tab></code>	next menu item
<code><ctrl-n></code>	next menu item
<code><up arrow></code>	previous menu item
<code><ctrl-p></code>	previous menu item
<code><left arrow></code>	left menu item
<code><right arrow></code>	right menu item
<code><Page Down></code>	scroll down page
<code><ctrl-f></code>	scroll down page
<code><Page Up></code>	scroll up page
<code><ctrl-b></code>	scroll up page
<code><End></code>	last menu item

<ctrl-e>	last menu item
<Home>	first menu item
<ctrl-a>	first menu item
<ctrl-q>	quit, return to previous menu
<Enter>	take selected item

Output Navigation

Some **config** operations may produce terminal output that does not fit on a single screen. Various keys are used to navigate through output. The output window can be scrolled up or down a line at a time or a screen at a time. You can also go to the first or the last screen of data.

The following summarizes the function of the output navigation keys. Some keys may not be available on certain terminals and, depending on the output, not all keys may be applicable:

<down arrow>	scroll down 1 line
<space>	scroll down 1 line
<tab>	scroll down 1 line
<ctrl-n>	scroll down 1 line
<up arrow>	scroll up 1 line
<ctrl-p>	scroll up 1 line
<Page Down>	scroll down 1 page
<ctrl-f>	scroll down 1 page
<Page Up>	scroll up 1 page
<ctrl-b>	scroll up 1 page
<End>	last output screen
<ctrl-e>	last output screen
<Home>	first output screen
<ctrl-a>	first output screen
<ctrl-q>	quit output display
<Enter>	quit output display

Kernel Config Utility Operations

Main Menu Functions

The `config` utility is primarily menu-driven. Screen 8-1 depicts the Main Menu screen which is used to get to the various sub-menus. A description of each sub-menu is provided following the main menu screen. Also refer to the `config(1M)` man page for additional information,

```

PowerMAX_OS Kernel Configuration Utility
Main Menu

>> Tunables  examine/modify system tunables
Modules     examine/enable/disable kernel modules
Adapters    examine/modify hardware adapters configuration
Class DPT   examine/modify Scheduling Class DPT's
Rebuild     rebuild kernel with current configuration
Options     utility options
Shell       temporary shell escape
Help        help for this menu
Quit        quit - exit utility

Press <Enter> to select item,  Navigate with arrow keys, <tab> or item name

```

Screen 8-1. Kernel Configuration Utility Main Menu

System Tunable Menu

Selecting `Tunables` from the Main Menu will cause `config` to enter the System Tunable Menu (Screen 8-2). This menu is used to look up, examine and modify system tunables. The operations available under this menu are described below.

```

PowerMAX_OS Kernel Configuration Utility
System Tunable Menu

>> Categories  list all tunable categories
Tunables      list all tunables for a specific category
Find          find tunable(s) by keyword
All Tunables  list all tunables
Describe      show full description/values for a specific tunable
Modify        modify current value for a specific tunable
Save          save to file full list of categories/tunables
Help          help for this menu
Return        return to main menu

Press <Enter> to select item,  Navigate with arrow keys, <tab> or item name

```

Screen 8-2. System Tunable Menu

Categories System tunables are separated into distinct functional categories. Every tunable is in exactly one category. This function will simply display the names of the different categories.

- Tunables** Display all tunables within a specified category. For each tunable within the category, its name, a short description and its currently configured value are displayed (values shown are generally in hexadecimal.) Specifying a category can be done by either providing its name or by selecting from a sub-menu of all available categories. This sub-menu mode is entered by depressing `<esc>` at the `Category Name (<esc> for menu) :` prompt.
- Find** Associated with each tunable is a small set of keywords. Keywords are used to locate a tunable (or group of tunables) if the exact name (or names) is not known. For example, you want to modify the tunable that controls the size of process core files but don't know the exact name(s) of the tunable(s). By doing a **Find** with the keyword **core**, you can see all of the tunables that are related to core files. For all tunables that are found by keyword, its name, a short description and its currently configured value are displayed.
- Specifying a keyword can be done by either providing it directly or by selecting from a sub-menu of all available keywords. This sub-menu mode is entered by depressing `<esc>` at the `Keyword (<esc> for menu) :` prompt.
- All** Displays all available tunables. For each tunable, its name, a short description and its currently configured value are displayed.
- Describe** Provides a more detailed description for a specified tunable. For the selected tunable, the following is displayed: its name, its category, the kernel module that this tunable is associated with, its minimum, maximum, default and current values and a short and long description.
- Specifying a tunable can be done by either providing its name or by selecting from a sub-menu. This sub-menu mode is entered by depressing `<esc>` at the `Tunable Name (<esc> for menu) :` prompt. If the sub-menu mode is entered, then the category must be selected first followed by selecting the tunable within that category.
- Some tunables are simply character strings. These tunables have no minimum, maximum and default values.
- Modify** Used to modify the current value for a specific tunable. For the specified tunable, the following is displayed: its name, its category, the kernel module that this tunable is associated with and its minimum, maximum, default and current values. You can then optionally change the current value. The new value cannot be less than the minimum or greater than the maximum.
- Note that this operation modifies the currently configured value but does **NOT** affect the running kernel. The kernel must be rebuilt and booted before a modification to a tunable impacts the running kernel. Specifying a tunable can be done by either providing its name or by selecting from a sub-menu. This sub-menu mode is entered by depressing `<esc>` at the `Tunable Name (<esc> for menu) :` prompt. If the sub-menu mode is entered, then the category must be selected first followed by selecting the tunable within that category.
- Save** Used to send a list of all tunables to an indicated file. This file can then be perused or printed. The output may be formatted in one of two ways: either

grouped by category or as a single (alphabetized) list of all tunables. You must specify the output format and the name of the output file.

- Help Displays built-in help.
- Return Returns to Main Menu.

Kernel Module Menu

Selecting Modules from the Main Menu will cause **config** to enter the Kernel Module Menu (Screen 8-3). This menu is used to lookup, examine and enable/disable kernel modules. The operations available under this menu are described below.

```

PowerMAX_OS Kernel Configuration Utility
Kernel Module Menu

>> Categories  (list all module categories)
Modules       (list all modules for a specific category)
Find          (find module(s) by keyword)
All Modules   (list all modules)
Describe      (show full description for a specific module)
Enable        (enable or disable a specific module)
Summarize     (list all enabled / disabled modules)
Save          (save to file full list of categories/modules)
Help          (help for this menu)
Return        (return to main menu)

Press <Enter> to select item,  Navigate with arrow keys, <tab> or item name

```

Screen 8-3. Kernel Module Menu

Categories Kernel modules are separated into distinct functional categories. Every module is in exactly one category. This function will simply display the names of the different categories.

Modules Display all modules within a specified category. For each module within the category, its name, a short description and its current enabled/disabled status are displayed.

Specifying a category can be done by either providing its name or by selecting from a sub-menu of all available categories. This sub-menu mode is entered by depressing **<ESC>** at the Category Name (**<esc>** for menu) : prompt.

Find Associated with each module is a small set of keywords. Keywords are used to locate a module (or group of modules) if the exact name (or names) is not known. For example, you want to disable the FDDI driver but don't know the exact name of the module. By doing a **Find** with the keyword **fddi**, you can see all of the modules that are related to FDDI.

For all modules that are found by keyword, its name, a short description and its current enabled/disabled status are displayed.

Specifying a keyword can be done by either providing it directly or by selecting from a sub-menu of all available keywords. This sub-menu mode is entered by depressing <esc> at the Keyword (<esc> for menu) : prompt.

- All Displays all available modules. For each, its name, a short description and its current enabled/disabled status are displayed.
- Describe Provides a more detailed description for a specified module. For the selected module, the following is displayed:
 - module name
 - category
 - size This is the total size of the object files associated with this module. If not a Dynamic Loadable Module (DLM), the static size of the kernel is increased by this amount if the module is toggled from disabled to enabled. While running, the module may require additional dynamic memory.
 - DLM Indicates whether or not this module is a DLM. A DLM is not loaded with the static kernel, but is loaded on demand.
 - Enabled Current enable/disable status.
 - Required Indicates whether the module must be enabled, or can optionally be disabled.
 - Depends List of modules that must also be enabled if this module is enabled. **config** will not verify dependencies, but **idbuild(1M)** (which is used indirectly by **config**) will enforce dependencies.

Abstract Description

Extended Description

Specifying a module can be done by either providing its name or by selecting from a sub-menu. This sub-menu mode is entered by depressing <esc> at the Module Name (<esc> for menu) : prompt. If the sub-menu mode is entered, then the category must be selected first followed by selecting the module within that category.

- Enable Used to enable a disabled module or disable an enabled module. For the specified module, the following is displayed: its name, its category, a short description, its module dependency list and its current enable/disable status. You may enable the module if it is disabled or disable the module if it is enabled. A required module (i.e. one that must always be enabled) cannot be modified.

Note that this operation modifies the currently enabled status but does NOT affect the running kernel. The kernel must be rebuilt and booted before a modification to a module impacts the running kernel.

Specifying a module can be done by either providing its name or by selecting from a sub-menu. This sub-menu mode is entered by depressing `<ESC>` at the `Module Name (<ESC> for menu):` prompt. If the sub-menu mode is entered, then the category must be selected first followed by selecting the module within that category.

Summarize	Used to see which set of modules are currently enabled and which are disabled. The first <code>output</code> window provides the names of all enabled modules. The second <code>output</code> window provides the names of all disabled modules.
Save	Used to send a list of all modules to an indicated file. File may then be perused or printed. The output can be formatted in one of two ways: either grouped by category or as a single (alphabetized) list of all modules. You must specify the output format and the name of the output file.
Help	Displays built-in help.
Return	Returns to Main Menu.

Kernel Hardware Adapters Menu

Selecting `Adapters` from the Main Menu will cause `config` to enter the Hardware Adapters Menu (Screen 8-4). This menu is used to examine, add, modify or delete adapters. The operations available under this menu are described below.

```

PowerMAX_OS Kernel Configuration Utility
Hardware Adapters Menu

>> Show      (all adapter types)
   Current   (display current adapter configuration)
   Add       (add adapter)
   Modify    (modify adapter)
   Delete    (delete adapter)
   Fields    (describe adapter fields)
   Help      (help for this menu)
   Return    (return to main menu)

Press <Enter> to select item,  Navigate with arrow keys, <tab> or item name

```

Screen 8-4. Hardware Adapters Menu

`Show` Display all of the adapters available with a short description for each.

`Current` Displays the current adapter configuration.

The current configuration can come from one of two sources:

1. The `Sadapters (4)` file (usually `/etc/conf/sadapters.d/kernel`).
2. As built into the currently running kernel.

The **Sadapters (4)** file reflects the adapter configuration if a new kernel was re-built with the current configuration.

The adapters configuration built into the kernel may differ because a different **Sadapters (4)** configuration was specified or adapters have been auto-configured (i.e. automatically added to the configuration).

This operation will display a sub-menu used to select the source for the current adapter configuration. In order to examine the adapter configuration in the running kernel, **/dev/kmem** access is required.

For each adapter, the following is displayed: global adapter number, adapter name, logical adapter number, bus type, interrupt type, slot and I/O addresses 1 and 2.

Add Used to add a new adapter to the current configuration(i.e. the **Sadapters (4)** file).

For the specified adapter type, example configurations are displayed. You must then provide all of the adapter fields. Verification is done before the adapter line is added.

Specifying an adapter type can be done by either providing it directly or by selecting from a sub-menu of all available adapters. This sub-menu mode is entered by depressing **<ESC>** at the Adapter Name (**<esc>** for menu) : prompt.

Modify Used to modify an existing adapter (as specified in the **Sadapters (4)** file).

For the selected adapter, you will be prompted to provide the adapter fields. Depressing **<Enter>** will leave the field unchanged. Verification is done before the adapter line is modified.

Specifying an adapter is done by selecting from a sub-menu of the currently configured adapters.

Delete This operation is used to remove an existing adapter (as specified in the **Sadapters (4)** file).

Verification is done before the adapter line is removed.

Specifying an adapter is done by selecting from a sub-menu of the currently configured adapters.

Fields A help screen that describes each of the adapter fields.

Help Displays built-in help.

Return Returns to Main Menu.

Scheduling Class Dispatcher Parameter Tables

Selecting `Class DPT` from the Main Menu will cause `config` to enter the Scheduling Class Configuration Menu (Screen 8-5). This menu is used to manipulate the Scheduling Class-specific Dispatcher Parameter Tables (DPTs). The operations available under this menu are described below.

```

PowerMAX_OS Kernel Configuration Utility
Scheduling Class Configuration Menu

Current Class: NONE

>> Select    the desired scheduling class
View        the existing dispatcher parameter table
Modify      the existing dispatcher parameter table
Insert      a new entry into the table
Delete      an existing entry from the table
Help        help for this menu
Return      return to main menu
Quit        terminate program

```

Screen 8-5. Scheduling Class Configuration Menu

Select	Select one of the available scheduling classes. Each scheduling class, except for the SYS class (System Class), has a DPT associated with it. Before any of the five table operations can be performed, config needs to know which table to act upon. The result of the Select option is a list of all the scheduling classes available in the system.
Generate	This builds a DPT whose entries are initialized with default values. The user is prompted for the desired number of entries and the starting global priority for the table. Each subsequent entry in the table will have a priority that is one greater than the previous entry. The rest of the fields will be initialized with default values. To see the effects of the user-provided numbers use the View option. To make additional changes to the new entries, use the Modify option.
View	View the contents of the class-specific DPT
Modify	Modify the values of the fields in one entry of the DPT. Changes can be made only to one entry at a time.
Insert	Insert entries into the class-specific DPT. As in the Generate option, the starting global priority and the number of entries desired are requested from the user. The effects can, likewise, be seen using the View option.
Delete	Delete one or more entries from the class-specific DPT. The number of lines to be deleted is requested from the user along with the entry at which the deletion will start. The number of lines to be deleted cannot be greater than the number of entries available from the point of deletion to the end of

Help	Built-in help.
Return	Returns to MAIN MENU.

Kernel Rebuild Menu

Selecting `Rebuild` from the Main Menu will cause `config` to enter the Kernel Build Menu (Screen 8-6). This menu is used to rebuild a kernel with the current configured tunables, modules, and adapters. Note that the entire kernel is rebuilt. You may not build just a single module. (Use the `idbuild(1M)` utility with the `-M` option if this is desired.) The operations available under this menu are described below.

```
PowerMAX_OS Kernel Configuration Utility
Kernel Build Menu

>> Boot          (defer kernel build to next system reboot)
    Dynamic      (build dynamic kernel immediately)
    Static       (build static kernel immediately)
    Strip Symbols (build static kernel immediately and strip(1)symbols)
    Help         (help for this menu)
    Return       (return to main menu)

Press <Enter> to select item,  Navigate with arrow keys, <tab> or item name
```

Screen 8-6. Kernel Build Menu

Boot Build a new kernel by invoking `idbuild(1M)` with no options. This causes a flag to be set. The kernel will be automatically rebuilt and installed on the next system reboot. The new kernel will be dynamic (i.e. contains DLMs).

Dynamic Build a new kernel by invoking `idbuild(1M)` with the option `-B`. This causes an immediate build of a kernel to be performed. In this case, the kernel is dynamic (i.e. contains DLMs).

`init(1M)` (with state 0) or `shutdown(1M)`, must be called to install the new kernel and then reboot.

Static Build a new kernel by invoking `idbuild(1M)` with the options `-BS`. This causes an immediate build of a kernel to be performed. In this case, the kernel is static (i.e. does not contain Dynamic Loadable Modules).

`init(1M)` (with state 0) or `shutdown(1M)`, must be called to install the new kernel and then reboot.

Strip Symbols

Build a new kernel with `idbuild(1M)` with the options `-BSs`. This causes an immediate build of the kernel which in this case will be a static kernel (i.e. does not contain Dynamic Loadable Modules). The `strip(1)` command will be executed to remove the symbol and debug sections from the new kernel.

Applications and commands that use the library function `nlist(3E)` to get information from a kernel object file using a name list will generate errors or produce incomplete results if used with a kernel object file that has had its symbol and debug sections removed with the `strip(1)` command. The Console Debugger commands that use kernel symbols will not work if the kernel symbols are not present and loaded. In addition, the following commands will not work with a kernel built with this option:

```
arp(1M)           errdead(1M)   hwstat(1M)       intstat(1M)
ipcs(1)           ktrc(1M)     machine(1)       mpstat(1)
netstat(1M)       nfsstat(1M)  physconfig(1M)  savecore(1M)
sbextract(1M)    smtpd(1M)    sysdef(1M)      trpt(1M)
xntpd(1M)
```

`init(1M)` (with state 0) or `shutdown(1M)`, must be called to install the new kernel and then reboot.

Help Displays built-in help.
Return Returns to Main Menu.

Kernel Options Menu

Selecting `Options` from the Main Menu will cause `config` to enter the Options Menu (Screen 8-7). The operations available under this menu are described below.

```

PowerMAX_OS Kernel Configuration Utility
Options Menu

>> Directory      (examine/modify root directory)
Help              (help for this menu)
Return           (return to main menu)

Press <Enter> to select item,  Navigate with arrow keys, <tab> or item name
    
```

Screen 8-7. Kernel Options Menu

Directory Examine and/or modify the current object directory. The current object directory is displayed. You may specify a new object directory. See the *Object Directory* section in this chapter for more information.

Help Displays built-in help.

Return Returns to Main Menu.

Other Menu Functions

Other miscellaneous functions provided in the main menu, `Help`, `Shell` and `Quit` are basically self-explanatory:

Help	Displays built-in help
Shell	<code>config</code> will do a shell escape. The shell specified in the environment variable <code>SHELL</code> will be invoked. After the shell exits, operation will return to the <code>config</code> utility at the Main Menu screen.
Quit	Exit the <code>config</code> utility.

Minimizing Kernel Size

Configuring the Kernel

The Operating System is a highly configurable set of kernel modules. The kernel configuration supplied with your system contains tunable parameters and device drivers that are suitable for most systems. You must reconfigure the kernel if you want to alter any of the tunable parameters or disable modules to reduce the size of your kernel.

The **Master(4)** and **System(4)** directories (`/etc/conf/mdevice.d` and `/etc/conf/sdevice.d` by default) contain an entry for every separately configurable module in the system. Some modules are required in every system. These are noted by an **r** flag in the *characteristics* field of the **Master** file. Most modules are optional and can be selectively enabled and disabled to meet specific system requirements.

The kernel must be rebuilt after you change a tunable parameter, install/de-install kernel packages, enable/disable modules, or modify the hardware configuration.

Kernel configuration, or reconfiguration, may be performed using the kernel configuration utility `config(1M)`. This menu-driven utility provides a front-end interface to the kernel configuration tools, including `idbuild(1M)` and `idtune(1M)`.

Required Modules

Table 8-1 describes the modules that **must** be enabled in every kernel. It lists the modules required for each of the Night Hawk, PowerMAXION and Power Hawk platforms. These lists describe the absolute minimum set of kernel modules that will build and link together correctly, but not necessarily boot and run correctly. A kernel with only the required modules enabled is not a very useful system. It has no disk or networking modules enabled, and only one of the modules needed to configure a console is enabled

Table 8-1. Required Modules

Module	Description	Night Hawk	Power-MAXION	Motorola SBCs
conssw	Console indirect driver routines	R	R	R
elf	ELF program-type routines	R	R	R
fs	generic file system / I/O routines	R	R	R
io	generic I/O routines	R	R	R
isa	ISA Bus Controller I/O bridge interface	N	N	R
kernel	kernel “glue”	R	R	R
kma	kernel memory allocator	R	R	R
mem	memory management routines	R	R	R
memfs	memory-based file system	R	R	R
mm	memory driver for /dev/mem & /dev/kmem	R	R	R
pci	PCI I/O bridge	N	R	R
pin	interrupt configuration table/routines	R	R	N
proc	process management	R	R	R
specfs	special-file (device) file system	R	R	R
sum	super-user privilege module	R	R	R
svc	kernel services	R	R	R
sysclass	kernel daemon scheduling class	R	R	R
util	miscellaneous kernel utilities	R	R	R
vme	device independent VME support module	O	O	R

where:

R - module is REQUIRED on this platform

N - module is NOT SUPPORTED on this platform

O - module is OPTIONAL on this platform

Optional Modules

The **config** utility provides descriptions of all kernel modules provided on your system, organized by category.

When looking for information about a particular module with **config**, select the Modules main menu. You can also look at the **Master** or **System** files for a module

directly. Do **not** change values in these files directly. Either use the **config** utility or the **idtools** command to make your changes.

Some modules have dependencies to other modules. The dependencies between modules are contained in a module's **Master** file. To look at the dependencies for a module using **config**, select the **Describe** menu under the **Modules** main menu.

idconfig will generate warning messages during the kernel build if you enable a module, and the module(s) it depends on are not enabled or installed on the system.

Device Drivers

All modules in the **Device Driver** category are optional. Many of these driver modules are provided in separate packages. Installing a particular driver package also enables the associated kernel module. You must disable the module if you do not want to include it in your kernel configuration.

The number and types of device adapters (or controllers) will vary from system to system and will also vary over time on a specific system. The kernel builds data structures for each of the adapters found in the **Sadapters(4)** file (**/etc/conf/sadapters.d/kernel**) by default during the kernel build or system initialization, whether or not they will actually be used. Some of these structures are extremely large, as is the case with the **hsa** adapter. To keep the amount of space used for these structures to its minimum value, do **not** configure any adapters unless they will actually be used by your application. Remember to deconfigure both the device driver module and the **Sadapters** file entry using **config** when an adapter type is no longer being used.

File Systems

Most of the modules in the **File Systems** category are optional. The **memfs** and **specfs** modules are required for all systems.

The **fdfs** module must be enabled to support the files called file descriptor files (**/dev/fd/0**, **/dev/fd/1**, **/dev/fd/2** and so on). For convenience in referring to standard input, standard output, and standard error, an additional set of names is provided by the **ttymap(1M)** command: **/dev/stdin** for **/dev/fd/0**, **/dev/stdout** for **/dev/fd/1**, and **/dev/stderr** for **/dev/fd/2**. The **namefs** module provides file system operations on mounted file descriptors. It is used with **fdfs**.

To use a **ufs** file systems, you must enable the **sfs** module as well as the **ufs** module. The majority of the code for these two file system types is shared and resides in the **sfs** module.

To use a **xfs** file system you will need to enable the **xfs** and **xfsth** modules. The **xfsth** module initiates **xfs** threads and must always be enabled to use **xfs** file systems.

The NFS software can be configured in multiple ways to suit the needs of your machine. It can be configured as:

- a client only NFS system
- a server only NFS system

- both a client and server NFS system

There are four NFS modules:

- `nfss` - code and structures common to both client and server NFS, static portion
- `nfscmn` - code and structures common to both client and server NFS, can be a DLM module
- `nfs` - code and structures for client NFS, can be a DLM module
- `nfssrv` - code and structures for server NFS, can be a DLM module

The `nfss` and `nfscmn` modules must always be enabled to use NFS. At least one of the `nfs` or `nfssrv` modules must be enabled. When the NFS software is installed on a machine, it is configured by default as both a client and a server NFS machine (all four modules enabled).

The kernel lock manager module `klm` must be enabled for client NFS configurations, but it is not used by the server NFS module. Networking must be enabled for all NFS configurations.

Memory Management

The `kma` and `mem` modules are required for all systems. The `segdev` special device segment driver is not required, but must be included on any system that will use the `ipc` module.

Networking

All of the networking modules are optional. There are four modules that must always be enabled if you wish to use networking: `net`, `inet`, `ip`, and `icmp`. These modules must also be enabled if you want to use NFS.

The `asyh`, `ppp`, and `slip` modules do not need to be enabled unless you have a point-to-point serial communications network.

The *Networking Administrations* manual provides additional information about configuring TCP/IP networks and NFS. If you do not want to enable NFS, you can disable the `des`, `klm`, `krpc`, and `ktli` modules in addition to the `nfs` and `nfss` modules.

By default, the debug option for `tcp` is disabled (even when the module is enabled). The data structures created when the debug option is enabled add substantially to the size of the kernel. Small system configurations should not configure the debug option unless networking problems have been encountered and you are trying to identify the cause.

Miscellaneous

All of the modules in this category are optional. Many can be disabled unless your application needs to use a particular feature supported by these modules.

The **mod** and **modksym** modules are required for dynamic kernels (DLMs). **modksym** provides access to the kernel symbol table and is used by several kernel administrative commands that execute during multiuser system initialization. Therefore most system configurations will need **modksym** enabled. For more information on kernel symbols, refer to the **Kernel Symbols** section presented later in this chapter.

The inter-process communications module **ipc** provides support for shared memory, semaphores, and messages. The services it provides are used by **fsck(1M)** and must be enabled on most systems.

kdb and **kdb_util** should not be enabled unless you want to use the **kdb(1M)** debugger.

The **lockstat** module provides an interface to the kernel lock statistics mechanism. This module can be disabled if lock statistics are not required.

The **prf** module provides kernel profiling information via the special file **/dev/prf**. This module can be disabled unless site-specific applications access this device or use the **profiler(1M)** utilities.

The **trace** module should not be enabled on a small system configuration unless you will be using the NightTrace **ktrace(1)** and **ntrace(1)** commands. The trace data buffers consume a very large amount of memory.

Scheduling Classes

There are multiple scheduling classes available. All are optional except for **sysclass**. The **sysclass** scheduling class is reserved for kernel daemons and cannot be used for applications. You must choose at least one other scheduling class to enable.

The **ad** class should not be enabled unless you will be running Ada applications.

By default, the scheduling class for the **init** process is time-share (**TS**). If the **ts** module is not enabled, you must change the value of the **Scheduling** tunable **INITCLASS** to one of the class names you have enabled.

Security

All of the security modules are optional except one. Exactly one of the **lpm** or **sum** modules must be enabled on every system. The **sum** module should be enabled on nonsecure systems. It provides a traditional “super-user” style privilege module. The **lpm** module should be enabled on secure systems. It provides a “least privilege” model of permission checking. Access to files is controlled by fixed and inheritable privileges. Users are assigned roles which allows execution with a specified set of privileges.

Streams

All of the streams modules are optional. However, the **connld**, **fifofs**, **iaf**, **ldterm**, **log**, **sad**, and **ttcompat** modules must be enabled if your system will have a console, user terminals, or network terminals.

The **uni** module need not be enabled unless you have a site specific non-multithreaded device driver on a multiprocessor system. All of the device driver modules supplied with the system are multithreaded and do not need the **uni** module.

VxVM Volume Manager

The optional modules **plex**, **vol**, **voles**, and **volspec** should not be enabled unless the Veritas Volume Manager software is required for your application. These modules consume large amounts of memory both in terms of the sizes of the **Driver.o** and **space.o** components, and the amount of dynamic memory allocation they request during system initialization. Do not enable these modules unless the volume manager software will actually be used by your application. Note that these modules will be enabled by default if this software package has been installed on your system.

Tunable System Parameters

Tunable system parameters are used to define limits, enable features and declare initial sizes of various tables in the kernel. The tunables described in this section affect the size of some of the larger data structures in the kernel. The default values for these tunables are acceptable for most configurations and applications. However, if you are trying to build a small system configuration, you may want to reduce the values of these tunables. You may have to experiment with different combinations of parameter values to find an optimal set. To modify kernel parameters, the kernel will have to be re-configured, and the system re-booted.

Most of the data structures allocated in module `space.c` files are dependent upon the value(s) of tunables.

All data structure space allocated in `space.c` files is included in the static portion of the kernel load image. A data structure will be placed into either the `.rodata`, `.data` or `.bss` section of the kernel based on its access attributes (read-only or read/write) and whether the structure was initialized in the `space.c` file.

Some kernel data structures are dynamically allocated during system initialization. Many of these are dependent on the amount of physical memory on the system in addition to the value of a tunable.

KMA_PAGEOUT_POOL This tunable controls the size of a small pool of overflow memory for the pageout daemon. This will be used by the kernel memory allocator if no memory is available through the regular mechanisms, rather than blocking in the pageout context. The size of this pool is controlled by the tunable **KMA_PAGEOUT_POOL**. Increasing this value may result in reduced memory for allocation, and hence, system overhead. This value is in number of bytes. If the size of the tunable is less than **PAGESIZE** (0x1000 = 4096) bytes, the size will automatically be increased to **PAGESIZE** bytes. This will usually be adequate for small system configurations.

NAUTOPUSH	<p>This tunable specifies the maximum number of streams devices that can be configured to be autopushed. Each entry of this kernel data structure contains a list of modules to push, major/minor numbers of the streams device and flags. The autopush(1M) command is executed at system startup (from an entry in the /etc/inittab(4) file) to initialize the autopush data structures from information in the file etc/ap/chan.ap. Each line in the file generates one autopush structure. You should set the value of the NAUTOPUSH tunable to be the number of entries in this file, plus a few as spare in case entries are added after the system is booted. You will receive the error message “Ran out of autopush structures” from the autopush command if this tunable is too small.</p>
NBUF	<p>The file system uses buffers and buffer headers to perform buffered block I/O. When the file system needs to transfer a file system data structure it uses a buffer header. When a buffer header is needed, but no free ones are available, the system dynamically allocates more headers in chunks of NBUF at a time. There is no limit to the number of buffer headers in the system, but the autotunable BUFHWM limits the amount of memory that may be used for buffers.</p> <p>A lower value for NBUF will increase the number of times the memory allocator must be called to allocate buffer headers. This will result in the least amount of memory that is reserved for buffer headers (buffer header space is never freed for other uses).</p>
NCALL	<p>Each entry of the callout table represents a function to be invoked at a later time by the callout clock handler portion of the kernel. Callout entries are used for timers, timeouts on interfaces such as server_block(2), as well as timeouts within device drivers. The amount of space allocated is controlled by the tunable NCALL which is the number of callout table entries to allocate.</p>
NDQUOT	<p>The tunable NDQUOT defines the number of entries to allocate for the SFS/UFS file system quota table. There is one entry per userid per filesystem in use. This tunable should be set to a value more than the maximum number of users that can be logged onto the system. If quotas are in effect, the table entries limit the amount of disk space a user can use. This tunable can usually be decreased for a small configuration. If there are no available entries, the message “dquot table full” is printed on the console terminal. If this occurs, the value of this tunable should be increased.</p>
NHBUF	<p>This file system tunable specifies the size of the hash table used to locate a buffer, given a device number and a block number. This value should be between 1/8 and 1/4 of the total number of buffers available (specified by the</p>

autotunable **BUFHWM**) with 1/4 being the ideal value. This must be a power of 2.

NPGOUTBUF

The **NPGOUTBUF** tunable defines the number of I/O buffers allocated for the pageout daemon. A set of I/O buffers are reserved for use by the pageout daemon. On systems where paging activity is high (which can be found by the ‘**sar -g**’ command (**sar(1M)**)) the amount of system time required for the pageout daemon can be reduced by increasing this value. If making the size of the kernel smaller is of higher priority than paging performance, the value of this tunable can be reduced.

NUMAIO

The **NUMAIO** tunable specifies the number of control blocks pre-allocated for asynchronous I/O to raw disk partitions. A control block is used for each outstanding I/O operation. Having too few control blocks may result in increased I/O overhead. List I/O uses multiple aio structures, strung on a list. This tunable may be decreased if the number of simultaneous asynchronous I/O operations is less than the default value.

PUTBUFSZ

This system parameter specifies the size of a circular buffer (**putbuf**) that is used for kernel informational, warning and panic messages. **putbuf** will hold the last **PUTBUFSZ** characters written to the console terminal. The contents of **putbuf** can be viewed using **crash(1M)** or **kdb**. In general, this buffer should only be increased if the kernel is generating many console messages and this buffer is too small to hold all of the relevant messages.

SEGKMEM_BYTES and SEGKMEM_PERCENT

These two tunables are used to determine the amount of kernel virtual address space available for the dynamic kernel memory allocator. Several data structures are allocated to manage this virtual space. The amount of global memory and the size of the **SEGKMEM_BYTES** and **SEGKMEM_PERCENT** tunables will affect the size of these structures. To decrease the amount of kernel virtual address space by a fixed amount, decrease **SEGKMEM_BYTES**. To decrease it by an amount of available physical memory, decrease **SEGKMEM_PERCENT**.

SHMMNI

The tunable **SHMMNI** defines the maximum number of system wide shared memory identifiers. For each **SHMMNI** identifier declared, a set of 3 data structures will be allocated: Shared Memory Directory Entry, Kernel Shared Memory Identifier Structure, and Common IPC Access Control Structure. This tunable can be decreased if the number of simultaneous shared memory identifiers that will be in use at one time is less than the default value.

TR_BUFFER_COUNT

This tunable defines the number of kernel trace buffers. The trace driver contains a trace header and a number of trace buffers. The size of each of these trace data buffers is

48 KB. The more trace buffers configured, the less likely trace event data will be lost. However, each buffer added will consume a large amount of additional physical memory. This module should not be enabled unless a problem has been seen and needs to be traced or you will be using the NightTrace `ktrace(1)` or `ntrace(1)` utilities.

Minimum Bootable Configuration

Single User Mode (init S)

This section describes the minimal set of modules that you will need to build a small configuration that will boot to single user mode and support a basic set of commands. It provides a template for you to add the modules that will be used by your specific application. The hardware configuration that this system is configured for consists of one or more CPUs with memory, a console, a SCSI controller board, and a disk with SFS and UFS file systems (or disk with XFS file system). This single user minimal configuration does not support multiple user logins, networking or NFS.

Table 8-1, shown earlier, describes the modules required for the various platforms. Table 8-2 lists the modules that were added to the required list of modules to build the minimum bootable kernel configuration.

Table 8-2. Minimum System Additional Modules - Single User

Module	Description
cons	console terminal driver
fdfs	file descriptor file system
fifofs	fifo (pipe) file system
fp	fixed priority scheduler
gd	generic disk driver
gently	generic controlling tty driver
hsa	SCSI adapter interface driver (for Night Hawk)
ipc	inter-process communications driver
ldterm	standard line discipline driver
name	system information provider
nullzero	zero memory driver (/dev/null, /dev/zero)
sad	streams administrative driver
scsi	generic SCSI tables and routines
segdev	special device segment driver
serial	device independent serial port driver

Table 8-2. Minimum System Additional Modules - Single User

sfs	secure file system
sysmsg	system messages via kernel console
ttcompat	ioctl compatibility routines
ufs	unix file system
via	SCSI adapter interface driver (for PowerMAXION Motorola SBCs)
xfs	resilient file system
xfsd	distributed resilient file system

Multi User Mode (init 2)

In order to bring the minimal single user system described in the previous section up to multi user mode, five additional modules must be enabled. This system will support multiple user logins, but it does not support networking or NFS.

Table 8-3. Minimum System Additional Modules - Multi User

Module	Description
conlld	connection line discipline driver
errlog	system error logging driver
iaf	identification and authentication facility
log	log driver
modksym	dynamic kernel and symbol support routines

Kernel Symbols

There are two types of symbols that can be included with the kernel:

.symtab and **.strtab**

These symbol sections are produced by the compiler. They provide the ability to debug the kernel using the console debugger and provide the symbols used by the `nlist(3C)` library routine.

.unixsyms

This symbol section is produced by the `unixsyms` command. They are optionally requested by configuring either of the `modksym` or `kdb` modules. The `.unixsyms` symbol section is required for dynamically loadable modules (a dynamic kernel). These symbols are used by the `getksym(2)` system call and allow the use of symbol names when performing I/O operations on

`/dev/kmem` kernel memory (using `ioctl(2)` system calls (`kmem(7)`)). In addition they provide the ability to debug the kernel using `kdb`.

The `.unixsyms` symbol section should be configured for most systems. The `getksym` and the `/dev/kmem ioctl` services are used by a significant number of commands including several that are executed when changing to the multi user `init 2` or `init 3` states. The default `/etc/inittab` entries for single user mode system initialization (`sysinit`) do not require the `.unixsyms` section.

By default, the kernel is built with both types of symbols included.

Loading Symbols

The `boot(8)` loader provides the ability to optionally load symbols with the kernel using the `boot` processor register options. This option only affects the `.symtab` and `.strtab` symbol sections generated by the compiler. The value of this register can be set using the console debugger command `pboot` (`load symbols = 0x80`).

Having the `.symtab` and `.strtab` symbol sections present in the kernel object file provides the ability to use the console debugger for debugging if a problem should arise. Symbols should be loaded for kernels that are being tested. After a kernel has been tested, a significant amount of kernel memory can be saved by reloading the system with the option to load symbols reset.

The `.unixsyms` section will always be loaded if it is present in the kernel object file.

Removing Symbols from the Kernel Object File

The amount of memory required to load the kernel is no different whether the `.symtab` and `.strtab` symbol sections are not present on disk, or whether they were present but not loaded. Embedded kernels, loaded from prom or systems where disk space is at a premium, may want to reduce the size of the kernel object file by removing these symbols.

The methods described below remove the `.symtab` and `.strtab` symbol sections from a kernel object file on disk.

- `config(1M)`: select the **Strip Symbols** option under the Rebuild Kernel menu. This option will build a static kernel and will then execute the `strip(1)` command to remove the `.symtab` and `.strtab` symbol sections.
- `idbuild(1M)`: request the `-s` option. This option will build a static kernel and will then execute the `strip` command to remove the `.symtab` and `.strtab` symbol sections.
- `strip(1)`: This command can be called directly by the user to remove the `.symtab` and `.strtab` symbol sections from a previously generated static or dynamic kernel object file.

Applications and commands that use the library function `nlist` to get information from a kernel object file using a name list will generate errors or produce incomplete results if used with a kernel object file that has had its `.symtab` and `.strtab` symbol sections

removed with the `strip` command. The following commands will not work with a stripped kernel:

<code>arp(1M)</code>	<code>errdead(1M)</code>	<code>hwstat(1M)</code>
<code>intstat(1M)</code>	<code>ipcs(1)</code>	<code>ktrc(1M)</code>
<code>machine(1)</code>	<code>mpstat(1)</code>	<code>netstat(1M)</code>
<code>nfsstat(1M)</code>	<code>physconfig(1M)</code>	<code>savecore(1M)</code>
<code>sbextract(1M)</code>	<code>smtpd(1M)</code>	<code>sysdef(1M)</code>
<code>trpt(1M)</code>	<code>xntpd(1M)</code>	

Process Scheduling

Introduction	9-1
Overview of the Process Scheduler	9-2
Time-Sharing Class and the Fixed Class	9-4
System Class	9-5
Fixed Priority Class	9-5
Configuring the Scheduler	9-5
Default Global Priorities	9-6
Tunable Parameters	9-7
Fixed Priority Parameter Table fp_dptbl	9-8
Fixed Class Parameter Table fc_dptbl	9-9
Kernel-Mode Parameter Table ts_kmdpris	9-9
Changing Scheduler Configuration	9-10
Procedure	9-10
Removing a Scheduler Class	9-11
Procedure	9-11
Installing a Time-Sharing Scheduler Class	9-11
Procedure	9-11
Installing a Fixed Class Scheduler Class	9-12
Procedure	9-12
Installing a Fixed Priority Scheduler Class	9-12
Procedure	9-12
Changing Scheduler Parameters with dispadmin	9-13
Managing Processors and Processes	9-15
Processor Administration Information	9-15
Binding Processes to Processors	9-15

Introduction

For the Operating System (OS), the schedulable entity is always a lightweight process (LWP). Scheduling priorities and classes are attributes of LWPs and not processes. When scheduling system calls accept a process on which to operate, the operation is applied to each LWP in the process. The operating system scheduler determines when LWPs run. It maintains priorities based on configuration parameters, process behavior, and user requests and it uses these priorities in conjunction with the processor bias to assign LWPs to the CPU.

The operating system gives users absolute control over the sequence in which certain LWPs run, and the amount of time each LWP may use the CPU before another LWP gets a chance.

By default, the scheduler uses the time-sharing class. The time-sharing class adjusts priorities dynamically in an attempt to provide good response time to interactive LWPs and good throughput to CPU-intensive LWPs.

The scheduler offers a fixed priority scheduling class as well as a time-sharing class. Fixed priority scheduling allows users to set fixed priorities on a per-process or LWP basis. In the default configuration, the highest-priority fixed priority LWP always gets the CPU as soon as it is runnable, even if system processes are runnable. An application can therefore specify the exact order in which LWPs run. An application may also be written so that its fixed priority LWPs have a guaranteed response time from the system.

For system environments in which real-time performance is not required, the default scheduler configuration works well and no fixed priority LWPs are needed: administrators should not change configuration parameters and users should not change scheduler properties of their applications. However, for real-time applications or applications with strict timing constraints, fixed priority LWPs are the only way to guarantee that the application's requirements are met.

NOTE

Fixed priority LWPs used carelessly can have a dramatic negative effect on the performance of time-sharing LWPs.

Administrators can manage the relationship between processes and processors. Processes can be bound to a specific processor or set of processors.

This chapter is addressed to system administrators who need to adjust the default configuration for the scheduler. Administrators should understand the scheduler for these reasons:

- The scheduler has an overriding effect on the performance and perceived performance of a system. The default scheduler is tuned to perform well in representative work environments, but you must understand how it operates to know whether you can reconfigure it to better suit local needs.
- A bug in a fixed priority program or a malicious fixed priority user can lock out all other processing, including kernel processing. Users need privilege to create fixed priority processes, and presumably only trustworthy users have this privilege. But administrators still should be aware that the scheduler functions introduce new ways to cause trouble, and should be prepared for accidents and for misuse of these functions.

See the “*Process Scheduling and Management*” chapter in the *Real-Time Guide* for explanations of the user commands and function calls that control process and LWP scheduling.

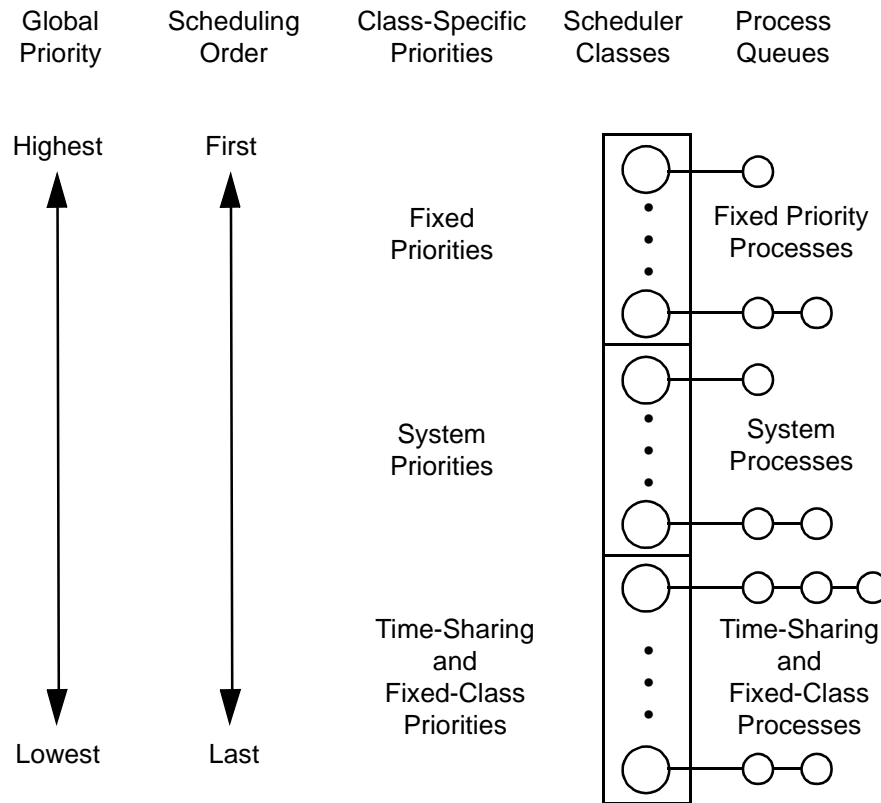
The rest of this chapter is organized as follows:

- The “*Overview of the Process Scheduler*” tells what the scheduler does and how it does it. It also introduces scheduler classes.
- “*Configuring the Scheduler*” describes how you can control the scheduler using tunable parameters and the scheduler parameter tables: **ts_dptbl(4)** for time-sharing parameters, **fp_dptbl(4)** for fixed priority parameters, and **fc_dptbl(4)** for fixed class parameters.
- “*Changing Scheduler Parameters with dispadmin*” tells how to display or change scheduler parameters in a running system. **dispadmin** changes do not survive a reboot. To make permanent changes in scheduler configuration, use **idtune(1M)** to change the scheduler parameter tables in the `/etc/conf/mtune.d` files.
- “*Managing Processors and Processes*” describes how you can manage the relationship between processors and processes. It tells how you can bind processes to processors.

Overview of the Process Scheduler

Figure 9-1 shows how the process scheduler works. Fixed Class Priorities overlap the default Time-Sharing Priorities.

When a process or LWP is created, it inherits its scheduler parameters, including scheduler class and a priority within that class. A process or LWP changes class only as a result of a user request. The system manages the priority of an LWP based on user requests and the scheduler class of the LWP.



163690

Figure 9-1. The Process Scheduler

In the default configuration, the initialization process belongs to the time-sharing class. Because processes inherit their scheduler parameters, all user login shells begin as time-sharing processes in the default configuration.

The scheduler converts class-specific priorities into global priorities. The global priority of an LWP determines when it runs—the scheduler always runs the runnable LWP with highest global priority. Numerically higher priorities run first. Once the scheduler assigns an LWP to the CPU, the LWP runs until it uses up its time slice, sleeps, or is preempted by a higher-priority LWP. LWPs with the same priority run round-robin.

Administrators specify default time slices in the configuration tables, but users may assign time slices to fixed priority LWPs.

You can display the global priority of a process or LWP with the `-cl` options of the **ps(1)** command. You can display configuration information about class-specific priorities with the **priocntl(1)** command and the **dispadmin(1M)** command.

By default, all fixed priority processes or LWPs have higher priorities than any kernel process, and all kernel processes have higher priorities than any time-sharing process.

NOTE

As long as there is a runnable fixed priority process or LWP for a given processor, no kernel process and no time-sharing process runs on that processor.

The following sections describe the default classes.

Time-Sharing Class and the Fixed Class

The goal of the time-sharing class is to provide good response time to interactive processes and LWPs and good throughput to CPU-bound processes and LWPs. The scheduler switches CPU allocation frequently enough to provide good response time, but not so frequently that it spends too much time doing the switching. Time slices are typically a few hundred milliseconds.

Previously, processes were selected for swapping in and out based primarily on their scheduling priority. Now, swapping out is based on a combination of factors:

- the length of time in memory since process birth or last swap in where greater time results in higher preference
- the resident set size (`rss`) of the process where the larger `rss` results in higher preference
- the minimum duration for which all LWPs in the process have been idle, stopped, or uninterruptibly blocked where greater duration results in increased preference.

Swapping in decisions are affected by these factors:

- the maximum scheduling priority of all LWPs within a process where higher priority results in greater preference
- the length of time the process has been swapped out where a greater time results in higher preference
- the `rss` of the process at the time of the last swap out where the larger `rss` results in reduced preference
- the minimum duration for which the LWPs in the process have been idle, stopped, or uninterruptibly blocked where greater duration results in increased preference.

The time-sharing class changes priorities dynamically and assigns time slices of different lengths. The scheduler raises the priority of an LWP that sleeps after only a little CPU use (an LWP sleeps, for example, when it starts an I/O operation such as a terminal read or a disk read); frequent sleeps are characteristic of interactive tasks such as editing and running simple shell commands. On the other hand, the time-sharing class lowers the priority of an LWP that uses the CPU for long periods without sleeping.

The default time-sharing class gives larger time slices to LWPs with lower priorities. An LWP with a low priority is likely to be CPU-bound. Other LWPs get the CPU first, but

when a low-priority LWP finally gets the CPU, it gets a bigger chunk of time. If a higher-priority LWP becomes runnable during a time slice, however, it preempts the running process or LWP.

The scheduler manages time-sharing processes and LWPs using configurable parameters in the time-sharing parameter table `ts_dptbl`. This table contains information specific to the time-sharing class.

The default fixed class is similar to the default time-sharing class except that the priorities and time slices given to fixed class processes or LWPs do not degrade over time. The `fc_dptbl` parameter table contains information specific to the fixed class.

System Class

The system class uses fixed priorities to run kernel processes such as servers and house-keeping processes like the paging demon. The system class is reserved for use by the kernel; users may neither add nor remove a process from the system class. Priorities for system class processes are set up in the kernel code for those processes; once established, the priorities of system processes do not change. (User processes running in kernel mode are not in the system class.)

Fixed Priority Class

With the fixed priority class, critical processes or LWPs can run in predetermined order. Fixed priorities never change except when a user requests a change. Contrast this fixed-priority class with the time-sharing class in which the system changes priorities to provide good interactive response time.

Privileged users can use the `run`, `rerun`, and `prIOCntl` commands or the `sched_set` scheduler and `prIOCntl` program interfaces to assign fixed priorities.

The scheduler manages fixed priority processes or LWPs using configurable parameters in the fixed priority parameter table `fp_dptbl`. This table contains information specific to the fixed priority class.

Configuring the Scheduler

The default configuration includes both the time-sharing and the fixed priority scheduler classes. The time-sharing class is tuned for representative operating system workloads. Such workloads have a high proportion of interactive processes, which sleep early and often. The fixed priority class is configured for applications that need it.

Systems used for real-time applications will require the fixed priority scheduling class to guarantee that the applications' timing deadlines are met.

For traditional time-sharing uses such as software development, office applications, and document production, fixed priority processes may be unnecessary. In addition, they may be undesirable. First, they consume memory that cannot be paged: the u-blocks of fixed priority processes are never paged out. Second, they introduce new ways to cause performance problems: a high-priority fixed priority process can block out all other processing. In a computing environment where only time-sharing is needed, you may want to remove the fixed priority scheduler class from your configuration, as described below in the section “*Changing Scheduler Configuration.*”

NOTE

Fixed priority processes can have a dramatic negative effect on time-sharing performance.

This section describes the parameters and tables that control scheduler configuration, and tells how to reconfigure the scheduler. A basic assumption is that your workload is reasonable for your system resources, such as CPU power, primary memory, and I/O capacity. If your workload does too much computation or too much I/O for your hardware, reconfiguring the scheduler won't help. See the “Managing System Performance” chapter for more information.

Default Global Priorities

The following table shows the scheduling order and global priorities for each scheduler class.

Scheduling Order	Global Priority	Scheduler Class
first	159 . . 100	Fixed Priority
	99 . . 60	System
	59 . . 0	Time-Sharing
last		

When your system is built, it constructs this information from the tunable parameters and scheduler parameter tables described in the following sections. Although you are not

forced to configure scheduler classes to produce consecutive, non-overlapping global priorities like the default priorities, we recommend that you do so for the sake of simplicity. Likewise, we recommend that you make all fixed priority global priorities greater than the global priorities of all other classes. These conventions simplify scheduler configuration, and they should be able to accommodate any requirements on the scheduler.

Kernel processes such as the swapper and the paging demon run in the system scheduler class. Kernel processes must compete with user processes for CPU time, and in the default configuration all fixed priority processes have higher priorities than all system processes. Therefore, fixed priority applications must be written carefully to ensure that the kernel gets the processing time it needs. Also, if you reconfigure the scheduler, make sure that the system class gets enough priority over the time-sharing class to give kernel processes the CPU time they need.

Tunable Parameters

This section describes the tunable parameters that control scheduler configuration. These parameters are specified in the `/etc/conf/mtune.d` files.

The following parameters are specified in the `kernel` file:

- **MAXCLSYSPRI** is the maximum global priority of processes in the system class. When the kernel starts system processes, it assigns their priorities using **MAXCLSYSPRI** as a reference point.

NOTE

MAXCLSYSPRI must be 39 or greater, because the kernel assumes it has at least that great a range of priorities below **MAXCLSYSPRI**. If you request a **MAXCLSYSPRI** below 39, it is changed to 39.

- The most important system processes get global priorities at or near **MAXCLSYSPRI**; the least important system processes get global priorities at or near **(MAXCLSYSPRI - 39)**. The default value of **MAXCLSYSPRI** is 100, which gives all system processes higher priorities than all user processes.
- **INITCLASS** is the scheduler class assigned to the `init` process. This scheduler class is inherited by all descendants of `init`, which normally include all user login shells. By default, **INITCLASS** is `TS`; that is, all login shells are time-sharing processes in the default configuration. **INITCLASS** is defined in `/etc/conf/patch.d/proc/space.c`.

The following parameters are specified in the `ts` file, which controls the time-sharing policy:

- **TSMAXUPRI** specifies the range within which users may adjust the priority of a time-sharing process using the `prIOCNTL` system call: the valid range is **-TSMAXUPRI** to **+TSMAXUPRI**. The default value of **TSMAXUPRI** is 20. (Configuring a value of 20 emulates the behavior of the older, less gen-

eral scheduler interfaces **nice** and **setpriority**, which continue to work as in previous releases.)

The value of **TSMAXUPRI** is independent of the configured number of global time-sharing priorities, though we recommend configuring at least 40 time-sharing priorities, as explained below in the section on **ts_dptbl**. In the default configuration, there are 60 time-sharing priorities. The system may use the remaining priorities depending on process behavior.

Fixed Priority Parameter Table **fp_dptbl**

The scheduler uses **fp_dptbl (4)**, the fixed priority scheduler (or dispatcher) parameter table, to manage fixed priority processes. A default version of **fp_dptbl** is delivered with the system, and an administrator may change it to suit local needs. **fp_dptbl** is specified in the **space.c** file in the **/etc/conf/pack.d/fp** directory. It is built into the kernel as part of system configuration if the **/etc/conf/sdevice.d/fp** file has a “Y” in the second column:

```
fp Y 1 0 . . . .
```

You may adjust the size and values of **fp_dptbl** depending on the applications on your system. Here is part of a simple **fp_dptbl**:

fp_glbpri	fp_qntm
100,	10,
101,	10,
102,	10,
103,	10,
104,	10,
105,	10

- The **fp_glbpri** column contains global priorities (the priorities that determine when an LWP runs). Higher numbers run first.
- The **fp_qntm** column contains the default time slice (or quantum) associated with the priority in the **fp_glbpri** column; this is the maximum amount of time an LWP with this priority may use the CPU before the scheduler gives another LWP a chance. This time slice is specified in clock ticks. (The system clock ticks HZ times per second, where HZ is a hardware-dependent constant defined in the **param_p.h** header file.)

The highest priority specified in this table is 105, so LWPs with priority 105 always run before any other LWPs. If it does not sleep, an LWP with priority 105 runs for 10 clock ticks before the scheduler looks for another LWP to run. (Because 105 is the highest priority, an LWP at this priority would be preempted after its time slice only if there were another LWP with priority 105.) LWPs at priority 104 run for 20 clock ticks, and so on.

The lowest fixed priority specified in this table is 100; a process with priority 100 runs for 10 clock ticks.

The default priority in the fixed priority class is the lowest priority configured in `fp_dptbl`. This is the priority assigned to an LWP if it is changed to a fixed priority LWP and no priority is specified. This is also the priority assigned to the `init` process and all its children if `INITCLASS` is set to `FP`.

Although `fp_dptbl` contains default time slices for fixed priorities, users with the appropriate privilege can set fixed priorities and time slices independently. Users can specify any time slice they want for a fixed priority LWP, including an infinite time slice. The system assumes that fixed priority LWPs voluntarily give up the CPU so other work can get done.

Fixed Class Parameter Table `fc_dptbl`

The scheduler uses `fc_dptbl(4)`, the fixed class scheduler (or dispatcher) parameter table, to manage fixed class LWPs. A default version of `fc_dptbl` is delivered with the system, and an administrator may change it to suit local needs. Save a backup of the default version of `fc_dptbl`. `fc_dptbl` is specified in the `space.c` file in the `/etc/conf/pack.d/fc` directory. It is automatically built into the kernel as part of system configuration.

You may change the size and values of `fc_dptbl` depending on your local needs, but only experienced administrators should make such changes. The default values have a long history of good performance over a wide range of environments. Changing the values is not likely to help much, and inappropriate values can have a dramatic negative effect on system performance.

The `fc_dptbl` has only two columns, one for global priorities and the other for the time slice (or quantum). For a discussion of the table entries, see the above discussion for the Time-Sharing Parameter Table.

Kernel-Mode Parameter Table `ts_kmdpris`

The scheduler uses the kernel-mode parameter table `ts_kmdpris` to manage sleeping time-sharing processes. A default version of `ts_kmdpris` is delivered with the system, and there is seldom a reason to change it. `ts_kmdpris` is specified in the `space.c` file in the `/etc/conf/pack.d/ts` directory. It is automatically built into the kernel as part of system configuration.

NOTE

The kernel assumes that it has at least 40 priorities in `ts_kmdpris`. It panics if it does not.

The kernel-mode parameter table is a one-dimensional array of global priorities. The kernel assigns these priorities to sleeping processes based on their reasons for sleeping. If a

user process sleeps because it is waiting for an important resource, such as an inode, it sleeps at a priority near the high end of the `ts_kmdpris` priorities, so that it may get and free the resource quickly when the resource becomes available. If a user process sleeps for a less important reason, such as a wait for terminal input, it sleeps at a priority near the low end of the `ts_kmdpris` priorities.

The default kernel-mode parameter table is simply a one-dimensional array of the integers from 50 through 99, which means that time-sharing processes sleep at priorities between the default real-time priorities fixed priorities and the default time-sharing priorities.

In the default configuration, the priorities in `ts_kmdpris` happen to be exactly the same as the priorities used by system class processes, because the tunable parameter **MAXCLSPRI** is the same as the highest priority in `ts_kmdpris`. This overlap is designed to be consistent with the scheduler behavior of previous releases of the operating system, because these priorities produce good performance in most environments. But the overlap is not necessary. The scheduler introduces a logical separation between the priorities of system processes and sleeping time-sharing processes; an administrator may configure a machine so that the two sets of processes have different ranges of global priorities.

Changing Scheduler Configuration

Changing scheduler configuration requires changing one or more of the tunable parameters or the configuration tables `fp_dptbl`, `ts_dptbl`, `fc_dptbl`, and `ts_kmdpris`. Changes made in this way are permanent. This is the only way to change the size of the configuration tables.

NOTE

The **config(1M)** utility has been changed to include a user-friendly interface for modifying the configuration tables `p_dptbl`, `ts_dptbl` and `fc_dptb`. Refer to Chapter 8 Configuring and Building the Kernel, and manual page **config(1M)**, for more information.

Procedure

Use the following procedure to change the scheduler configuration:

1. Identify the appropriate tunable file in the `/etc/conf/mtune.d` directory or the appropriate configuration table in the `/etc/conf/pack.d` directory.
2. Change the tunable parameter(s)
3. Rebuild the kernel.

Changes made in this way are permanent. This is the only way to change the size of the configuration tables.

See the section below on the **dispadm** command for a way to make a temporary change on a running system.

Removing a Scheduler Class

For systems that do not need fixed priority processes, it may make sense to remove the fixed priority class, thereby making it impossible to create fixed priority processes. By not having fixed priority processes, you avoid their non-pageable u-blocks and you avoid the possibility of a runaway process monopolizing the machine.

Procedure

Use the following procedure to remove the fixed priority scheduler class:

1. In the file `/etc/conf/sdevice.d/fp` change the line

```
fp Y 1 0 . . . .
```

so that there is an “N” in the second column:

```
fp N 1 0 . . . .
```

2. Rebuild the kernel.

Installing a Time-Sharing Scheduler Class

By default, the time-sharing, the fixed class, and the fixed priority scheduler class is installed. Therefore, you need to install this class only if you first remove it.

Procedure

Use the following procedure to install the time-sharing class:

1. Ensure that the `Driver.o` module is in the `/etc/conf/pack.d/ts` directory.
2. In the `/etc/conf/sdevice.d/ts` file, edit the line

```
ts N . . .
```

to

```
ts Y . . .
```

3. (The module is now automatically configured unless it is explicitly excluded, that is, by having N in the second column.)
4. Verify that the value of `INITCLASS` in `/etc/conf/proc/space.c` to make sure that your configuration is assigning the appropriate default scheduler class.
5. Build the kernel.

Installing a Fixed Class Scheduler Class

By default, the fixed class scheduler class is installed. Therefore, you need to install this class only if you first remove it.

Procedure

Use the following procedure to install the fixed class:

1. Ensure that the **Driver.o** module is in the **/etc/conf/pack.d/fc** directory.

2. In the **/etc/conf/sdevice.d/fc** file, edit the line

```
fc N . . .
```

to

```
fc Y . . .
```

3. (The module is automatically configured unless it is explicitly excluded, that is, by having N in the second column.)

4. Verify that the value of **INITCLASS** in **/etc/conf/proc/space.c** to make sure that your configuration is assigning the appropriate default scheduler class.

5. Build the kernel.

Installing a Fixed Priority Scheduler Class

By default fixed priority scheduler class is installed. Therefore, you need to install this class only if you first remove it.

Procedure

Use the following procedure to install the fixed priority class:

1. Make sure that the **Driver.o** module is in the **/etc/conf/pack.d/fp** directory.

2. In the **/etc/conf/sdevice.d/fp** file, edit the line

```
fp N . . .
```

to

```
fp Y ...
```

(The module is not configured unless it is explicitly included.)

3. Verify that the value of **INITCLASS** in `/etc/conf/pack.d/proc/space.c` to make sure that your configuration is assigning the appropriate default scheduler class.
4. Build the kernel.

Changing Scheduler Parameters with `dispadmin`

The `dispadmin(1M)` command allows you to change or retrieve scheduler information in a running system. Changes made using `dispadmin` do not survive a reboot. To make permanent configuration changes, you must change the scheduler parameter tables in the `/etc/conf/pack.d` directory as described in the previous section on configuration. However, you can use `dispadmin` to get an effect equivalent to changing configuration tables by calling `dispadmin` from a startup script that changes the configuration automatically at boot time.

The `dispadmin` command has three forms:

- `dispadmin -l` lists the configured scheduler classes.
- `dispadmin -g [-r res] -c class` gets scheduler parameters for the specified class. By default, time slices are printed in milliseconds. You may optionally retrieve time slices at a resolution specified by the `-r` option.
- `dispadmin -s config_file -c class` sets scheduler parameters for the specified class from `config_file` (your configuration file).

Here is the output of the `-l` option for the default configuration.

```
$ dispadmin -l
CONFIGURED CLASSES
=====
SYS      (System Class)
FC       (Time Sharing)
TS       (Time Sharing)
FP       (Fixed Priority)
```

The `-g` option gets current scheduler parameters for the specified class and writes them to the standard output. Scheduler parameters are class specific. The parameters for the default classes are described in the previous sections about the scheduler parameter tables; `ts_dptbl` holds time-sharing parameters, `fc_dptbl` holds fixed class parameters, and `fp_dptbl` hold fixed priority parameters.

The following screen shows part of the output of `dispadmin -g` for the fixed priority class:

```
$ dispadmin -c FP -g # list fixed priority parameters
# Fixed Priority Dispatcher Configuration
RES=1000

# TIME QUANTUM          PRIORITY
# (fp_quantum)         LEVEL
1000 # 0
1000 # 1
1000 # 2
1000 # 3
1000 # 4
1000 # 5
1000 # 6
1000 # 7
1000 # 8
1000 # 9
800 # 10
800 # 11
800 # 12
800 # 13
800 # 14
800 # 15
800 # 16
800 # 17
800 # 18
800 # 19
... # ...
100 # 50
```

The following screen shows part of the output of `dispadmin -g` for the time-sharing class:

```
$ dispadmin -c TS -g # list time-sharing parameters
# Time Sharing Dispatcher Configuration
RES=1000

# ts_quantum ts_tqexp ts_slpret ts_maxwait ts_lwait PRIORITY LEVEL
1000 0 10 5 10 # 0
1000 0 11 5 11 # 1
1000 1 12 5 12 # 2
1000 1 13 5 13 # 3
1000 2 14 5 14 # 4
1000 2 15 5 15 # 5
1000 3 16 5 16 # 6
1000 3 17 5 17 # 7
1000 4 18 5 18 # 8
1000 4 19 5 19 # 9
800 5 20 5 20 # 10
800 5 21 5 21 # 11
800 6 22 5 22 # 12
800 6 23 5 23 # 13
800 7 24 5 24 # 14
800 7 25 5 25 # 15
800 8 26 5 26 # 16
800 8 27 5 27 # 17
800 9 28 5 28 # 18
800 9 29 5 29 # 19
... ... ... ... # ...
100 40 55 5 55 # 50
```

By default, **dispadmin** reports time slices in milliseconds. If you specify the **-r res** option, **dispadmin** reports time slices in units of *res* intervals per second. For example, a *res* of 1000000 reports time slices in microseconds (millionths of a second).

The **-s config_file** option uses *config_file* to set scheduler parameters for the specified class. The configuration file must be in the class-specific format produced by the **-g** option. The meanings of the parameters are described in the previous sections about the scheduler parameter tables; **ts_dptbl** holds time-sharing parameters, **fc_dptbl** holds fixed class parameters, and **fp_dptbl** holds fixed priority parameters.

The following examples show how to set the parameters for the default classes as specified in the configuration files **fp_config**, **fc_config**, and **ts_config**. The examples presuppose that these files are in the correct formats.

```
$ dispadmin -c RT -s fp_config # set real-time parameters
$ dispadmin -c FP -s fp_config # set fixed priority parameters
$ dispadmin -c FC -s fc_config # set fixed class parameters
$ dispadmin -c TS -s ts_config # set time-sharing parameters
```

The files that specify the new scheduler parameters must have the same number of priority levels as the current table that is being overwritten. To change the number of priority levels, you must change the configuration file **/etc/conf/pack.d/ts/space.c** or **/etc/conf/pack.d/fc/space.c** or **/etc/conf/pack.d/fp/space.c** as described in the previous section about configuration.

Managing Processors and Processes

Processor Administration Information

Processors are identified with a processor ID number that gives them a unique tag within the system. The state of the processors in your system can be examined by using the **psrinfo(1M)** command or the **processor_info(2)** system call. They report which processors are on-line/available.

Binding Processes to Processors

By default, an LWP can execute on any processor in the system. You can use the **run(1)**, **rerun(1)**, and **pbind(1M)** commands, **mpadvise(3C)** library routine, or the **processor_bind(2)** system call to bind a process (all of the associated LWPs) or LWP to a processor. This restricts the LWPs to execute only on the specified processor. Once bound, all child processes created by this process or timeout routines requested by this process are bound by default to the same processor. The **-P** option of the **ps(1)** command can be used to display the processor binding status of all processes in the system. The **pbind(1M)** command can also be used to unbind a process (all the associated LWPs) or LWP from a processor. Once unbound, the process or LWP can execute on any processor in the system.

Backup and Restore Services

Replace with Part 3 tab for 0890430

Part 3 - Backup and Restore Services

Part 3 Backup and Restore Services

Note

The Backup and Restore services are supported by PowerMAX OS release 4.2 and earlier. This service is **not** available with any PowerMAX OS release after release 4.2.

Chapter 10 Archiving and Restoring Data 10-1
Chapter 11 Backup and Restore Services 11-1

Archiving and Restoring Data

What Is A Backup Operation?	10-1
Archiving and Restoring Data On Your System	10-1
Comparison of fsdump/fsrestore & backup/restore	10-2
Tape Archiving Using tar	10-3
Making A Tape Archive	10-4
Create A New Tape Using Full Pathname(s)	10-6
Create A New Tape Using Partial Pathname(s)	10-6
Listing The Contents Of A Tape Archive	10-6
Restoring Files From A Tape Archive	10-7
Restore A Complete tar Tape with a Full Pathname	10-7
Restore Specific Files from a tar Tape with a Full Pathname	10-8
Restore a Complete tar Tape with a Partial Pathname	10-8
Restore Specific Files From tar Tape with a Partial Pathname	10-8
Tape Archiving Using cpio	10-8
Archiving and Restoring Files Using fsdump/fsrestore and xfdump/xfsrestore	10-10
Daily And Weekly Backups	10-10
Backing Up Selected System Files	10-11
The fsdump and xfdump Commands	10-11
Making the Daily Backup Tape	10-12
Making the Weekly Backup Tape	10-13
Monthly Backups	10-14
Saving root	10-14
Restoring Files	10-14
Restoring an Entire File System	10-15
Restoring an Individual File	10-16
Restoring an Individual Directory	10-16
Using the Interactive Restore	10-17
Restoring the root Partition	10-17
Additional Features of fsdump/fsrestore and xfdump/xfsrestore	10-17
File System Restore Utility	10-21
Introduction	10-21
Booting Up Procedure	10-22
Night Hawk Systems	10-22
Power Hawk Systems	10-22
File System Restore Operation	10-23

What Is A Backup Operation?

NOTE

If you are running your system in compliance with the security criteria described in the “Security Administration” part of *System Administration*, Volume 1, the `fsdump` and `xfsdump` commands will not be available for use in multi-user state. See the chapter “Trusted Backup and Restore” in this book for information on backing up and restoring files with Mandatory Access Control levels. See the “Security Administration” part of *System Administration*, Volume 1 for a list of the software packages that should be installed on the secure system, and the facilities (such as the Backup Service) that are available in single-user mode only.

Note

The Backup and Restore services are supported by PowerMAX OS release 4.2 and earlier. This service is **not** available with any PowerMAX OS release after release 4.2.

What's so important about backups? Ask the employee who accidentally removed the file containing the speech to be delivered (tomorrow) to the senior management team, or the student who somehow clobbered the term paper it took five weeks to write.

Backup operations allow you to copy the data on your system to another storage medium, such as DAT or cartridge tapes (archiving). When the need arises, you can recover (restore) information that may have been lost as a result of human or mechanical error.

Archiving and Restoring Data On Your System

There are several means of archiving and restoring data on your system.

- `cpio` and `tar` - basic UNIX system utilities for archiving files. Use `cpio` regularly to archive your root filesystem so that you can restore it, either partially or fully, if it becomes corrupted. `cpio` is also especially suitable for transferring files between different types of UNIX machines.

- **fsdump** and **fsrestore** - utilities for archiving **ufs** and **sfs** file systems.
- **xfsdump** and **xfsrestore** - utilities for archiving **xfs** (and **xfsd**) file systems.
- Basic **backup** and **restore** - a simple system of archiving system files and user directories. (Refer to Chapter 11 for further details.)
- Extended **backup** and **restore** - a comprehensive set of utilities for scheduling and archiving files, file systems, partitions, and whole disks of data. (Refer to Chapter 11 for further details.)

The `tar` archive (**tar**) command and the `copy` file archives `in` and `out` (**cpio**) command allow you to copy individual files or directories to magnetic tape and restore them when necessary.

The first part of this chapter describes how to create **tar** and **cpio** tapes and extract information from them. The second part explains the recommended procedures for backing up and restoring files on a regular basis using **fsdump** and **fsrestore** or **xfsdump** and **xfsrestore**.

Comparison of **fsdump/fsrestore** & **backup/restore**

As system administrator you must decide whether to use the **fsdump/fsrestore** and **xfsdump/xfsrestore** sets of commands or the **backup/restore** set of commands for your system backups. The backup tapes created by **fsdump** can be restored using **fsrestore** or **xfsrestore**. The backup tapes created by **xfsdump** can only be restored using **xfsrestore**. They are not compatible with **backup/restore** and vice versa.

Each set of commands provides the ability to backup and restore your disks, however each contain capabilities that are not supported by the other. Each support full and incremental backups. None of the sets support a trusted backup operation. For information about trusted backup and restore operations, see man pages **tcpio(1M)** and **rtcpio(1M)**. You must evaluate each set of commands to determine which provides the best set of capabilities for your environment.

The **fsdump/fsrestore** and **xfsdump/xfsrestore** sets of commands are functionally compatible with the **fdump/restore** commands used in the CX/UX operating system. The **fsdump/fsrestore** commands support backup of UNIX (**ufs** and **sfs**) file systems (with **sfs** file systems, only file data is saved. Some trusted information, such as security levels or access control lists, are not saved). The **xfsdump/xfsrestore** commands support **xfs** (and **xfsd**) file systems. These commands:

- support multiple levels of dumps (0 - 9), including overlapping incremental dumps
- support multiple dumps on the same tape
- support an interactive restore interface that allows you to walk through the file structure on the dump tape using commands such as **cd**, **ls**, **pwd**

- allows use of a script file that is executed at the end of the tape for use with tape drives that have automatic tape changing capabilities
- do not support trusted backup and restore operations
- are described in more detail in online man pages **fsdump(1M)**, **fsrestore(1M)**, **xfsdump(1M)** and **xfrestore(1M)**

There are two versions of the **backup/restore** set of commands: the basic and the extended packages. Procedures for using these commands are provided in Chapter 11.

The **backup/restore** commands:

- support backup of UNIX (ufs) and secure (sfs) file systems
- supports saving a specific user's home directory
- provide the ability to dump files based on supplied filenames (with shell pattern matching) as well as interactively
- will optionally send mail when the backup or restore operation has completed
- will restore a partition as of a specified date
- will not overwrite a file unless an overwrite option is specified
- do not support trusted backup and restore operations
- are described in more detail in online man pages **backup(1)**, **backup(1M)**, **restore(1)** and **restore(1M)**

Tape Archiving Using tar

You may use the **tar** command to copy single files or entire directory structures to tape. If necessary, the files can be read back in, individually or as directories, by specifying their names. The **tar** command, which is very simple to use, also allows you to:

- transfer files from one file system to another
- transfer a directory, with its subdirectories and files, from one file system to another.

This section describes how to make a **tar** tape and restore files from it (it also includes specific examples of tape archiving that you should consult when you use the **tar** command), and how to list the pathnames of all files on tape.

A file on a **tar** tape has a header block that describes the file and all of the file's data blocks. At the end of the tape there are two empty blocks.

The header blocks include the:

- file name
- read, write, and execute permissions

- user ID of the file’s owner
- group ID associated with the file
- file size in bytes (0 for linked files)
- checksum.

Table 10-1 describes the options and modifiers of the **tar** command (see the **tar (1)** man page for a complete description of options and modifiers).

Table 10-1. tar Command Options and Modifiers

tar options [modifiers] [pathnames]		
Options and Modifiers		Description
options	-c	Create a new tape containing any files specified by the optional [pathnames] .
	-r	The named <i>files</i> are written on the end of an existing archive.
	-x	Read (extract) the components of the tape into the current directory. Files are loaded into the system with their original pathnames. You may include [pathnames] to specify that tar extract only certain files. If the [pathnames] are directories, tar recursively extracts the contents of the directories, including all subdirectories.
	-t	List the pathnames of the files optionally specified by [pathnames] . If you omit [pathnames] , then tar shows a list (table of contents) of all files on the tape.
	-u	If the [pathnames] are not on the tape or they changed since they were placed on the tape, then add the [pathnames] to the tape.
modifiers	v	Typically used with the -c or -x options. Show the full pathname (verbose) of every file processed.
	f	Use the next argument (a file name) as the name of the archive rather than /dev/rmt/0ms , which is the default. If the name of the archive is - , tar reads from the standard input or writes to the standard output, as appropriate.
	b	Use the next argument (an integer) as blocking factor. The b modifier is valid only with the tape drive, and the default factor is 1.

Making A Tape Archive

To create a **tar** tape, you must either specify the full pathnames of the files in question or be in the directory that contains the files to archive. You specify the full pathname if you intend to restore the files in exactly the same position. On the other hand, if the files are to be placed into a new file system or directory, you perform the **tar** procedure using partial pathnames.

If material is archived using full pathnames, the **tar** tape includes the entire pathnames in its records. If files are archived using partial pathnames, the **tar** tape records only the partial pathnames beginning at the current directory.

The following examples show two methods of archiving material. In all cases, if the command line names a directory, the entire contents (all files and subdirectories) of that directory are archived.

Create A New Tape Using Full Pathname(s)

```
# tar -cv (full pathnames of files or directories being archived)
```

This version of **tar** (with the **-c** option) begins a new tape with the specified files, overwriting the previous contents. In this example, **tar** labels each archived file by its full pathname.

Create A New Tape Using Partial Pathname(s)

```
# cd (full pathname of source directory)
# tar -cv (name of file or partial pathname of material to archive)
```

In this case, file names are recorded starting below the current directory. For example, suppose the directory **/users/joe** includes three files —**program**, **letter**, and **status**— and a subdirectory called **assignments**. You execute the following commands.

```
# cd /users/joe
# tar -cv program letter status assignments
```

The three files will be archived as simply **program**, **letter**, and **status**; however, all the files in the **assignments** directory will be archived with the partial pathnames

```
assignments/file1
assignments/file2
```

and so on. This is because the subdirectory **assignment** is located directly below the current directory **joe**.

Listing The Contents Of A Tape Archive

The **-t** option is used to list the files contained on a **tar** tape. The general form of the command is:

```
# tar -t [pathnames]
```

This lists the pathnames of all files on the **tar** tape that lie below the directory specified in the **pathname** arguments. You may specify more than one directory in the **pathname** arguments, but multiple directories must be separated by a single space. If no directory is specified, **tar** lists the **pathnames** of all files on the current **tar** tape.

To obtain a more detailed listing of the files, you may include the **v** (verbose) modifier in the command line, as follows:

```
# tar -tv [pathnames]
```


Restoring Files From A Tape Archive

Use the **-x option** to restore files from a **tar** tape for storage on disk. The command for restoring these files is:

```
# tar -xv [pathnames]
```

Individual files are restored by specifying the **[pathnames]** under which they were archived. If you specify a directory in this argument, **tar** restores all the subdirectories and files that lie below that directory on the tape. If you do not specify a pathname, the **-x** option causes **tar** to restore all the directories and data files on the tape.

WARNING

If the full pathname of a restored file is the same as that of a file in the current file system, the tape version always replaces the disk version.

Files archived with partial pathnames may be restored to any directory in the current file system (subdirectories will be created automatically when restored). If you specify partial pathnames in the pathname argument, the corresponding files are restored to the current directory.

The following examples illustrate four methods of restoring files from a **tar** tape. The next two examples describe the steps that restore files that were archived with full pathnames. Then, there are two examples to describe the steps required to restore files archived with partial pathnames to a specific destination directory.

NOTE

If the system is configured with RSTCHOWN in the Restricted mode, files restored from a **tar** tape by any user other than “root” will be owned by the user doing the restore. The group membership of the file will be the default group for the user. If RSTCHOWN is in the Compatibility mode (default mode), files restored will retain their original ownership and group membership.

Restore A Complete tar Tape with a Full Pathname

```
# tar -xv
```

This command line restores files archived with full pathnames to the file system maintained on disk. Since no pathnames are specified, **tar** restores every file on the tape with its original full pathname. If the full pathname of a file on tape matches the full pathname of a file on disk, the tape version always replaces the disk version.

NOTE

If a **tar** tape contains files archived with partial pathnames, the above command line restores those files to the current directory.

Restore Specific Files from a tar Tape with a Full Pathname

```
# tar -xv (full pathnames of files and directories)
```

To restore only specific files, you must include the appropriate full pathnames in the above command. As in the previous example, if the full pathname of a file on disk matches the full pathname of a file on tape, the tape version always replaces the disk version.

Restore a Complete tar Tape with a Partial Pathname

```
# cd (full pathname of destination directory)
# tar -xv
```

Files archived with partial pathnames are always restored to the directory you are currently in when you issue the **tar** command. Therefore, before restoring such files you must move to the desired directory. In the above example, all files on the **tar** tape are restored to a specific directory.

Restore Specific Files From tar Tape with a Partial Pathname

```
# cd (full pathname of destination directory)
# tar -xv (partial pathnames of files to be restored)
```

The steps listed above cause **tar** to restore specified files to a single destination directory.

Tape Archiving Using cpio

The **cpio** command copies files to tape on an individual basis. Since **cpio** requires individual pathnames, it is often used with the **find** command. To create a **cpio** tape of an entire file system, use the following format:

```
find file_system -print | cpio [options] > device
```

The **find** command locates all files in the file system. The pipe symbol passes each of the selected files through the **find** command to the **cpio** command for backup. Table 10-2 describes the options of the **cpio** command (see the **cpio(1)** man page for a more complete description of options and arguments associated with the **cpio** command).

Table 10-2. `cpio` Command Options and Arguments

Options & Arguments		Description
options	<code>-o</code>	Copy out.
	<code>-i</code>	Copy in.
	<code>-p</code>	Pass (copy files to another directory on the same system. Destination pathnames are interpreted relative to the named <i>directory</i>).
	<code>-B</code>	Input/output blocked 5,120 bytes per record. If a tape is created with the <code>-B</code> option, it must be read with the <code>-B</code> option.
	<code>-c</code>	Write header information in ASCII character form for portability.
	<code>-d</code>	Make directories as needed.
	<code>-t</code>	Print a table of contents of the tape.
	<code>-u</code>	Overwrite existing files with the same name.
	<code>-m</code>	Retain previous file modification times.
	<code>-v</code>	Verbose mode.
devices	<code>/dev/rmt/0ms</code>	Device name of tape.

The following command shows the typical procedure used to create a backup tape with `cpio`:

```
# find /users_3 -print | cpio -ocvB > /dev/rmt/0m
```

NOTE

If the system is configured with RSTCHOWN in the Restricted mode, files restored from a `cpio` tape by any user other than “root” will be owned by the user doing the restore. The group membership of the file will be the default group for the user. If RSTCHOWN is in the Compatibility mode (default mode), files restored will retain their original ownership and group membership.

Archiving and Restoring Files Using `fsdump/fsrestore` and `xfsdump/xfrestore`

There are three basic times when you should back up files: daily, weekly, and monthly. For each backup, it is recommended that you use the `fsdump` or `xfsdump` command. (If desired, you may back up daily files with a `tar` command. `tar`, however, cannot back up files that require more than one tape. If the `fsdump` or `xfsdump` command runs out of space on a tape, it prompts you to load another tape.)

This section explains how to use `fsdump` or `xfsdump` for your daily, weekly, and monthly backups and lists procedures that tell you how to restore user files. Although it is highly unlikely that the `root` file system will be destroyed, you should also have some protection against losing your `root` files.

Note

`fsdump`, `fsrestore`, `xfsdump`, and `xfrestore` are located in directory `/usr/sbin`. You should add this directory to your `PATH` environment variable.

Daily And Weekly Backups

You should back up the file systems on a daily and weekly basis. Perform daily and weekly backups after working hours or during times of minimal system use; a regular backup schedule allows users to plan their work accordingly.

Make sure all users are logged off before running a backup. This practice insures a complete backup of all user files.

For `ufs` and `sfs` file systems, to obtain a valid tape backup, you should first check that the file systems are consistent by running `fsck`. If you find inconsistencies, refer to Chapter 4 in this book before continuing.

If the `fsck` indicates that the file systems are consistent, then run `fsdump`. The preferred method to running `fsdump` is in either single or multi-user mode with the file system unmounted. You may also run `fsdump` with the file system mounted but this method is only recommended during a period of no file system activity.

For `xfs` file systems `fsck` is not used. No action is required to ensure file system consistency, except in cases where the system has been brought down without the file system being unmounted tidily. In this situation it is necessary for the file system to be mounted once before running `xfsdump`. This is because any necessary recovery takes place automatically during the mount operation. If there is an entry for the `xfs` file system in the `/etc/vfstab` file, then the act of bringing up the system is sufficient to ensure recovery.

Backing Up Selected System Files

Certain system files are site-dependent and periodically updated. You should have backup copies of these files (Refer to Table 10-3).

Table 10-3. System Files Requiring Regular Backups

System File	Description
<code>/var/adm/dumpdates</code>	File where dump history is kept.
<code>/etc/vfstab</code>	Defines characteristics of file systems.
<code>/etc/gettydefs</code>	Define terminal configuration and capabilities.
<code>/etc/group</code>	Lists all user groups on the system.
<code>/etc/hosts</code>	Lists IP addresses, hostnames, hostname aliases, and optional comments.
<code>/etc/inittab</code>	Define terminal configuration and capabilities.
<code>/etc/conf</code>	System configuration directory.
<code>/etc/passwd</code> <code>/etc/shadow</code>	Identifies every authorized user on the system.
<code>/etc/rc*.d</code>	Site specific initialization scripts.
<code>/etc/ttytype</code>	Data base of terminal types by port.

The first time you back up the selected system files, you should create a directory for storing these files.

```
# mkdir /users/sysbackups
```

You can also save time by creating a shell script that copies all the selected system files to another file system, as shown in Figure 10-1.

After you create your shell script, you must adjust the permissions for this file (see the *User's Guide* for details about permissions):

```
# chmod a+x sysfiles
```

The fsdump and xfsdump Commands

The **fsdump** and **xfsdump** commands copy a device to tape. The dump may be either full or incremental. In general, the **fsdump** and **xfsdump** processes write the i-nodes and data blocks of each file to the tape. They also write some information about unallocated i-nodes and blocks to the tape. The **fsrestore** command is the counterpart to **fsdump**. The **xfsrestore** command is the counterpart to **xfsdump**. **fsrestore** and **xfsrestore** may read an entire file system or individual files from a tape created with **fsdump**. **xfsrestore** may read an entire file system or individual files from a tape created with **xfsdump**. The **fsdump**, **fsrestore**, **xfsdump** and **xfsrestore** commands use `/dev/rmt/0ms` as the default tape dump device. The order of options does not matter but the arguments must be in the same order as the corresponding options.

```

# cd /usr/local
# cat >sysfiles
cp /etc/passwd /users/sysbackups/passwd
cp /etc/shadow /users/sysbackups/shadow
cp /etc/group /users/sysbackups/group
cp /etc/inittab/ /users/sysbackups/inittab
cp /etc/gettydefs /users/sysbackups/gettydefs
cp /etc/vfstab /users/sysbackups/vfstab
cp -R /etc/conf /users/sysbackups/conf
cp /etc/ttytype /users/sysbackups/ttytype
cp /var/adm/dumpdates /users/sysbackups/dumpdates

(Also include any site-dependent files)

<ctrl-d>
#

```

Figure 10-1. Creating a Backup Script

NOTE

You should use the `-c` option with `fsdump` and `xfsdump` whenever you store backups on cartridge tapes. This will use the space on the tape more efficiently.

Making the Daily Backup Tape

Each day, you should dump all `users` files that have changed during the last 24 hours. An incremental `fsdump` or `xfsdump` will save only the files that have changed since the last `fsdump` or `xfsdump`. Follow the guidelines below to run the daily backup using the shell script illustrated in Figure 10-1.

1. Insert the tape into the `/dev/rmt/0ms` tape drive (wait for the drive to become ready before proceeding to step 2). If you wish to use a different tape drive you must specify the `-f tape` option in step 3 below.
2. Execute the shell script you created in Figure 10-1, (which copies system files into the selected file system).
3. As `root`, execute this command:

```

# fsdump -9u /dev/dsk/1s2
OR
# xfsdump -9u /dev/dsk/1s2

```

`fsdump` must be used if the file system type is `ufs` or `sfs`. `xfsdump` must be used if the file system type is `xfs`. The `/var/adm/dumpdates` file (Figure 10-2) contains the date of your last dump of the `users` file system.

`fsdump` and `xfsdump` check the “time-last-updated” fields corresponding to each file in `users`. Then they check `/var/adm/dumpdates`, as shown below, for the date of the last `/users` dump. If a

file changed since the specified date, **fsdump** or **xfsdump** copies the file to the tape. Because you use the **-9** option, **fsdump** or **xfsdump** does an incremental save. Because you included the **-u** option **fsdump** or **xfsdump** writes the current date to the **/var/adm/dumpdates** file.

<code>/dev/rdisk/1s0</code>	0	Thur Aug 31 10:16:41 1995
<code>/dev/rdisk/1s2</code>	0	Thur Aug 31 10:25:00 1995
<code>/dev/rdisk/0s2</code>	0	Thur Aug 31 10:33:12 1995
<code>/dev/rdisk/2s2</code>	0	Thur Aug 31 10:21:00 1995
<code>/dev/rdisk/0s0</code>	0	Thur Aug 31 10:13:00 1995
		——Date of Last Dump
		——Full Dump Level

Figure 10-2. Sample `/var/adm/dumpdates` File

4. Remove the tape. You should record the successful completion of the backup in the system log.

Making the Weekly Backup Tape

To run your weekly backup, follow the procedures below.

1. Insert the tape into the `/dev/rmt/0ms` tape drive (wait for the drive to become ready before proceeding to step 2). If you wish to use a different tape drive, you must specify the `-f tape` option in step 3 below.
2. Execute the shell script that copies system files into the `/users` file system.
3. As **root**, issue this command:

```
# fsdump -5u /dev/dsk/1s2
OR
# xfsdump -5u /dev/dsk/1s2
```

fsdump must be used if the file system type is **ufs** or **sfs**. **xfsdump** must be used if the file system type is **xfs**. The **-5** option backs up all files in `/users` that changed during the past week, and the **-u** option writes the current date in `/var/adm/dumpdates`. When **fsdump** or **xfsdump** has consumed 2300 feet of tape, it prompts you to put another tape in the tape drive.

4. Remove the tape. You should record the successful completion of the backup in the system log.

Monthly Backups

Each month, back up all files on the system using the following procedure.

1. Insert the tape into the `/dev/rmt/0ms` tape drive (wait for the drive to become ready before proceeding to step 2). If you wish to use a different tape drive, you must specify the `-f tape` option in step 2 below.
2. As **root**, issue this command:

```
# fsdump -0u /dev/dsk/1s2
OR
# xfsdump -0u /dev/dsk/1s2
```

fsdump must be used if the file system type is **ufs** or **sfs**. **xfsdump** must be used if the file system type is **xfs**. The `-0` option causes the entire `/users` file system to be saved.

3. Remove the tape from the tape drive and label it with the current date. You should record the successful completion of the backup in the system log.
4. Using a separate tape for each file system, repeat Steps 1 - 3 for all of your file systems.

Saving root

To copy `/dev/dsk/0s0` to tape, use:

```
# fsdump -0u /dev/dsk/0s0
OR
# xfsdump -0u /dev/dsk/0s0
```

If you use **fsdump** or **xfsdump**, you can use the procedure documented under *Restoring the root Partition* in this chapter for restoring files to root.

fsdump must be used if the file system type is **ufs** or **sfs**. **xfsdump** must be used if the file system type is **xfs**.

Restoring Files

One of your system administration tasks is to assist users who encounter system problems resulting in the loss of some or all of their files. This section contains material designed to help you restore user files to the file system.

To locate a file on the tape, you must know the file's full pathname and the most recent date of revision. The user will usually be able to tell you the last time the file was updated.

If you use the **fsdump** command for a backup, you can use **fsrestore** or **xfsrestore** to retrieve files from the tape. The **fsrestore** command may be used to restore an entire file system or individual files from a **fsdump** tape.

If you use the **xfsdump** command for a backup, you can use **xfsrestore** to retrieve files from the tape. The **xfsrestore** command may be used to restore an entire file system or individual files from a **xfsdump** tape.

Because directories are dumped before files, **fsrestore** and **xfsrestore** can display a table of contents without reading the entire backup tape.

The syntax for the **fsrestore** command is:

```
# fsrestore options name
```

The syntax for the **xfsrestore** command is:

```
# xfsrestore options name
```

Restoring an Entire File System

If, for example, you want to restore the **/users** file system using **fsrestore** or **xfsrestore** perform the following:

```
# cd /users
# fsrestore r
OR
# xfsrestore r
```

with **/users** mounted on **/dev/dsk/2s2**, the **fsrestore** (or **xfsrestore**) command overlays the entire **/users** file system with the information on the tape. You should instead restore the file system to a safe temporary location and, after verifying that the restored data is accurate, move it to **/dev/dsk/2s2**. To restore **/dev/dsk/2s2**, use the following procedure.

1. Make a temporary file system to hold **/dev/dsk/2s2**. In the examples below, we have used **/dev/dsk/1s2** as the temporary partition to hold **/dev/dsk/2s2**'s data.

```
# /sbin/newfs /dev/dsk/1s2
```

2. Mount **/dev/dsk/1s2** in a temporary directory.

```
# mkdir /mnt
# /sbin/mount /dev/dsk/1s2 /mnt
```

3. Change the temporary directory.

```
# cd /mnt
```

4. Restore **/dev/dsk/2s2** from the tape onto **/dev/dsk/1s2**.

```
# fsrestore -r
OR
# xfsrestore -r
```

5. Move the restored data stored in `/mnt` to `/users`.

```
# fsdump -0f- /mnt | (cd /users; fsrestore -xf-)
OR
# xfsdump -0f- /mnt | (cd /users; xfsrestore -xf-)
```

6. Remove the contents of the temporary `/mnt` directory.

```
# cd /mnt
# rm -r *
```

7. Unmount the temporary directory.

```
# /sbin/umount /dev/dsk/1s2
```

Restoring an Individual File

To restore a particular file from a dump tape, use the following command syntax:

```
# fsrestore -x name
OR
# xfsrestore -x name
```

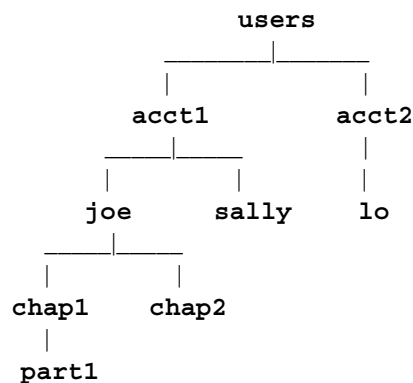
This command extracts the file name from the tape. If name is a directory, then `fsrestore` and `xfsrestore` recursively extract the entire contents of the directory (unless the `-h` option is specified).

Make sure you specify a file or a directory name. Otherwise, `fsrestore` and `xfsrestore` extract the contents of the tape beginning with the `root` directory. For example, if the full pathname is `/users/acct1/legal/joe/doc`, enter:

```
acct1/legal/joe/doc
```

Restoring an Individual Directory

If your backup tape contains the hierarchy shown below, you may specify that `fsrestore` or `xfsrestore` extract only the file `joe`.



In this case, enter

```
# fsrestore -xh acct1/joe
OR
# xfsrestore -xh acct1/joe
```

The **-h** option extracts only the directory **acct1/joe**. **chap1** and **chap2** will not be restored.

Using the Interactive Restore

To use the interactive restore command you may also request:

```
# fsrestore -i
OR
# xfsrestore -i
```

which allows interactive restoration of files from a **fsdump** or **xfsdump** tape. After receiving the interactive **fsrestore>** (or **xfsrestore>**) prompt, you may enter one of these requests:

- **ls**
- **extract**
- **add**
- **delete**
- **pwd**
- **help**
- **cd**
- **verbose**
- **setmodes**
- **what**
- **quit**

If you need details about the above requests, use the **help** option.

Restoring the root Partition

If the system is unusable because of corrupt or non-existing essential files on the **root**, **/usr**, **/var** or **/stand** partitions, you may restore using the File System Restore utility described later in this chapter.

Additional Features of fsdump/fsrestore and xfsdump/xfsrestore

Options are available in the **fsdump/fsrestore** and **xfsdump/xfsrestore** sets of commands that are very useful when performing system backups. These options are

described in the table below. For more details, refer to the **fsdump(1M)**, **fsrestore(1M)**, **xfsdump(1M)**, and **xfsrestore(1M)** man pages.

Table 10-4. Additional Options Available

fsdump/xfsdump Options	Description
B bsize	Specifies tape size in blocks. One block equals 1024-bytes.
H host (fsdump only)	The H option causes fsdump to dump across the network to a remote system specified by the host argument. This feature is useful when the local system has no tape drive. The f option must be used with the H option.
f dump-file	Use dump-file as the file to dump to instead of /dev/rmt/0ms .
M msize	This option specifies tape size in megabytes (MB).
0-9	The dump level.
R rfile	Writes tape position information to rfile when the backup is completed. This option is useful when you want to write multiple backups on a tape.
S shell-script	Executes shell-script when end-of-tape is reached instead of notifying the operator. This option is useful if you have multiple tape drives that can swap tapes from user commands.
fsrestore/xfsrestore Options	Description
H host (fsrestore only)	The H option causes fsrestore to restore a dump across the network from a remote system specified by the host argument. This feature is useful when the local system has no tape drive. The f option must be used with the H option.
f dump-file	Use dump-file as the file to restore from instead of /dev/rmt/0ms .
r	Restore the entire tape. Loads the tape's full contents into the current directory.
V (xfsrestore only)	A backup tape created by xfsdump can be verified using the V option of xfsrestore . This option specifies that each file on the tape is to be compared with the corresponding disk file relative to the current directory.

In addition, the **fsdump** and **fsrestore** scripts **rdump** and **restore** are provided for users who are familiar with the CX/UX OS **rdump** and **restore** utility.

The following is a summary of how to use the options described above and provides examples on their use.

1. The **H host** option (**fsdump** and **fsrestore** only):

The **H** option allows the administrator to backup a system that does not have a tape drive. An example of the use of the **H** option follows:

Assume host orlando has no tape drive while host seminole is equipped with a DAT tape drive. Both host systems are connected to the same network and users on one host can run **rsh** (remote shell) to the other host and vice versa. If the DAT tape drive on seminole is `/dev/rmt/1m`, then on host orlando, **root** can execute the following command to backup the `/dev/dsk/1s2` partition.

```
orlando:/
# fsdump OHf seminole /dev/rmt/1m /dev/dsk/1s2
```

By using the **H** option from **fsrestore**, **root** on orlando can restore the `/dev/dsk/1s2` partition from seminole as follows:

```
orlando:/
# fsrestore rHf seminole /dev/rmt/1m
```

This restores the data on `/dev/dsk/1s2` to the current directory. In order for the **H** option to work, it is necessary to permit **root** on orlando to execute commands on seminole as **root**. Both hosts should be connected to the same network as dumping across the network may stress the network's capacity.

2. The **M msize** option:

This option makes it possible to specify the larger sizes that are capable with the DAT and 8mm tape drives. By default, **fsdump** and **xfsdump** will assume that the total capacity of the storage device is around 42 megabytes. Since most DATs are capable of storing 1 gigabyte, the default would waste considerable tape capacity. Therefore, when using a DAT drive it is suggested that you execute a command string similar to the one shown below to backup `/dev/dsk/1s0` to the DAT device `/dev/rmt/1m`:

```
# fsdump OMf 1024 /dev/rmt/1m /dev/dsk/1s0
OR
# xfsdump OMf 1024 /dev/rmt/1m /dev/dsk/1s0
```

3. The **B bsize** option:

This option functions like the **M** option except you specify the number of blocks, where a block is equal to 1024 bytes.

4. The **R rfile** option:

This option allows you to keep track of how much unused tape remains if you need to backup several file systems to a single tape. To use this option you should initialize **rfile** by executing the following command:

```
# echo "1 0M" > /tmp/rfile
```

This tells **fsdump** and **xfsdump** that this is the first tape and it is 0 megabytes into the tape. Next invoke **fsdump** or **xfsdump** with the **M** option. If the letter after the tape consumed field was B then **fsdump** or **xfsdump** would have to be invoked with the **B** option.

```
# fsdump 0MRf 1024 /tmp/rfile /dev/rmt/1mn /dev/dsk/1s0
OR
# xfsdump 0MRf 1024 /tmp/rfile /dev/rmt/1mn /dev/dsk/1s0
```

The device specified is the non-rewinding device. The n after 1m indicates non-rewinding. If the size of /dev/dsk/1s0 was 300 megabytes /tmp/rfile might contain the following

```
# cat /tmp/rfile
1 311M
#
```

Following this, the next file system could be backed up using:

```
# fsdump 0MRf 1024 /tmp/rfile /dev/rmt/1mn /dev/dsk/2s0
OR
# xfsdump 0MRf 1024 /tmp/rfile /dev/rmt/1mn /dev/dsk/2s0
```

5. The **S** shell-script option:

This option allows you to backup a file system to a tape drive that can hold more than 1 tape and can swap tapes by a user command. The shell-script is executed when **fsdump** or **xfsdump** consumes all of the current tape. For example, if /dev/rmt/2* is the stacker DAT tape drive and is capable of holding 6 tapes, the shell-script "changetape" might contain the following:

```
#!/bin/sh
echo "switching to tape $1"
mt -f /dev/rmt/2mn offline
if [ $? = 0 ]; then
    echo "tape $1 is loaded"
    exit 0
else
    echo "error: could not switch to tape $1"
    exit 1
fi
```

The command used to change tapes in the stacker is:

```
# mt -f /dev/rmt/2mn offline
```

To backup a file system using the stacker, execute the following:

```
# fsdump 0Sf /tmp/changetape /dev/rmt/2mn /dev/dsk/1s0
OR
# xfsdump 0Sf /tmp/changetape /dev/rmt/2mn /dev/dsk/1s0
```

By combining this option with the **R** option, you should be able to save all your backups on a set of tapes.

File System Restore Utility

Introduction

You can restore corrupt or non-existing essential **root**, **/usr**, **/var** or **/stand** partitions using the File System Restore utility that is available on the Base Installation Tape. This utility supports tapes made by **fsdump**, **xfsdump** and **cpio**. Before executing the restore utility, you should be familiar with the following information and guidelines:

- When you run the File System Restore utility, you will be given the option of running the **format (1M)** command to format the disk and to select partition sizes. You should run format only if the entire disk is being fully restored.
- Backup tapes created using **fsdump** can be restored using either **fsrestore** or **xfsrestore**. Backup tapes created using **xfsdump** can only be restored using **xfsrestore**.
- Trusted restore operations are not supported by any of the currently supported restore methods. **fsdump** supports backup of **sfs** file systems, however only file data is saved. Some trusted information, such as security levels or access control lists, is not saved. Thus **fsrestore** and **xfsrestore** cannot restore any of the trusted information.
- The File System Restore utility supports backup tapes with more than one backup image as the utility allows for fast forwarding of the tapes.
- If you use the **find** command to generate pathnames for the **cpio** command, use the **find** command **-mount** option to restrict the search to the file system containing the directory specified.
- You will be prompted from what directory to execute the restore command. The correct response depends on how the backups were made. For example, if you have a **cpio** backup image of the **/usr** file system that was generated relative to **/**, then you should select to execute the **cpio** command from the **/** directory when restoring the **/usr** file system. However, if you generated the backup relative to the **/usr** file system, then you should select to execute the command from the **/usr** file system.

If for example, you have a **fsdump** or **xfsdump** backup image of the **/usr** file system you should select to execute the **fsrestore** or **xfsrestore** command from the directory that corresponds to the file system being restored, in this case **/usr**. The same is true for the **/**, **/var**, and **/stand** file systems.

Booting Up Procedure

Night Hawk Systems

To begin the file system restore utility on a halted system, insert the Base Installation Tape and execute the following console commands: (where *c* = controller or slot number, and *u* = device unit designation of tape drive containing the Base Installation Tape.

```
#> p boot 0
#> fd mt (c,u,1)
#> fb
```

Power Hawk Systems

First the resident console must be loaded off of the distribution media. This must be done using the Motorola **ppcbug** product. **ppcbug** is the resident debug/self-test program initially loaded when the Power Hawk hardware is reset. Depending upon the firmware setup, **ppcbug** may attempt to auto-boot or may just go to a debug prompt. If it attempts to auto-boot, depress the Escape (ESC) key until the PPC1-BUG> prompt is received at the system console. The **ppcbug pboot** command is used to load the console off of the distribution media. This must be done on a tape drive connected to the internal SCSI controller as **ppcbug** is not capable of communicating with VME controllers. Place the Base Installation Tape containing the file system restore utility into the tape drive and execute the following:

```
PPC1-BUG> p boot 0,x0
```

where *x* is the SCSI ID of the selected tape drive.

The console is copied to the target disk during the installation procedure. Once this has been done, the console can be loaded from that disk without having to load it from tape each time. Once the console is loaded, it will enter the halt state and the boot up may continued.

To begin system installation on a halted system, insert the Base Installation Tape and execute the following console commands: (where *d* = is the logical tape drive designation of the tape drive containing the Base Installation Tape. This number is found from the output of the “**fd -l**” command.)

```
#> fd -l
.....
fd          disk          tape
0  (0,0,x,0) FUJITSU M2624S-512 (0,5,x,0) ARCHIVE VIPER 150 21247
1  (2,0,x,1) FUJITSU M2624F-512

#> p boot 0
#> fd mt (d,1)
#> fb
```


File System Restore Operation

The File System Restore utility is a menu driven utility. As noted earlier, this utility resides on the Base Installation tape. Also residing on the same tape is the system software installation program. The first screen (see Screen 10-1) permits you to select the install option or the restore option. (Note that Version number will change with each release.)

```

PowerMAX OS Version 2.2 System Installation Tape

You may choose to install the PowerMAX OS system
software or restore a file system from a backup tape at this time.
Please choose one:

1  INSTALL      System Software Installation
2  RESTORE      File System Restore

=> Select menu mode. [?,??]: 2

```

Screen 10-1. System Installation, Main Menu

The File System Restore Menu screen (see Screen 10-2) informs the administrator how to get help (?), quit (q) and what the values in brackets [xxx] represent.

If the user selects to quit the file system restore, or a fatal error occurs, the file system restore operation will be suspended by executing a sub-shell. When exiting the shell, the last operation will be repeated and the file system restore utility will continue. Typing **.restart** from the shell will restart the restore procedure from the beginning.

```

File System Restore Procedure

To aid you in restoring file systems from backups, a series of
prompts preceded by '=>' will follow. At any of these prompts
you may enter:

    '?'      for help
    'q'      to quit

Values shown in brackets, '[' ]' are valid choices for that
prompt. If a default value is shown in parenthesis, it may be
selected simply by pressing 'Return'.

=> Press 'Return' to begin the restore process. (default: begin) [?,q]

```

Screen 10-2. File System Restore, Main Menu

Screen 10-3 depicts the screen that prompts the administrator for the ID of the disk to be restored.

```
More than one hard disk was found on this system. Please
identify the disk to be restored.
 1 Disk0      (NCR, pci0 slot 0, SCSI 0/0)
 2 Disk1      (NCR, pci0 slot 0, SCSI 1/0)

=> Select disk to be restored. [?,??,q]: 2
```

Screen 10-3. Identification of Disk Requiring Restoring, Menu

Screen 10-4 primarily provides information on how to create file systems. The actual creation of the file system(s) is done in the following screen.

```
File systems may now be created for /, /usr, and /var. If / is
type xfs, the /stand file system may also be created.

Three backup formats are currently supported: cpio(1), fsrestore(1M),
and xfsrestore(1M).

The fsrestore command restores file systems using tapes created by
fsdump(1M). fsrestore supports restoring ufs and sfs file systems.

The xfsrestore command restores file systems using tapes created by
xfsdump(1M) or fsdump(1M). xfsrestore supports restoring ufs, sfs,
or xfs file systems.

The cpio command supports restoring ufs, sfs, and xfs file systems.

None of these commands support trusted restore operations.
```

Screen 10-4. Create File System Help

Screen 10-5 asks the user which of spurious “:” the file systems should be recreated. It starts by asking if the root file system is of type ‘xfs’. If the answer is “y” (yes), the set of file systems that can be restored is /, /stand, /usr, and /var. If the answer is “n” (no), the set of file systems is /, /usr and /var. Only answer “y” (yes) for file systems being fully restored.

```
=> Is the root file system of type 'xfs' ? [y,n,?,q] n
=> Do you wish to create the '/' file system ? (default: y) [y,n,?,q]
=> Enter the file system type for '/'. (default: ufs) [?,q]
=> Do you wish to create the '/usr' file system ? (default: y) [y,n,?,q]
=> Enter the file system type for '/usr'. (default: ufs) [?,q]
=> Do you wish to create the '/var' file system ? (default: y) [y,n,?,q]
=> Enter the file system type for '/var'. (default: ufs) [?,q]
```

Screen 10-5. Create File System

In Screen 10-6, the user is asked to verify that the information concerning the file system to be created is correct. Otherwise, the administrator can restart the menu and correct the erroneous information. Note that a dash “-” will appear in the FS Type to be created column for a partition not selected to be created. If selected, the file system type (ufs, sfs, or xfs) will appear instead.

```

*****
IMPORTANT:  Please verify that the following information concerning
-----    the file systems to be created is correct. If it is not,
           this portion of the restore process will be restarted so
           you can correct it.

Disk: Disk1 (NCR, pci0 slot 0, SCSI 1/0)

      Partition  File System  FS Type
      -----
           0      /           ufs
           1      (swap)      -
           2      /usr        ufs
           3      /var        ufs
           6      (console)   -

*****
=> Is this information correct? [y,n,?,q]

```

Screen 10-6. Verify File System Creation

Screen 10-7 depicts the prompt that provides the administrator with a choice to run the **format (1M)** command.

CAUTION

Do not run format unless the entire disk is being fully restored.

```

The installation disk MUST be formatted and partitioned correctly for
the restore to succeed. This is done with the format(1M) command.

=> Do you wish to run format now? (default: y) [y,n,?,q]

```

Screen 10-7. Option to Run **format (1M)** Command

Screen 10-8 shows the “making” and “mounting” of the various file systems. Note that **/root** is the temporary mount point on the ramdisk for the file system to be restored.

```
* Making the / file system (ufs).
* Mounting the / file system under /root.
* Making the /usr file system (ufs).
* Mounting the /usr file system under /root/usr.
* Making the /var file system (ufs).
* Mounting the /var file system under /root/var.

Please verify that the backup tape is in the drive and press 'Return' when
ready. (default: go) [?,q]
```

Screen 10-8. Creating File Systems

Screen 10-9 shows the prompt that allows the administrator to advance the tape.

```
The image to be restored may not be the first file on tape.
=> Do you wish to advance forward the tape? [y,n,?,q] n
```

Screen 10-9. Option to Advance Tape

Screen 10-10 depicts the menu that allows the administrator to select the appropriate directory before beginning the restore operation.

```
The files on the backup image must be relative to one of
the following directories. The restore command will be
executed from within this directory.

1 /
2 /usr
3 /var
4 /stand

=> Select a directory. [?,??,q]: 1
```

Screen 10-10. Selecting a Directory, Menu

Screen 10-11 depicts the menu that allows the administrator to select the tape archive format: **cpio**, **fsrestore**, or **xfrestore**.

```
The backup tape must be in one of the following formats:  
1  cpio  
2  fsrestore  
3  xfsrestore  
  
=> Select archive format of backup tape. [?,??,q]: 2
```

Screen 10-11. Selecting Tape Archive Format, Menu

Screen 10-12 shows the prompt that allows the administrator to enter the appropriate argument string. In this example, the argument string for **fsrestore** needs to be entered as **fsrestore** was chosen as the archive format in the previous menu. Refer to Table 10-2 or the **cpio(1)** man page for details on **cpio** arguments.

```
Please enter argument string for the fsrestore command.  
Use '/dev/tape' for the tape device name [i.e. -rf /dev/tape]:  
  
=> Enter argument string: -rf /dev/tape
```

Screen 10-12. Option to Enter fsrestore Argument String

Screen 10-13 shows the prompt that requests that the administrator verify that information previously entered is correct.

```
*****
IMPORTANT:   Please verify that the following information is correct.
-----
              If it is not, this portion of the restore process will be
              restarted so you can correct it.

              File Images to Forward:    0 (restoring from 1st image on tape)

              Restore Command:          fsrestore -rf /dev/tape

              Executing from Directory:  / (mounted under /root)

*****
=> Is this information correct? [y,n,?,q] y
```

Screen 10-13. Verification of Information Entered

Screen 10-14 shows the prompt that allows the administrator to restore another file system if required.

```
* Changing directory to / mounted under /root.
* Waiting for tape to position.
* Executing: /sbin/fsrestore -rf /dev/tape. Time started: 10:15:06
Warning: ./usr: File exists
Warning: ./var: File exists
Warning: ./lost+found: File exists

The restore command succeeded.
=> Do you wish to restore another file system? [y,n,?,q] n
```

Screen 10-14. Option to Restore Additional File System

Screen 10-15 shows the prompt allowing the administrator to examine the file systems before halting the system.

```
                This completes the file system restore procedure.

At this time you may quit the procedure (enter 'q' at the next prompt)
and inspect the file systems restored. Note that the file systems
just restored are mounted under the '/root' directory. You may
then continue this procedure by exiting the shell (type 'exit') or
restart the procedure from the beginning (type '. restart').

=> Press 'Return' to halt the system. (default: halt) [?,q]
.
* Synchronizing the system.
* Unmounting file systems.
CPU halted
```

Screen 10-15. Completion of File System Restore

Backup and Restore Services

What Is A Backup Operation?	11-1
Archiving and Restoring Data On Your System	11-1
Using the Basic Backup Service	11-2
Editing Backup and Ignore Files for Basic Backups	11-2
The Backup File	11-2
The Ignore File	11-2
Media Preparation and Handling for Basic Backups	11-3
Backing Up to Tape	11-3
Using the backup Command	11-3
Performing Complete Backups to Tape	11-3
Performing Partial Backups to Tape	11-3
Backing Up Selected Files to Tape	11-3
Backing Up User Home Directories to Tape	11-4
Using the Extended Backup Service	11-4
Suggestions for Performing Extended Backup Operations	11-4
The System Administrator's Tasks	11-5
The Operator's Tasks	11-6
Establishing a System Backup Plan	11-6
Backup Plan Information	11-6
Getting Backup Plan Information	11-7
Using bkgreg to Set Up Backup Tables	11-8
What Is a Backup Table?	11-8
Validating Backup Tables	11-9
Displaying the Contents of the Backup Tables	11-9
Specifying Custom Backup Tables	11-11
Using bkgreg to Customize Backup Tables	11-11
Adding or Changing Backup Table Entries	11-11
Adding an Operation Entry	11-11
Modifying an Existing Operation Entry	11-12
Removing an Operation Entry	11-12
Specifying Backup Methods	11-13
Determining Which Backup Method to Use	11-13
Full File System Method	11-14
Incremental File System Method	11-14
Excluding Files from Incremental Backups	11-15
Creating an Exception List	11-15
Creating a Customized Exception List	11-15
Modifying an Exception List	11-16
Converting Exception Lists from Earlier Backup Services	11-18
Full Image Method	11-20
Full Disk Method	11-20
Full Data Partition Method	11-20
Requesting Migrations for Backed-Up Information	11-21
When to Do a Migration	11-21
Migrating an Archive	11-21
Requesting Core File System Backups	11-22
Restrictions on Backing Up Core File Systems	11-22

- Specifying Originating Objects 11-22
- Specifying Destination Devices 11-23
- Specifying the Rotation Period 11-24
- Establishing Dependencies and Priorities 11-25
- Creating Tables of Contents 11-26
- Using backup to Perform Backup Operations 11-27
 - Selecting an Operator Mode 11-28
 - Background Mode 11-28
 - Interactive Mode 11-29
 - Running Automatic Backups 11-30
 - Previewing Backup Operations 11-31
 - Requesting Limited Backups 11-32
- Using bkoper to Monitor Backup Jobs 11-33
 - When Backup Jobs Need Assistance 11-34
 - Using bkstatus to Check Job Status 11-36
 - Displaying the Backup Status Table 11-37
 - Controlling Jobs in Progress 11-38
- Using bkhistory to Display the Backup History Log 11-39
 - Customizing the Backup History Log Display 11-40
 - Customizing the Contents of the Display 11-40
 - Customizing the Format of the Display 11-41
 - Truncating the Backup History Log 11-42
- Backing Up Data through OA&M Menus 11-42
 - Basic Backup Service 11-42
 - Extended Backup Service 11-43
- Quick Reference to the Extended Backup Service 11-44
- What Is a Restore Operation? 11-48
 - Before You Begin 11-48
- Basic Restore Service 11-48
 - Specifying Pattern Matching on the restore Command Line 11-48
 - Media Handling 11-49
 - Restore Examples 11-49
 - Displaying an Index of Archived Files 11-49
 - Restoring Everything from the Archive 11-49
 - Restoring Selected Files 11-50
- Extended Restore Service 11-50
 - Who Is Responsible for Servicing Restore Requests? 11-50
 - Using rsnotify to Arrange for Restore Requests 11-51
 - Deciding Which Restore Command to Use 11-51
 - How the Extended Restore Service Works 11-51
 - Using urestore to Restore a Directory or File 11-52
 - Using restore to Restore Other Disk Objects 11-52
 - Restoring a Specific Version of an Object 11-53
 - Restoring an Object to a New Location 11-53
 - Checking the Status of Restore Requests 11-54
 - Using restore or urestore to Check Status 11-54
 - Using rsstatus and ursstatus to Check Status 11-54
 - The Status Table 11-54
 - Customizing the Display of the Status Table 11-55
 - Servicing Pending Restore Requests 11-56
 - Restricting Restores 11-57
 - Removing and Canceling Restore Jobs 11-57
 - Basic Options 11-58
 - Restoring ffile and incfile Backup Archives Using cpio 11-58

Using the Restore Service Through OA&M Menus	11-60
Basic Restore Service	11-60
Extended Restore Service	11-60
Quick Reference to the Extended Restore Service	11-62
The Administrator's Tasks	11-62
The Operator's Tasks	11-63
The User's Tasks	11-64

What Is A Backup Operation?

NOTE

If you are running your system in compliance with the security criteria described in the “Security Administration” part of *System Administration*, Volume 1, the Backup Service will not be available for use in multi-user state. See the chapter “Trusted Backup and Restore” in this book for information on backing up and restoring files with Mandatory Access Control levels. See the “Security Administration” part of *System Administration*, Volume 1 for a list of the software packages that should be installed on the secure system, and the facilities (such as the Backup Service) that are available in single-user mode only.

Note

The Backup and Restore services are supported by PowerMAX OS release 4.2 and earlier. This service is **not** available with any PowerMAX OS release after release 4.2.

What's so important about backups? Ask the employee who accidentally removed the file containing the speech to be delivered (tomorrow) to the senior management team, or the student who somehow clobbered the term paper it took five weeks to write.

Backup operations allow you to copy the data on your system to another storage medium, such as DAT or cartridge tapes (archiving). When the need arises, you can recover (restore) information that may have been lost as a result of human or mechanical error.

Archiving and Restoring Data On Your System

There are several means of archiving and restoring data on your system.

- Basic **backup** and **restore** - a simple system of archiving system files and user directories.
- Extended **backup** and **restore** - a comprehensive set of utilities for scheduling and archiving files, file systems, partitions, and whole disks of data.

- **cpio** and **tar** - basic UNIX system utilities for archiving files. Use **cpio** regularly to archive your root filesystem so that you can restore it, either partially or fully, if it becomes corrupted. **cpio** is also especially suitable for transferring files between different types of UNIX machines. (Refer to Chapter 10 *Archiving and Restoring Data* for further information.)
- **fsdump** and **fsrestore** - utilities for archiving **ufs** file systems. (Refer to Chapter 10 *Archiving and Restoring Data* for further information.)

This chapter explains how to use shell commands to backup your system and users' data. The manual pages provide detailed descriptions of the shell commands.

If the Operations, Administration and Maintenance (OA&M) package (a non-graphical menu interface) is installed on your system you can use it to complete many of these tasks. (See “*Backing Up Data through OA&M Menus*” later in this chapter.)

Using the Basic Backup Service

The basic backup service is a simple system of archiving system files and directories, either completely or partially. Examples of how to use the **backup** command and its options are in the section “*Using the backup Command*”.

Editing Backup and Ignore Files for Basic Backups

When performing a complete or partial backup, the backup process searches through two special files to determine which files should, or should not be backed up. These are the **Backup** and **Ignore** files.

If a directory or file name is specifically listed in both **/etc/Backup** and **/etc/Ignore**, the **Backup** file takes precedence and the file is backed up.

The Backup File

You can edit **/etc/Backup** to include

- a list of all directories which contain files to be backed up
- the names of specific files to be backed up

For example, you may add the directory **/home/myacct/messages** to **/etc/Backup** to ensure that all files in this directory are backed up.

The Ignore File

You can edit **/etc/Ignore** to list directories and files to exclude from backup.

Thus, after adding `/home/myacct/messages` to the `/etc/Backup` file, you might add `/home/myacct/messages/junkmail` to `/etc/Ignore` to specify that this one file in `/home/myacct/messages` should not be backed up.

Media Preparation and Handling for Basic Backups

The `backup` program will give you an estimated number of diskettes or tapes that will be needed to perform the backup.

The `backup` program can handle multi-volume backups to a raw device. It will prompt you when it is ready for the next diskette or tape.

Backing Up to Tape

Tapes do not need to be formatted, but they should be rewound to the beginning of the tape before issuing the `backup` command.

Using the backup Command

The following sections contain information on how to use the `backup` command and its options.

Performing Complete Backups to Tape

To backup all user files and system files (except system software) to tape, enter

```
backup -t -c -d device
```

where `-t` specifies that the backup is to tape, `-c` indicates that a complete backup is to be performed, and `-d device` specifies the full path name of the tape device.

Performing Partial Backups to Tape

To backup all files (except system software) which have changed since the last complete or partial backup, to tape, enter

```
backup -t -p -d <device>
```

where `-t` specifies that the backup is to tape, `-p` indicates that a partial backup is to be performed, and `-d device` specifies the full path name of the tape device.

Backing Up Selected Files to Tape

To backup selected files to tape, enter

```
backup -t -f "file1 file2" -d <device>
```

where **-t** specifies that the backup is to tape, **-f** has the full path name of one or more files to be backed up (within double quotes if more than one file), and **-d** *device* specifies the full path name of the tape device.

Metacharacters may be used, for example

```
backup -t -f "/sample/*" -d <device>
```

In this example, all files in the **/sample** directory would be backed up to tape.

Backing Up User Home Directories to Tape

To backup selected user home directories to tape, enter

```
backup -t -u "user1 user2" -d <device>
```

where **-t** specifies that the backup is to tape, **-u** specifies the user name of one or more users (within double quotes if more than one user), and **-d** *device* specifies the full path name of the tape device.

If **-u** all is specified, all user home directories on the system will be backed up.

Using the Extended Backup Service

The extended backup service allows you to make copies of the data on your system and the partitioning information on your disk so you can automatically restore all disk formats or any data later lost from your system. Thus, the data to be copied can be in the form of a file system or a data partition.

The data or partitioning information to be copied is referred to as an "originating object." Your copy (or "archive volume") of an originating object is stored on media such as diskettes or cartridge tapes that are known as the "destination media." In the course of a typically large backup job, a computer operator must mount several such media on the "destination device" (that is, the tape drive or diskette drive).

The extended backup service runs attended or unattended backups, and also includes tools for creating on-line history reports of your backups and status reports on current backup jobs.

Suggestions for Performing Extended Backup Operations

This section suggests a way to approach your backup duties. It describes the

- type of planning required

- steps involved in preparing for a backup
- steps for backing up a system

Most of the effort required to use the backup and restore services is needed to prepare the backup procedures. Once this is done, performing backup and restore operations is straightforward.

The system administrator and computer center operator play different roles in the extended backup service. On small systems, one individual may perform both roles; on large systems, different people are usually assigned these areas of responsibility. The responsibilities of each are described below.

The System Administrator's Tasks

The following is a detailed list of the administrator's responsibilities for the extended backup service.

1. Establish a backup plan (see "*Establishing a System Backup Plan*"). This plan should specify the following:
 - backup policies for a site based on factors such as resources, the needs of the users, and management directives
 - a list of file systems and data partitions that should be backed up, and for each file system or data partition, the intervals at which the backups should be done, the backup method to be used, and the types of destination devices to be used
 - how long and where destination media are to be kept before being reused
2. Set up backup tables that provide the computer with instructions for implementing your plan. (See "*What Is a Backup Table?*" below for instructions.)
3. If your plan includes incremental file system backups and you want to exclude certain files from those jobs, you must create an exception list. (See "*Excluding Files from Incremental Backups*" under "*Specifying Backup Methods*," below, for instructions.)
4. Either schedule backup jobs that will be invoked automatically, along with reminder messages (see "*Establishing a System Backup Plan*"), or invoke backup jobs manually (see the appropriate section of "*Using the Extended Backup Service*" earlier in this chapter for instructions).
5. When performing backup operations, keep the following considerations in mind:
 - You must have enough destination media to complete the backup.
 - You must allow enough time to complete the backup.
 - All removable media, such as cartridge magnetic tapes, must be labeled using **labelit (1M)**.

6. Check the status of backup jobs (see “*Using bkoper to Monitor Backup Jobs*”).
7. Evaluate your backup operations by examining the backup history log (see “*Using bkhistory to Display the Backup History Log*”). You may want to revise your plan on the basis of this evaluation.
8. Check whether the originating and destination devices have entries in the device database `/etc/device.tab`. If not, you must add entries to this table before the backups will work. You can check whether the devices are present by using the `getdev` command. This lists all the devices in the device database. To add devices to the database, use the `putdev` command as described in the “Managing Storage Devices.” chapter in this book.

The Operator’s Tasks

The operator performs any attended or demand backup jobs scheduled by the administrator. During interactive backups, the operator must respond to system prompts, and mount and remove destination media.

Establishing a System Backup Plan

Your first task regarding backups is to establish a system backup plan that specifies the following information:

Backup Plan Information

- Which objects need to be backed up?
- How often should the objects be backed up?
- Which is the most appropriate type of destination medium for storing the archive volume being made?
- How many destination media do you need for various backup methods (see “*Previewing Backup Operations*”)?
 - The number of cartridge tapes and/or diskettes you need will depend on the size of your local system. Each tape must be labeled before using.
 - As an example of the ratio of file system blocks to backup media, suppose you need to back up the `/usr` file system on your computer and it contains 13,294 blocks. You will have to allocate either one cartridge tape or a combination of 45 diskettes for daily incremental backups and 40 diskettes for monthly full file backups.
- How much time do you need for various backup methods?

- As a rule of thumb, allow eight to ten times as much time in your plan for a full file backup as you do for an incremental file backup.
- To calculate the amount of time you will need, figure out how many destination media you will need for a particular operation (see the previous item in this list), and then calculate the amount of time required when you know how long it will take to write to a certain type of medium.
- How many files are being copied during your incremental file backups?
 - If the majority of files in a file system are being copied during incremental file backups, or if an incremental file backup takes almost as long as a full file backup, consider scheduling full file backups more frequently.
- How much time are you willing to devote to restoring a file system?
 - If your plan relies heavily on incremental file system backups, you should keep in mind that all the destination media for each incremental backup may be needed to restore a file system to a consistent state.
 - Which are the most appropriate methods for backing up an object?
 - Should a backup be invoked automatically or manually?
 - In what order should various backup operations be performed?
 - Whether to validate the archive during backups (see the section “*Specifying Backup Methods*”).
 - Where and how long should backup archives be kept?
 - You may want to save daily incremental backups for a week, weekly full backups for a month, and monthly full backups for a year or indefinitely.
 - You may want to save some volumes off site to ensure that no one will accidentally overwrite an important archive.
 - Where should the historical records of completed backups be stored?

Getting Backup Plan Information

To answer these questions, begin by observing how the resources on your computer (such as disk space and CPU time) are used.

- You need to know which file systems and data partitions are used and how they are used.
- Consider the approximate rate of change in the file systems studied.
- Notice whether changes occur throughout the file system or in only a small percentage of the files.

- If change is frequent and widespread, it may be best to schedule nightly incremental backups and weekly full backups.
- If change is concentrated in just a few files, a weekly incremental backup and a monthly full backup may be sufficient.
- It is a good idea to re-evaluate, periodically, your system resources and how they are used. Keep in mind that if these periodic reevaluations show the use of your computer is changing, you may revise your backup plan.

Using bkreg to Set Up Backup Tables

After you have established a plan, your next job is to prepare for backup operations by setting up your backup tables. In these tables you must define the parameters that control backup jobs, such as which file system is to be backed up and the day of the week on which the backup is to be done. This section explains how to

- define the necessary parameters in your backup tables
- validate the contents of your backup tables
- change those parameters

What Is a Backup Table?

Each backup table defines the following parameters:

- the backup operation tag
- the rotation period (the length of time after which a backup operation should be repeated)
- the days of the week on which the backup operation is to be done
- the backup method to be used
- the priority level of the backup operation
- the origination device from which the backup is to be made
- the destination media on which the backup is to be written
- the volume name of each tape if CMTs are used as the destination media

The system supplies a backup table with default values that can be changed. You can use this table or create your own. (If you create your own, you can also assign default values that can later be changed.) Either type of backup table can be set up and maintained through the **bkreg** command.

Validating Backup Tables

Before the operations listed in a backup table can be performed, the backup service checks for the consistency of several items (such as partition information) between the destination device and the backup table. Consistency is validated when the **backup** command is invoked. If **backup** is invoked and any of the consistency checks fail, **backup** terminates.

1. If you prefer to validate the consistency of items in the system-supplied backup table manually, enter

```
backup -n -e
```

This command processes the backup operation without actually running it. By doing this before running the backup, you can avoid a **backup** failure.

2. You can perform the same step for a custom backup table by entering

```
backup -n -e -t table
```

Displaying the Contents of the Backup Tables

If you want to check your backup tables before requesting a validation check, request a display of the contents of your tables. A display consists of a set of entries, each of which defines a backup operation. The following fields are available for display:

rperiod	the number of weeks in the rotation period
cweek	the week in the rotation period to which the current week corresponds
tag	a unique identifier associated with the backup operation
oname	the pathname of the originating object
odevice	the device name of the originating device
olabel	the volume label of the originating device
week	the weeks, during the rotation period, in which the backup operation is performed
day	the days of the week on which the backup is performed
method	the backup method used
moptions	the options associated with a particular backup method
priority	the priority level for this backup operation
depends	tags for backup operations that must be completed successfully before this backup operation can begin
dgroup	the device group for a destination device

ddevice	the device name of a destination device
dchar	characteristics of a destination device
dlabel	the volume labels for a destination device

1. To display a complete table, enter

bkreg -A

The **-A** option can be followed by one or more of the following options: **-h**, **-s**, **-v**, **-t**, and **-c**.

The output for this command is a set of extremely long lines; it is best used as input to a filter. To obtain a display that is easier to examine, enter

bkreg -C fields

2. Specify only those fields from the list above that you want to see. For example, you may want to display only the backup tags, the weeks of the rotation period, the backup methods used, dependencies, and priorities. If you enter

bkreg -C tag,week,method,depend,prio

the following information is displayed:

```
Tag:Weeks:Method:Depends:Pri
rootsun:1-8:ffile::
rootsp:r,8:ffile::20
usrdai:1-8:ffile::10
usrsun:1-8:ffile::
medrecs3:demand:ffile:mainofc:4
mainofc:demand:ffile::6
newusr2:demand:ffile::2
```

The **-C** option can be followed by one or more of the following options: **-h**, **-v**, **-t**, **-F**, and **-c**.

3. You may also tailor the information displayed by using either the **-O** or the **-R** option to the **bkreg** command.
 - **bkreg -O** allows you to display a summary of all originating objects in the table.
 - **bkreg -R** accesses a summary of all destination devices in the table.

The **-O** and **-R** options, like the **-A** option, can be followed by one or more of the following options: **-h**, **-s**, **-v**, **-t**, and **-c**.

Specifying Custom Backup Tables

The system-supplied backup table is named `/etc/bkup/bkreg.tab`.

For a small system with limited backup operations, this table may be adequate. For large systems consisting of many computers, however, it may be more effective to create several backup tables of your own and sort the required backup operations into the different tables. For example, an administrator responsible for ten computers linked in a group called “services” and twelve computers linked in a group called “accounting” might set up two backup tables called `services` and `acctg`.

Using bkreg to Customize Backup Tables

These tables would be created by issuing the `bkreg` command followed by the `-t` option. The `-t` option would appear as follows:

```
bkreg -t /etc/bkup/services . . .
bkreg -t /etc/bkup/accts . . .
```

NOTE

The `-t` option to the `bkreg` command can be used in combination with any other options (such as `-a` or `-e`) whenever you want to work with a custom backup table.

You can create your custom backup table in any directory, but it might be convenient to put all backup tables in the `/etc/bkup` directory, where the system-supplied backup table resides.

Adding or Changing Backup Table Entries

There are three ways to change the contents of a backup table. You can

- add a new backup operation to the table
- modify the instructions for a backup operation already defined in the table
- remove a backup operation from the table

Adding an Operation Entry

1. To add a new backup operation to the table, enter

```
bkreg -a tag
```

where `tag` identifies a backup operation. `tag` can be an alphanumeric string of any length. The `-a` option must be followed by the `-o`, `-c`, `-m`, and `-d` options and, if you choose, any of the following options: `-b`, `-t`, `-P`, and `-`

- D. If you do not include **-b**, **-t**, **-P**, or **-D**, default values for them will be used.
2. The following example shows how to add an entry to a custom backup table.

```
bkreg -a acct5 -o /usr:/dev/rusr/c1d0s2:usr \  
-c 1:0 -m incfile -b "-t -x" \  
-d /dev/fd/0::cap=1422:acctwkly1,acctwkly2,acctwkly3 \  
-t /etc/bkup/wklybu.tab
```

This command line allows you to add an entry for a backup operation named **acct5** to a backup table named **wklybu.tab**. (If **wklybu.tab** does not already exist, it will be created.) The originating object to be backed up is the **/usr** file system on the **/dev/rusr** device. The backup operation defined here will be performed every week on Sunday using the incremental file backup method.

The method options specify that a table of contents will be created on a destination device. The backup will be done to the next available destination device using the devices labeled **acctwkly1**, **acctwkly2**, and **acctwkly3**.

These devices have a capacity of 1422 blocks each.

Modifying an Existing Operation Entry

1. To modify an existing definition (in a backup table) of a backup operation, enter

```
bkreg -e tag
```

where *tag* identifies the backup operation you want to modify.

tag can be an alphanumeric string of any length. The **-e** option should be followed by any other options for which you want to change the value. You can replace the values of arguments you previously specified for the following options: **-b**, **-c**, **-o**, **-D**, and **-P**.

2. In the following example command line, the values previously assigned with the **-o**, **-b**, and **-d** options are changed.

```
bkreg -e acct5 -t /etc/bkup/wklybu.tab \  
-o /usr:/dev/rusr:usr -m incfile -b "-t -x" \  
-d /dev/fd/0::cap=1422:acctwkly1,acctwkly2,acctwkly3
```

Removing an Operation Entry

To remove the entry for an operation from a backup table, enter

```
bkreg -r tag
```

where *tag* is the tag of the backup operation you want to remove.

tag can be an alphanumeric string of any length.

Specifying Backup Methods

The extended backup service uses a “backup table” to define the parameters governing the results of a backup. And one of the most important parameters you must define in the backup table is the extended backup method to be used.

There are six methods for backing up files, directories, file systems, and data partitions

- full file system (**ffile**)
- incremental file system (**incfile**)
- full image (**image**)
- full disk (**disk**)
- full data partition (**fdp**)
- migration backups

You can limit your backups to one of these methods or you can use several methods, at different times, to achieve a comprehensive backup strategy for your system.

Determining Which Backup Method to Use

Each backup method is characterized by how it answers the following questions:

- What type of information is copied? (For example, does this method allow you to copy files, data partitions, or both?)
- How much information is copied? (For example, does this method allow you to copy only individual files or a complete file system?)
- How is information copied? (For example, is information copied as it appears logically on the originating device or is it copied in raw format, that is, byte by byte?)
- How long does a backup take when you use this method?
- What kind of procedure must be followed to restore information that has been backed up with this method? (See the second part of this chapter that discusses Restore Services for details.)

You should consider each of these questions when deciding which method to use.

NOTE

If you are backing up the core file system, you must use either the full file backup method or the incremental file backup method; any other method may corrupt your archive volume. For details about core file systems, see “*Requesting Core File System Backups*” later in this chapter.

The following sections will tell you how to perform the six different backup methods listed above. All backup methods use the **bkreg** command and its various options. (See the **bkreg (1M)** manual page for complete information on the command and its options.)

Full File System Method

To request a full file system backup, enter

```
bkreg -m ffile
```

The full file system backup method copies all directories and files of a specified mounted file system to a destination device. The files are copied in the hierarchical order reflected by the directory structure.

Incremental File System Method

To request an incremental file system backup, enter

```
bkreg -m incfile
```

Incremental file backups copy only those files and directories that have been changed since one of the following:

- the last full file or full image backup
- the last full file or full image backup plus the last incremental file backup
- the last *n* days

Therefore, an incremental file backup must be preceded by at least one full backup that can be used as a base. Incremental file backups are useful for file systems that are changed frequently.

The following additional method options are available with the incremental file system backup method:

-p mode	Shows which type of incremental file backup is to be performed, where <i>mode</i> is optional and can be any of the following:
0	the previous full file or full image
<i>n</i> (>0)	the last <i>n</i> days
no mode specified	the previous full file or full image

- x Ignores the exception list; backs up all changed or modified files. (See “*Excluding Files from Incremental Backups*” below.)

Excluding Files from Incremental Backups

The incremental backup method is not recommended for all types of files.

- It is not useful for copying temporary files (such as `/usr/tmp`, `/etc/utmp`, and `/tmp`), or files that can be reconstructed from other files (such as any `.o` files, `core` files, `a.out` files, and `nohup.out` files).
- Files created by users (such as `dead.letter`, `junk`, `trash`, and `testing`) might not need to be backed up
- Files that are routinely changed or created daily might not need to be backed up because they are invalid or outdated on subsequent days.

The incremental file backup method excludes these types of files automatically.

Creating an Exception List

Each time you run an incremental file backup, **backup** examines a list of files to be excluded. Use the exception list provided by the system (`/etc/bkup/bkexcept.tab`) or create your own.

Each entry in the exception list specifies, in the format of a *pattern*, a set of one or more files. A *pattern* may consist of either the name of a single file or a string that includes one or more shell special characters (`*`, `?`, or `[]`) and thus represents multiple files. Special characters are similar to those use in shell commands.

Consider the following when creating an exception list:

1. Review your list to make sure it contains all the files you want to exclude and doesn't specify files you want to have copied. Modify the exception list accordingly, if necessary. (For instructions, see “*Modifying an Exception List*” below.)
2. If you do not want any files to be excluded from your incremental file backup (that is, if you want the exception list to be ignored), enter

```
bkreg -b -x
```

when setting up your backup table.

For details about exception lists, see the **bkexcept (1M)** manual page.

Creating a Customized Exception List

You can create your own custom exception list through the **bkreg** command, as shown in the following example:

```
bkreg -e acct3 -m incfile -b "-e file"
```

where *file* is the full pathname of the exception list to be created.

Once this entry has been created, the backup service will use the specified exception list for the backup selected.

Modifying an Exception List

Use the same procedures to modify either a system-supplied exception list or one you have created. The **-t** option to the **bkexcept** command must be used each time a list other than the default list is referenced. If the **-t** option is not used, the default list is referenced.

1. To add entries to an exception list, enter

```
bkexcept -a pattern
```

where *pattern* is a string that represents a file or a set of files, as defined on the **bkexcept (1M)** page. Items in a list of patterns must be separated by commas or by blank spaces. (Items separated by spaces must be enclosed in double quotes.) For example, suppose you want to add entries for the following items (that is, you want to exclude these items from your incremental backups):

- all subdirectories and files under the **/tmp** directory
- all subdirectories and files under the **/usr/tmp** directory
- any file named **junk**
- the user file named **/usr/accts/clerk3/oldfile**

Enter the following command line:

```
bkexcept -a \  
/tmp/\*,/usr/tmp/\*,\*/junk,/usr/accts/clerk3/oldfile
```

NOTE

If a command does not fit on one line, escape the new line character between multiple lines by entering a backslash (\) at the end of every line except the last. If the command syntax requires a space between elements in the command, leave a space before each backslash.

As shown in this example, special characters (such as *) must be preceded by a shell escape character (such as the backslash shown above). The escape character prevents the shell from expanding the special character. If you prefer not to use escape characters in your command line, enclose the string of arguments to **-a** in quotation marks, as follows:

```
bkexcept -a "/tmp/*,/usr/tmp/*,*/junk,/usr/accts/  
clerk3/oldfile"
```

Another way to avoid using escape characters on the command line is by entering a list of *patterns* from standard input. To do this, first enter a dash in place of the *pattern* argument on the command line. Then specify the

desired *patterns* on separate lines. To end your list, type `<CTRL><d>`. The following example shows how to specify *patterns* in this way.

```
$ bkexcept -t /etc/save.d/except -a -
/tmp/*
/usr/tmp/*
/usr/spool/*
*/trash
/usr/accts/clerk3/oldfile
<CTRL><d>
$
```

2. To remove an entry from your exception list, enter

```
bkexcept -r pattern...
```

where *pattern* matches an entry in the exception list. For example, to remove `/usr/spool/*` and `/usr/rje/*` from the exception list, enter the following:

```
bkexcept -r /usr/spool/\\* ,/usr/rje/\\*
```

You can remove entries by

- listing *patterns* on a command line (as shown above)
- specifying them on separate lines, using a dash for the value of *pattern*.
- To end your list, enter `<CTRL><d>`, as shown in the following example:

```
$ bkexcept -r -
/tmp/*
/usr/tmp/*
/usr/spool/*
*/trash
/usr/accts/clerk3/oldfile
<CTRL><d>
$
```

- When you have added or removed entries in the exception list, you may want to examine the list. To display the full contents of the exception list (in ASCII collating order), enter

```
bkexcept
```

with no options.

- To tailor the display, enter

```
bkexcept -d patterns
```

The following example shows how such a display appears on your screen. If you enter

```
bkexcept -d/usr,/usr/spool
```

your output might appear as follows:

```
Files in the exception list beginning with /usr:  
/usr/adm/*  
/usr/asp/*  
/usr/at/*  
/usr/crash/*  
/usr/games/*  
/usr/news/*  
/usr/rje/*  
/usr/spool/*  
/usr/tmp/*  
  
Files in the exception list beginning with /usr/spool:  
/usr/spool/*
```

- For exception list files containing only a <CR> character, enter

```
bkexcept -d/usr
```

This returns successfully with a null exception list. For files of zero length (no characters), **bkexcept -d/usr** returns the message

```
search of table failed
```

Converting Exception Lists from Earlier Backup Services

Previous versions of the backup service created exception lists using **ed** syntax. The **bkexcept -C** command translates the entries in these earlier exception lists to the new shell pattern format. The translation is not perfect; not all **ed** patterns have equivalent shell patterns. For those patterns that have no equivalents, an attempt at translation is made, and the translated version is flagged with the word **QUESTIONABLE**.

Because the translation is not perfect, you need to edit the output of the **bkexcept -C** command before adding it to the default exception list for the current backup service. The following procedure explains how to do this.

1. The translation of the exception list is directed to standard output. Redirect the standard output to a file, such as **checkfile** in the following example:

```
bkexcept -C/etc/save.d/except > checkfile
```

The exception list from the previous version of the backup service specified in this example is **/etc/save.d/except**. Before being converted, the contents of this file appear as follows:

```
# Patterns of filenames to be excluded from saving by savefiles.
# These are ed(1) regular expressions.
/.news_time$
/.yesterday$
/a.out$
/core$
/dead.[a-l]*$
/ed.hup$
/nohup.out$
/tmp/
.o$
^/etc/mnttab$
^/etc/save.d/timestamp/
^/etc/utmp$
^/etc/wtmp$
^/usr/adm/
^/usr/at/
^/usr/crash/
^/usr/dict/
^/usr/spool/
^/usr/tmp/
```

After this file has been converted by **bkexcept -C**, and you have redirected the output to **checkfile**, the contents of **checkfile** appear as follows:

```
QUESTIONABLE: # Patterns of filenames to be excluded from saving by
savefiles.
QUESTIONABLE: # These are ed(1) regular expressions.
*/.news_time
*/.yesterday
*/a.out
\core
*/dead.[a-l]*
*/ed.hup
*/nohup.out
*/tmp/*
*.o
/etc/mnttab
/etc/save.d/timestamp/
/etc/utmp
/etc/wtmp
/usr/adm/*
/usr/at/*
/usr/crash/*
/usr/dict/*
/usr/spool/*
/usr/tmp/*
```

2. Review the contents of the new file (**checkfile** in this example). Notice that the **QUESTIONABLE** flag is used for two purposes: to mark comments in the old file, and to draw your attention to entries that may not have been translated properly.
3. Review the entries that may not have been translated properly. Revise those entries that are not in the correct format by deleting the **QUESTIONABLE** flag and modifying the pattern as necessary. If you decide that a translation is adequate, you need only remove the **QUESTIONABLE** flag.

4. After editing the entries in the converted file (**checkfile** in this example), add the contents of this file to the current exception list (**/etc/bkup/bkexcept.tab**). To do this, specify the converted file on the **bkexcept -a** command line, as shown in the following example:

```
bkexcept -a - < checkfile
```

Full Image Method

A full image backup copies an entire file system, byte-for-byte, starting with the first block and ending with the last. A full image backup differs from a full file backup because it does not copy the file system according to its directory structure. Instead, a full image backup copies the data blocks in the order in which they appear on the disk, from the first segment to the last segment.

The **fdisk** method backs up the disk VTOC (Volume Table of Contents) and file system characteristics [as defined by **mkfs (1M)**]. If the disk is destroyed, the contents can be recovered by

- restoring the disk configuration using the **fdisk** method
- restoring data partitions using the appropriate backup method for the partition

To use this method, enter

```
bkreg -m fimage
```

Full Disk Method

The full disk backup method allows you to copy all the information required (by the restore service) to recover the format of an entire disk. Typically, the disk format is restored, followed by individual file systems and, finally, by the data partitioning information.

Because **fdisk** only backs up disk format information, you should use one of the other methods to back up the data on the disk.

To use this method, enter

```
bkreg -m fdisk
```

Full Data Partition Method

A full data partition backup allows you to copy a data partition that contains objects other than file systems, such as databases. Also, it allows you to copy a raw data partition, byte-for-byte, starting with the first block of the data partition and ending with the last.

To use this method, enter

```
bkreg -m fdp
```


Requesting Migrations for Backed-Up Information

Migration is the process of moving an existing archive to a new location. The archive may have been created by any of the preceding backup methods and its new location is recorded in the backup history log. Any restore operation done with a migrated object must be done with the restore method appropriate for the backup method by which the archive was originally created.

A migration cannot be done until a backup operation has been performed. The fact that an operation has been performed implies that an entry for that operation exists in the backup table. That entry includes values for the originating device and destination device.

When to Do a Migration

Migrations are useful when factors such as staffing, machine cycles, and the availability of destination devices vary over time. For example, you may want to schedule an automatic incremental file system backup to a spare disk partition at a specified time, and later have an operator move the resulting archive to tape.

Specifically, suppose you want an incremental file backup for the operation known as `/usr:/dev/usr:usr` as the originating device as the destination device; you also want the backup performed the Friday of week one of the rotation period.

To request this operation,

1. Add an entry to the backup table.
2. Enter `bkreg` with these options:

```
bkreg -a mktg3 -c 1:f -m incfile e-o /usr:/dev/usr:usr -d :
/dev/usr:
```

Once this entry exists in the backup table, the `crontab` is updated to run `backup` at the required time, the information saved by the backup (the archive) will be stored in the designated destination device.

Migrating an Archive

1. To migrate the archive, edit the entry for the relevant operation in the backup table, specifying a migration, and the new destination for that archive. Edit the entry for the `mktg3` operation by entering a command line such as the following:

```
bkreg -e mktg3 -m migration e
-o oname:/dev/usr:e
-d ctape1::mktgwklyA,mktgwklyB
```

As shown here, the destination device specified for the incremental backup is also specified as the originating device for the migration. `oname` is ignored by the backup command for this type of migration.

2. Once you have edited the specified entry in the backup table, an operator can request the migration with the `backup` command, as long as the migration requested is limited to the originating device you have specified in the table. In this example scenario, you enter

```
backup -i -o oname
```

3. This is what happens as a result of the previous command line:

- An existing archive is stored on a new destination device (from which it can be restored in the same way any other archive is restored).
- The backup history log is updated to show the new location of information that has been migrated.
- The table of contents (for the archive that has been migrated) is updated to show the new location of the archive.

Requesting Core File System Backups

Core file systems are different from other file systems because they contain system software and, therefore, they must always remain mounted even when they are being backed up and restored. (For a description and list of core file systems, see the “Managing File System Types” chapter in this book.)

A backup done on a core file system should be done on a “demand only” basis; specify demand in the **bkreg** table entry for each core file system backup.

Restrictions on Backing Up Core File Systems

Unlike other file systems, core file systems must be backed up only with the **-m ffile** option or the **-m incfile** option to the **bkreg** command. Do not use the full image backup method on a core file system because changes made to files in this system during this type of backup can corrupt the resulting archive. Running an incremental file backup or a full file backup is less dangerous to a core file system because changes made to a core file system during one of these types of backups will not corrupt your destination archive; at worst, you may get an intermediate file copy, or one that is not up to date.

Suppose you want to add an entry with the tag **usrdai** to the default backup table. The originating object to be backed up is the **/usr** file system on the **/dev/rusr** device (which is labeled **/usr**). You want to use the incremental file backup method (with the **-m incfile** option) on the next available diskette device using the three diskette volumes **usrdai1**, **usrdai2**, and **usrdai3**. (These volumes have a capacity of 1422 blocks.) Enter the following command line:

```
bkreg -a usrdai -o /usr:/dev/rusr:usr \  
-c demand -m incfile \  
-d diskette::cap=1422:usrdai1,usrdai2,usrdai3
```

See “Incremental File Backups” under “Extended Backup Methods and When to Use Them” for information on the options listed in the above example.

Specifying Originating Objects

You can define the originating object for a backup operation by entering

```
bkreg -o orig
```

The *orig* argument takes the following form:

```
oname:odevice[:olabel]
```

The following is an example of the `-o orig` option:

```
-o /usr:/dev/rusr/:usr
```

Each component of the argument *orig* is defined below:

<i>oname</i>	The pathname of the originating object. For file system partitions, this is usually the node name on which the file system is mounted. For data partitions, it is any valid pathname. This value is provided to the backup method and validated by the backup command. The data partition backup methods (invoked with the -m fdp and -m fdisk options to bkreg) do not require the name of the originating object; the ffile and incfile methods require <i>oname</i> .
<i>odevice</i>	The raw disk partition device name for the originating object.
<i>omname</i>	The volume label for the originating object. For file system partitions, it corresponds to the <i>volumename</i> specified with the labelit command. A data partition may have an associated volume name. If it does, the name is known only externally (taped on the device); run the getvol (1M) command to validate the name. Not all file system types support <i>omname</i> . See the “Managing Storage Devices” chapter in this book for a list of those that do and for details about volume names and the labelit command.

Specifying Destination Devices

All backup operations require a destination device on which an archive volume can be stored. To specify a destination device, enter

```
bkreg -d ddev
```

where *ddev* takes the form

```
dgroup:ddevice[:dchar][:dlabel]
```

Here, both *dgroup* and *ddevice* must be specified and *dchar* and *dlabel* are optional. Colons separate fields and must be included as shown above. *dgroup* is the device group for the destination device. (For a description of device groups, see the “Managing Storage Devices” chapter in this book.)

If *dgroup* is not specified, *ddevice* must be specified and any available destination device in *ddevice* will be used.

dchar describes the characteristics of a destination device and, if specified, it overrides the default characteristics for the device and group. These characteristics are found in the device table, **/etc/device.tab**. The critical characteristics are type and capacity; these should be specified with *dchar*. (For details about the format and meaning of *dchar*, see the “Managing Storage Devices” chapter earlier in this manual.)

NOTE

If device characteristics for a backup or restore device are not specified in `/etc/device.tab`, they must be specified in the backup register table.

If the destination device is a disk partition, you must make a special entry for it in the device table. This entry must include all the fields in a normal entry plus the following two fields: `type` (where `type=dpart`) and `capacity` (where `capacity=n` and `n` is the maximum number of bytes allowed for that partition). No other fields are allowed in this entry.

`dlabel` is a list of names of the destination media. The items in this list must be separated either by commas or by blank spaces. (Items separated by blanks must be enclosed in double quotes.) Each name in the list corresponds to a `volumename` specified with the `labelit` command. (For details about destination device labels and the `labelit` command, see the “Managing Storage Devices” chapter in this book.) If `dlabel` is omitted, the `backup` and `restore` commands do not validate the labels on the destination devices.

Specifying the Rotation Period

A rotation period is the number of weeks between invocations of a backup operation. The rotation period for every backup operation is assumed to be one week (the default period) unless it is specified otherwise in the backup table. To set the rotation period for the system-supplied backup table, enter

```
bkreg -p period
```

where *rperiod* is an integer between 1 and 52 that represents the number of weeks between backups.

1. To set the rotation period in a custom backup table, enter

```
bkreg -t -p rperiod
```

NOTE

The `-p` option to `bkreg` cannot be used with any other options on the same command line.

2. By default, a rotation period always begins on a Sunday, but you can change that parameter by entering

```
bkreg -c weeks:days
```

where *weeks* is a list of one or more integers between 1 and 52, that cannot exceed the value set by the `-p period` option.

days is a list of integers between 0 (Sunday) and 6 (Saturday), or a list of characters from *s*, *m*, *t*, *w*, *th*, *f*, *sa*.

Both the *weeks* argument and the *days* argument can be specified as either a list of individual items (such as 1, 3, 5) or as a range of items (such as 1-3).

The items in each list can be separated by either commas or blank spaces (in which case the list is enclosed in double quotes).

For example, if you want a backup operation to be performed every Sunday, Tuesday, Wednesday, Thursday, and Saturday on the first, second, third, and fifth weeks of the year, specify these times as shown below:

```
bkreg -a svcs -m incfile -c 1-3,5:0,2-4,6
```

- Another way of defining the rotation period for a backup operation is by requesting that this operation be performed only on demand. Operations for which this request has been made are not performed regularly; they can be performed only when the **backup** command is invoked with **-c demand**. The following example command line shows how to request demand only status for all the backup operations listed in the **services** table.

```
bkreg -c demand -t /etc/bkup/services
```

Establishing Dependencies and Priorities

Some backup operations should be started before others begin. Some backup operations should not be run at all until other backups have been completed. The backup service lets you handle both these situations by providing a way for you to establish priorities and dependencies when setting up backup tables.

- You may want to list your backup operations in the order you want them to run. To do this, assign a priority level to each backup operation defined in the backup table by using the **-P** option and specifying a priority level. The priority level is an integer from 0 to 100 where 0 is the lowest priority and 100 is the highest priority. For example

```
bkreg -P 50
```

If a set of backup operations is to be performed at the same time, each backup operation is not started until all others with a higher priority are completed. All backup operations with the same priority can be done simultaneously, unless the priority is 0. All backups with a priority of 0 are performed sequentially in an unspecified order.

- You can also specify that a backup operation not be started until a set of other backup operations is completed successfully. These dependencies must be identified in the backup table. To add a list of dependencies to the backup table, enter

```
v bkreg -D depends
```

depends is a list of the operation tags for the operations on which a particular backup operation depends. Items in the list must be separated with either commas or blank spaces. (Items separated by blanks must be enclosed in double quotes.) Establishing dependencies is particularly useful using the migration method.

3. A backup operation's dependencies take precedence over a backup operation's priorities. For example, an administrator may have a backup operation called **acctswkly** which is dependent on completion of the backup operation **SysengFri**. However, **SysengFri** has a priority of 40, which is less than the priority of **acctswkly1** (50). According to the rules of priorities, **SysengFri** should not begin before **acctswkly1**. But because dependencies take precedence over priorities, the **SysengFri** backup operation will be performed before the **acctswkly1** backup.

To check the priorities and dependencies of these backup operations, enter

```
bkreg -C tag,prio,depend
```

This is what would result

```
acctswkly1:50:SysengFri
SysengFri:40:
```

Creating Tables of Contents

Another item you can specify in the backup table is a table of contents that lists the files and directories on a particular destination device.

NOTE

A table of contents is used by the restore service to locate files and directories to be restored.

1. Tables of contents can be provided only for backup operations performed with the incremental file, full file, or full image backup methods. Tables of contents are created and changed by using the **-s** and **-t** arguments to the **-m method** option. Because arguments to the **-m method** option must always be introduced on the command line by the **-b** flag, you would enter the **-s** and **-t** arguments as follows:

```
bkreg -m ffile -b "-s -t"
```

2. You can specify either a long-form or a short-form table of contents. The short form of a table of contents shows the names of the directories or files, and volumes that have been stored on a particular destination device. The long form of the table contains the same information as the short form, as well as the information about those directories or files that is normally provided by the **ls -l** command.

By default, the system creates the short form of the table of contents.

To create the long form, enter

```
bkreg -m method -b -l
```

method may be **ffile**, **incfile**, or **fimage**.

3. You can store a table of contents in any of three ways:

- on-line
- on removable destination volumes
- both on-line and on destination volumes

Alternatively, you can specify that no table of contents be stored. The location of a table of contents is controlled by three factors:

- the system default
- the **-s** argument to **bkreg -b**
- the **-t** argument to **bkreg -b**

These three factors can be used in any of the following combinations:

- To store a table of contents on-line only, use the system default; do not specify **-s** or **-t** after the **bkreg -b** flag.
- To store a table of contents on removable destination devices only, enter both **-s** and **-t** after the **-b** flag, as follows:

```
bkreg -m method -b "-s -t"
```

- To store a table of contents both on-line and on removable destination devices, enter **-t** after the **-b** flag, as follows:

```
bkreg -m method -b -t
```

- To request that no table of contents be stored, enter **-s** after the **-b** flag, as follows:

```
bkreg -m method -b -s
```

- To validate the archive during backup, enter **-v** after the **-b** flag as follows:

```
bkreg -m method -b -v
```

Using backup to Perform Backup Operations

You are almost ready to start running backup operations. Before you do, you will need to answer three questions:

- Which operator mode do you want to use?
- How much destination device space or how many destination archive volumes will you need?
- Do you want to limit your backup operation to a subset of the information normally copied during a defined rotation period (such as only today's files)?

This section defines each of these questions and explains how to find answers to them.

After you have selected an operator mode, set aside the required number of destination volumes, and decided which, if any, of the backup table values you want to override, you are ready to begin. You have already requested a backup method in your backup table. The rest of this section provides detailed descriptions of running backup operations using each backup method. It also explains how to back up a core file system (core file systems have special needs not associated with other file systems).

Selecting an Operator Mode

Once your backup tables have been created, you must decide whether your backup operation requires operator assistance. Usually an operator is needed to mount destination devices. Some backup operations, however, are small enough that they can be done without any help from an operator. To accommodate both situations, the backup service allows you to run backup operations attended or unattended. For an attended backup, an administrator (or operator) issues the backup command and provides any necessary assistance during the course of the backup job.

Unattended backups are useful if your backup jobs do not require an operator to mount multiple destination devices and if you want your site's backup jobs to be done during off-hours when the computer center is unstaffed.

An unattended backup operation is run without the help of an operator; it is invoked by the system at a time you have specified in the backup table. Attended backups are useful when operators are present and when you want flexibility in the time of day that backup jobs occur.

There are three operator modes in which you can run backups:

<code>background</code>	an operator is available but not at the terminal (the default mode)
<code>interactive</code>	an operator is available at the terminal throughout a backup
<code>automatic</code>	no operator is available

This section describes each mode and explains when you might want to use it.

Background Mode

If **backup** is invoked with no options, the background mode is used by default (this mode is normally set up to be run by **cron**). In the background mode, whenever a backup oper-

ation requires an operator's assistance, a **mail** message is sent to the operator's mailbox. The mail message reads as follows:

The following backup requires operator intervention:

job_id	tag	time	volume
back-441	acct3	09:35	/usr:/dev/rusr:usr

When a new medium is needed for a backup operation running in background mode, the backup operation is suspended until the operator receives mail, issues the **bkoper** command, and responds to the prompts, thereby enabling the job to proceed. Note that in this mode, this type of suspension delays completion of any scheduled backup operations that depend on this backup or that have a lower priority.

Interactive Mode

If an operator will be present at the terminal during backup jobs, you may want to run your jobs in interactive mode. When you use this mode, the system sends all prompts directly to the standard output of the terminal where the **backup** command was issued. Interactive mode allows the operator on duty to respond to the prompts as they arrive at the operator terminal, to insert or remove destination media as required, and to oversee the progress of the backup operation.

NOTE

If you are backing up data to a non-removable device, such as another hard disk, you will not receive any prompts to change.

1. To request interactive mode, enter

```
backup -i
```

2. If an operator will be present at the terminal during backup jobs, and wants to monitor an incremental file backup or a full file backup closely, the operator can ask the backup service to post the progress of the operation. This request is made by running a backup job in verbose mode.

This mode is a form of interactive mode, so to request it, use both the **-i** and the **-v** options, by entering

```
backup -iv
```

When a backup is run with this command line, the name of every file and directory being backed up is displayed on standard output.

3. If you want to track the progress of a backup in more detail, request special verbose mode by entering

```
backup -is
```

When a backup is run with this command, a dot is displayed as every 100 (512-byte) blocks are transferred to the destination device. If the data you are backing up is smaller than 100 blocks, you will not see any dots.

Running Automatic Backups

Many systems are small enough to be backed up (either fully or incrementally) overnight, to a single medium, without the assistance of an operator. This is convenient in a range of situations: you may want to do incremental dumps on a small system or backups of a medium-size system to a large capacity tape drive. In any such case, follow the steps described below.

1. Set up the backup register table (with the **bkreg** command) to implement a schedule of backups appropriate for your system.

For example, suppose you want to run a full backup, once a week, to a diskette labeled `wkly` and a daily incremental backup (from the last full—weekly—backup) to a diskette labeled `daily`. The file system being backed up is `/home`.

2. Enter

```
df /home
```

to get the name of the device on which `/home` resides.

3. Use the device name (`/dev/dsk/0s9` in this example) in the **bkreg** command line, as follows:

```
bkreg -a home.full -o /home:/dev/dsk/0s9 -c 1:0 \  
-m ffile -d diskette::wkly -b "-v -o"  
bkreg -a home.inc -o /home:/dev/dsk/0s9 -c 1:1-6 \  
-m incfile -d diskette::daily -D home.full -b  
"-i -v -o -p 0"
```

4. Label your media with the **labelit** command. For example:

```
labelit /dev/diskette home daily
```

5. Edit the cron table file (`cron.tab`) so the owner of the `root` login can run the **backup** command at a specified time. For example, suppose you want the **backup** command to be run every day at 3:30 A.M. and to have any output messages mailed to `root`. You will need the following entry in your cron table:

```
30 03 * * * echo "1\\\xd3 | backup -i -o /home 2>&1 |  
mail root
```

If you want to override label checking, use the following entry, instead.

```
30 03 * * * echo "1\\\xd3 | backup -i -o /home 2>&1  
| mail root
```

NOTE

Now your backup operation is set up to run. Check every day to make sure a properly labeled medium has been inserted; without one, your operation will fail.

6. Attempt a trial run to verify that tapes will be labeled and backups will be done properly. For example:

```
backup -i -o /home
```

7. Check the status of your backup operations every day. To verify the successful completion of an operation, use the **bkstatus** and **bkhistory** commands.

If a backup operation seems to be hanging and you think there's a problem with the archive medium, enter

```
ps -ef
```

to verify that this is the case. If it is, resolve the problem with the medium and cancel the current operation as follows:

```
$ bkstatus  
$ backup -C -j job-ID
```

Then, if you still want to run a backup, restart an interactive operation:

```
$ backup -i -o /home
```

Previewing Backup Operations

There may be times when you want to know the schedule of backup operations for a day without invoking any jobs. The backup service provides two preview capabilities

- preview the set of backup operations for a day
 - estimate the number of destination media required for a backup
1. To display the current day's backup operations in the order in which they would proceed if invoked (that is, according to priorities and dependencies), enter

```
backup -n
```

A list of operations will appear, as in the following example:

Tag	Orig.Name	Orig.Device	Dest.Group	Dest.Device	Pri	Depends On
rootdai	/	/dev/root	ctape		0	
usrdai	/usr	/dev/usr	ctape		0	

2. You can preview the same information for any day in a rotation period by adding the `-c week:day` option, as shown in the following example:

```
backup -n -c 3:5
```

This command line requests a list of backup operations scheduled for Friday (5) of the third (3) week in the rotation period.

3. After previewing the day's schedule of backups, you should find out whether you have enough space on your destination device or archive volumes before initiating a backup.

You can find out how many devices you need by entering

```
backup -ne
```

4. You can obtain a report that lists the scheduled backup operations for a specified day, along with an estimate of the number of devices required for each operation by entering

```
backup -ne -c
```

In addition to providing this information, the `-ne` option validates the consistency between corresponding items in the backup table and on the destination device. If any of the validation checks fail, the service sends an error message to standard error. This capability enables you to correct problems before initiating an actual backup.

Requesting Limited Backups

When you issue the `backup` command, usually all operations defined in your backup table for the current rotation period are performed. There may be times, however, when you want to run a backup without having all operations performed. For example, you may want to run only backup operations defined for demand (that is, operations that are not scheduled to be run regularly). This section describes ways in which you can limit the backup operations to be performed.

- To run only those backup operations scheduled for the current day, enter

```
backup -c
```

- To invoke backup operations scheduled for a day other than the current day, enter

```
backup -c week:day
```

where *week* and *day* are integers that represent the desired week of the rotation period and day of the week.

For example, if the current week is the eighth week of the rotation period, but you want to run the backup operations scheduled for Thursday of the seventh week, enter:

```
backup -c 7:4
```

- To run backups scheduled to run only on demand, enter

```
backup -c demand
```

- To run backup operations defined in a custom backup table, enter

```
backup -t table
```

- To back up objects on a particular originating device, enter

```
backup -o orig
```

where *orig* must be of the form *oname:odevice:[omname]*.

- To request that mail be sent to a specified user when the entire backup operation is complete, enter

```
backup -m user
```

The above options can be used in various combinations on the **backup** command line. For example, suppose you want to invoke those backups listed on your custom backup table (**/etc/bkup/accts.tab**) that are scheduled for Friday of the second week in the rotation period. In addition, when the backup job has been completed, you want mail to be sent to the user with login *supv3*. Invoke this operation by entering

```
backup -t /etc/bkup/accts.tab -c 2:5 -m supv3
```

Using bkoper to Monitor Backup Jobs

Backup jobs may require operator assistance for such tasks as

- mounting diskettes, cartridge tapes, and 9-track tapes
- checking the labels on destination media to verify that correct volumes are being used

An operator may perform a backup in any of three modes:

- background mode
- interactive mode
- automatic mode

(See “*Selecting an Operator Mode*” earlier in this chapter.) This section explains how an operator interacts with a backup operation being run in background mode.

When Backup Jobs Need Assistance

1. When a backup job in background mode cannot proceed further without the assistance of an operator, the **backup** command sends a **mail** message to the operator requesting assistance for that job.
2. To find out what needs to be done, enter

bkoper

The **bkoper** command responds by printing a list of backup jobs for which assistance is needed, such as those shown in the following example:

```
1. back-111 usrsun /dev/dsk/0s9 disk /dev/dsk/1s9 usrsave
2. back-112 fs2daily /dev/dsk/0s8 ctape /dev/rmt/0m-
```

Each entry contains the following:

- operation number (the initial digit followed by a period)
- backup job ID
- (the *back-*nmn** label)
- operation tag
- originating device
- destination device group
- destination device name
- archive volume label

In the example above, the dash displayed as the last item of the second entry shows that no specific volume label is required for the backup operation listed. Backup operations are numbered in the order in which they appear in the backup table.

NOTE

If you want to interrupt your session with **bkoper** to perform a task at the shell level, type **!** and enter the command you want.

To quit the **bkoper** session completely, type **q**.

3. The **backup** command then displays the following question:

```
Which prompt do you want to respond to?
Type [q] to quit bkoper
Type [h] to display the list of backup operations
Type a number or <CR> to service a backup operation
?
```

4. To find out what kind of assistance is needed for the first operation listed, press <CR> or type 1.
5. The **bkoper** command will respond by explaining the task that needs to be done. For example, if you press <CR> after seeing the list displayed above, you might receive a message such as the following:

```
Insert a diskette into the diskette drive. The diskette should
be internally labeled as follows:
```

```
    usrsave

Type [go] when ready
or [f] to format the diskette
or [q] to quit.
```

6. You can mount the volume shown (**usrsave**) or mount an unlabeled volume. You could also enter

```
bkoper -b -o
```

to override the request and mount any volume, regardless of whether or not it has been used before.

If the **-b -o** method option has been used, the previous display will appear as follows:

```
Insert a diskette into the diskette drive. The diskette should
be internally labeled as follows:
```

```
    usrsave

Type [go] when ready
or [f] to format the diskette
or [o] to use the current label anyway
or [q] to quit.
```

7. After performing the task requested, press <CR> to find out what kind of assistance is needed for the next operation listed.

8. If you want to service an operation other than the current one (that is, other than the one at the top of the list), you can do so with either the `p` (print) keyletter or the `t` (type) keyletter, followed by the number of the relevant operation.

For example, if you want to service the second operation before the current one, enter

`p2`

9. If, after the original list of operations has appeared, servicing is required for other operations (and you are still interacting through the `bkoper` command), the following message appears:

There are new backup operations requiring service.

10. When you have finished servicing all the operations that require assistance, the following message is displayed:

No more backup operations are waiting for operator action at this time.

Using `bkstatus` to Check Job Status

You can check the status of backup jobs (and the backup operations included in each job) by using the `bkstatus` command.

Each backup job progresses through the states listed below.

<code>pending</code>	The <code>backup</code> command has been invoked and the operations listed in the backup table for the specified day are scheduled to occur.
<code>active</code>	All backup operations have been assigned destination devices and archiving is currently underway, or a suspended backup has been resumed.
<code>waiting</code>	The backup job is waiting for operator assistance with a task, such as the mounting of a new tape.
<code>suspended</code>	The backup job has been suspended by an invocation of <code>backup -S</code> .
<code>failed</code>	The backup job has failed or has been canceled.
<code>completed</code>	The backup job has been completed successfully.

The status of backup operations is recorded in the `/etc/bkup/bkstatus.tab` table. In the table, each state is represented by the first letter of the relevant status: `p` (pending), `a` (active), `w` (waiting), `s` (suspended), `f` (failed), or `c` (completed). By default, only one week's worth of backup status information is saved on this table.

Displaying the Backup Status Table

You may display the backup status table in any of several ways. For all information about backup operations in progress (those labeled a, p, w, or s), enter

```
bkstatus
```

1. To include information about backups with a status of f (failed) or c (complete), enter

```
bkstatus -a
```

A display such as the following will appear on your screen:

Jobid Status	Tag	User	Oname	Odevice	Start Time	Dest	
-							
back-459	UsDly	oper1	/usr/spool	/dev/dsk/0s8	-	diskette	f
back-459	PtsDly	oper1	VTOCcd0	/dev/dsk/0s7	-	diskette	c
back-395	SysDai	oper2	/sys	/dev/dsk/0s9	May 26 16:45	diskette	c

2. If you don't need a full report, you can limit the types of information that are displayed.

For example, you can restrict a report to information about only specified jobs by entering

```
bkstatus -j jobids
```

This command will not show those operations that were completed or failed. In addition, all *jobids* must be of the form *back-number*.

3. To restrict a report to information about jobs in particular states, enter

```
bkstatus -s states
```

where *states* is a list of keyletters that are concatenated, comma-separated, or blank-separated (items separated by blanks are enclosed in double quotes), as shown in the examples below:

- apf
- a,p,f
- "a p f"

All three examples specify that the report should include information about only those backup operations that are active or pending, or that have failed.

4. To restrict the report to information about backup operations invoked by specified users, enter

```
bkstatus -u users
```

where *users* is a list of user logins separated by commas or blank spaces. (Items separated by blanks are enclosed in double quotes.) The report will not include completed or failed operations.

5. If you want backup status information to be saved for more than one week, enter

```
bkstatus -p n
```

where *n* is the number of weeks for which you want information to be saved.

You may find it useful to save status information for longer periods, such as a month, so you can examine patterns of servicing particular backup jobs.

Controlling Jobs in Progress

There may be times when you want to suspend a job, cancel a job, or resume a suspended job. You can request these actions by using the **backup** command options **-S**, **-C**, and **-R** (respectively), followed by the appropriate job ID.

- For example, suppose the backup job with the job ID `back-3288` is in progress when you need the destination device for another purpose. Suspend the job by entering

```
backup -S -j back-3288
```

Entering **backup -S** without a job ID suspends all outstanding backup operations that were begun by the user entering this command.

- Whenever the backup service receives a suspend request, it remembers the destination device currently in use, rewinds the destination device (if appropriate), and yields control of it. When the destination device becomes available for the backup again, you can resume the job by invoking

```
backup -R -j back-3288
```

Entering **backup -R** without a job ID resumes all outstanding backup operations that were begun by the user entering this command. The backup service re-validates the volume label and begins the backup from the beginning of the current volume, not from the point where the suspend request was received.

- To cancel the backup job begun in the example above, enter

```
backup -C -j back-3288
```

Entering **backup -C** without a job ID cancels all outstanding backup operations that were begun by the user entering this command.

In this case, the backup service rewinds the destination medium currently in use, and yields control of the destination device. In addition, the status display shows this backup job has been canceled.

If the **-j** option is used, only the backup job with the specified id will be affected. If the **-u** option is used, only backup jobs issued by the users with the specified logins will be affected. If **-A** is used, all backup jobs are affected.

Using bkhistor to Display the Backup History Log

The **backup** command automatically records all backup operations that have been completed successfully in a backup history log (**/etc/bkup/bkhistor.tab**). You can examine the log's contents through the **bkhistor** command. Entering **bkhistor** without any options displays a summary of log's contents that includes the following information:

tag	A unique identifier associated with a backup operation								
date	The date and time when a backup operation was performed								
method	The backup method used for the backup (incfile , ffile , fimage , fdp , or fdisk)								
destination	The name of the device on which the backup archive was created.” CHECK THIS TECHNICAL CHANGE: used to read “dmname”								
dlabels	The labels of the volumes on which the backup was done								
vols	The number of volumes required to hold the entire backup archive								
TOC	Whether a backup archive contains a table of contents and, if so, in what form. The four possible values of TOC are								
	<table> <tr> <td>online</td> <td>The TOC is present online (on the hard disk).</td> </tr> <tr> <td>Arch</td> <td>The TOC is present in a volume on a destination device (a removable medium).</td> </tr> <tr> <td>Both</td> <td>The TOC is present both online and on a destination device (a removable medium).</td> </tr> <tr> <td>None</td> <td>No TOC exists.</td> </tr> </table>	online	The TOC is present online (on the hard disk).	Arch	The TOC is present in a volume on a destination device (a removable medium).	Both	The TOC is present both online and on a destination device (a removable medium).	None	No TOC exists.
online	The TOC is present online (on the hard disk).								
Arch	The TOC is present in a volume on a destination device (a removable medium).								
Both	The TOC is present both online and on a destination device (a removable medium).								
None	No TOC exists.								

Backups are listed alphabetically by operation tag. When a backup operation has been performed more than once during the period reported in the display, the most recent backup is listed first. Screen 11-1 is a sample display of information from a backup history log:

Tag	Date	Method	Destination	Dlabels	Vols	TOC
rootdai	Feb24 12:26 1992	incfile	/usr2/tmp/mnt	cpio1, cpio3	2	Online
rootsun	Feb24 12:31 1992	ffile	diskette	!cpio1, rep1, !cpio3, !rep4	4	Archive
usr2dai	Mar02 23:41 1992	ffile	diskette	med1, med6	3	None
usrdai	Jan28 20:29 1992	incfile	file	med1, med2, med3	3	Archive

Screen 11-1. Sample Display of a Backup History Log

Note that some entries in the **Dlabel** field begin with the **!** character. This shows that the volume listed has been reused.

Customizing the Backup History Log Display

The **bkhistory** command allows you to customize both the contents and the format of a display.

Customizing the Contents of the Display

The default display contains full details about completed backup operations. You can restrict the type of information that is displayed by using the **-d**, **-t**, or **-o** options to the **bkhistory** command.

1. To restrict a display to information about backups performed on specified dates, enter

```
bkhistory -d dates
```

where *dates* is a list of one or more dates, separated by commas.

The dates must conform to the syntax used with the **date (1)** command, with one exception: the only argument required is *month*.

2. To restrict a display to backup operations with specified tags, enter

```
bkhistory -t tags
```

3. To restrict a display to backup operations with specified originating devices, enter

```
bkhistory -o orig
```

where *orig* is in the following format: *oname:odevice:[omname]*.

- You may also combine these options, as shown in the following example:

```
bkhistory -d 01,02,03 -o "/usr:/dev/dsk/0s2 \
/back:/dev/dsk/0s8"
```

This command line restricts the display to backup operations completed in January, February, and March from two originating devices: **/usr:/dev/dsk/0s2** and **/back:/dev/dsk/0s8**.

Customizing the Format of the Display

In the default display, each field is labeled by a header (such as **Tag**) and has a specified length. Entries that exceed the designated field length wrap to the next line within the field.

You can change the format of the display by using the **-f**, **-h**, or **-l** options.

- To suppress the headers in the display, enter

```
bkhistory -h
```

This option is useful when the contents of the display are to be filtered by another process, such as an editor.

- To suppress field wrap and specify a character for separating fields in your display, enter

```
bkhistory -h -f c
```

where the value of *c* is the character that will appear as the field separator.

This option cannot be used without the **-h** option. The fields in the display appear together in one line. For example, to display the output shown in Screen 11-1 in this format, enter

```
bkhistory -h -f|;
```

The following display will appear:

```
rootdai;Feb 24 12:26 1992;incfile;/usr2/tmp/mnt;cpio1,cpio3,2;Online
rootsun;Feb 24 12:31 1992;ffile;diskette;!cpio1,rep1,!cpio3,!rep4;4;Archive
usr2dai;Mar 02 23:41 1992;ffile;diskette;med1,med6;3;None
usrdai;Jan 28 20:29 1992;incfile;file;med1,med2,med3;3;Archive
```

To avoid confusion, when selecting a separator, do not choose a character that is likely to appear in a field. For example, do not use a colon as a field separator if the display will contain dates in which a colon is used to separate hours from minutes.

- To produce the long form of the display, enter

bkhistory -l

The long form includes the information shown in Screen 11-1, along with the information produced by the **ls -l** command. A display produced in this format looks like the following example:

```

Tag DlabelFile information
rootdai cpio1 -r--r--r-- root sys 2898 Mar 30 11:42 /etc/passwd
rootdai cpio1 -rw-r--r-- root sys 329 Jan 17 16:05 /etc/group
rootdai cpio1 -rwxr--r-- root other 646452 Mar 29 12:10 /unix
rootdai cpio3 -rwxr-xr-x bin bin 33746 Mar 07 1987 /lib/cc
rootdai cpio3 -rw-rw-r-- root other 18616 Mar 29 12:10 /lib/lib1d.a
rootdai cpio3 -rwsr-xr-x root sys 11242 Oct 09 17:30 /bin/newgrp
usrdai med4 -rwxr-xr-x root other 851 Mar 29 12:10 /usr/oam/bin/
add
usrdai med4 -rwxr-xr-x root other 759 Mar 29 11:46 /usr/oam/bin/
res

```

The entries in this display have values in the **File information** field because the **-l** option to the **bkreg** command was specified for the operations described.

Truncating the Backup History Log

Without some type of control, the backup history log would grow without bounds as more and more backups were completed. Therefore, by default, the system removes any entries more than one week old. You can override this default and save history information for additional weeks by entering

```
bkhistory -p period
```

where *period* is the number of weeks for which you want information to be saved in the backup history log. Keep in mind, however, the longer the history log, the greater the number of automatic restore operations that can be done.

Backing Up Data through OA&M Menus

The system administration menus are only available if the Operations, Administration and Maintenance (OA&M) package is installed on your system. To help you plan, prepare for, and execute backups, the menus described below provide guidance through the steps in each process.

Basic Backup Service

To access the system administration menus for the basic backup service, type **sysadm backup_service/basic**. The following menu will appear on your screen:

```

2 Backup to Removable Media
Backup History
Personal Backup
Schedule Backup to Tape
System Backup

```

Depending upon which option you select from this menu, further menus or forms are presented from which you may make selections, or specify parameters.

Extended Backup Service

The extended backup service is only available if the Extended Backup and Restore (bkrs) package is installed on your system.

To access the system administration menus for the extended backup service, type **sysadm backup_service/extended**. The following menu will appear on your screen:

```

1      Extended Backup Service
backup  - Start Backup Jobs
history - Backup History Management
reminder - Schedule Backup Reminder
respond - Respond to Backup Job Prompts
schedule - Schedule Automatic Backups
setup   - Backup Control Table Management
status  - Backup Status Management

```

Table 11-1 shows how the tasks listed on the **backup_service/extended** menu correspond to the shell commands discussed throughout this chapter.

Table 11-1. The backup_service/extended Menu

Task to Be Performed	sysadm Task	Shell Command
Start backup jobs	backup	backup (1M)
Request backup history information	history	bkhistory (1M)
Schedule a backup reminder message	reminder	crontab (1)
Respond to backup job prompts	respond	bkoper (1M)
Schedule automatic backup jobs	schedule	crontab (1)

Table 11-1. The backup_service/extended Menu (Cont.)

Task to Be Performed	sysadm Task	Shell Command
Backup table management	setup	bkreg (1M)
		bkexcept (1M)
Backup status management	status	bkstatus (1M)

Some of the menu items in the table above have their own underlying menus.

Quick Reference to the Extended Backup Service

- Adding an entry to a backup table:

bkreg -a tag

where *tag* identifies a backup operation. The **-a** option must be followed by the **-o**, **-c**, **-m**, and **-d** options and, if you choose, any of the following options that are not required: **-b**, **-t**, **-P**, and **-D**.

- Adding files to a custom exception list for incremental backups:

bkexcept -t file -a pattern . . .

where *file* is the full path of the custom exception list, and *pattern* is a list of files and/or sets of files specified by the shell special characters *****, **?**, or **[]**. Items in this list must be (a) separated by commas or (b) separated by blank spaces and enclosed in quotes.

- Adding files to the system-supplied exception list for incremental backups:

bkexcept -a pattern . . .

where *pattern* is a list of files and/or sets of files specified by the shell special characters *****, **?**, or **[]**. Items in this list must be (a) separated by commas or (b) separated by blank spaces and enclosed in quotes.

- Checking the status of backup jobs:

The status of a backup operation is shown by one of the following keyletters: **p** (pending), **a** (active), **w** (waiting), **s** (suspended), **f** (failed), or **c** (completed).

- Displaying backup operations that are pending, active, waiting, or suspended.

bkstatus

- Displaying the status of backup operations that have either failed or been completed:

bkstatus -a

- Interrupting a backup job:

```
backup -S -C -R [-j jobid] [-u users] [-A]
```

where **-S** suspends the backup job with the specified *jobid*, **-C** cancels the backup job with the specified *jobid* or cancels the backup job issued by *users* with the specified logins, **-R** resumes the backup job with the specified *jobid*, and **-A** suspends, resumes, or cancels all running backup jobs.

- Defining the number of weeks for which backup status information will be saved:

```
bkstatus -p n
```

where *n* is the number of weeks.

- Defining and/or limiting the backup operations to be invoked:

```
backup -t table -o orig -c week:day|demand -m user
```

where *table* is the full path of a custom backup table, and *orig* is the list of originating objects from which backups are to be made. *orig* must be of the form *oname:odevice:[omname]*. *week* is an integer specifying the week and *day* is an integer specifying the day of the rotation period on which backups will be performed. *user* is the login name of the user who is to be notified by **mail** when the backup job is complete.

- Displaying the contents of a backup table:

```
bkreg -C fields -A | -O | -R -t table
```

where **-C** produces a summary display of the specified *fields*, **-A** displays all fields, **-O** displays a summary of all originating objects, and **-R** displays a summary of all destination devices. **-t *table*** is the name of the backup table.

- Editing an existing entry in a backup table:

```
bkreg -e tag
```

where *tag* identifies a backup operation. If any of the options **-b**, **-c**, **-d**, **-m**, **-o**, **-D**, or **-P** are present, they replace the current settings for the specified entry in the table.

- Displaying the contents of the backup history log:

```
bkhistory
```

- Invoking backup operations that are run only on demand:

```
backup -c demand
```

- Limiting the growth of the backup history log:

```
bkhistory -p period
```

where *period* is the number of weeks for which information will be saved.

- Previewing backup operations:

```
backup -n -e -c week:day|demand
```

where **-n** alone displays the current day's backup operations, **-e** estimates the number of destination device volumes required, and **-c week:day|demand** specifies the week and day of the rotation period (or the demand operations) to be previewed.

- Removing an entry from a backup table:

```
bkreg -r tag -t table
```

- Removing files to a custom exception list for incremental backups:

```
bkexcept -t file -r pattern...
```

where *file* is the full path of the custom exception list, and *pattern* is a list of files and/or sets of files specified by the shell special characters *****, **?**, or **[]**. Items in this list must be (a) separated by commas or (b) separated by blank spaces and enclosed in quotes.

- Removing files from the system-supplied exception list for incremental backups:

```
bkexcept -r pattern...
```

where *pattern* is a list of files and/or sets of files specified by the shell special characters *****, **?**, or **[]**. *pattern* must match, exactly, an entry in the exception list.

- Requesting the long form of the backup history display:

```
bkhistory -l
```

- Restricting the status information that is displayed:

```
bkstatus [-j jobids] [-s states] [-u users]
```

where *jobids* is a list of job IDs, *states* is a list of keyletters representing operation status, and *users* is a list of user login names.

- Selecting an operator mode while invoking the current day's backup operations:

```
backup [-a | -i]
```

where the default system response (in the absence of options) is to send a **mail** message to the operator when a backup operation needs assistance; **-i** prompts the operator at the terminal, and **-a** assumes no operator is present and fails any operations requiring assistance.

- Setting a rotation period for a backup table:

```
bkreg -p period -w cweek -t table
```

where *period* is the number of weeks in the rotation period, *cweek* is the current week of the rotation period, and *table* is the name of a backup table if a custom backup table is to be used

- Servicing backup operations:

bkoper

initiates an interactive session by displaying a list of backup operations

- Storing a table of contents online only: Tables of contents are stored online by default.
- Storing a table of contents on removable destination devices only, by specifying both **-t** and **-s**:

bkreg -m method -b "-t -s"

- Storing a table of contents both online and on removable destination devices by specifying **-t**:

bkreg -m method -b -t

- Storing a table of contents neither online nor on removable destination devices by specifying **-s**:

bkreg -m method -b -s

- Validating an archive during a backup:

bkreg -m method -b -v

where *method* is one of the following backup methods: **incfile**, **ffile**, **fimage**, **fdisk**, **fdp**.

- Tailoring the display of the backup history log contents:

bkhistory -d dates -o orig -t tags

where *dates* is a list of dates that restricts the report to backup operations performed on the specified dates, *orig* restricts the report to the specified originating devices, and *tags* is a list of operation tags.

- Translating an exception list from **ed** syntax to **cpio** format:

bkexcept -C old_file > new_file

where *old_file* is the filename of the exception list in **ed** command syntax, and *new_file* is a temporary file you edit before giving it to **/etc/bkup/bkexcept.tab** for input. After editing the file you can enter **bkexcept -a -< new_file**.

- Validating the contents of a custom backup table:

backup -n -t table

where *table* is the name of a custom backup table.

What Is a Restore Operation?

NOTE

If your system is being run in compliance with the security criteria described in the “Security Administration” part in *System Administration*, Volume 1, the Restore Service described in this chapter will be available in single-user mode only; see the “*Trusted Backup and Restore*” chapter in *System Administration*, Volume 1, for information on backing up and restoring files with Mandatory Access Control levels.

Restoring lost data is an important administrative task--**and** one you'll probably need to do more frequently than you might suspect. If you have used the backup service to archive system and user files, you can use the restore service to restore files, directories, file systems, data partitions, disks, and partitioning information from archive volumes.

Depending on the software packages installed on your system, two different restore services may be available: basic, or extended service.

If the Operations, Administration and Maintenance (OA&M) package (a non-graphical menu interface) is installed on your system, you can use it to complete many of these tasks.

Before You Begin

Be sure to read Chapter 10, *Archiving and Restoring Data*, before beginning any restore operations. It gives you information on how to archive and restore data.

Basic Restore Service

The basic restore service is available on all systems, and can be executed using the **restore** command with one or more options. Note that the basic **restore** command is different from, and offers a smaller set of options, than the extended **restore** command. The basic restore service should be used to restore data that was backed up using the basic backup service.

Specifying Pattern Matching on the restore Command Line

When doing a restore, one or more patterns can be specified on the command line. for example

```
restore [option...] [pattern...]
```

Here, *pattern* is a file name.

These patterns are matched against the files on the tape or diskette. When a match is found, that file is restored. Since backups are done using full pathnames, the file is restored to its original directory.

- Metacharacters can be used in *pattern* to match more than one file. Patterns containing metacharacters should be in quotes to prevent the characters from being expanded before they are passed to the command.
- If no patterns are specified, the default is to restore all files.
- If a pattern does not match any file on the tape, a message is printed.

Media Handling

When the end of a disk or tape is reached, you are prompted for the next media. If you need to, you can exit at this point by typing `q`.

CAUTION

Quitting before the entire restore is completed may cause files to be corrupted if a file happens to span multiple tapes or diskettes.

In order for multi-volume restores to work correctly, a raw device must be specified with the `-d device` option.

Restore Examples

The following examples show several common ways to use the basic `restore` command.

Displaying an Index of Archived Files

To display an index to the files on the backup medium, enter

```
restore -i -d device
```

Restoring Everything from the Archive

1. To restore all files from backup tape(s), enter

```
restore -t -c -d device
```

where `-t` specifies that the restore is from tape, `-c` specifies that a complete restore is to be done, and `-d device` specifies the full pathname of the tape device.

2. To restore all files from backup diskette(s), enter

```
restore -c -d device
```

where **-c** indicates that a complete restore is to be done, and **-d device** specifies the full pathname of the diskette device.

Note that you must also specify the **-o** option if existing disk files that have the same name as the files being restored are to be overwritten.

Restoring Selected Files

To restore selected files from tape, and overwrite existing files with the same name, enter

```
restore -t -o -d device file
```

where **-t** specifies that the restore is from tape, **-o** indicates that any files on disk that have the same name as the files being restored will be overwritten by the restore files, **-d device** specifies the full pathname of the tape device, and *file* is one or more full pathnames of files to be restored. If *file* is a list of files, the names must be separated by blank space and enclosed in double quotes.

Metacharacters may also be used in *file*. When used, they must also be enclosed in double quotes. For example:

```
restore -t -d device "/sample/*"
```

In this example, all files in the **/sample** directory will be restored from tape.

Extended Restore Service

The commands that make up the extended restore service are contained in the Extended Backup and Restore (bkrs) package — if this package is installed on your system the extended restore service is available to you. If you use the extended backup service to do a backup, and you need to do a restore, you should use the extended restore service to restore the data. The extended restore service enables you to retrieve copies (archives) of files, directories, file systems, data partitions, disks, and partitioning information that have been preserved on archive media such as diskettes and tapes. There are various types of restore operations; the type you do depends on the type of archive you have.

Who Is Responsible for Servicing Restore Requests?

As system administrator, you may be responsible for establishing backup and restore policies for your computer site. As part of this work, one of the first decisions you should make is who will be responsible for servicing restore requests. On large central server systems the tasks associated with the restore service might be performed by more than one person: a system administrator and a computer operator, for example. If you decide to

delegate restore jobs to an operator, you'll have to arrange to have all restore requests sent to this person.

Using rsnotify to Arrange for Restore Requests

1. To arrange for all restore requests to be sent to one person, enter

```
rsnotify -u login_name
```

where *login_name* is the login of the designated operator.

2. Once you've run this command, the designated operator will receive requests for restores and mail about pending jobs from users who are not themselves responsible for performing backups and restores. (The operator can also scan for pending restore requests by running the **rsstatus** command.)

3. To find out if an operator is already assigned, enter

```
rsnotify
```

without options. The login name of the designated operator and the date the assignment was made will be displayed.

4. To change the assigned operator, enter

```
rsnotify -u login_name
```

again.

5. If no operator is assigned, requests for restore jobs will be mailed to **root**.

Deciding Which Restore Command to Use

Which of the **restore** commands you choose depends on how you created the archive of the object you're now trying to restore.

- If you created it by executing the **ffile** or **incfile** method of backup, use the **urestore** command to restore individual files and directories. Use the **restore** command to restore full file systems.
- If you use other backup methods, however, you can use the **restore** command.

How the Extended Restore Service Works

The restore process begins when someone, either a user or an administrator, requests that a particular object be restored. All users can request restores of any files and directories they own, but only administrators can request restores of data partitions, file systems, or entire disks. Because of this there are two commands for requesting restores. The first, **ure-**

store, can be used by anyone to request a restore of one of their own files or directories. (An administrator can use **urestore** for any file or directory, regardless of who owns it.)

The second, **restore**, can be used only by an administrator to restore file systems, disks, or data partitions, or to display partitioning information.

Using urestore to Restore a Directory or File

When issuing the **urestore** command, you must have read permission for the parent directories of the file or directory to be restored, as well as write permission for the immediate parent directory. In addition, if the request is made by anyone other than the administrator or the operator, that person must have owned the file or directory at the time the archive was made. The following options are available with the **urestore** command:

- c** *job_ID* Cancels the previously issued restore request, *job_ID*.
- d** *date* Restores a file or directory as of *date*, which may or may not be the date of the latest archive version. The value of *date* is the same ten-character string used to specify a month, day, hour, minute, and year with the **date (1)** command: **MMddhhmmyy**.
- m** If the restore cannot be done immediately, notify the person requesting the restore (via **mail**) when the request is complete.
- n** When issued with the **-D** or **-F** option (one of which must be used), **-n** lists all archived versions of a file or directory contained in the backup history log. The **-n** option does not restore the file or directory.
- o** *target* Restores the archive to the *target* location.
- D** *directory* Restores *directory* and all the files underneath it. More than one directory can be specified, each directory being separated by a space.
- F** *file* Restores *file*. More than one file can be specified, each file to be restored being separated by a space.

Using restore to Restore Other Disk Objects

An administrator or operator can restore a data partition, a file system partition, or an entire disk by using the **restore** command, along with the following options:

- A** *partdev* Initiates a restore of the entire disk *partdev*.
- P** *partdev* Initiates a restore of the data partition *partdev*.
- S** *oname* Initiates a restore of the file system partition *oname*.
- n** When issued with the **-A**, **-P**, or **-S** option (one of which must be used), **-n** lists all archived versions (of the disk object) contained

in the backup history log. The **-n** option does not restore any disk objects.

partdev takes the form **/dev/rdisk/c?d?s?** (**c?t?d?s?** for SCSI devices). If **-A** is specified, the full disk method is used to repartition the disk. When *partdev* is used as an argument to **-A**, the value of *partdev* is **/dev/rdisk/c?d?** *partdev* should be the name of the originating object as set in the backup table for the partition or disk you are restoring).

CAUTION

Restoring a disk overwrites any data on the disk and resets all partitioning information for the disk to that of the archive.

Restoring a Specific Version of an Object

Sometimes, such as when the most recent archive of an object has been corrupted, you will want to restore a version of an object that's older than the version in the most recent archive.

1. Enter

```
restore -n
```

to get a list, organized by date, of all the archives available for the desired object.

Don't forget to specify the type of object you want to restore by including the **-A**, **-P**, or **-S** option. See **restore (1M)** for details.

2. Enter

```
restore -d MMddhhmmyy
```

where *MMddhhmmyy* is the date (month, day, hour, minute, and year) of the archive containing the version you want to have restored.

For example, to request an archive made at 16:30 (that is, 4:30 P.M.) on October 6, 1993, you would enter

```
restore -d 1006163093
```

Restoring an Object to a New Location

By default, the restore service restores an object to its original location. Sometimes, however, this may not be desirable. For example, you may have a new object in that location that you don't want to overwrite.

To restore an object to a new location, enter

```
restore -o target
```

where *target* is the name of the new destination location.

Checking the Status of Restore Requests

You can check the status of restore requests by using options to any of four commands: **restore**, **urestore**, **rsstatus**, or **ursstatus**.

Using restore or urestore to Check Status

To check restore request status using either **restore** or **urestore**, enter

```
restore (or urestore) -s
```

This displays a “.” for each 100 blocks transferred from the archive.

Or, you may enter

```
restore (or urestore) -v
```

This displays the name of each directory or file to be transferred from the archive.

NOTE

These options take effect only when the required archive volume is online and the restore operation is processing.

Using rsstatus and ursstatus to Check Status

To check restore requests using **rsstatus**, enter

```
rsstatus
```

This lists all pending restore requests in details, and may be issued only by an administrator or an operator.

To check restore requests using **ursstatus**, enter

```
ursstatus
```

This lists pending file and directory restore requests for the invoking user. The information gathered by **ursstatus** or **rsstatus** is recorded in the `/etc/bkup/rsstatus.tab` table. This is default information.

The Status Table

The **ursstatus** report contains only the first five fields.

The **rsstatus** report appears in the following format on your screen:

Jobid Labels	Login	File	Date	Target	Bkp date	Method	Dtype
rest-11111a tlabl1	user1	/home/user1	Dec 22	/home/user1	Sun Dec	incfile	dtabl1
		/oam/ISSUES	1992 17:00: 00	/bkISSUES	22 1991		
rest-32984c tlabl1	user1	/home/user1	Dec 22	/home/user1	Sun Dec	incfile	dtabl1
		/oam/bkrs	1992 17:00: 00	/oam/bkrs	22 1991		
rest-04818a tlabl1	user2	/home/user2	Dec 22	/home/user2	Sun Dec	ffile	dtabl2
		/rsstatus.c	1992	/rsstatus.c	22 1991		

The fields shown above represent

Job_ID	The job ID of the restore request
Login	The login name of the person requesting the restore
File	The full pathname of the file to be restored
Date	The specified date from which an object should be restored. (The restore service selects archives made on a backup date as near as possible to the date specified.)
Target	The full pathname of the location where the restored object should be placed
Backup_Date	The date of the last backup performed before the date specified in a restore request
Method	The method used to perform the backup
Dtype	The destination device, such as a diskette or cartridge tape, on which the backup archive was created
Labels	The labels for the archive volumes to be restored

Customizing the Display of the Status Table

NOTE

Be sure to refer to the **rsstatus (1M)** manual page for full details on using the options described below.

The default display provides all the available information about pending restore requests. If you do not want to see a default status report, however, you can customize both the contents and the format of the report. In the default **rsstatus** display, each field has a title and a specific length. Data entries that exceed the specified field length wrap to the next line within the field.

1. You can restrict the fields displayed by entering

```
rsstatus -d [dtype] -j -u
```

The report displayed will contain only those entries that satisfy all three specifications.

Or, you could use only one of the options.

2. You can change the format of the **rsstatus** display by entering

```
rsstatus
```

with either **-h**, **-f**, or **-s**.

Servicing Pending Restore Requests

Restore operations that cannot be done immediately are posted to the restore status table and are considered “pending.” They must be serviced, through the **rsoper** command, by the operator who received mail about them. To satisfy a pending restore request, locate and install the correct archive volume, as follows:

1. Display the restore status table by entering **rsstatus** with the desired options.
2. Note the name of the object to be restored and the label of the archive volume for it.

NOTE

If there is no label information in the status table, you may have to determine which archive volume contains the backup from which to restore by guessing or relying on other information. For example, the date of a backup may allow you to identify an archive volume.

3. Mount the archive volume.
4. Enter

```
rsoper -d ddev
```

where *ddev* is the name of the device that is to read the archive. *ddev* takes the following form:

```
ddevice [ : dchar ] [ : dlabels ]
```

dlabel must be specified if there is more than one label. It must include a device name and it may include device characteristics and volume labels.

NOTE

If the history log has been removed, all the *ddev* fields (*ddevice*, *dchar*, *dlabels*) must be specified.

5. After the volume is processed, check the restore status with the **rsstatus** command.

The restore service compares the information you have entered on the **rsoper** command line with the information in the restore status table and on the archive volume.

- If the information on the command line matches the information in the restore status table, the restore operation begins.
- If the information on the command line does not match that in the restore status table, the information on the command line predominates and the restore operation begins.
- Then the restore service attempts to resolve any restore requests that can be satisfied by the archive volume described by **-d ddev**.

Restricting Restores

By default, when **rsoper** is invoked, the restore service attempts to complete any restore requests on the archive volume mounted. You can restrict restore jobs to those with specific job IDs by entering

```
rsoper -j
```

You can also restrict restore jobs to specific user logins through the **-u** option.

Removing and Canceling Restore Jobs

Entries in the restore status table may be deleted for either of two reasons:

- to remove a restore request that has been satisfied
- to cancel a job that cannot be completed or cannot be serviced at all
- To remove an entry for a pending restore request, enter

```
rsoper -r job_IDs
```

where *job_IDs* is a list of pending restore requests that have been serviced.

This command notifies the people who issued these requests that the restore operations have been done successfully and that the entries for them have been removed from the status table.

- To cancel a request, enter

```
rsoper -c job_IDs
```

where *job_IDs* is a list of pending restore requests to be canceled.

This command notifies the people who issued these requests that the requests cannot be serviced and have been canceled.

Basic Options

Three options allow you to describe the archive volume being mounted for use by a restore operation.

1. Enter

```
rsoper -t
```

This tells the service that the volume inserted in the destination device contains a table of contents for the archive.

2. Enter

```
rsoper -o oname:odev
```

This specifies the originating file system partition or data partition to be restored. *oname* is the name of the originating file system; the value of *oname* may be null. *odev* is the device name of the originating file system or data partition.

3. Enter

```
rsoper -m method
```

This specifies that the first archive volume in the destination device was created by the backup *method* specified.

The following example illustrates the use of some of these options. Suppose the backup history log no longer contains a log of the backup operations for which archives are being requested. To obtain the desired data, you must request an incremental file restore from the `/usr2` file system. To request this, enter

```
rsoper -d /dev/diskette2::arc.dec79.a,arc.dec79.b, \
      arc.dec79.c -m incfile -o /usr2
```

The `/usr2` file system archive is found on diskettes labeled `arc.dec79.a`, `arc.dec79.b`, and `arc.dec79.c`.

Restoring ffile and incfile Backup Archives Using cpio

For `ffile` and `incfile` methods, the backup archive consists of an optional `labelit` header, backup archive header and a `cpio` archive header. You can retrieve the archive by using `dd (1M)` and `cpio (1M)` commands.

1. For example, to retrieve the archive from the cartridge tape drive `/dev/rmt/0m`, enter

```
dd if=/dev/rmt/0m skip=3 | cpio idum
```

The tape will contain a number of blocks of header information which must be skipped by using the **dd (1M)** command before running **cpio (1M)**.

2. To determine how many blocks to skip, run the command

```
od -x ddev | grep '^000.000'
```

where **ddev** is the input device special.

This lists the first few bytes from each of the first eight blocks of the tape.

3. To identify the block at which the **cpio** archive starts, look for a line starting with the backup header magic number

```
000?000 9180 67de . . . . .
```

where ? is a digit.

4. Now look for the first line that follows the backup header that starts with the magic number of a **cpio** archive header:

```
000?000 3037 3037 3031
```

where ? is a digit and is the number of blocks you will need to skip. Normally ? will be between 3 and 5.

The **od (1M)** command in the example above might produce the following output:

000	0000	566f	6c63	6f70	7900	766f	6c31	0000	0000
000	3000	f49180	67defP	0000	0200	29bd	2b98	0000	0001
000	4000	f43037	3037	3031f1	3030	3030	3037	6330	3030
000	5000	2077	6572	6520	7072	6f76	6964	6564	2062
000	6000	5022	0a23	0989	0977	6865	7265	2061	7474
000	7000	6465	762f	7253	412f	6330	6430	392f	6465

In this example, the backup header starts at block 3 (the first block is block 0), and the first **cpio** header at block 4, so it is necessary to skip four blocks:

```
dd if=/dev/rmt/0m skip=4 | cpio -idum
```

If the pattern `f43037 3037 3031f1` is seen before the backup header, it should be ignored. You can use this method of retrieving the backup archive provided that it occupies a single volume.

Using the Restore Service Through OA&M Menus

The system administration menus are only available if the OA&M package is installed on your system. The Restore Service Menus can help you plan, prepare for, and execute restores, by guiding you through the steps of each process.

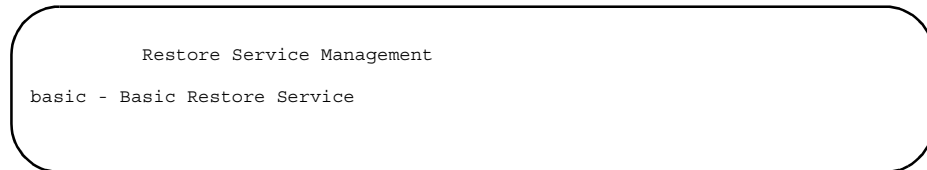
Some of the `sysadm` menus do not offer all the functionality of the commands discussed in this chapter. If you are not an experienced administrator, however, you may find it easier to use the menus, which provide prompts and help messages that are not available with the shell commands.

Basic Restore Service

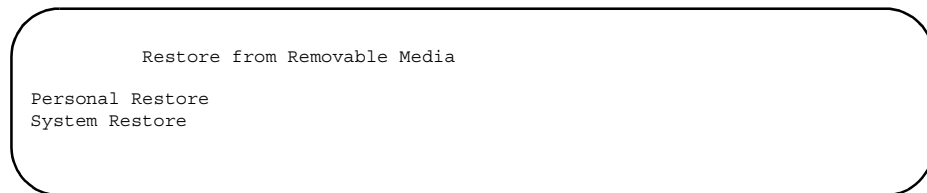
1. To access the system administration menus for the basic restore service, enter

```
sysadm restore_service
```

The following menu will appear on your screen:



2. After selecting `basic`, a menu appears offering the following selections.



Depending upon which option you select from this menu, further menus or forms are presented from which you may make selections, or specify parameters.

Extended Restore Service

The extended restore service is only available if the Extended Backup and Restore (bkrs) package is installed on your system. If this package is installed, you can select the appropriate restore task from the OA&M menus.

To access the system administration menu for using the restore service, enter


```
sysadm restore_service/extended
```

The following menu will appear on your screen:

```

1      Extended Restore Service

operator - Set/Display the Restore Operator
respond  - Respond to Restore Job Prompts
restore  - Restore from Backup Archives
status   - Modify/Report Pending Restore Request Status

```

The following table shows how the tasks listed on the **Basic Restore Service** and **Extended Restore Service** menus correspond to the tasks discussed throughout this chapter.

Task to Be Performed	<i>sysadm</i> Task	Shell Command
Restore from backup archives	restore	restore (1M) urestore (1M)
Respond to pending restore requests	respond	rsoper (1M)
Cancel unsatisfied requests or remove satisfied restore requests	status	rsoper (1M)
Set up and / or display the login name of the operator who will monitor restore requests	operator	rsnotify (1M)
Obtain status information on pending restore request	status	rsstatus (1M) ursstatus (1M)

Quick Reference to the Extended Restore Service

The Administrator's Tasks

- Assigning an operator to service restore operations:

```
rsnotify -u login_name
```

where *login_name* is the login name of the designated operator.

- Displaying the name of the operator assigned to service restore operations:

```
rsnotify
```

- Initiating a restore of a data partition:

```
restore -P partdev
```

where *partdev* is the name of a data partition.

- Initiating a restore of a file system partition:

```
restore -S oname
```

where *odevice* is the name of the file system partition to be restored.

- Initiating a restore of an entire disk:

```
restore -A partdev
```

where *partdev* is the name of the disk to be restored.

- Restoring an object from an archive volume of a particular date:

```
restore -d MMddhhmmyy (or) urestore -d MMddhhmmyy
```

where *MMddhhmmyy* is the date (month, day, hour, minute, and year) of the archive to be used for the restore.

- Restoring an object to a new disk location:

```
restore -o target (or) urestore -o target
```

where *target* is the complete pathname of the destination location on the disk.

The Operator's Tasks

- Displaying a list of all archived versions of an object:

```
restore -n (or) urestore -n
```

- Displaying the complete contents of the restore status table:

```
rsstatus
```

- Displaying a list of restore jobs that can be satisfied by a specified device type or archive volume:

```
rsstatus -d [dtype] [:dlabels]
```

where *dtype* is a description of a device type (diskette, ctape, and so on) and *dlabels* is the label of a particular archive volume (such as bk0010, bk0011, and so on)

- Displaying the status of particular **fdisk**, **fimage**, **ffile**, or **fdp** restore jobs:

```
rsstatus -j job_IDs
```

where *job_IDs* is a list of one or more job IDs for requested restores.

- Displaying the status of restore requests made by specific users

```
rsstatus -u users
```

where *users* is a list of one or more login IDs.

- Removing a pending restore request from the restore status table and designating it canceled:

```
rsoper -c job_IDs
```

where *job_IDs* is a list of one or more job IDs for restore.

- Removing a pending restore request from the restore status table and designating it complete:

```
rsoper -r job_IDs
```

where *job_IDs* is a list of one or more job IDs for requested restore.

- Servicing a pending restore request:

```
rsoper -d ddev
```

where *ddev* describes the device to be used to read the archive containing the file system or data partition to be restored.

- Suppressing field wrap and specifying an output field separator on the restore status display:

```
rsstatus -f c
```

where *c* is the character that will appear as the field separator in the display of the restore status table.

The User's Tasks

- Canceling a previously requested restore:

```
urestore -c job_ID
```

where *job_ID* is the job ID of a job to be canceled.

- Displaying the status of particular incfile restore jobs:

```
rsstatus -j job_IDs
```

where *job_IDs* is a list of one or more job IDs for requested restore.

- Initiating a restore of a directory:

```
urestore -D directory
```

where *directory* is the name of the directory to be restored.

- Initiating a restore of a file:

```
urestore -F filename
```

where *filename* is the name of the file to be restored.

Print Service Administration

Replace with Part 4 tab for 0890430

Part 4 - Print Service Administration

Part 4 Print Service Administration

Chapter 12	Basic Print Service.....	12-1
Chapter 13	Advanced Print Service.....	13-1

Introduction	12-1
Overview of the Print Service	12-1
Functions	12-2
Getting Started	12-2
Choose Your Printer Sites	12-2
Physically Connect Printers to Computers	12-4
Install the LP Print Service	12-5
Configure Printers for the Printer Service	12-5
Configuring a Directly Connected Printer	12-5
Choose the Device Level Range	12-6
Allocate the Device	12-7
Parallel Port Printer Setup (Power Hawk Only)	12-8
Step 2: Define Printer Attributes	12-8
Attribute 1: Printer or Queue Name	12-8
Attribute 2: Device Path	12-9
Attribute 3: Printer Type	12-9
Attribute 4: Content Type	12-9
The Default Content Type: simple	12-10
Attribute 5: Printer Interface	12-10
Examples of Defining Printer Attributes	12-11
Step 3: Notify the Printer to Accept Jobs	12-11
Step 4: Turn on the Printer (Logically)	12-11
Other Printer Configuration Options	12-12
Printer Port Characteristics	12-12
Printer Types	12-14
Content Types	12-14
Using the Default Content Type	12-16
Printer Interface	12-16
Character Sets, Fonts, or Print Wheels	12-16
Alerting to Mount a Print Wheel	12-19
Forms Allowed	12-20
Printer Fault Alerting	12-22
Printer Fault Recovery	12-24
User Access Restrictions	12-25
Inclusion of Banner Page in Output	12-26
Controlling Copying of Files	12-27
Printer Description	12-27
Default Printing Attributes	12-28
System Default Destination	12-29
Printer Class Membership	12-29
Putting It All Together	12-30
Example 1: Configuring a PostScript Printer on a Serial Port	12-30
Example 2: Configuring a PostScript Printer on a Parallel Port	12-30
Example 3: Adding a Printer with the Standard Interface	12-30
Example 4: Configuring an Alert	12-30
Example 5: Configuring a Secure Printer	12-31
Examining a Printer Configuration	12-31

Display Status of Printer Service	12-32
Making Printers Available	12-32
Accepting Print Requests for a New Printer	12-33
Enabling and Disabling a Printer	12-33
Allowing Users to Enable and Disable a Printer	12-34
Managing the Printing Load	12-35
Rejecting Requests for a Printer or Class	12-36
Accepting Requests for a Printer or Class	12-36
Moving Requests to Another Printer	12-36
Examples	12-37
Example 1	12-37
Example 2	12-37
Example 3	12-38
Starting and Stopping the LP Print Service	12-38
Manually Stopping the Print Service	12-38
Manually Starting the Print Service	12-38
Managing Classes of Related Printers	12-39
Adding a New Class	12-39
Listing Printers in Classes	12-39
Modifying Membership of a Class	12-39
Removing a Class	12-40
Managing Active Print Requests	12-40
Canceling Print Requests	12-40
Putting a Request on Hold	12-40
Releasing Held Print Requests	12-41
Moving Requests to A New Destination	12-41
Moving a Request Around in the Queue	12-42
Changing the Priority for a Request	12-42
Moving a Request to the Head of the Queue	12-42
Troubleshooting	12-42
Problem: No Output (Nothing Is Printed)	12-43
Is the Printer Connected to the Computer?	12-43
Is the Printer Enabled?	12-43
Is the Baud Rate Correct?	12-43
For a PostScript Printer: Do You Have Access to It?	12-43
Problem: A Print Request That Won't Cancel	12-44
Problem: No Output and No Notification from the lp Command	12-44
Problem: Printer Stops Working after Printing 2 or 3 Pages	12-44
Problem: Illegible Output	12-44
Is the Baud Rate Correct?	12-44
Is the Parity Setting Correct?	12-45
Are the Tabs Set Correctly?	12-45
Have You Selected the Correct Printer Type?	12-45
Problem: The Printing Is Legible but the Spacing Is Wrong	12-46
Double Spaced Text?	12-46
Text Has No Left Margin, Is Run Together, or Is Jammed Up?	12-46
Text Zig Zags Down the Page?	12-46
Letters Are Poorly Spaced?	12-46
A Combination of Problems?	12-46
Problem: Wrong Character Set or Font	12-47
Problem: An Attempt to Dial Out Fails	12-47
Problem: The Printer Is Idle	12-47
Problem: Warning Messages About Unrecognized Options	12-48
Problem: Error Messages About Options That Can't Be Handled	12-48
Administering LP through OA&M Menus	12-49

Quick Reference to LP Print Service Administration 12-50

Introduction

The LP Print Service, originally called the LP spooler, is a set of software utilities that allows you, minimally, to send a file to be printed while you continue with other work. (The term “spool” is an acronym for “simultaneous peripheral output on-line,” and “LP” originally stood for Line Printer, but has come to include many other types of printing devices.) The Print Service has many optional enhancements, however; you can make yours as simple or as sophisticated as you like.

The information presented in this chapter includes the following:

- a description of how the LP Print Service works
- instructions for setting up the Print Service and an LP network
- troubleshooting guidelines
- instructions for stopping and starting the Print Service manually
- instructions for configuring a Print Service for the unique requirements of your users (such as the need for particular pre-printed forms and filters)
- instructions for supporting PostScript printers
- instructions for writing customized filters and interface programs

This chapter describes the shell commands available to administer the LP Print Service. Complete details about the commands described here are available in the manual pages for them. Problems encountered by the LP Print Service are listed in the “Troubleshooting” section of this chapter.

If the Operations, Administration and Maintenance (OA&M) package (a non-graphical menu interface) is installed on your system, you can use it to complete many of these tasks. (See “*Administering LP through OA&M Menus*” later in this chapter.)

Overview of the Print Service

The Print Service consists of both hardware and software. You must have at least one computer and one printing device for an LP Print Service. Beyond that, there is no limit to the number of pieces of hardware you may include. The software consists of the LP Print Service utilities and any filters (programs that process the data in a file before it is printed) that you may provide. If you make different types of printers and/or filters available with

your service, users may choose from several formats. You may also offer your users a choice between plain paper and pre-printed forms (such as invoices or checks).

Functions

Whether your Print Service is simple (such as a one-computer/one-printer configuration that prints every file in the same format on the same type of paper) or sophisticated (such as a computer network with multiple printers and a choice of printing formats and forms), the LP software helps you maintain it by performing several important functions:

- scheduling the print requests of multiple users
- scheduling the work of multiple printers
- starting programs that interface with the printers
- filtering users' files (if necessary) so they will be printed properly
- keeping track of the status of jobs
- keeping track of forms and print wheels currently mounted and alerting you to mount needed forms and print wheels
- alerting you to printer problems

Getting Started

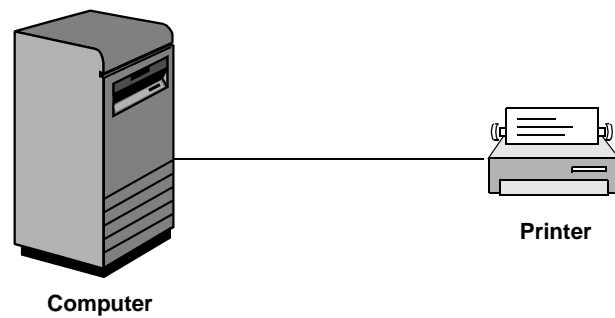
The following is a summary of the procedure you will need to complete to set up the LP Print Service. Instructions for each step are provided in subsequent sections.

1. Choose your printer sites.
2. Physically connect printers to computers.
3. Install the LP Print Service software.

Choose Your Printer Sites

Depending on the sensitivity of the data your printers will be used to print, you should give consideration to the physical location and security of those printers. Decisions you make about where to place your printers and how to connect them to your computers depend on how they will be used. (For details, see “*Configuring a Directly Connected Printer*,” in this chapter, and “*Configuring a Network Printer*,” and “*Configuring a Dial-up Printer*,” in the chapter “Advanced Print Service”.)

- You may want to connect a particular printer directly to the “home” computer of the users who will use that printer most often. Directly connected printers (refer to Figure 12-1) provide a greater level of security than remotely connected printers, because you can monitor status and output and administer them more closely.



162960

Figure 12-1. A Directly Connected Printer

An environment that includes more than one computer, each of which is directly connected to a printer, is said to have a “distributed printing configuration.”

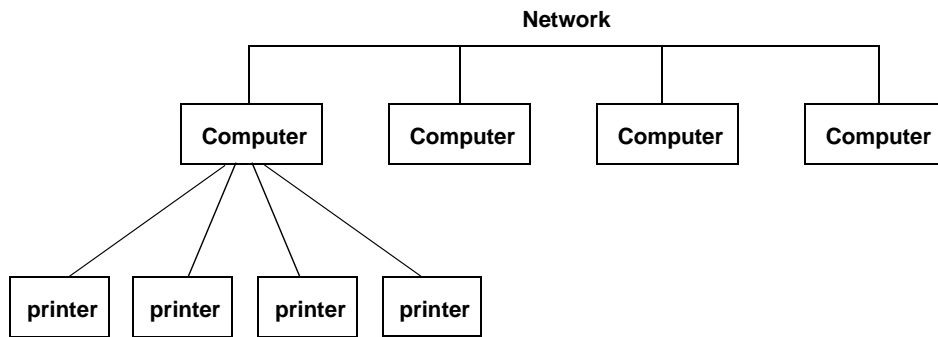
- You may want to have all your printers in one physical location, such as in a computer center, connected to one computer. Users on other computers who want to use a printer may access it through a network linking their own computers to the computer serving the printers.

NOTE

To set up a printer network, you must install the LP Networking Service. If you install this software on a system with a B2-level security rating, that rating will be invalidated.

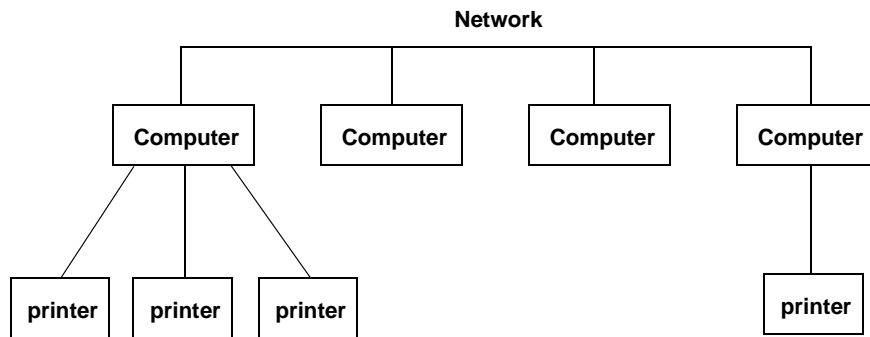
An environment in which one computer serves several printers (which can be accessed only through a computer-to-computer network) is described as a “printer server configuration.” Figure 12-2 shows a sample printer server configuration.

- You may want to link most of your printers to a dedicated printer server computer, while allowing other printers to be connected to your computer. If so, you can arrange your computers and printers in a network configuration, as shown in Figure 12-3.



162970

Figure 12-2. Printer Server Configuration



162979

Figure 12-3. Network Configuration

Physically Connect Printers to Computers

Follow the instructions in your printer manual or your computer installation manual to connect your printer (and any optional hardware).

Install the LP Print Service

NOTE

If the Enhanced Security Utilities are installed and running on your system, you must boot your system to Single-user mode before you can install the LP Print Service software. See the “Booting and System States” part of *System Administration* Volume 1, for a description of single-user mode.

You can install the Commands Networking Extension Package by re-executing the installation of the Networking Set and selecting custom installation rather than automatic, and including `netcmds` among the packages to be installed. (See the *Release Notes* for complete details.)

Other software packages you may want to install are the Advanced Commands package, and the BSD Compatibility package. The BSD package gives you access to an older BSD version of the Documenter's Workbench commands, including `troff`. These packages are not required by the LP Print Service, but will allow you to make use of `troff`.

Configure Printers for the Printer Service

To configure a printer is to define the environment in which it will operate. The LP Print Service allows you to set up a variety of printing environments:

- You can connect a printer directly to the home computer of the people who use it.

If you have the Commands Networking Extension Package installed you can also set up these printing environments:

- You can connect several printers to one computer that then functions as a “printer server.” Users access these printers via a network connection between their home computers and the server.
- You can connect printers to a remote server. Users access them by dialing up the server instead of reaching it through a network.

In addition, you can set up a printing environment that includes more than one of these arrangements.

Configuring a Directly Connected Printer

Probably the most common way of connecting a printer and a computer is by connecting them physically, as shown in Figure 12-3. To configure a directly connected printer, complete the following steps in the order shown:

1. Define the device.
2. Allocate the printer.
3. Define printer attributes.
4. Notify the printer to accept print jobs.
5. “Enable” the printer (turn it on, logically).

The rest of this section provides instructions for each of these steps.

Choose the Device Level Range

NOTE

This section assumes you understand Mandatory Access Control (MAC) security levels. If you do not, read the “*Mandatory Access Control*” material in the “Security Administration” part of “*System Administration*, Volume 1.

On a system on which files have different MAC levels, every item that contains data (such as a file or pipe) has a MAC security level associated with it. These security levels reflect the sensitivity of the data; a higher security level means that the data is more sensitive to disclosure. Each process or user of the system also has a security level. The system compares the user's security level with the security level of the data to determine if the user can access the data.

When a file is printed, the computer system can no longer maintain access control for the data. Unless the printed copy has some indication of the sensitivity of the data, it is not possible to ensure that access to the data is controlled properly. Thus, the LP Print Service automatically prints security level information on each page of paginated output and on the banner and trailer pages for each job. Users can override the printing of security level information on the printed output, but not on the banner and trailer pages.

Each printer on your system has a security level range associated with the device special file used as the printer's pathname. The printer can print a file only if the level of the file is within the security level range of the device. The security level range is set with the **putdev** and **admalloc** commands and should reflect the physical security of the printer itself. For example, if the printer is in an open location, it should not be used to print sensitive information. Conversely, if a printer is accessible only to trusted staff members, you may want to set its level range to reserve it for printing sensitive material. Before you configure the printer and allocate a device for it, you should decide what security level range is appropriate for it.

Set the printer's level range with the **putdev** command when the device special file is allocated for use. You must allocate the device special file with **admalloc** before you can use **lpadmin** to configure the printer.

For details about **putdev** and **admalloc**, see “*Define the Device*” and “*Allocate the Device*” below.

Step 1: Define the Device

To define a device as an LP device, run the **putdev** command, and define a minimum of the following attributes:

- `range='hilevel-lolevel'` The value of `range` is the MAC-level (sensitivity level) of the device. Only print jobs within this range will be printed on a printer configured with this device. (See “Choose the Device Level Range” above.)
- `state=public`
- `mode=static`
- `startup=y` By defining this attribute as `y`, you ask to have the device allocated automatically when the system is booted.
- `startup_owner='lp>rw-'`
- `startup_group='lp>---'`
- `startup_other='>---'`
- `startup_level='hilevel'` Must be within the `range` defined above. We recommend you set the value of `startup_level` as `hilevel`.

For example:

```
putdev -a tty011 \
cdevice='/dev/tty0-11' \
type=tty \
mode=static \
state=public \
range='SYS_RANGE_MAX-USER_LOGIN' \
startup=y \
startup_owner='lp>rw-' \
startup_group='lp>---' \
startup_other='>---' \
startup_level='SYS_RANGE_MAX' \
other='>y' \
ual_enable=y
```

For details about **putdev**, see the “Managing Storage Devices” chapter in *System Administration*.

Allocate the Device

The device can now be allocated for use via **admalloc**:

```
admalloc -w SYS_RANGE_MAX -u lp,lp /dev/tty0-11
```

(This is done only once, after the device has been defined using **putdev**.)

Parallel Port Printer Setup (Power Hawk Only)

Use the following procedure to configure a printer connected to the parallel port of a Power Hawk (chassis or desktop). This procedure assumes the `lpt` optional product software has previously been installed. (Refer to the Release Notes or the “Installing Add-On Software” chapter in Volume 1 for further details on installing optional software.)

```
# /etc/conf/bin/idmknod -M lpt  
  
# ls -ail /dev/lp* (to verify file generation)
```

```
7648 crw-rw-rw- 1 root sys 10, 0 Oct 26 10:17 /dev/lpt
```

Perform steps 2 through 4 to complete the setup procedure.

Step 2: Define Printer Attributes

Printer attributes are defined with the `lpadmin` command. (See the online manual page `lpadmin (1M)` for complete details.)

Define the following five attributes for the Print Service at your site:

Attribute 1: Printer or Queue Name

Use the `-p printer` option of `lpadmin` to define the printer name or queue name.

The printer name identifies the printer you are configuring. It will be used by you (as the administrator), when you need to add or change configuration information for the printer or when removing the printer, and by users when requesting or canceling print jobs, or checking the status of the printer.

On systems that support the print server feature, the queue name is the name of a staging area to contain print jobs ready to be serviced.

There are no default names; choose any name you like, but it is good practice to choose one that is meaningful to users. For example, `laser` is a good name for a laser printer. The name may contain the file-system dependent maximum of alphanumeric characters and underscores. Table 12-1 shows the default filename lengths for several file system types:

Table 12-1. Maximum Filename Lengths for Different File System Types

File System Type	Maximum Filename Length
<code>sfs</code>	255
<code>ufs</code>	255
<code>xf</code> s	255

Don't try to fit a lot of descriptive information into the name; there is a better place for this information. (See “*Printer Description*” in “*Other Printer Configuration Options*” later in this chapter.) Nor should you try to make the name identify the type of printer; users who need to use a particular type of printer can specify it by type rather than name (see the “*Printer Types*” section later in this chapter).

Attribute 2: Device Path

Use the `-v device` option to `lpadmin` to identify the device-path.

The *device-path* attribute defines the path of the device that is the printer. This is the path to the special device file associated with the connecting port. The following are examples of typical device-paths of special device files:

<code>/dev/lp</code>	(parallel port)
<code>/dev/tty0-00</code>	(serial port)
<code>/dev/tty0-01</code>	(serial port)

(For details about using these files, see “*Printer Port Characteristics*” under “*Other Printer Configuration Options*” later in this chapter.)

Attribute 3: Printer Type

Use the `-T printer-type-list` option to `lpadmin` to identify the printer type.

A printer type is the generic name for a printer. Typically it is derived from the manufacturer's name. For example, the ACME Computer Co.'s 356 Dot Matrix Printer might have the type 356. Assigning a “type” for each printer is important because the LP software extracts information about printers from the `terminfo` database on the basis of type. This information includes a list of the printer's capabilities that is used to check the configuration information you supply to the Print Service. (By checking the information you provide against the known capabilities of the type of printer you are configuring, the Print Service can catch inappropriate information you may have supplied.) The `terminfo` database also specifies the control data needed to initialize a particular printer before printing a file.

While you are not required to specify a printer type, it is good practice to do so; you enhance your system's ability to serve your users by classifying, on the basis of type, the printers available through the Print Service.

If you give a list of printer types, separate the names with commas. If you do not define a printer type, the default unknown will be used.

Attribute 4: Content Type

Use the `-I content-type-list` option to `lpadmin` to identify the types of files it can print.

Most printers can print files of two types: the same type as the printer type (if the printer type is defined) and the type `simple`, (meaning an ASCII file) which is the default content type for all printers.

The Default Content Type: simple

Files of content type **simple** are assumed to contain only two types of characters, printable ASCII characters and the following control characters:

backspace	moves the carriage back one space, except at the beginning of a line
tab	moves the carriage to the next tab stop; by default, stops are spaced every 8 columns on most printers
linefeed	moves the carriage to the beginning of the next line (may require special port settings for some printers—see “ <i>Printer Port Characteristics</i> ” later in this chapter)
form feed	moves the carriage to the beginning of the next page
carriage return	moves the carriage to the beginning of the same line (may fail on some printers)

The word “carriage” may be archaic for modern laser printers, but these printers do actions similar to those done by a carriage. If a printer can handle several types of files, including **simple**, you must include **simple** explicitly in the content type list; If you *don't* want a printer to accept files of type **simple**, specify a blank *content-type-list* (**-I ""**) on the **lpadmin** command line.

Attribute 5: Printer Interface

You may use one of three different options to the **lpadmin** command to identify an interface program for the printer:

- e printer** (use the existing interface program of *printer*)
- i interface** (use *interface* as the interface program)
- m model** (copy *model*, an existing model interface program)

If you do not specify an interface program via one of these options, the standard one provided with the LP Print Service will be used. This should be sufficient for most of your printing needs. (See the **lpadmin (1M)** manual page for complete details.)

An interface program is the program the LP Print Service uses to manage the printer each time a file is printed. It has several tasks:

- to initialize the printer port (the connection between the computer and the printer)
- to initialize the printer (restore it to a normal state in case a previously printed file has left it in an unusual state) and set the character pitch, line pitch, page size, and character set requested by the user
- to print a banner page
- to run a filter that prepares the file for printing
- to manage printer faults

If you prefer, however, you can change it to suit your needs, or completely rewrite your own interface program, and then specify it when you add a new printer. See “*Customizing the Print Service*” in the chapter “Advanced Print Service”.) for details on how to customize an interface program.

Examples of Defining Printer Attributes

The following examples show the `lpadmin` command being used to set up two of the most frequently used printer environments.

- Configure a PostScript printer on a serial port.

```
lpadmin -p ps1 -v /dev/tty0-00 -T PS -I PS -m PS
```

The string “`-T PS -I PS -m PS`” is essential.

- Configure a PostScript printer on a parallel port.

```
lpadmin -p ps1 -v /dev/lp -T PS-b -I PS -m PS
```

The string “`-T PS-b`” is also necessary for a PostScript printer on a serial port running in batch mode.

If the Enhanced Security Utilities are installed and running on your system, you should configure your Print Service to take advantage of the available security features. For example, you might configure a printer that is part of the evaluated configuration with the required interface (specified with `-m B2`).

```
lpadmin -p lp1 -v /dev/tty0-11 -T 572 -m B2
```

See the “Security Administration” part of *System Administration* for a description of the evaluated system.

Step 3: Notify the Printer to Accept Jobs

Once the printer has been configured, it will almost be ready to start forwarding print requests. Run the `accept` command so the Print Service can start queuing jobs.

For example, suppose you want to start forwarding requests to a printer named `ps1`. Enter

```
accept ps1
```

Step 4: Turn on the Printer (Logically)

Run the `enable` command so users can start sending job requests to the printer.

To continue the example above, enable `ps1` by entering

```
enable ps1
```

Other Printer Configuration Options

The preceding sections described the requirements for basic printer configurations. You can, however, define many more attributes for your system if you want to offer your users a large choice of options for their print jobs. The following is a list of these optional attributes:

- printer port characteristics
- printer type
- content type
- printer interface
- character sets or print wheels
- alerting to mount a print wheel
- forms allowed
- printer fault alerting
- printer fault recovery
- restrictions on user access
- inclusion of banner page in output
- copying of files to the spool area
- printer description
- default printing attributes
- printer class membership
- system default destination

You need to specify very little of this information to add a new printer to the LP Print Service. The more information you provide, however, the better the printer will satisfy various users' needs.

The descriptions in the sections below will help you understand what this printer configuration information means and how it is used, so that you can decide how to configure your printers. In each section you will also be shown how to specify this information when adding a printer. While you can follow each of the sections in order and correctly configure a printer in several steps, you may want to wait until you've read all the sections before adding a printer, so that you can do it in fewer steps.

Printer Port Characteristics

Printers connected directly to computers and those connected over some networks require that the printer port characteristics be set by the interface program. These characteristics define the low level communications with the printer. Included are the baud rate; use of XON/XOFF flow control; 7, 8, or other bits per byte; type of parity; and output post-pro-

cessing. The standard interface program will use the **stty** command to initialize the printer port, minimally setting the baud rate and a few other default characteristics.

The default characteristics applied by the standard interface program are listed below

Default	Meaning
9600	9600 baud rate
cs8	8-bit bytes
-cstopb	1 stop bit per byte
-parenb	no parity generation
ixon	enable XON/XOFF flow control
-ixany	allow only XON to restart output
opost	post-process data stream as listed below:
-olcuc	don't map lowercase to uppercase
onlcr	map linefeed into carriage-return/linefeed
-ocrnl	don't map carriage-return into linefeed
-onocr	output carriage-returns even at column 0
nl0	no delay after linefeeds
cr0	no delay after carriage-returns
tab0	no delay after tabs
bs0	no delay after backspaces
vt0	no delay after vertical tabs
ff0	no delay after form-feeds

You may find that the default characteristics are sufficient for your printers. However, printers vary enough that you are likely to find that you have to set different characteristics. See the description of the **stty** command in the manual page to find the complete list of characteristics.

If you have a printer that requires printer port characteristics other than those handled by the **stty** program, you will have to customize the interface program. See the section “*Customizing the Print Service*” in “Advanced Print Service” for help.

When you add a new printer, you may specify an additional list of port characteristics. The list you give will be applied after the default list, so that you do not need to include in your list items that you don't want to change. Specify the additional list as follows:

```
lpadmin -p printer-name -o "stty='stty-option-list'"
```

Note that both the double quotes and single quotes are needed if you give more than one item in the *stty-option-list*.

As one example, suppose your printer is to be used for printing graphical data, where line-feed characters should be output alone, without an added carriage-return. You would enter the following command:

```
lpadmin -p printer-name -o "stty=-onlcr"
```

Note that the single quotes are omitted because there's only one item in the list.

As another example, suppose your printer requires odd parity for data sent to it. You would enter the following command:

```
lpadmin -p printer-name -o "stty='parenb parodd cs7'"
```

Printer Types

You can assign several types to a printer, if your printer is capable of emulating more than one kind of printer.

NOTE

The only types of printers allowed on a system on which a B2-level security rating is to be maintained are the Siemens 9022 and <<which others?>> printers.

For example, if your printer can emulate an IBM Proprinter XL, an Epson FX86e, and an HP LaserJet II. The **terminfo** database names these types `593ibm`, `593eps`, and `593hp`, respectively. If you specify more than one printer type, the LP Print Service will use one of them, as appropriate, for each print request.

The following example shows how to use **lpadmin** to associate the type `593ibm` with the printer named `laser`.

```
lpadmin -p laser -T 593ibm
```

NOTE

If you specify more than one printer type, you must specify **simple** as the content type.

Content Types

Some printers, though, can accept (and print properly) several different types of files. When adding this kind of printer, specify the names of the content types the new printer accepts by adding these names to the list. (By default, the list contains only one type: **simple**.) If you're adding a remote printer, list the content types that have been established for it by the administrator of the system on which it resides.

The *content-type-list* is a list of names separated by commas or spaces. If you use spaces to separate the names, enclose the entire list (but not the **-I**) in quotes.

Content type names may look a lot like printer type names, but you are free to choose names that are meaningful to you and the people using the printer. (The names **simple** and **any** are recognized as having particular meanings by the LP Print Service; be sure to use them consistently. The name **terminfo** is also reserved, as a reference to *all* types of printers.) The names must contain no more than 14 characters and may include only letters, digits, and underscores. The following table lists and describes some accepted content types.

Types	Description
troff	Device independent output from troff (troff filters are available in the Advanced Commands package)
otroff	CAT typesetter instructions generated by BSD or troff (old troff)
daisy	Print files intended for a Diablo 630 (“daisy-wheel”) printer.
dmd	Print the contents of a bit-mapped display from a terminal.
tek4014	Print files formatted for a Tektronix 4014 device.
tex	DVI format files
plot	Plotting instructions for Tektronix displays and devices
raster	Raster bitmap format for Varian raster devices
cif	Output of BSD cifpbt
fortran	ASA carriage control format
postscript	PostScript language
pcl	HP Laserjet native output format
simple	ASCII file

When a file is submitted to the LP Print Service for printing with the printer specified by the **-d** any option of the **lp** command, the Print Service searches for a printer capable of handling the job. The Print Service can identify an appropriate printer through either the content type name or the printer type name. Therefore, you may specify either name (or no name) when submitting a file for printing. If the same content type is printable by several different types of printers, you should use the same content type names when you add those printers. This makes it easier for the people using the printers, because they can use the same name to identify the type of file they want printed regardless of the printing destination.

Most manufacturers produce printers that accept simple ASCII files. While these printers are different types (and thus have different initialization control sequences), they may all be capable of handling the same type of file, which we call **simple**. As another example, several manufacturers may produce printers that accept ANSI X3.64 defined escape sequences. However, the printers may not support all the ANSI capabilities; they may support different sets of capabilities. You may want to differentiate them by assigning different content type names for these printers.

Using the Default Content Type

However, while it may be desirable (in situations such as these) to list content types for each printer, it is not always necessary to do so. If you don't, the printer type will be used as the name of the content type the printer can handle. If you have not specified a printer type, the LP Print Service will assume the printer can print only files of content type **simple**. This may be sufficient if you require users to specify the proper printer explicitly and if files are properly prepared for the printer before being submitted for printing.

Printer Interface

Whether and how you specify an interface program on the **lpadmin** command line depends on the program being used.

- If you are using the standard interface program, you needn't specify it when adding a printer.
- If you want to specify one of the system supplied interface programs such as **standard**, the default, give the program name (after **-m**).
- If you are using a non-standard interface program on a local printer, refer to it by either (a) specifying its full pathname, or (b) referring to another printer using the same interface program.
- If you want to specify a customized interface program on the local system, give the printer name (after **-p**), and the pathname of the interface program (after **-i**).
- If you want to specify a customized interface program on another printer, give the name of the printer you're adding (after **-p**), and the name of the printer that is using the customized interface program (after **-e**).

Character Sets, Fonts, or Print Wheels

NOTE

Although your users may use character sets, font cartridges, or print wheels that have been mounted on a printer on a server system (by the administrator of the server system), you cannot mount a character set, font cartridge, or a print wheel on a printer on a client system.

Printers differ in the way they can print in different font styles. Some have changeable print wheels, some have changeable font cartridges, others have preprogrammed, selectable character sets.

When adding a printer, you may specify what print wheels, font cartridges, or character sets are available with the printer.

NOTE

If you're adding a client printer and you want your users to be able to use character sets or print wheels that have been mounted by the administrator of the server system, you must list those character sets and print wheels, just as you would list the character sets and print wheels on a local printer.

Only one of these (a print wheel, a font cartridge, or a character set) is assumed to apply to each printer. From the point of view of the LP Print Service, however, print wheels and changeable font cartridges are the same because they require you to intervene and mount a new print wheel or font cartridge. Thus, for ease of discussion, only print wheels and character sets will be mentioned.

When you list the print wheels or character sets available, you will be assigning names to them. These names are for your convenience and the convenience of the users. Because different printers may have similar print wheels or character sets, you should use common names for all printers. This allows a user to submit a file for printing and ask for a particular font style, without regard for which printer will be used or whether a print wheel or selectable character set is used.

If the printer has mountable print wheels, you need only list their names. If the printer has selectable character sets, you need to list their names and map each one into a name or number that uniquely identifies it in the **terminfo** database. Use the following command to determine the names of the character sets listed in the **terminfo** database.

```
tput -T printer-type csnm 0
```

Printer-type is the name of the printer type in question. The name of the 0th character set (the character set obtained by default after the printer is initialized) will be printed. Repeat the command, using 1, 2, 3, and so on in place of the 0, to see the names of the other character sets. In general, the **terminfo** names should closely match the names used in the user documentation for the printer. However, because not all manufacturers use the same names, the **terminfo** names may differ from one printer type to the next.

NOTE

For the LP Print Service to be able to find the names in the **terminfo** database, you must specify a printer type. See “*Printer Types*” earlier in this section.

To specify a list of print wheel names when adding a printer, enter the following command.

```
lpadmin -p printer-name -S print-wheel-list
```

The *print-wheel-list* is a comma or space separated list of names. If you use spaces to separate the names, enclose the entire list (but not the **-S**) in quotes.

To specify a list of character set names and to map them into **terminfo** names or numbers, enter the following command:

```
lpadmin -p printer-name -S character-set-list
```

The *character-set-list* is also a comma or space separated list; however, each item in the list looks like one of the following:

```
csN=character-set-name  
character-set-name1=character-set-name2
```

The *N* in the first case is a number from 0 to 63 that identifies the number of the character set in the **terminfo** database. The *character-set-name1* in the second case identifies the character set by its **terminfo** name. In either case the name to the right of the equal sign (=) is the name you may use as an alias of the character set.

NOTE

You do not have to provide a list of aliases for the character sets if the **terminfo** names are adequate. You may refer to a character set by number, by **terminfo** name, or by your alias.

For example, suppose your printer has two selectable character sets (sets #1 and #2) in addition to the standard character set (set #0). The printer type is 5310. You enter the following commands to determine the names of the selectable character sets.

```
tput -T 5310 csnm 1  
english  
tput -T 5310 csnm 2  
finnish
```

The words *english* and *finnish*, the output of the commands, are the names of the selectable character sets. You feel that the name *finnish* is adequate for referring to character set #2, but better names are needed for the standard set (set #0) and set #1. You enter the following command to define synonyms.

```
lpadmin -p printer-name -S "cs0=american, english=british"
```

The following three commands will then produce identical results. (The **lp** command routes print jobs to the printer, and in these examples, **lp** will route the print job to any printer with the capability of handling the *cs1* character set.)

```
lp -S cs1 -d any . . .  
lp -S english -d any . . .  
lp -S british -d any . . .
```

If you do not list the print wheels or character sets that can be used with a printer, then the LP Print Service will assume the following: a printer that takes print wheels has only a single, fixed print wheel, and users may not ask for a special print wheel when using the printer; and a printer that has selectable character sets can take any *csN* name or **terminfo** name known for the printer.

Alerting to Mount a Print Wheel

NOTE

This section does not apply if you are making a client printer available to users on your system.

If you have printers that can take changeable print wheels, and have listed the print wheels allowed on each, then users will be able to submit a print request to use a particular print wheel. Until it is mounted though (see the “*Mounting a Form or Print Wheel*” section), a request for a print wheel will stay queued and will not be printed. You could periodically monitor the number of print requests pending for a particular print wheel, but the LP Print Service provides an easier way: You can ask to be alerted when the number of requests waiting for a print wheel has exceeded a specified threshold.

You can choose one of several ways to receive an alert.

- You can receive an alert via electronic mail. See the description of the **mail** command in the manual page for a description of mail on the operating system.
- You can receive an alert written to any terminal on which you are logged in. See the description of the **write** command in the manual page. (If the Enhanced Security Utilities are installed and running on your system, the **write** command will not work and therefore will not be available as an alert mechanism.)
- You can receive an alert through a program of your choice.
- You can receive no alerts.

NOTE

If you elect to receive no alerts, you are responsible for checking to see whether any print requests haven't printed because (a) the proper print wheel isn't mounted, or (b) the proper form isn't mounted. You're also responsible for finding out whether any printer faults have occurred and, if so, for fixing them. (The LP Print Service will not continue to use a printer that has a fault.)

In addition to the method of alerting, you can also set the number of requests that must be queued before you are alerted, and you can arrange for repeated alerts every few minutes until the print wheel is mounted. You can choose the rate of repeated alerts, or you can opt to receive only one alert for each print wheel.

To arrange for alerting to the need to mount a print wheel, enter one of the following commands:

```
lpadmin -S print-wheel-name -A mail -Q requests -W minutes
lpadmin -S print-wheel-name -A write -Q requests -W minutes
lpadmin -S print-wheel-name -A 'command' -Q requests -W minutes
```

The first two commands direct the LP Print Service to send you a mail message or write the message directly to your terminal, respectively, for each alert. The third command directs the LP Print Service to run the *command* for each alert. The shell environment currently in effect when you enter the third command is saved and restored for the execution of *command*; this includes the environment variables, user and group IDs, and current directory. The argument *requests* is the number of requests that need to be waiting for the print wheel before the alert is triggered, and the argument *minutes* is the number of minutes between repeated alerts.

If you do not want the Print Service to issue an alert when a print wheel needs to be mounted, enter the following:

```
lpadmin -S print-wheel-name -A none
```

NOTE

If you want to send an alert to another user, either as mail or as a message, use the third command with the option **-A 'mail login-ID'** or **-A 'write login-ID'**. If you do not specify a *login-ID*, the mail or message will be sent to your current login. (This may not be your login name if you have used the **su** command to change it.)

When you start receiving repeated alerts, you can direct the LP Print Service to stop sending you alerts (for the current case only), by executing the following command.

```
lpadmin -S print-wheel-name -A quiet
```

Once the print wheel has been mounted and unmounted again, alerts will resume if too many requests are waiting. Alerts will also start again if the number of requests waiting falls below the **-Q** threshold and then rises up to the **-Q** threshold again. This happens if waiting requests are canceled or if the type of alerting is changed.

If *print-wheel-name* is **all** in any of the commands above, the alerting condition will apply to all print wheels for which an alert has already been defined.

If you don't define an alert method for a print wheel, you will not receive an alert to mount it. If you do define a method without the **-W** option, you will be alerted once for each occasion.

Forms Allowed

NOTE

For information about how to define, mount, and set up alerting to mount a form, see *“Providing Forms”* in the chapter *“Advanced Print Service”*. This section does not apply if you're making a server printer available to the users on your system.

You can control the use of preprinted forms on any printer, including server printers. (Although you cannot mount forms on client printers, your users may use forms on server printers.)

The LP Print Service will use a list of forms allowed or denied on a printer to warn you against mounting a form that is not allowed on the printer. However, you have the final word on this; the LP Print Service will not reject the mounting. The LP Print Service will, however, reject a user's request to print a file on a printer using a form not allowed on that printer. If, however, the printer is a local printer and the requested form is already mounted, the request will be printed on that form.

Also, if the printer does not have sufficient capabilities to handle a form, the form will not be added to the list of "allowed" forms (that is, the command will be rejected).

The method of listing the forms allowed or denied for a printer is similar to the method used to list users allowed or denied access to the **cron** and **at** facilities. (See the description of the **crontab** command in the manual page. Briefly, the rules are as follows:

- An allow list is a list of forms that are allowed to be used on the printer. A deny list is a list of forms that are not allowed to be used on the printer.
- If a form is in the deny list, access is denied.
- If a form is in the allow list, access is allowed.
- If a form is in neither list, access is denied.
- If neither the allow list or the deny list is present, access is denied.
- If only the allow list is present and the form is not in the list, access is denied.
- If only the deny list is present and the form is not in the list, access is allowed.
- Specifying **all** in the allow list allows all forms; specifying **all** in the deny list denies all forms.

You can add names of forms to either list using one of the following commands:

```
lpadmin -p printer-name -f allow:form-list
lpadmin -p printer-name -f deny:form-list
```

The *form-list* is a comma or space separated list of names of forms. If you use spaces to separate names, enclose the entire list (including the **allow:** or **deny:** but not the **-f**) in quotes.

The first command shown above adds names to the allow list and removes them from the deny list. The second command adds names to the deny list and removes them from the allow list. To make the use of all forms permissible, specify **allow:all**; to deny permission for all forms, specify **deny:all**.

If you do not use this option, the LP Print Service will consider that the printer denies the use of all forms. It will, however, allow you to mount any form that is within the capabilities of the printer. (See the "Mounting a Form or Print Wheel" section in "Advanced Print Service".)

Printer Fault Alerting

NOTE

This section does not apply if you are making a server printer accessible to users on your system.

The LP Print Service provides a framework for detecting printer faults and alerting you to them. Faults can range from simple problems, such as running out of paper or ribbon, or needing to replace the toner, to more serious faults, such as a local power failure or a printer failure. The range of fault indicators is also broad, ranging from dropping carrier (the signal that indicates that the printer is on line), to sending an XOFF, to sending a message. Only two classes of printer fault indicators are recognized by the LP Print Service itself: a drop in carrier and an XOFF not followed in reasonable time by an XON. However, you can add filters that recognize any other printer fault indicators, and rely on the LP Print Service to alert you to a fault when the filter detects it.

NOTE

For a description of how to add a filter, see “*Providing Filters*” in the chapter “Advanced Print Service”. For a description of how a filter should let the LP Print Service know a fault has occurred, see “*Customizing the Print Service*” in the chapter “Advanced Print Service”.

You can choose one of several ways to receive an alert to a printer fault. See “*Alerting to Mount a Print Wheel*” above, for a list.

NOTE

If you elect to receive no alerts, you will need a way of finding out about the faults and fixing them; the LP Print Service will not continue to use a printer that has a fault.

In addition to the method of alerting, you can also arrange for repeated alerts every few minutes until the fault is cleared. You can choose the rate of repeated alerts, or you can opt to receive only one alert per fault.

NOTE

Without a filter that provides better fault detection, the LP Print Service cannot automatically determine when a fault has been cleared except by trying to print another file. It will assume that a fault has been cleared when it is successfully able to print a file. Until that time, if you have asked for only one alert per fault, you will not receive another alert. If, after you have fixed a fault, but

before the LP Print Service has tried printing another file, the printer faults again, or if your attempt to fix the fault fails, you will not be notified. Receiving repeated alerts per fault, or requiring manual re-enabling of the printer (see the “*Printer Fault Recovery*” section below), will overcome this problem.

To arrange for alerting to a printer fault, enter one of the following commands:

```
lpadmin -p printer-name -A mail -W minutes
lpadmin -p printer-name -A write -W minutes
lpadmin -p printer-name -A 'command' -W minutes
```

The first two commands direct the LP Print Service to send you a mail message or write the message directly to your terminal, respectively, for each alert. The third command directs the LP Print Service to run the *command* for each alert. The shell environment currently in effect when you enter the third command is saved and restored for the execution of *command*. The environment includes environment variables, user and group IDs, and current directory. The *minutes* argument is the number of minutes between repeated alerts.

If you do not want the LP Print Service to issue an alert when a fault occurs, enter the following:

```
lpadmin -p printer-name -A none
```

NOTE

If you want mail sent or a message written to another user when a printer fault occurs, use the third command with the option -A '*mail login-ID*' or -A '*write login-ID*'. If you do not specify a *login-ID*, the mail or message will be sent to your current login name. This may not be your login if you have used the **su** command to change it.

Once a fault occurs and you start receiving repeated alerts, you can direct the LP Print Service to stop sending you alerts (for the current fault only), by executing the following command:

```
lpadmin -p printer-name -A quiet
```

NOTE

Use the alert type **quiet** only to terminate an active alert; do not specify **quiet** as the alert type for a new printer.

If the *printer-name* is **all** in any of the commands above, the alerting condition will apply to all printers.

If you don't define an alert method, you will receive mail once for each printer fault. If you define a method without the **-W** option, you will be alerted once for each fault.

Printer Fault Recovery

NOTE

This section does not apply if you are making a remote printer accessible to users on your system.

When a printer fault has been fixed and the printer is ready for printing again, the LP Print Service will recover in one of three ways:

- it will continue printing at the top of the page where printing stopped
- it will restart printing at the beginning of the print request that was active when the fault occurred
- it will wait for you to tell the LP Print Service to re-enable the printer

NOTE

The ability to continue printing at the top of the page where printing stopped requires the use of a filter that can wait for a printer fault to be cleared before resuming properly. Such a filter probably has to have detailed knowledge of the control sequences used by the printer so it can keep track of page boundaries and know where in a file printing stopped. None of the filters supplied with the LP Print Service can do this. If a proper filter is not being used, you will be notified in an alert if recovery cannot proceed as you want.

To specify the way the LP Print Service should recover after a fault has been cleared, enter one of the following commands:

```
lpadmin -p printer-name -F continue
lpadmin -p printer-name -F beginning
lpadmin -p printer-name -F wait
```

These commands direct the LP Print Service, respectively, to continue at the top of the page, restart from the beginning, or wait for you to enter an **enable** command to re-enable the printer (see the “*Enabling and Disabling a Printer*” section in this chapter for information on the **enable** command).

If you do not specify how the LP Print Service is to resume after a printer fault, it will try to continue at the top of the page where printing stopped, or, failing that, at the beginning of the print request.

If the recovery is **continue**, but the interface program does not stay running so that it can detect when the printer fault has been cleared, printing will be attempted every few minutes until it succeeds. You can force the LP Print Service to retry immediately by issuing an **enable** command.

User Access Restrictions

You can control which users are allowed to use a particular printer on your system. For instance, if you're designating one printer to handle sensitive information, you don't want all users to be able to use the printer. Another time you might want to do this is when you're authorizing the use of a high quality printer which produces expensive output.

The LP Print Service will use a list of users allowed or denied access to a printer. The LP Print Service will reject a user's request to print a file on a printer he or she is not allowed to use.

NOTE

If your users have access to remote printers, or if users on other systems have access to printers on your system, make sure that the allow and deny lists for those printers on your computer match the allow and deny lists on the remote system where the remote printers reside. If these two sets of lists don't match, your users may receive conflicting messages (some accepting jobs, and others refusing jobs) when submitting requests to remote printers.

The method of listing the users allowed or denied access to a printer is similar to the method used to list users allowed or denied access to the **cron** and **at** facilities, and the method described above in the "Forms Allowed" section. Briefly, the rules are as follows:

- An allow list is a list of users allowed to use the printer. A deny list is a list of users denied access to the printer.
- If a user is in the deny list, access is denied.
- If a user is in the allow list, access is allowed.
- If a user is in neither list, access is denied.
- If neither the allow list or the deny list is present, access is denied.
- If only the allow list is present and the user is not in the list, access is denied.
- If only the deny list is present and the user is not in the list, access is allowed.
- Specifying **all** in the allow list allows everybody access to the printer; specifying **all** in the deny list denies access to everybody except the user **lp**.

You can add names of users to either list using one of the following commands:

```
lpadmin -p printer-name -u allow:login-ID-list
lpadmin -p printer-name -u deny:login-ID-list
```

The *login-ID-list* is a comma or space separated list of users' login names. If you use spaces to separate logins, enclose the entire list (including the **allow:** or **deny:** but not the **-u**) in quotes. Each item in the *login-ID-list* may take any of the following forms:

<i>login-ID</i>	a user on the local system
<i>system-name!login-ID</i>	a user on system <i>system-name</i>
<i>system-name!all</i>	all users on system <i>system-name</i>
all!login-ID	a user on all systems
all	all users on the local system
all!all	all users on all systems

NOTE

It is strongly recommended to use **all!all** since other login ID lists may cause printer access problems.

The first command shown above adds the names to the allow list and removes them from the deny list. The second command adds the names to the deny list and removes them from the allow list.

If you do not use this option, the LP Print Service will assume everybody on the local system may use the printer.

NOTE

The restrictions applied by the printer's level range override any permissions granted by the **lpadmin** command. If users try to print jobs that lie outside the printer's level range, the jobs will be rejected, even if the users are on the **allow** list for the printer.

Inclusion of Banner Page in Output

Banner and trailer pages are always printed for each job. The **-o nobanner** option of **lpadmin** is silently ignored when the Enhanced Security Utilities are installed.

The banner and trailer pages are marked with information on the security level of the output, printed as a fully qualified level name. This information is never truncated. If it is longer than one page, extra banner and trailer pages are printed to contain it. Paginated output may also contain security level information on each page of the output. Users may delete this information from the output with the **-o nolabels** option of **lp**. Use of this option is an auditable event.

In addition to the print job number and the security level, the banner and trailer pages always contain a random number generated by the system. This number can be used to detect the true start and end of a print job, in case a malicious user includes false banner or trailer pages in a print job. This random number is never visible to the user until the job is printed.

Controlling Copying of Files

Files submitted for printing are not always automatically copied to the spool area. If a user submits a print job with the **-c** option of **lp**, the system copies the file(s) to be printed.

You can configure the LP service so that files are always copied to the spool area by using the **-O** option of **lpadmin**. This option takes one of two arguments:

- O copy** forces the system to copy all files submitted for printing,
- O nocopy** leaves the copying of files to the discretion of the user. This is the default. If **-O nocopy** is used, the files are copied only if the user submits the print job with **lp -c**.

(Copying files to a spool area ensures that the files cannot be modified in the time between the submission of the **lp** request and the actual printing of the job.)

The copying of files is actually controlled by the value of the **copy-files** keyword in the file **/etc/default/lp**. The initial value is **nocopy**; to change the value to **copy**, use the following command:

```
lpadmin -O copy
```

The LP system checks the value of this keyword before printing each job. If it is **copy**, the file to be printed is automatically copied to the spool area. If the value of **copy-files** is **nocopy**, the files are copied only if the user submitted the job with **lp -c**.

NOTE

If you specify **-O copy**, print jobs may fail if the spool area becomes full. If you use this option, you should ensure that the spool area is large enough to handle your projected workload.

Printer Description

An easy way to give users of the LP Print Service helpful information about a printer is by adding a description of it. This description can contain any message you'd like, including the number of the room where the printer is found, the name of the person to call with printer problems, and so forth.

Users can see the message when they use the **lpstat -D -p printer-name** command.

To add a description of a printer, enter the following command.

```
lpadmin -p printer-name -D 'text'
```

The *text* is the message. You'll need to include the quotes if the message contains blanks or other characters that the shell might interpret if the quotes are left out.

Default Printing Attributes

The attributes of a printing job include the page size, print spacing (character pitch and line pitch), and **stty** options for that job. If a user requests a job to be printed on a particular form, the printing attributes defined for that form will be used for that job. When, however, a user submits a print request without requesting a form, the Print Service uses one of several sets of default attributes.

- If the user has specified attributes to be used, those attributes will take precedence.
- If the user has not specified attributes, but the administrator has executed the **lpadmin -o** command, the default attributes for that command will take precedence.
- If neither of the above, the attributes defined in the **terminfo** database for the printer being used will take precedence.

The LP Print Service lets you override the defaults for each printer. Doing so can make it easier to submit print requests by allowing you to designate different printers as having different default page sizes or print spacing. A user can then simply route a file to the appropriate printer to get a desired style of output. For example, you can have one printer dedicated to printing wide (132-column) output, another printing normal (80-column by 66-line) output, and yet another printing letter quality (12 characters per inch, 8 lines per inch) output.

You can independently specify four default settings, page width, page length, character pitch, and line pitch. You can scale these to fit your needs: The first two can be given in characters and lines, or inches or centimeters. The last two can be given as characters and lines per inch or per centimeter. In addition, the character pitch can be specified as *pica* for 10 characters per inch (cpi.), *elite* for 12 cpi, or *compressed* for the maximum cpi the printer can provide (up to a limit of 30 cpi).

Set the defaults using one or more of the following commands:

```
lpadmin -p printer-name -o width=scaled-number
lpadmin -p printer-name -o length=scaled-number
lpadmin -p printer-name -o cpi=scaled-number
lpadmin -p printer-name -o lpi=scaled-number
```

Append the letter *i* to the *scaled-number* to indicate inches, or the letter *c* to indicate centimeters. The letter *i* for character pitch (cpi) or line pitch (lpi) is redundant. You can also give *pica*, *elite*, or *compressed* instead of a number for the character pitch.

If you don't provide defaults when you configure a printer, then the page size and print spacing will be taken from the data for your printer type in the **terminfo** database. (If you do not specify a printer type, then the type will be *unknown*, for which there is an entry in the **terminfo** database.) You can find out what the defaults will be by first defining the printer configuration without providing your own defaults, then using the **lpstat** command to display the printer configuration. The command

```
lpstat -p printer-name -l
```

will report the default page size and print spacing.

System Default Destination

You can define the printer or class to be used to print a file when the user has not explicitly asked for a particular destination and has not set the LPDEST shell variable. The printer or class must already exist.

Make a printer or class the default destination by entering the following command:

```
lpadmin -d printer-or-class-name
```

If you later decide there should be no default destination, enter a null *printer-or-class-name* as in the following command.

```
lpadmin -d
```

If you don't set a default destination, there will be none. Users will have to explicitly name a printer or class in each print request, (unless they specify the **-T content-type** option, as shown in the examples below) or will have to set the LPDEST shell variable with the name of a destination.

Printer Class Membership

It is convenient to treat a collection of printers as a single class. The benefit is that a user can submit a file for printing by a member of a class, and the LP Print Service will pick the first printer in the class that it finds free. This allows faster turn-around, as printers are kept as busy as possible.

Classes aren't needed if the only purpose is to allow a user to submit a print request by type of printer. The **lp -T content-type** command allows a user to submit a file and specify its type. If no printer default destination is set, the first available printer that can handle the type of the file will be used to print it. The LP Print Service will avoid using a filter, if possible, by choosing a printer that can print the file directly over one that would need it filtered first.

Grouping printers as a class can be useful if you have a series of printers that should be used in a particular order. If you have a high speed printer and a low speed printer, for instance, you probably want the high speed printer to handle as many print requests as possible, with the low speed printer reserved for use when the other is busy. Because the LP Print Service always checks for an available printer in the order the printers were added to a class, you could add the high speed printer to the class before the low speed printer, and let the LP Print Service route print requests in the order you wanted.

Until you add a printer to a class, it doesn't belong to one. If you want to do so, use the following command:

```
lpadmin -p printer-name -c class-name
```

If the class *class-name* doesn't exist yet, it will be created. If you want to remove a printer from a class without deleting the printer, enter the following command:

```
lpadmin -p printer-name -r class-name
```

The class name may contain a maximum of 14 alphanumeric characters and underscores.

NOTE

Class names and printer names must be unique. Because they are, a user can specify the destination for a print request without having to know whether it's a class of printers or a single printer.

Putting It All Together

It is possible to add a new printer by completing a number of separate steps, shown in the commands described above. You may find it easier, however, to enter one or two commands that combine all the necessary arguments. Below are some examples. In all these examples, it is assumed you have defined the device special file in the Device Database with the `putdev` command and that you have allocated the device, with the `admalloc` command, at an appropriate security level range. See the “Introduction to Security” chapter in “*System Administration*” Volume 1, for information on these steps.

Example 1: Configuring a PostScript Printer on a Serial Port

To configure a PostScript printer on a serial port, enter

```
lpadmin -p ps1 -v /dev/tty0_00 -T PS -I PS -m PS
```

In this command line, it's necessary to include `-T PS`, `-I PS`, and `-m PS`.

Example 2: Configuring a PostScript Printer on a Parallel Port

To configure a PostScript printer on a parallel port, enter

```
lpadmin -p ps1 -v /dev/lp -T PS-b -I PS -m PS
```

(You must also specify `-T PS-b` for a PostScript printer on a serial port running in “batch-mode.”)

Example 3: Adding a Printer with the Standard Interface

Add a new printer called `lp1` (of the type 455) on printer port `/dev/tty0_00`. It should use the standard interface program, with the default page size of 90 columns by 71 lines, and linefeeds should *not* be mapped into carriage return/linefeed pairs. (The following command line is split into two lines for readability.)

```
lpadmin -p lp1 -v /dev/tty0_00 -T 455 \  
-o "width=90 length=71 stty=-onlcr"
```

Example 4: Configuring an Alert

When you added the `lp1` printer in Example 5, you did not set the alerting. Do this now: have the LP Print Service alert you—by writing to the terminal on which you are logged in—every 10 minutes after a fault until you fix the problem.

```
lpadmin -p lp1 -Awrite -W 10
```

Example 5: Configuring a Secure Printer

To configure a B2-evaluated printer with the required B2 interface (specified with **-m**), enter

```
lpadmin -p lp1 -v /dev/term/tty0-11 -T 572 -m B2
```

Examining a Printer Configuration

Once you've defined a printer configuration, you'll probably want to review it to see if it is correct. If, after examining the configuration, you find you've made a mistake, just reenter the command that applies to the part that's wrong.

Use the **lpstat** command to examine both the configuration and the current status of a printer. Use the short form of this command to see if the printer exists and if it is busy, idle, or disabled. Use the long form of the command to get a complete configuration listing.

Enter one of the following commands to examine a printer.

```
lpstat -p printer-name
lpstat -p printer-name -l
```

(The second command is the long form.) With either command you should see one of the following lines of output.

```
printer printer-name now printing request-id. enabled since date.
printer printer-name is idle. enabled since date.
printer printer-name disabled since date.
reason
printer printer-name waiting for auto-retry.
reason
```

The waiting for auto-retry output shows that the LP Print Service failed in trying to use the printer (because of the *reason* shown), and that it will try again later.

With the long form of the command, you should also see the following output:

```
Form mounted: form-name
Content types: content-type-list
Printer type: printer-type
Description: comment
Connection: connection-info
Interface: pathname
On fault: alert-method
After fault: fault-recovery
Users allowed:
    user-list
Forms allowed:
    form-list
Banner required
Character sets:
    character-set-list
Default pitch: integer CPI, integer LPI
Default page size: scaled-decimal-number wide, scaled-decimal-number long
Default port settings: stty-option-list
```

Display Status of Printer Service

You can display the complete status of the LP Print Service by using the **lpstat** command. For example, the following command

```
lpstat -t
```

will display information on the following:

- The forms that are currently available to you.
- The availability of each printer.
- A list of currently pending print requests.
- The available character sets and printwheels.

Making Printers Available

There are two steps in making a printer ready for use after you've defined the printer configuration. First, you must instruct the LP Print Service to accept print requests for the new printer. To do this, run the **accept** command. Second, you must activate or enable the new printer. To do this, run the **enable** command. These tasks are separate steps because you may have occasion to want to do one but not the other.

NOTE

By default, the **lp** command sends mail about the status of print requests to the **lp** login name (which should no longer be used for administration of the LP Print Service). Therefore, to be sure you receive mail from **lp**, forward that mail to the login name you'll be using while doing LP Print Service administration.

Accepting Print Requests for a New Printer

Telling the LP Print Service to accept print requests for the new printer is done with the **accept** command. You will read more about this command in a later section, “*Managing the Printing Load*.” For now, all you need to know is that you should enter the following command to let this printer be used.

```
accept printer-or-class-name
```

As you can see, this command is needed to let the LP Print Service start accepting print requests for a class, too. To prevent the Print Service from accepting any more requests, execute the following command.

```
reject printer-or-class-name
```

Enabling and Disabling a Printer

Because you may want to make sure, before printing begins, that the correct form is loaded in your printer, the correct print wheel or font cartridge is in place, and the printer is on-line, the LP Print Service will wait for an explicit signal from you before it starts printing files. Once you have verified that all the necessary components are in place, you can request the beginning of printing by issuing the **enable** command for a particular printer, as follows:

```
enable printer-name
```

If you want to enable several printers simultaneously, list the printers (separating the names with spaces) on the same line as the **enable** command. Don't enclose the list in quotes.

The **enable** command recognizes the attributes of a print device and passes them to the Print Service. Therefore if you want to change the definition or allocation for a device, you must do so before running **enable**. To make changes for a printer already enabled, disable the device, make the desired changes, and run **enable**. The new device attributes will become effective when **enable** is executed.

Disabling a printer stops further print requests from being printed. (It does not, however, stop the LP Print Service from accepting new print requests for the printer.) From time to time you may want to disable a printer. For example, you may want to interrupt a print request, change a form or print wheel, or change a device definition, in any of which cases you should disable the printer first. Normally, disabling a printer also stops the request that's currently being printed, placing it back in the queue so it can be printed later. You

can, however, have the LP Print Service wait until the current request finishes, or even cancel the request outright.

If you change the printer level range, the device definition and the device allocation for a printer with an existing set of attributes, the new attributes take effect when the device is re-enabled.

To disable a printer, enter one of the following commands:

```
disable -r "reason" printer-name  
disable -W -r "reason" printer-name  
disable -c -r "reason" printer-name
```

The first command disables the printer, stopping the currently printing request and saving it for printing later. The other commands also disable the printer, but the second one makes the LP Print Service wait for the current request to finish, while the third cancels the current request.

NOTE

The **-c** and **-W** options are not valid when the **disable** command is run to stop a remote printer because, when run for a remote printer, **disable** stops the transferring (rather than the actual printing) of print requests.

The *reason* is stored and displayed whenever anyone checks the status of the printer. You can omit it (and the **-r** option) if you don't want to specify a reason.

Several printers can be disabled at once by listing their names in the same line as the **disable** command.

NOTE

You can only enable or disable local printers; the loading of forms, print wheels, and cartridges in a remote printer and the enabling of that printer are the responsibility of the administrator of the remote system. You can, however, enable or disable the transfer of print requests to the remote system on which a printer is located. Only individual printers can be enabled and disabled; classes cannot.

Allowing Users to Enable and Disable a Printer

NOTE

Users without the appropriate privileges can enable or disable printers only if their security level is within the range of the printer.

You may want to make the **enable** and **disable** commands available for use by other users. This availability is useful, for instance, if you have a small organization where anyone who spots a problem with the printer should be able to disable it and fix the problem. This is *not* a good idea if you want to keep others from interfering with the proper operation of the Print Service.

If you want to allow others access to the **enable** and **disable** commands, you must change the access mode of `/usr/bin/enable` so that **other** can execute the commands. Clearing the bit removes this privilege.

In addition, you must change the level of the **enable** command from `SYS_PRIVATE` to `SYS_PUBLIC`.

To allow everybody to run **enable** and **disable**, enter the following:

```
chmod 111 /usr/bin/enable
chlvl SYS_PUBLIC /usr/bin/enable
```

To prevent others from running **enable** and **disable**, enter the following:

```
chmod 110 /usr/bin/enable
chlvl SYS_PRIVATE /usr/bin/enable
```

Managing the Printing Load

Occasionally you may need to stop accepting print requests for a printer or move pending print requests from one printer to another. There are various reasons why you might want to do this, such as the following:

- the printer needs periodic maintenance
- the printer is broken
- the printer has been removed
- you've changed the configuration so that the printer is to be used differently
- too many large print requests are queued for one printer and should be spread around

If you are going to make a big change in the way a printer is to be used, such as stopping its ability to handle a certain form, changing the print wheels available for it, or disallowing some users from using it, print requests that are currently queued for printing on it will have to be moved or canceled. The LP Print Service will attempt to find alternate printers, but only if the user doesn't care which printer is to be used. Requests for a specific printer won't be automatically moved; if you don't move them first, the LP Print Service will cancel them.

If you decide to take a printer out of service, to change its configuration, or to lighten its load, you may want to move print requests off it and reject additional requests for it for awhile. To do so, use the **lpmove** and **reject** commands. If you do reject requests for a printer, you can accept requests for it later, by using the **accept** command.

Rejecting Requests for a Printer or Class

To stop accepting any new requests for a printer or class of printers, enter one of the following commands.

```
reject -r "reason" printer-name  
reject -r "reason" class-name
```

You can reject requests for several printers or classes in one command by listing their names on the same line, separating the names with spaces. The *reason* will be displayed whenever anyone tries to print a file on the printer. You can omit it (and the **-r**) if you don't want to specify a reason.

Although the **reject** command stops any new print requests from being accepted, it will not move or cancel any requests currently queued for the printer. These will continue to be printed as long as the printer is enabled.

Accepting Requests for a Printer or Class

After the condition that led to rejecting requests has been corrected or changed, enter one of the following commands to start accepting new requests.

```
accept printer-name  
accept class-name
```

Again, you can accept requests for several printers or classes in one command by listing their names on the same line.

You will always have to use the **accept** command for a new printer or class after you have added it, because the LP Print Service does not initially accept requests for new printers or classes.

Moving Requests to Another Printer

If you specify **-d any** when you run the **lp** command to queue a job, the Print Service schedules the job for a particular printer. If another becomes available first, the job is sent to the latter printer. If a job is scheduled for a given printer and you run **lpmove** to get jobs off that printer, that job will be moved off and the destination will change from any to the printer you've specified on the **lpmove** command line. Users may not have intended this side effect. If not, run the following command:

```
lp -i request-ID -d any
```

This command will change the destination for the requested job to the original destination: any (that is, any available printer).

If you have to move requests from one printer or class to another, enter one of the following commands


```
lpmove request-id printer-name
lpmove printer-name1 printer-name2
```

You can give more than one request ID before the printer name in the first command.

When you issue a command to move a request, the request on the source printer is compared to capabilities (including printer range) on the destination printer. The request is moved only when capabilities match.

The first command above moves the listed requests to the printer *printer-name*. The second command tries to move *all* requests currently queued for *printer-name1* to *printer-name2*. If some requests cannot be printed on the new printer, they will be left in the queue for the original printer. When the second command is used, the LP Print Service also stops accepting requests for *printer-name1* (the same result you would obtain by running the command **reject** *printer-name2*).

Examples

Here are some examples of how you might use these three commands:

Example 1

You've decided it is time to change the ribbon and perform some preventive maintenance on printer `lp1`. First, to prevent the loss of print requests already queued for `lp1`, you move all requests from printer `lp1` to printer `lp2`.

```
lpmove lp1 lp2
```

After the requests are moved, make sure the LP Print Service does not print any more requests on `lp1` by disabling it.

```
disable lp1
```

Now you may physically disable the printer and start working on it.

Example 2

You've finished changing the ribbon and doing the other work on `lp1`; now it's time to bring it back into service. Execute the following commands in any order:

```
accept lp1
enable lp1
```

See the “*Enabling and Disabling a Printer*” section under “*Making Printers Available*” in this chapter.

Example 3

You notice that someone has queued several large files for printing on the printer `laser1`. Meanwhile `laser2` is idle because no one has queued requests for it. Move the two biggest requests (`laser1-23` and `laser1-46`) to `laser2`, and reject any new requests for `laser1` for the time being.

```
lpmove laser1-23 laser1-46 laser2
reject -r "too busy--will reopen later" laser1
```

Starting and Stopping the LP Print Service

Under normal operation, you should never have to start or stop the LP Print Service manually. It is automatically started each time the operating system is started, and stopped each time the operating system is stopped. If, however, you need to stop the LP Print Service without stopping the operating system as well, you can do so by following the procedure described below.

Stopping the LP Print Service will cause all printing to cease within seconds. Any print requests that have not finished printing will be printed in their entirety after the LP Print Service is restarted. The printer configurations, forms, and filters in effect when the LP Print Service is stopped will be restored after it is restarted.

Manually Stopping the Print Service

To stop the LP Print Service manually, enter the following command:

```
lpshut
```

The message "Print services stopped." will appear, and all printing will cease within a few seconds. If you try to stop the LP Print Service when it is not running, you will see the message "Print services already stopped."

Manually Starting the Print Service

To restart the LP Print Service manually, enter the following command:

```
/usr/lib/lpsched
```

The message "Print services started." will appear. It may take a minute or two for the printer configurations, forms, and filters to be reestablished, before any saved print requests start printing. If you try to restart LP when it is already running, you will see the message "Print services already active."

NOTE

The LP Print Service does not have to be stopped to change printer configurations or to add forms or filters.

Managing Classes of Related Printers

A group of printers can be defined to constitute a single, named *class*. When users submit a file for printing by a class, LP picks the first printer in the class that it finds free.

Classes are used to establish a priority ordering of equivalent printers. For example, group a high-speed printer and a low-speed printer in a single class; the high-speed printer handles as many requests as possible and the low-speed printer is reserved for use when the other is busy. This keeps both printers as busy as possible.

You must add each printer to the system before adding it to a class. See the section “*Configuring a Directly Connected Printer*” for information on adding a printer.

Adding a New Class

Selecting **add** displays a form requesting you to supply the new class name and the list of printers in the class. Class names and printer names must be unique.

Adding a new class can also be accomplished by giving the command:

```
/usr/sbin/lpadmin -p printer-name -c class-name
```

where *class-name* does not yet exist.

Listing Printers in Classes

Selecting **list** displays a form requesting you to supply the names of one or more print classes. After you press Enter, the system displays the printers that comprise each class.

Modifying Membership of a Class

To modify class membership using the menu, select **modify** from the **Classes** menu. This displays a form asking you to specify the name of the class and whether you want to add or remove printers from that class (**add** is the default). After you choose, you are asked for the names of the printers to be added or removed.

By command, you add a printer to a class using the same command used to create a new class (see above).

To remove a printer from a class by command, enter:

```
/usr/sbin/lpadmin -p printer-name -r class-name
```

When the last printer in a class is removed, the class is automatically removed.

Removing a Class

Selecting **remove** displays a form requesting you to supply the names of the classes you wish removed. You can only remove a class if it has no pending print requests. If there are pending requests, you have to first move them to another printer or class (using the **lpmove** command), or cancel them (using the **cancel** command).

Removing the last remaining printer of a class automatically removes the class as well. The removal of a class, however, does not cause the removal of printers that were members of the class.

To remove a class by command, enter:

```
/usr/sbin/lpadmin -x class-name
```

Managing Active Print Requests

Occasionally you may need to stop accepting print requests for a printer or move print requests from one printer to another. There are various reasons why you might want to do this, such as the need for periodic maintenance, a broken printer, a removed printer, a changed configuration or an imbalance in request loads.

Canceling Print Requests

You can cancel print requests for specific printers by selecting the **cancel** option and entering the printers affected.

You can also cancel requests by issuing these commands:

```
lpstat -o
```

gets the request identification (request-id). Then use

```
cancel request-id
```

Putting a Request on Hold

Any request that has not finished printing can be put on hold. You can stop its printing, if it currently is printing, and keep it from printing until you resume it. A user can also put his

or her own request on hold and then resume it, but cannot resume a print request you have put on hold.

Use the following command to place a request on hold:

```
lp -i request-id -H hold
```

Releasing Held Print Requests

You can release held print requests by entering the following command:

```
lp -i request-id -H resume
```

Once resumed, a request continues to move up the queue and will eventually print. If it had been printing when you held it, it will be the next request to print. Normally it will start printing from the beginning, with page one, but you can have it start printing at a later page. Enter the following command to resume the request at a different page:

```
lp -i request-id -H resume -P starting-page-
```

The final dash is needed to specify the starting page and all subsequent pages.

You can also issue the command:

```
lp -i request-id -H immediate
```

to release the held request and move it up for immediate printing.

NOTE

The ability to print a subset of pages requires the presence of a filter that can handle this. The default filter used by LP does not. An attempt to resume a request on a later page will be rejected if an appropriate filter is not being used.

Moving Requests to A New Destination

If you have to move requests from one printer or class to another, either select **move** from the **Print Requests** menu or enter one of the following commands:

```
/usr/sbin/lpmove request-id printer-name  
/usr/sbin/lpmove printer-name1 printer-name2
```

You can give more than one request ID before the printer name in the first command.

The first command above moves the listed requests to the printer named. The latter command moves *all* requests currently queued for the first printer to the second printer. When the latter command is used, LP will also no longer accept requests for the first printer.

Enter

```
lpstat -l -o
```

to check that requests were moved.

If you select **move** from the menu, you will fill in a form with the printers to be moved from, the IDs of the requests to be moved (the default is “all”), and the new printer destination identification.

Moving a Request Around in the Queue

Changing the Priority for a Request

Print requests that are still waiting to print can be reassigned a new priority. This overrides any existing priorities and will reposition the request in the queue to put it ahead of lower priority requests or behind any others at the same or higher priority.

Enter the following command to change the priority of a request:

```
lp -i request-id -q new-priority-level
```

You can change only one request at a time with this command.

If a request is already printing, you cannot change its priority.

Moving a Request to the Head of the Queue

You can move a print request to the head of the queue where it will be the next one eligible for printing. If it must start printing immediately but another request is currently printing, you can hold the other request as described above.

Enter the following command to move a print request to the head of the queue:

```
lp -i request-id -H immediate
```

Only a user with appropriate privileges can move a request like this; regular users cannot use the **-H immediate** option.

If you set more than one request for immediate printing, they will print in the reverse order set; that is, the request moved to the head of the queue most recently will print first.

Troubleshooting

Here are a few suggestions of what to do if you are having particular problems.

Problem: No Output (Nothing Is Printed)

The printer is sitting idle; nothing happens. First, check the documentation that came with the printer to see if there is a self-test feature you can invoke, or if there are additional troubleshooting suggestions; make sure the printer is working before continuing.

There are three possible explanations when you don't receive any output: (1) the printer might not be connected to the computer; (2) the printer might not be enabled; (3) the baud rate for the computer and the printer might not be set correctly; or (4) the owner and mode of your port might not be set correctly. The rest of this section describes each of these situations in detail.

Is the Printer Connected to the Computer?

The type of connection between a computer and a printer may vary. Consult the manufacturer's installation manual for instructions on connecting printers. A null modem attachment may be required on your connection.

Is the Printer Enabled?

The printer must be "enabled" in two ways: First, the printer must be turned on and ready to receive data from the computer. Second, the LP Print Service must be ready to use the printer. If you receive error messages when setting up your printer, follow the "fixes" suggested in the messages. When the printer is set up, issue the commands

```
accept printer-name
enable printer-name
```

where *printer-name* is the name you assigned to the printer for the LP Print Service. Now submit a sample file for printing:

```
lp -d printer-name file-name
```

Is the Baud Rate Correct?

If the baud rate (the rate at which data is transmitted) is not the same for both the computer and the printer, sometimes nothing will be printed (see below).

For a PostScript Printer: Do You Have Access to It?

If the Enhanced Security Utilities are installed and running on your system, you—as an administrator—will not have access to PostScript filters because these filters carry MAC labels of `USER_PUBLIC`.

At times you may want access to these filters. For example, you may want to use them as a way of verifying they're working properly. To make them accessible, temporarily change the level of the filters (located in `/usr/lib/lp/postscript`) to `SYS_PUBLIC`.

Problem: A Print Request That Won't Cancel

If a particular request keeps getting canceled incorrectly, try the following command:

```
lpshut; cd /var/spool/lp; find requests tmp! -type d\  
! -name.SEQF -exec rm {}; /usr/lp/lpsched
```

Problem: No Output and No Notification from the lp Command

If you do not receive mail messages from the **lp** command when your print requests fail to print, the reason may be that you have not had mail from **lp** forwarded to the login name you use while doing LP Print Service administration. Because, by default, **lp** sends all messages to the **lp** login name, you must arrange to have mail forwarded to your LP administration login before you start sending requests to the Print Service. Otherwise, you will not receive mail from **lp** when your requests fail.

Problem: Printer Stops Working after Printing 2 or 3 Pages

An HP Laserjet printer on a serial connection without a PostScript card will be disabled by the LP scheduler after printing only two or three pages of a multipage document. This happens because the HP Laserjet printer drops DTR (data terminal ready) and DCD (data carrier detect) signals during flow control. The operating system detects the loss of carrier and hangs up the line.

This problem can be circumvented by setting the line to ignore the loss of carrier by executing **stty clocal** on the line. You can reconfigure the printer to set the line from the specified printer destination by executing the following command:

```
lpadmin -p printer_name -o stty="'clocal -onlcr'
```

Problem: Illegible Output

The printer is printing, but the output is not readable. There are four possible explanations for this situation: (1) the baud rate for the printer might not match the baud rate for the computer, (2) the parity setting of the computer might be incorrect, (3) the tabs might be set incorrectly, (4) the printer type might not have been set correctly. The rest of this section describes each of these situations in detail.

Is the Baud Rate Correct?

Usually, when the baud rate of the computer doesn't match that of the printer, you'll get some output but it will not look at all like what you submitted for printing. Random characters will appear, with an unusual mixture of special characters and unlikely spacing.

Read the documentation that came with the printer to find out what its baud rate is. It should probably be set at 9600 baud for optimum performance. If it isn't set to 9600 baud, you can have the LP Print Service use the correct baud rate (by default it uses 9600). If the printer is connected via a parallel port, the baud rate is irrelevant.

To set a different baud rate for the LP Print Service, enter the following command:

```
lpadmin -p printer-name -o stty=baud-rate
```

Now submit a sample file for printing (explained earlier in this section).

Is the Parity Setting Correct?

Some printers use a “parity bit” to ensure that the data received for printing has not been garbled in transmission. The parity bit can be encoded in several ways; the computer and the printer must agree on which one to use. If they do not agree, some characters either will not be printed or will be replaced by other characters. Generally, though, the output will look approximately correct, with the spacing of “words” typical for your document and many letters in their correct place.

Check the documentation for the printer to see what the printer expects. The LP Print Service will not set the parity bit by default. You can change this, however, by entering one of the following commands:

```
lpadmin -p printer-name -o stty=oddp  
lpadmin -p printer-name -o stty=evenp  
lpadmin -p printer-name -o stty=-parity
```

The first command sets odd parity generation, the second sets even parity. The last command sets the default, no parity.

If you are also setting a baud rate other than 9600, you may combine the baud rate setting with the parity settings, as in the sample command below.

```
lpadmin -p printer-name -o “stty='evenp 1200'”
```

Are the Tabs Set Correctly?

If the printer doesn't expect to receive tab characters, the output may contain the complete content of the file, but the text may appear in a chaotic looking format, jammed up against the right margin.

Have You Selected the Correct Printer Type?

See “*Problem: Wrong Character Set or Font*” below.

Problem: The Printing Is Legible but the Spacing Is Wrong

The output contains all of the expected text and may be readable, but the text appears in an undesirable format: double spaced, with no left margin, run together, or zig-zagging down the page. These problems can be fixed by adjusting the printer settings (if possible) or by having the LP Print Service use settings that match those of the printer. The rest of this section provides details about solving each of these types of problems.

Double Spaced Text?

Either the printer's tab settings are wrong or the printer is adding a linefeed after each carriage return. (The LP Print Service has a carriage return added to each linefeed, so the combination causes two linefeeds.) You can have the LP Print Service not send tabs or not add a carriage return by using the `stty -tabs` option or the `-onlcr` option, respectively.)

```
lpadmin -p printer-name -o stty=-tabs
lpadmin -p printer-name -o stty=-onlcr
```

Text Has No Left Margin, Is Run Together, or Is Jammed Up?

The printer's tab settings aren't correct; they should be set every eight spaces. You can have the LP Print Service not send tabs by using the `-tabs` option.

```
lpadmin -p printer-name -o stty=-tabs
```

Text Zig Zags Down the Page?

The `stty onlcr` option is not set. This is set by default, but you may have cleared it accidentally.

```
lpadmin -p printer-name -o stty=onlcr
```

Letters Are Poorly Spaced?

A request for Courier font on a PostScript printer can sometimes result in letters that are too widely separated or that overwrite each other. This can happen when the font is not resident on the printer or has not been downloaded correctly.

A Combination of Problems?

If you need to use several of these options to take care of multiple problems, you can combine them in one list, as shown in the sample command below. Include any baud rate or parity settings, too.

```
lpadmin -p printer-name -o "stty='-onlcr -tabs 2400'"
```

Problem: Wrong Character Set or Font

If the wrong printer type was selected when you set up the printer with the LP Print Service, the wrong “control characters” can be sent to the printer. The results are unpredictable and may cause output to disappear or to be illegible, making it look like the result of one of the problems described above. Another result may be that the wrong control characters cause the printer to set the wrong character set or font.

If you don't know which printer type to specify, try the following to examine the available printer types. First, if you think the printer type has a certain name, try the following command:

```
tput -T printer-type longname
```

The output of this command will appear on your terminal: a short description of the printer identified by the *printer-type*. Try the names you think might be right until you find one that identifies your printer.

If you don't know what names to try, you can examine the **terminfo** directory to see what names are available. Warning: There are probably many names in that directory. Enter the following command to examine the directory.

```
ls -R /usr/share/lib/terminfo/*
```

Pick names from the list that match one word or number identifying your printer. Try each of the names in the other command above.

When you have the name of a printer type you think is correct, set it in the LP Print Service by entering the following command:

```
lpadmin -p printer-name -T printer-type
```

Problem: An Attempt to Dial Out Fails

The LP Print Service uses the Basic Networking Utilities to handle dial out printers. If a dialing failure occurs and you are receiving printer fault alerts, the LP Print Service reports the same error reported by the Basic Networking software for similar problems. (If you haven't arranged to receive fault alerts, they are mailed, by default, to the user **lp**.) See the “Error Messages” chapter in this book for explanations.

Problem: The Printer Is Idle

There are several reasons why you may find a printer idle and enabled but with print requests still queued for it:

- The print requests need to be filtered. Slow filters run one at a time to avoid overloading the system. Until a print request has been filtered (if it needs slow filtering), it will not print. Use the following command to see if the first waiting request is being filtered.

```
lpstat -l -o
```

- The printer has a fault. After a fault has been detected, printing resumes automatically, but not immediately. The LP Print Service waits about five minutes before trying again, and continues trying until a request is printed successfully.

You can force a retry immediately by enabling the printer as follows:

```
enable printer-name
```

- A dial out printer is busy or doesn't answer, or all dial out ports are busy. As it does following a fault, the LP Print Service waits five minutes before trying to reach a dial out printer again. If the dial out printer can't be reached for an hour or two (depending on the reason), the LP Print Service finally alerts you to a possible problem. You can force a retry immediately by enabling the printer as follows:

```
enable printer-name
```

Problem: Warning Messages About Unrecognized Options

Some systems may issue warnings about unrecognized options, such as the `locale=` or `flist=` options, when processing print requests from remote systems running a more recent version of the LP Print Server. The request will be printed normally, however.

You can suppress these warnings by adding the following two lines to the section annotated as "adding simple options," in the printer interface program used by the printer issuing the warnings.

```
locale=*) ;;  
flist=*) ;;
```

An example of how to do this can be found in the `standard` interface program. (Printer interface programs are found in the `/usr/lib/lp/model` directory.)

Problem: Error Messages About Options That Can't Be Handled

You may try to use the `-o` option with the `lp(1)` command to modify printer output in a manner you expect the printer interface script will support. However, the print request fails with a message that indicates that the option can't be handled. It is possible that the `lp` administrator has not enabled that particular option for general use with the `lpadmin -o` option command. Refer to online manual page `lpadmin(1M)` for additional information.

Administering LP through OA&M Menus

The system administration menus are only available if the Operations, Administration and Maintenance (OA&M) package is installed on your system. To access the system administration menu for the LP Print Service, type **sysadm** and select the **printers** item from the main menu. The menu shown in Screen 12-1 will then appear.

```

2          Line Printer Services Configuration and Operation

classes    - Manage Classes of Related Printers
filters    - Manage Filters for Special Processing
forms      - Manage Pre-Printed Forms
operations - Perform Daily Printer Service Operations
printers   - Configure Printers for the Printer Service
priorities - Assign Print Queue Priorities to Users
requests   - Examine and Manipulate Print Requests
status     - Display Status of the Print Service

```

Screen 12-1. Main Menu for Print Service

NOTE

If, in addition to the LP Print Service you install the Commands Networking Extension package, the following entry will also appear in the above menu:

```

systems    - Configure Connections to Remote
Systems

```

The following table shows how the tasks listed on the **printers** menu correspond to the shell commands discussed throughout this chapter.

Task to Be Performed	sysadm Task	Shell Command
Group printers into classes	classes	lpadmin (1M)
Provide pre-processing software for files to be printed	filters	lpfilter (1M)
Define pre-printed forms for print requests	forms	lpforms (1M)
Control (turn on/off) queuing of requests; enable & disable printers; mount forms and fonts; start & stop Print Service; and report status of printers, classes, & forms	operations	accept & reject [see accept (1M)], enable & disable [see enable (1)], lpadmin (1M) , lpsched & lpshut [see lpsched (1M)], lpstat (1)
Configure printers for Print Service	printers	lpadmin (1M)

Task to Be Performed	sysadm Task	Shell Command
Define levels of priority available to users requesting print jobs	priorities	lpusers (1M)
Identify active printers, print wheels & character sets, mounted forms, and pending requests	status	lpstat (1)
Submit and cancel print requests	requests	lp & cancel [see lp (1)], lpmove [see lpsched (1M)]
Set up communication to remote Print Service	systems	lpssystem (1M)

Quick Reference to LP Print Service Administration

LP administrative commands are found in the `/usr/sbin` and `/usr/bin` directories. (If you expect to use them frequently, you might find it convenient to include these paths in your `PATH` variable.) To use these commands, you must be an LP administrator.

You'll also need to use commands designed for users as well as administrators, such as the commands for disabling and enabling a printer.

- Defining a device special file for a printer in the Device Database:

`putdev(1M)`

- Allocating a device special file for use:

`admalloc(1M)`

- Activating a printer:

`/usr/bin/enable`

See `enable (1M)`.

- Canceling a request for a file to be printed:

`/usr/bin/cancel`

See `lp (1)`.

- Sending a file (or files) to a printer:

`/usr/bin/lp`

See `lp (1)`.

- Reporting the status of the LP Print Service:

`/usr/bin/lpstat`

See `lpstat (1)`.

- Deactivating a specified printer(s):

/usr/bin/disable

See **enable (1M)** .

- Permitting job requests to be queued for a specific destination:

/usr/sbin/accept

See **accept (1M)** .

- Preventing jobs from being queued for a specified destination:

/usr/sbin/reject

See **accept (1M)** .

- Setting up or changing printer configurations:

/usr/sbin/lpadmin

See **lpadmin (1M)** .

- Setting up or changing filter definitions:

/usr/sbin/lpfilter

See **lpfilter (1M)** .

- Setting up or changing preprinted forms:

/usr/sbin/lpforms

See **lpforms (1M)** .

- Mounting a form:

/usr/sbin/lpadmin

See **lpadmin (1M)** .

- Moving output requests from one destination to another:

/usr/sbin/lpmove

See **lpmove (1M)** .

- Starting the LP Print Service scheduler:

/usr/lib/lp/lpsched

See **lpsched (1M)** .

- Stop the LP Print Service scheduler

/usr/sbin/lpshut

See **lpsched (1M)** .

- Setting or changing the default priority and priority limits that can be requested by users of the LP Print Service:

`/usr/sbin/lusers`

See `lpusers(1M)`.

Introduction	13-1
Configuring a Network Printer	13-2
Printer Server Setup	13-2
Configure the Server Printer	13-2
Advertise the LP Print Service	13-2
Define Attribute Mapping	13-3
Mapping Client User lp to Server lp	13-3
Mapping Client LIDs to Server LIDs	13-4
Attribute Mapping for the Connection Server	13-4
Configure the Client Communication Parameters	13-4
Printer Client Setup	13-4
Advertise the LP Print Service	13-4
Define Attribute Mapping	13-5
Configure Server Communication Parameters	13-5
Configure the Printer Client Attributes	13-5
Example of a Server Configuration	13-6
Examples of Client Configurations	13-6
Notify the Printer to Accept Jobs	13-6
Turn on the Printer (Logically)	13-6
Putting It All Together	13-7
Example 1: Configuring a Remote Printer	13-7
Example 2: Configuring a Remote PostScript Printer	13-7
Configuring a Dial-up Printer	13-7
Sharing Printers	13-8
Providing Forms	13-9
Defining a Form	13-10
Removing a Form	13-12
Restricting User Access	13-12
Alerting to Mount a Form	13-13
Mounting a Form or Print Wheel	13-15
Unmounting a Form or Print Wheel	13-15
Setting the Default Destination	13-16
Examining a Form	13-16
Providing Filters	13-17
What Is a Filter?	13-17
Task 1: Converting Files	13-17
Example 1	13-18
Example 2	13-18
Task 2: Handling Special Modes	13-18
Task 3: Detecting Printer Faults	13-19
Will Any Program Make a Good Filter?	13-19
Defining a Filter	13-20
Defining Options with Templates	13-23
Template Keywords	13-24
Example 1	13-26
Example 2	13-27
Example 3	13-27

- Command to Enter 13-28
- Removing a Filter 13-28
- Examining a Filter 13-28
- Restoring Factory Defaults 13-29
- A Word of Caution 13-29
- Managing the Printer Queue 13-30
 - Assign Print Queue Priorities to Users 13-30
 - Setting A User's Priority Limits. 13-30
 - Setting a Default Limit 13-30
 - Listing User's Priorities 13-31
 - Removing User's Priorities 13-31
 - Setting the System Priority Level 13-31
 - Cleaning Out the Request Log 13-31
- PostScript Printers 13-33
 - How to Use a PostScript Printer 13-34
 - Support of Non-PostScript Print Requests 13-34
 - Additional PostScript Capabilities Provided by Filters 13-35
 - The Administrator's Duties 13-36
 - Installing and Maintaining PostScript Printers 13-36
 - Installing and Maintaining PostScript Filters 13-37
 - Installing and Maintaining PostScript Fonts 13-38
 - Managing Printer-Resident Fonts 13-39
 - Installing and Maintaining Host-Resident Fonts 13-40
 - Where Are Fonts Stored? 13-41
 - Adding an Entry to the Map Table 13-41
 - Downloading Host-Resident Fonts 13-42
- Customizing the Print Service 13-43
 - Adjusting the Printer Port Characteristics 13-45
 - Adjusting the Terminfo Database 13-46
 - How to Write an Interface Program 13-48
 - What Does an Interface Program Do? 13-48
 - How Is an Interface Program Used? 13-49
 - Customizing the Interface Program 13-50
 - An Example: Adding a Printer with a Customized Interface 13-52
- Troubleshooting 13-53
 - Networking Problems 13-53
 - Jobs Backed Up in the Client Queue? 13-53
 - Jobs Backed Up in the Server Queue? 13-53
 - Conflicting Messages about the Acceptance of Jobs? 13-53
- Administering LP through OA&M Menus 13-54
- Quick Reference to LP Print Service Administration 13-55

Introduction

NOTE

The Commands Networking Extension package (package identifier `net_cmds`) from the Networking Set must be installed on your computer in order to operate printers on a network. The Network Support Utilities package (package identifier `nsu`) is part of the Foundation Set. Both these packages are prerequisites for the LP Networking Service.

Be sure to read the “Reportscheme Source” chapter in the *Network Administration* manual.

Ensure that the Listener Program is properly installed. (See the *System Administration Manual* for more information.)

When the LP Networking Service is installed and running on your system, you can offer your users access to printers not directly connected to their home computers. Within an LP network, users can access any printer on a computer to which their home machines are linked.

NOTE

If you install the LP Networking Service on a system with a B2-level security rating, that rating will be invalidated.

This can be advantageous for several reasons. For example, if only one of several printers in a network has a particular typesetter needed for some print jobs, users on other machines won't be able to use it unless it's available, as a remote printer, through a network. Another advantage you may appreciate as an administrator is the ability to connect all your printers to a single machine linked to others in a network. Users on any network machine (or “printer client”) can submit print requests to any printer on this machine, the “printer server.”

NOTE

Before you add a remote printer to your system, be sure communications between your system and the network have been set up properly and verified. See *Network Administration* for details. None of the following procedures will work unless intersystem communications have been set up properly.

This chapter covers the following topics:

- configuring a Print Service for the unique requirements of your users (such as the need for particular pre-printed forms and filters)
- support of PostScript printers, including downloading of Type 1 fonts
- writing customized filters and interface programs

If the Operations, Administration and Maintenance (OA&M) package (a non-graphical menu interface) is installed on your system you can use it to complete many of these tasks. (See “*Administering LP Through OA&M Menus*” later in this chapter.)

Configuring a Network Printer

This section describes how to configure machines as network servers and clients.

Printer Server Setup

The printer server is the computer to which the printer is directly connected.

Configure the Server Printer

The procedure for configuring a printer server is the same as that for any directly connected printer. See “*Configuring a Directly Connected Printer*” in “Basic Print Service” for instructions.

Advertise the LP Print Service

Because printer clients need to access printer servers, you need to configure the local port monitor for the network you share to accept service requests and to notify the LP Print Service of such requests. If your printer servers and clients are all running current operating systems, run the following command to advertise the Print Service:

```
pmadm -a -p netname -s lp -i root -v 'nlsadmin -V' \  
-m 'nlsadmin -o /var/spool/lp/fifos/listenS5'
```

where *netname* is the name of a network such as `tcp`.

If any of the printer clients that will be calling the printer server are BSD systems, you must also execute the following commands. First, to find out your system's TCP/print_service address, run the **-A** option with the **lpssystem** command, as follows:

```
lpssystem -A
```

Next, to configure your local port monitor, execute this command:

```
pmadm -a -p tcp -s lpd -i root -V `nlsadmin -V`  
-m `nlsadmin -o /var/spool/lp/fifos/listenBSD -A`xaddress`
```

Finally, because changes have been made to the port monitor's administrative files, the port monitor needs to be notified. The following command will do this:

```
sacadm -x -p netname
```

NOTE

This procedure may have been done automatically upon installation of the Networking Set. You can find out what services have been configured on your system by executing **pmadm -l** and look for **listen BSD** or **listen s5**.

Define Attribute Mapping

LP makes use of the attribute mapping features of the OS. Attribute mappings preserve the security of your system by providing a mechanism for preserving the identity of a user across a network; if an attribute mapping fails, the jobs requested will be kicked back by the server.

Setting up an attribute mapping arrangement can be a simple or complex task, depending on your needs. Here we offer an example of a simple transparent level ID (LID) mapping. If you want to set up a more complicated mapping arrangement, see the following on-line manual pages **attradmin(1M)**, **uidadmin(1)**, **idadmin(1M)**, **attrmap(3I)**, and **namemap(3I)**.

Mapping Client User lp to Server lp

LP makes use of the attribute mapping files of the **cr1** authentication schemes. Minimally, all client **lp** users must be mapped to the server **lp** user. This is done by the following procedure on your server machine:

```
idadmin -S cr1 -I "M1@M2" # user lp is mapped  
idadmin -S cr1 -a -r "lp@*" -l lp # to local user lp
```

Additional mappings of client users to server users are also supported. See the "Network Services" chapter in *Network Administration* for details about configuring the **cr1** authentication scheme.

Mapping Client LIDs to Server LIDs

LP provides for the mapping of a client's LIDs to semantically equivalent LIDs on the server. Minimally, all the system supplied LIDs must be mapped. The following is an example of how to set up transparent LID mapping.

```
attradmin -A LID -I "M1:M2"  
attradmin -A LID -a -r "[12345678]@" -l %1# map system LIDS to same  
attradmin -A LID -a -r "*" -l %1# transparent LID mapping.
```

See the "Network Services" chapter in *Network Administration* for more information about attribute mapping.

Attribute Mapping for the Connection Server

The attribute mapping for the connection server must be configured. (This may already have been done on your system by your network administrator.) The following example shows how to set up the connection server for transparent LID mapping.

```
attradmin -A LIDAUTH -I "M2:M1"  
attradmin -A LIDAUTH -a -r ":[12345678]" -l %1 # map system LIDS to same  
attradmin -A LIDAUTH -a -r "*" -l %1 # transparent LID mapping.
```

Configure the Client Communication Parameters

To configure the parameters of communication with your client machine, run this command on your printer server:

```
lpssystem client-name
```

Various options are available for this command; see the **lpssystem (1M)** on-line manual page for details.

Printer Client Setup

Now you're ready to set up the client machine.

Advertise the LP Print Service

Advertise the LP Print Service on the printer client machine that will send jobs to the server machine. On those client machines, run

```
pmadm -a -p netname -s lp -i root -v 'nlsadmin -V' \
-m 'nlsadmin -o /var/spool/lp/fifos/listenS5'
```

where *netname* is the name of a network such as `tcp`.

Define Attribute Mapping

The procedure for defining attribute mapping on a client machine is the same as that followed on a printer server machine. See “*Define Attribute Mapping*” under “*Printer Server Setup*” above for instructions.

Configure Server Communication Parameters

To configure the parameters of communication with your server machine, run this command on the client machine:

```
lpsystem server_name
```

Various options are available with this command; see the **lpsystem(1M)** on-line manual page for details.

Configure the Printer Client Attributes

Define the following three attributes of the client environment by running the **lpadmin** command with the options listed below.

- *printer-name*: Specify (with **-p**) the name used on the client machine to identify the server printer.
- *server-name!server_printer_name*: Specify (with **-s**)
 - *server_name*—the name of the system on which the printer resides (see details below this list)
 - *server_printer_name*—the name used on the server system to identify the server printer. For systems that support print queues, the desired *server_queue_name* must be used in place of the *server_printer_name*.
- *range*: If the Enhanced Security Utilities are installed and running on your system, specify (with **-R**) the valid range of device levels defined for the *client*. (Defining this attribute is optional.)

For *server_printer_name*, you can usually specify the same name used by the server to identify that printer. To avoid unnecessary complexity, this is recommended. However, if you already have a printer (or printer class) on your system with that name, choose a different name for the server. To assign a different name to a server printer, run the **lpadmin** command, specifying the local name for the printer (with the **-p** option) and the server’s name for the printer after the name of the server (with the **-s** option).

Example of a Server Configuration

For example, suppose you want your users to have access to a printer called `psjet2` that resides on a server called `newyork`. Because you already have a printer called `psjet2` on your own (client) system, you want to give the users of your system a different name by which to request the server printer: `psjet3`. Request the new name by entering the following:

```
lpadmin -p psjet3 -s newyork!psjet2
```

For details, see *Network Administration*.

Examples of Client Configurations

As an example of a client configuration you might set up, suppose you want to configure a printer (known locally as `r1`) as the remote PostScript printer `ps1` on a remote system called `bullwinkle`. Enter the following:

```
lpadmin -p r1 -s bullwinkle!ps1 -T PS -I simple,postscript,plot
```

(“-T PS” is optional; “-I simple,postscript,plot” is essential.)

You can also, for example, configure a printer known locally as `r1` as the remote printer `lp1` on `bullwinkle`, and restrict jobs for `r1` to dominate `USER_LOGIN`. To do this, enter the following:

```
lpadmin -p r1 -s bullwinkle!lp1 -R SYS_RANGE_MAX, \
USER_LOGIN
```

For details, see *Network Administration*.

Notify the Printer to Accept Jobs

Once the client machine has been configured, it will almost be ready to start forwarding print requests from users to the remote printer. Run the **accept** command so the client can start receiving jobs.

For example, suppose you want to start forwarding requests to a printer named `r1`. Enter

```
accept r1
```

Turn on the Printer (Logically)

Run the **enable** command so the client can start sending job requests to the printer.

To continue the example above (under “*Notify the Printer to Accept Jobs*”), enable the `r1` you just notified by entering

```
enable r1
```


Putting It All Together

Example 1: Configuring a Remote Printer

To configure a printer known locally as `r1` to be the remote printer `lp1` on a system called `iowa`, and restrict jobs for `r1` to dominate `USER_LOGIN`, enter

```
lpadmin -p r1 -s iowa!lp1
```

```
lpadmin -p r1 -s iowa!lp1 -R USER_LOGIN,SYS_RANGE_MAX
```

Example 2: Configuring a Remote PostScript Printer

To configure a printer known locally as `r1` to be the remote PostScript printer `ps1` on a system called `iowa`, enter

```
lpadmin -p r1 -s system-name!ps1 -T PS -I simple, \
postscript, troff
```

Here the `-T` option, followed by `PS`, is optional. The string

```
-I simple, postscript, troff
```

is essential.

Configuring a Dial-up Printer

Why would you want to use a printer that is not directly connected to your computer?

- The environment where a printer is located is so far from the computer that a direct connection is not possible or practical. For example, you might have one printer in use with a single terminal at a branch office located a few miles from your main site.
- You may want to share a printer with computers that are not on a common network.

The LP Print Service establishes a connection to the printer as necessary to print requests; at the end of each request the connection is dropped, making the printer available to the next machine that calls it. Thus the printer gets shared by the users of all the computers, more or less equally.

There are two methods for connecting printers not directly connected to your system: you can attach them directly to a network or you can reach them through a dial-up modem. The LP Print Service uses the Basic Networking Utilities (BNU) to handle both methods.

When a modem connection is used, the printer must be connected to a dialed modem, and the dial-out modem must be connected to the computer. Whether printers are connected to a modem or directly to a network, the connection must be described to the Basic Network-

ing Utilities. For instructions on describing either type of connection, see the “Network Services” chapter in *Network Administration*.

To make a printer connected in one of these ways available to your users, enter the following command:

```
lpadmin -p printer-name -U dial-info
```

Dial-info is either the telephone number to be dialed to reach the printer's modem, or the system name entered in the Basic Networking **Systems** file for the printer.

NOTE

The **-U** option provides a way to link a single printer to your Print Service. It does not allow you to connect with a Print Service on another system.

A note on printers connected to a modem or directly to a network: if the printer or port is busy, the LP Print Service will automatically retry later. This retry rate is 10 minutes if the printer is busy, and 20 minutes if the port is busy. These rates are not adjustable. However, you can force an immediate retry by issuing an **enable** command for the printer. If the port or printer is likely to be busy for an extended period, you should issue a **disable** command.

The **lpstat -p** command reports the reason for a failed dial attempt. Also, if you are alerted to a dialing fault (see the “*Printer Fault Alerting*” section in the “Basic Print Service” chapter), the alert message will give the reason for the fault. These messages are identical to the error messages produced by the Basic Networking Utilities (BNU) for similar problems.

In summary, to add printers to your system, run the **lpadmin** command, specifying either **-v** (for a directly connected printer) or **-U** (for a network printer).

Sharing Printers

If you have access to other systems via the Remote File Sharing Utilities (RFS), you may want to share printers with those systems by running the print service over RFS. You can do so by following these instructions.

On the server machine:

1. Set up the LP print service on the server machine as you would on any machine. (The server is the computer that does all the spooling.) Make sure that the printer works and that you are able to print text on it.
2. Share **/var/spool/lp**, **/etc/lp**, and **/var/lp** with all the *client(s)* that will be using this printer.
3. In **/etc/rfs/auth.info/uid.rules**, map the user ID (UID) of **lp** to itself, so the entry in **uid.rules** appears as follows: **map lp**.

On the client machines:

1. Make sure the scheduler is not running on the client machines. (Use the command `lpstat -r` to find out.) If it is running, shut it down with `lpshut`: it can cause print jobs to be lost.

On the client machines you need only `lp`, `lpstat`, and other LP print service commands.

2. Mount the resources that were shared by the server on the client's `/var/spool/lp`, `/etc/lp`, and `/var/lp`.

On client machines, the `-c` option of `lp` should be used for any user file not in a shared resource. This will force a copy of the file to be sent to the server machine. The LP print service cannot access local files that are not in a shared resource.

Providing Forms

A form is a sheet of paper, on which text or graphical displays have already been printed, that can be loaded into a local printer (that is, a printer on your system) for use in place of plain stock. Common examples of forms include company letterhead, special paper stock, invoices, blank checks, vouchers, receipts, and labels.

Typically, several copies of a blank form are loaded into a printer, either as a tray of single sheets or as a box of fan-folded paper. An application is used to generate data that will be printed on the form, thereby filling it out.

The LP Print Service helps you manage the use of preprinted forms, but does not provide your application any help in filling out a form; this is solely your application's responsibility. The LP Print Service, however, will keep track of which print requests need special forms mounted and which forms are currently mounted. It can alert you to the need to mount a new form.

This section tells you how you can manage the use of preprinted forms with the LP Print Service. You will see how you can

- define a new form
- change the Print Service's description of an existing form
- remove the Print Service's description of a form
- examine the Print Service's description of a form
- restrict user access to a form
- arrange alerting to the need to mount a form
- inform the Print Service that a form has been mounted

Defining a Form

When you want to provide a new form, the first thing you have to do is define its characteristics. To do so, enter information about each of the nine required characteristics (page length, page width, and so on) as input to the **lpforms** command (see below for details). The LP Print Service will use this information for two purposes: to initialize the printer so that printing is done properly on the form, and to send you reminders about how to handle that form. Before running the **lpforms** command, gather the following information about your new form:

Page length	The length of the form, or of each page in a multi-page form. This can be expressed as the number of lines, or the size in inches or centimeters.
Page width	The width of the form, expressed in characters, inches, or centimeters.
Number of pages	The number of pages in a multi-page form. The LP Print Service uses this number with a filter (if available) to restrict the alignment pattern to a length of one form. (See the description of alignment patterns below.) If no filter is available, the LP Print Service does not truncate the output.
Line pitch	A measurement that shows how closely together separate lines appear on the form. It can be expressed in either lines per inch or lines per centimeter.
Character pitch	A measurement that shows how closely together separate characters appear on the form. It can be expressed in either characters per inch or characters per centimeter.
Character set choice	The character set, print wheel, or font cartridge that should be used when this form is used. A user may choose a different character set for his or her own print request when using this form, or you can insist that only one character set be used.
Ribbon color	If the form should always be printed using a certain color ribbon, then the LP Print Service can remind you which color to use when you mount the form.
Comment	Any remarks that might help users understand what the form is, when it should be used, and so on.
Alignment pattern	A sample file that the LP Print Service uses to fill one blank form. When mounting the form, you can print this pattern on the form to align it properly. You can also define a content type for this pattern so that the printer service knows how to print it.

NOTE

The LP Print Service does not try to mask sensitive information in an alignment pattern. If you do not want sensitive information printed on sample forms—very likely the case when you align checks, for instance—then you should mask the appropriate data.

When you've gathered this information about the form, enter it as input to the **lpforms** command. You may want to record this information first in a separate file so you can edit it before entering it with **lpforms**. You can then use the file as input instead of typing each piece of information separately after a prompt. Whichever method you use, enter the information in the following format:

```

Page length: scaled-number
Page width: scaled-number
Number of pages: integer
Line pitch: scaled-number
Character pitch: scaled-number
Character set choice: character-set-name[,mandatory]
Ribbon color: ribbon-color
Comment:
informal notes about the form
Alignment pattern: [content-type]
alignment-pattern

```

Although these attributes are described in detail on the previous page, a few points should be emphasized here. First, the phrase [mandatory] is optional and, if present, means that the user cannot override the character set choice in the form. Second, *content-type* can be given optionally, with an alignment pattern. If this attribute is given, the Print Service uses it to determine, as necessary, how to filter and print the file.

With two exceptions, the information in the above list may appear in any order. The exceptions are the alignment pattern (which must always appear last) and the *comment* (which must always follow the line with the **Comment:** prompt). If the *comment* contains a line beginning with a key phrase (such as Page length, Page width, and so on), precede that line with a > character so the key phrase is hidden. Be aware, though, that any initial > will be stripped from the comment when it is displayed.

Not all of the information has to be given. When you don't specify values for the items listed below, the values shown beside them are assigned by default.

Item	Default
Page length	66 lines
Page width	80 columns
Number of pages	1
Line pitch	6 per inch
Character pitch	10 per inch
Character set choice	any
Ribbon color	any
Comment	(no default)
Alignment pattern	(no default)

To define the form, use one of the following commands

```
lpforms -f form-name -F file-name  
lpforms -f form-name -
```

where *file-name* is the full path for the file.

The first command gets the form definition from a file; the second command gets the form definition from you, through the standard input. A *form-name* can be anything you choose, as long as it contains a maximum of 14 alphanumeric characters and underscores.

If you need to change a form, just reenter one of the above commands. You need only provide information for items that must be changed; items for which you don't specify new information will stay the same.

Removing a Form

The LP Print Service imposes no fixed limit on the number of forms you may define. It is a good idea, however, to remove forms that are no longer appropriate. If you don't, users will see a long list of obsolete forms when choosing a form, and may be confused. In addition, because the LP Print Service must occasionally look through all the forms listed before performing certain tasks, the failure to remove obsolete forms may require extra, unnecessary processing by the Print Service.

To remove a form, enter the following command:

```
lpforms -f form-name -x
```

Restricting User Access

If your system has a form that you don't want to make available to everyone, you can limit its availability to selected users. For example, you may want to limit access to checks to the people in the payroll department or accounts payable department.

The LP Print Service restricts the availability of a form by using the lists (provided by you) of users allowed or denied access to that form. If a user is not allowed to use a particular form, the LP Print Service will reject his or her request to print a file with it.

The method used to allow or deny users access to a form is similar to the method used to allow or deny users access to the **cron** and **at** facilities. (Refer to the **crontab(1)** on-line manual page.) The rules are listed under "*User Access Restrictions*" in the "Basic Print Service" chapter.

If users on your system are to be able to access forms on a remote printer, it's necessary for all the users included on the allow list for the local system to be included on the allow list for the remote system, as well.

If, on the other hand, a local user is to be denied permission to use forms on a remote printer, it's not necessary for the deny lists in both the local and remote Print Services to include that user. By being included in only one of these deny lists, a user can be denied access to remote forms. As a courtesy to your users, however, it's a good idea to make sure

that any local users who are included in a deny list on a remote system are included in the corresponding deny list on your local system. By doing this you can make sure that whenever a user on your system requests a form that he or she is not authorized to use, he or she is immediately informed that permission to use the form is being denied. If the local Print Service does not “know” that a user is denied permission to use a particular remote form, there will be a delay before the user receives a “permission denied” message from the remote system.

You can add names of users to either list, using one of the following commands:

```
lpforms -f form-name -u allow:user-list  
lpforms -f form-name -u deny:user-list
```

The *user-list* is a comma or space separated list of names of users. If you use spaces to separate the names, enclose the entire list (including the **allow:** or **deny:** but not the **-u**) in quotes. Each item in the list can include a system name, as shown under “*User Access Restrictions*” in the “Basic Print Service” chapter. The first command adds the names to the allow list and removes them from the deny list. The second command adds the names to the deny list and removes them from the allow list. Specifying **allow:all** will allow everybody; specifying **deny:all** will deny everybody.

If you do not add user names to the allow or deny lists, the LP Print Service will assume that everybody may use the form.

NOTE

The access restrictions imposed by the security level range of the printer override any permissions granted a user by this command. If users are listed on the **allow** list for a form but their security levels are not within the security level range of the printer used for the forms, they will not be able to print jobs.

Alerting to Mount a Form

If you define more forms than printers, you will obviously not be able to print files on all the forms simultaneously. This means that some print requests may be held in a queue until you mount the forms they need. How will you know when to mount a particular form? One method would be to monitor, periodically, the number of print requests pending for that form. The LP Print Service, however, provides an easier way: You can ask to be alerted when the number of requests waiting for a form has exceeded a specified threshold.

You can choose one of several ways to receive an alert. See “*Alerting to Mount a Print Wheel*” earlier in this chapter for a description of alert methods.

NOTE

If you elect to receive no alerts, you are responsible for checking to see whether any print requests haven’t been printed because the proper form isn’t mounted.

In addition to the method of alerting, you can also set the number of requests that must be queued before you are alerted, and you can arrange for repeated alerts every few minutes until the form is mounted. You can choose the rate of repeated alerts, or choose to receive only one alert for each form.

To arrange for alerting to the need to mount a form, enter one of the following commands:

```
lpforms -f form-name -A mail -Q requests -W minutes  
lpforms -f form-name -A write -Q requests -W minutes  
lpforms -f form-name -A 'command' -Q requests -W minutes
```

The first two commands direct the LP Print Service to send you a mail message or write the message directly to your terminal, respectively, for each alert. The third command directs the LP Print Service to run the *command* for each alert. The shell environment in effect when you enter the third command is saved and restored for the execution of *command*; this includes the environment variables, user and group IDs, and the current directory.

In each command line, the argument *requests* is the number of requests that need to be waiting for the form to trigger the alert, and the argument *minutes* is the number of minutes between repeated alerts.

NOTE

If you want mail sent or a message written to another user for an alert, use the third command with the option **-A 'mail login-ID'** or **-A 'write login-ID'**. If you do not specify a login-ID, the mail or message will be sent to your current login. This may not be your login if you have used the **su** command to change it.

If you want the Print Service to issue no alert when the form needs to be mounted, execute the following command:

```
lpforms -f form-name -A none
```

When you start receiving repeated alerts, you can direct the LP Print Service to stop sending you alerts (for the current case only) by issuing the following command:

```
lpforms -f form-name -A quiet
```

NOTE

Use the alert type *quiet* only to terminate an active alert; do not specify *quiet* as the alert type for a new printer.

Once the form has been mounted and unmounted again, alerts will resume if too many requests are waiting. Alerts will also start again if the number of requests waiting falls below the **-Q** threshold and then rises up to the **-Q** threshold again. This happens when waiting requests are canceled, and when the type of alerting is changed.

If *form-name* is any, the alerting condition defined in this command applies to any form for which an alert has not yet been defined. If *form-name* is all, the alerting defined in this command applies to all forms.

If you don't define an alert method for a form, you will not receive an alert to mount it. If you define a method without the **-W** option, you will be alerted once for each occasion.

Mounting a Form or Print Wheel

Before LP starts printing files that need a pre-printed form or print wheel, you have to mount it on a printer. If alerting has been set on the form or print wheel, you will be alerted when enough print requests are queued waiting for it to be mounted (see previous section on "alerting"). Mounting a form or print wheel involves first loading it onto the printer and then telling LP that it is mounted. It is sound practice to disable the printer first.

Enter the following to inform LP that the form or wheel is mounted:

```
/usr/sbin/lpadmin -p printer-name -M -S print-wheel-name\  
-f form-name -a -o filebreak
```

Omit **-S print-wheel-name** if you are mounting just a form, or omit **-f form-name -a -o filebreak** if you are mounting just a print wheel.

If an alignment pattern has been registered with the form, you can ask that this be repeatedly printed after you've mounted the form, until you have adjusted the printer so that the alignment pattern looks correct.

The **-o filebreak** option tells LP print service to add a "form-feed" after each copy of the alignment pattern, if there is one. You will be asked to press the <CR> key before each copy of the alignment pattern is printed.

The actual control sequence used for the "form-feed" depends on the printer involved and is obtained from the Terminfo database. If the alignment pattern already includes a form-feed, leave out the **-o filebreak** option.

Until you've mounted a form (or printwheel) on a printer, only print requests that don't require the form (or printwheel) will be sent to it.

Unmounting a Form or Print Wheel

Selecting **unmount** allows you to specify that a form or font on a specified printer is to be unmounted.

If you want to unmount a form or print wheel use the following command:

```
/usr/sbin/lpadmin -p printer_name -M -S none -f none
```

Omit "**-S none**" if you just want to unmount a form. Omit "**-f none**" if you just want to unmount a print wheel.

Setting the Default Destination

You can specify the destination of print requests when a destination is not specifically given on the command line. The printer or class must already exist. Use the **lpstat -t** command to list your printers.

Make a printer or class the default destination by entering the following command:

```
/usr/sbin/lpadmin -d printer-or-class-name
```

If you later decide that there should be no default destination, enter a null *printer-or-class-name* as in the following command:

```
/usr/sbin/lpadmin -d
```

If you don't set a default destination, there will be none. Users will have to explicitly name a printer or class in each print request, or will have to set the LPDEST shell variable with the name of a destination.

Examining a Form

Once you've defined a form to the LP Print Service, you can examine it with one of two commands, depending on the type of information you want to check. The **lpforms** command displays the attributes of the form. (The display produced by **lpforms** can be used as input; you may want to save it in a file for future reference.) The **lpstat** command displays the current status of the form.

Enter one of the following commands to examine a defined form.

```
lpforms -f form-name -l  
lpforms -f form-name -l > file-name  
lpstat -f form-name  
lpstat -f form-name -l
```

The first two commands present the definition of the form; the second command captures this definition in a file, which can be used later to redefine the form if you inadvertently remove the form from the LP Print Service. The last two commands present the status of the form, with the second of the two giving a long form of output, similar to the output of **lpforms -l**:

```
Page length: scaled-number  
Page width: scaled-number  
Number of pages: integer  
Line pitch: scaled-number  
Character pitch: scaled-number  
Character set choice: character-set[,mandatory]  
Ribbon color: ribbon-color  
Comment:  
comment  
Alignment pattern: [content-type]  
content
```

To protect potentially sensitive content, the alignment pattern is not shown if the `lpstat` command is used.

Providing Filters

This section explains how you can manage the use of filters with the LP Print Service. You will see how you can

- define a new filter
- change a filter
- remove a filter
- examine a filter

The “*Customizing the Print Service*” section at the end of this chapter describes how you can write a filter. First, let’s see what a filter is and how the LP Print Service can use one.

What Is a Filter?

A filter is a program that you can use for any of three purposes:

- To convert a user’s file from one data format to another so that it can be printed properly on a given printer
- To handle the special modes of printing that users may request with the `-y` option to the `lp` command (such as two-sided printing, landscape printing, draft or letter quality printing)
- To detect printer faults and notify the LP Print Service of them, so that the Print Service can alert you

Not every filter can perform all three tasks. Given the printer-specific nature of these three roles, the LP Print Service has been designed so that these roles can be implemented separately. This separation allows you or a printer manufacturer (or another source) to provide filters without having to change the LP Print Service.

A default filter is provided with the LP Print Service to provide simple printer fault detection; it does not convert files or handle any of the special modes. It may, however, be adequate for your needs.

Let’s examine the three tasks performed by filters more closely.

Task 1: Converting Files

For each printer (local or remote) you can specify what file content types it can print. When a user submits a file to print on any printer, and specifies its content type, the Print Service will find a printer that can handle files of that content type. Because many applica-

tions can generate files for various printers, this is often sufficient. However, some applications may generate files that cannot be printed on your printers.

By defining and creating a filter that converts such files into a type that your printers can handle, you can begin to support more applications in the LP Print Service. (The LP Print Service comes with a few filters for converting various types of files into PostScript.) For each filter you add to the system, you must specify one or more types of input it can accept and the type of output it can produce (usually only one).

When a user specifies (by executing `lp -T`) a file content type that no printer can handle, the Print Service tries to find a filter that can convert the file into an acceptable type. If the file to be printed is passed through a filter, the Print Service will then match the output type of that filter with a printer type or the input type of another filter. The LP Print Service will continue to match output types to input types in this way, thus passing a file through a series of filters, until the file reaches a printer that accepts it.

Below are some examples.

Example 1

The user Chris has run a spreadsheet program and has generated a file containing a copy of a spreadsheet. Chris now wants to print this file using the LP Print Service. You have only HP DeskJet 500 printers on your system. Fortunately, the spreadsheet application understands how to generate output for several printers, and Chris knows it's necessary to request output that can be handled by the HP DeskJet 500. When Chris submits the file for printing, the LP Print Service queues it for one of the printers; no filter is needed.

Example 2

A user named Marty has created a graphic image that can be displayed on a Tektronix 4014 terminal. Marty now wants to print this image, but all of the printers are PostScript printers. Fortunately, your system provides a filter called `posttek` that converts Tektronix type files to PostScript. Because you set the printer type to PostScript, the LP Print Service recognizes that it can use the `posttek` filter to convert Marty's output before printing it.

Task 2: Handling Special Modes

Another important role that filters can perform is the handling of special printing modes. Each filter you add to the filter table can be registered to handle special modes and other aspects of printing:

- special modes
- printer type
- character pitch
- line pitch
- page length
- page width
- pages to print
- character set
- form name
- number of copies

A filter is required to handle the special modes and printing of specific pages; the LP Print Service provides a default handling for all the rest. However, it may be more efficient to have a filter handle some of the rest, or it may be that a filter has to know several of these aspects to fulfill its other roles properly. A filter may need to know, for example, the page size and the print spacing if it is going to break up the pages in a file to fit on printed pages. As another example, some printers can handle multiple copies more efficiently than the LP Print Service, so a filter that can control the printer can use the information about the number of copies to skip the LP Print Service's default handling of multiple copies.

We'll see below how you can register special printing modes and other aspects of printing with each filter.

Task 3: Detecting Printer Faults

Just as converting a file and handling special printing modes is a printer-specific role, so is the detecting of printer faults. The LP Print Service attempts to detect faults in general, and for most printers it can do so properly. The range of faults that the Print Service can detect by itself, however, is limited. It can check for hang-ups (loss of carrier, the signal that indicates the printer is on-line) and excessive delays in printing (receipt of an XOFF flow-control character to shut off the data flow, with no matching XON to turn the flow back on). However, the Print Service can't determine the cause of a fault, so it can't tell you what to look for.

A properly designed filter can provide better fault coverage. Some printers are able to send a message to the host describing the reason for a fault. Others indicate a fault by using signals other than the dropping of a carrier or the shutting off of data flow. A filter can serve you by detecting more faults and providing more information about them than you would otherwise receive.

Another service a filter can provide is to wait for a printer fault to clear and then to resume printing. This service allows for more efficient printing when a fault occurs because the print request that was interrupted does not have to be reprinted in its entirety. Only a real filter, which has knowledge of the control sequences used by a printer, can "know" where a file breaks into pages; thus only such a filter can find the place in the file where printing should resume.

The LP Print Service has a simple interface that allows a filter to send you fault information and to restart printing if it can. The alerting mechanism (see the "*Printer Fault Alerting*" section under "*Configuring Your Printers*" in this chapter) is handled by the LP Print Service; the interface program that manages the filter takes all error messages from the filter and places them in an alert message that can be sent to you. Thus you'll see any fault descriptions generated by the filter. If you've set the printer configuration so that printing should automatically resume after a fault is cleared, the interface program will keep the filter active, so that printing can pick up where it left off.

Will Any Program Make a Good Filter?

It is tempting to use a program such as **troff**, **nroff**, or a similar word-processing program as a filter. However, the **troff** and **nroff** programs have a feature that allows references to be made in a source file to other files, known as "include files." The LP Print Service does not recognize include files; it will not enqueue any that are referenced by a source file when that file is in a queue to be printed. As a result, the **troff** or **nroff** pro-

gram, unable to access the include files, may fail. Other programs may have similar features that limit their use as filters.

Here are a few guidelines for evaluating a program for use as a filter:

- Only programs capable of reading data from standard input and writing data to standard output may be used as filters.
- Examine the kinds of files users will submit for printing that will require processing by the program. If they stand alone (that is, if they do not reference other files that the program will need), the program is probably okay.

Check also to see if the program expects any files other than those submitted by a user for printing. If it does, those files must be in the directory of the person using the filter, or they must be readable by all users authorized to use the filter. The latter prerequisite is necessary because filters are run with the user ID and group ID of the user who submitted the print request.

- If referenced files are permitted in the files submitted for printing, or if the program will need files other than those submitted by a user, then the program, unable to access the additional files, is likely to fail. We suggest you don't use the program under consideration as a filter; instead, have users run the program before submitting files for printing.

Referenced files that are always specified by full pathnames *may* be okay, but only if the filter is used for local print requests. When used on requests submitted from a remote machine for printing on your machine, the filter may still fail if the referenced files exist only on the remote machine.

Defining a Filter

When adding a new filter, the first thing you must do is to define the characteristics of its use. To do this, issue the **lpfilter** command with arguments that specify the values of the following filter characteristics:

- the name of the filter (that is, a command name)
- the types of input it will accept
- the types of output it will produce
- the types of printers to which it will be able to send jobs
- the names of specific printers to which it will send jobs
- the “type” of the filter (whether it's a `fast` filter or a `slow` filter)
- options

Each of these characteristics is described below.

Command: This is the full path of the filter program. If there are any fixed options that the program always needs, include them here.

Input types: This is the list of file content types that the filter can process. The LP Print Service doesn't impose a limit on the number of input types that can be accepted by a filter, but most filters can take only one. Several file types may be similar enough so that the filter can deal with them. You can use whatever names you like here, using a maximum of 14 alphanumeric characters and dashes (not underscores). Because the LP Print Service uses these names to match a filter with a file type, you should follow a consistent naming convention. For example, if more than one filter can accept the same input type, use the same name for that input type when you specify it for each filter. These names should be advertised to your users so they know how to identify the type of a file when submitting that file for printing.

Output types: This is the list of file types that the filter can produce as output. For each input type the filter will produce a single output type, of course; the output type may vary, however, from job to job. The names of the output types are also restricted to 14 alphanumeric characters and dashes.

These names should either match the types of printers you have on your system, or match the input types handled by other filters. The LP Print Service groups filters together in a shell pipeline if it finds that several passes by different filters are needed to convert a file. It's unlikely that you will need this level of sophistication, but the LP Print Service allows it. Try to find a set of filters that takes (as input types) all the different files your users may want printed, and converts those files directly into types your printers can handle.

Printer types: This is a list of printer types into which the filter can convert files. For most filters this list will be identical to the list of output types, but it can be different.

For example, you may have a printer that is given a single type for purposes of initialization (see the "*Printer Types*" section under "*Other Printer Configuration Options*" in the "Basic Printer Service" chapter), but which can recognize several different types of files. In essence this printer has an internal filter that converts the various types into one with which it can deal. Thus, a filter may produce one of several output types that match the "file types" that the printer can handle. The filter should be marked as working with that printer type.

As another example, you may have two different models of printers that are listed as accepting the same types of files. However, due to slight differences in manufacture, one printer deviates in the results it produces. You label the printers as being of different printer types, say A and B, where B is the one that deviates. You create a filter that adjusts files to account for the deviation produced by printers of type B. Because this filter is needed only for those printer types, you would list it as working only on type B printers.

For most printers and filters you can leave this part of the filter definition blank.

Printers:

You may have some printers that, although they're of the correct type for a filter, are in other ways not adequate for the output that the filter will produce. For instance, you may want to dedicate one printer for fast turn-around; only files that the printer can handle without filtering will be sent to that printer. Other printers, of identical type, you allow to be used for files that may need extensive filtering before they can be printed. In this case, you would label the filter as working with only the latter group of printers.

In most cases a filter should be able to work with all printers that accept its output, so you can usually skip this part of the filter definition.

Filter type:

The LP Print Service recognizes "fast" filters and "slow" filters. Fast filters are labeled "fast" because they incur little overhead in preparing a file for printing, and because they must have access to the printer when they run. A filter that is to detect printer faults has to be a fast filter. A filter that uses the `PRINTER` keyword as a filter option must be installed as a fast filter.

Slow filters are filters that incur a lot of overhead in preparing a file and that don't require access to a printer. The LP Print Service runs slow filters in the background, without tying up a printer. This allows files that don't need slow filtering to move ahead; printers will not be left idle while a slow filter works on a file if other files can be printed simultaneously.

Slow filters that are invoked by modes (via the `-y` option), must be run on the computer where the print request was issued. The LP Print Service can't pass values for modes to server machines. It can, however, match a file content type (specified after the `-T` option of the `lp` command) to a content type on a server machine. Therefore, if you want to activate special modes on a server machine, you must do so by specifying content types that will allow the LP Print Service to match input types and output types.

Options:

Options specify how different types of information should be transformed into command line arguments to the filter command. This information may include specifications from a user (with the print request), the printer definition, and the specifications implemented by any filters used to process the request.

There are thirteen sources of information, each of which is represented by a "keyword." Each option is defined in a "template," a statement in the following format:

keyword-pattern=replacement

This type of statement is interpreted by the LP Print Service to mean "When the information referred to by *keyword* has the value matched by *pattern*, take the *replacement* string, replace any

asterisks it contains with the *pattern* specified or expand any regular expressions it contains, and append the result to the command line.”

The options specified in a filter definition may include none, all, or any subset of these thirteen keywords. In addition, a single keyword may be defined more than once, if multiple definitions are required for a complete filter definition. (See “*Defining Options with Templates*” below.)

When you’ve gathered enough information to define the above characteristics of your filter, you are ready to run the **lpfilter** command, using your data as arguments. Because there are so many arguments, and because some of them may need to be entered more than once (with different values), we recommend you record this information first in a separate file and edit it, if necessary. You can then use the file as input to the **lpfilter** command and avoid typing each piece of information separately.

Whether you store the information in a file or enter it directly on the command line, use the following format:

```
Command: command-pathname [options]
Input types: input-type-list
Output types: output-type-list
Printer types: printer-type-list
Printers: printer-list
Filter type: fast or slow
Options: template-list
```

The information can appear in any order. Not all the information has to be given. When you do not specify values for the items listed below, the values shown beside them are assigned by default.

Item	Default
Command:	(no default)
Input types:	any
Output types:	any
Printer types:	any
Printers:	any
Filter type:	slow
Options:	(no default)

As you can see, the default values define a very flexible filter, so you probably have to supply at least the input and output type(s). When you enter a list, you can separate the items in it with blanks or commas, unless it is a *templates-list*; items in a *templates-list* must be separated by commas.

Defining Options with Templates

A template is a statement in a filter definition that defines an option to be passed to the filter command based on the value of one of the characteristics of the filter. A filter definition may include more than one template. Multiple templates may be entered on a single line

and separated with commas, or they may be entered on separate lines, preceded by the **Options:** prefix.

The format of a template is as follows:

keyword-pattern=replacement

The *keyword* identifies the type of option being registered for a particular characteristic of the filter.

Let's look at an example of how an option is defined for a particular filter. Suppose you want to have the Print Service scheduler assign print requests to filters on the basis of the following criteria:

- If the type of OUTPUT to be produced by the filter is *impress*, then pass the **-I** option to the filter.
- If the type of OUTPUT to be produced by the filter is *postscript*, then pass the **-P** option to the filter.

To specify these criteria, provide the following templates as options to the **lpfilter** command.

Options: OUTPUT impress=-I, OUTPUT postscript=-P

NOTE

If the **Options:** line becomes too long, put each template on a separate line, as follows:

Options: OUTPUT impress=-I
Options: OUTPUT postscript=-P

In both templates, the keyword is defined as OUTPUT. In the first template, the value of *pattern* is *impress* and the value of the *replacement* is **-I**. In the second template, the value of *pattern* is *postscript* and the value of the *replacement* is **-P**.

Template Keywords

The following thirteen *keywords* are available for defining **Options** in a filter definition:

Characteristic	<i>keyword</i>	Possible <i>patterns</i>	Example
Content type (input)	INPUT	<i>content-type</i>	troff
Content type (output)	OUTPUT	<i>content-type</i>	postscript
Printer type	TERM	<i>printer-type</i>	att495
Printer name	PRINTER	<i>printer-name</i>	lp1
Character pitch	CPI	<i>scaled-decimal</i>	10
Line pitch	LPI	<i>scaled-decimal</i>	6

Characteristic	<i>keyword</i>	Possible <i>patterns</i>	Example
Page length	LENGTH	<i>scaled-decimal</i>	66
Page width	WIDTH	<i>scaled-decimal</i>	80
Pages to print	PAGES	<i>page-list</i>	1-5, 13-20
Character set	CHARSET	<i>character-set</i>	finnish
Form name	FORM	<i>form-name</i>	invoice2
Number of copies	COPIES	<i>integer</i>	3
Special modes	MODES	<i>mode</i>	landscape

To find out which values to supply for each type of template (that is, for the *pattern* and *replacement* arguments for each *keyword*), see the source of information listed below.

- The values for the INPUT and OUTPUT templates come from the file type that needs to be converted by the filter and the output type that has to be produced by the filter, respectively. They'll each be a type registered with the filter.
- The value for the TERM template is the printer type.
- The value for the PRINTER template is the name of the printer that will be used to print the final output.
- The values for the CPI, LPI, LENGTH, and WIDTH templates come from the user's request, the form being used, or the default values for the printer.
- The value for the PAGES template is a list of pages that should be printed. Typically, it is a comma separated list of page ranges, each of which consists of a dash separated pair of numbers or a single number (such as 1-5,6,8,10 for pages 1 through 5, 6, 8, and 10). However, whatever value was given in the **-P** option to a print request is passed unchanged.
- The value for the CHARSET template is the name of the character set to be used.
- The value for the FORM template is the name of the form requested by the **-f** option of the **lp** command.
- The value of the COPIES template is the number of copies that should be made of the file. If the filter uses this template, the LP Print Service will reduce to 1 the number of copies of the filtered file *it* will have printed, since this "single copy" will really be the multiple copies produced by the filter.
- The value of the MODES template comes from the **-y** option of the **lp** command (the command used to submit a print request). Because a user can specify several **-y** options, there may be several values for the MODES template. The values will be applied in the left-to-right order given by the user.

The *replacement* part of a template shows how the value of a template should be given to the filter program. It is typically a literal option, sometimes with the place-holder ***** included to show where the value goes. The *pattern* and *replacement* can also use the regular expression syntax of **ed(1)** for more complex conversion of user input options into

filter options. All of the regular expression syntax of **ed(1)** is supported, including the `\(. . . \)` and `\n` constructions, which can be used to extract portions of the *pattern* for copying into the *replacement*, and the `&`, which can be used to copy the entire *pattern* into the *replacement*.

NOTE

If a comma or an equals sign (=) is included in a *pattern* or a *replacement*, escape its special meaning by preceding it with a backslash (`\`). Note that some regular expressions include commas that will have to be escaped this way. A backslash in front of any of these characters is removed when the *pattern* or *replacement* is used.

The following examples show how this works.

Example 1

Suppose you already added a filter called `col` with the following definition:

```
Input types:      N37, Nlp, simple
Output types:    simple
Command:         /usr/bin/col
Options:         TERM 450 = -b, MODES expand = -x
Options:         INPUT simple = -p -f
```

NOTE

If you provide more than one definition (that is, more than one line) for any filter characteristic other than `OPTIONS`, only the second definition will be used by the Print Service.

After you have “registered” this definition with the Print Service by entering it as input with the `lpfilter` command, users' print requests will be handled as follows:

- If a user enters the command

```
lp -y expand report.dec10
```

the filter command will be run with the following arguments:

```
/usr/bin/col -x -p -f
```

- If a user enters the command

```
lp -T N37 -y expand report.dec10
```

the filter command will be run with the following arguments:

```
/usr/bin/col -x
```

Qualifier: The default printer is not of type 450.

- If a user enters the command

```
lp -y expand -T 450 report.dec10
```

the filter command will be run with the following arguments:

```
/usr/bin/col -b -x
```

Example 2

The filter program is called `/usr/lib/lp/postscript/dpost`. It takes one input type, `troff`, produces an output type called `postscript`, and works with any printer of type `PS` (for PostScript). You've decided that your users need give just the abbreviations `port` and `land` when they ask for the paper orientation to be portrait mode and landscape mode, respectively. Because these options are not intrinsic to the LP Print Service, users must specify them using the `-y` option to the `lp` command.

The filter definition would look like this:

```
Input types: troff
Output types: postscript
Printer types: PS
Filter type: slow
Command: /usr/lib/lp/postscript/dpost
Options: LENGTH * = -l*
Options: MODES portrait = -op, MODES landscape = -ol
```

A user submitting a file of type `troff` for printing on a PostScript printer (type `PS`), with requests for landscape orientation and a page length of 60 lines, would enter the following command:

```
lp -T troff -o length=60 -y landscape -d any
```

Then this filter would be invoked by the LP Print Service to convert the file as follows:

```
/usr/lib/lp/postscript/dpost -l60 -ol -p1
```

Example 3

You add the following option template to the previous example:

```
Options: MODES group\=\([1-9]\) = -n\1
```

This template is used to convert a MODES option of the form

```
-y group=number
```

into filter options

`-nnumber`

So if a user gives the following command

```
lp -y group=4
```

the `dpst` command would include the following options:

`-n4`

For additional examples, run the command

```
lpfilter -f filter -l
```

where *filter* is the name of the factory installed PostScript filters. (For a list of PostScript filters, see “*PostScript Printers*” later in this chapter.)

Command to Enter

Once a filter definition is complete, enter one of the following commands to add the filter to the system.

```
lpfilter -f filter-name -F file-name  
lpfilter -f filter-name -
```

The first command gets the filter definition from a file, and the second command gets the filter definition from the standard input. A *filter-name* can be any string you choose, with a maximum of 14 alphanumeric characters and underscores.

If you need to change a filter, just reenter one of the same commands. You need only provide information for those items that must be changed; items for which you don't specify new information will stay the same.

Removing a Filter

The LP Print Service imposes no fixed limit on the number of filters you can define. It is a good idea, however, to remove filters no longer applicable, to avoid extra processing by the LP Print Service which must examine all filters to find one that works in a given situation.

To remove a filter, enter the following command:

```
lpfilter -f filter-name -x
```

Examining a Filter

Once you've added a filter definition to the LP Print Service, you can examine it by running the `lpfilter` command. The output of this command is the filter definition dis-

played in a format that makes it suitable as input. You may want to save this output in a file that you can use later to redefine the filter if you inadvertently remove the filter from the LP Print Service.

To examine a defined filter, enter one of the following commands:

```
lpfilter -f filter-name -l
lpfilter -f filter-name -l >file-name
```

The first command presents the definition of the filter on your screen; the second command captures this definition in a file for future reference.

Restoring Factory Defaults

The software is shipped from the factory with a default set of filters. As you add, change, or delete filters, you may overwrite or remove some of these original filters.

NOTE

All the filters delivered with LP are PostScript filters. Because B2 level security cannot be maintained on a PostScript printer, use of these filters will invalidate a B2 rating.

If, after changing them, you want to restore some or all of them, enter the following command:

```
lpfilter -f filter-name -i
```

Replace *filter-name* with the name of the filter to restore, or the word `all` to restore all the default filters.

A Word of Caution

Adding, changing, or deleting filters can cause print requests still queued to be canceled. This is because the LP Print Service evaluates all print requests still queued, to see which are affected by the filter change. Requests that are no longer printable, because a filter has been removed or changed, are canceled (with notifications sent to the people who submitted them). There can also be delays in the responses to new or changed print requests when filters are changed, due to the many characteristics that must be evaluated for each print request still queued. These delays can become noticeable if there is a large number of requests that need to be filtered.

Because of this possible impact, you may want to make changes to filters during periods when the LP Print Service is not being used much.

Managing the Printer Queue

Assign Print Queue Priorities to Users

LP provides a simple priority mechanism that people can use to adjust the position of a print request in the queue. Each print request can be given a priority level (a number from 0 to 39, where 0 is the highest), by the person who submits it. In this way, if you decide that your print request is of too low priority, you can set a higher priority (lower value) when you submit the file for printing.

You can define the following characteristics of this scheme:

- Each user can be assigned a personal priority limit. A user cannot submit a print request with a priority higher than the limit.
- A default priority limit can be assigned for users not assigned a personal limit.
- A default priority can be set for print requests to which the user does not assign a priority.

You may find that you want a critical print request to print ahead of any others, perhaps even if it has to preempt the currently printing request. You can have LP give “immediate” handling to a print request and put another print request on “hold.”

Setting A User’s Priority Limits

To set the priority limit for a user with the following command:

```
/usr/sbin/lpusers -q priority-level -u user-name
```

Set the limit for a group of users by listing their names after the **-u** option. Separate multiple names with a comma or space. If you use a space to separate users, enclose the whole list in quotes. To modify a user's priority limit, re-enter the **lpusers** command with a new limit.

Setting a Default Limit

To set the default limit, which applies to those users who have not been given a specific limit, type the following command:

```
/usr/sbin/lpusers -q priority-level
```

The *priority-level* is a number from 0 to 39, with the lower numbers having higher priority. If you do not set a default priority, LP will use the default of 20.

Listing User's Priorities

You can examine all the settings you have assigned for priority limits and defaults by using the following command:

```
/usr/sbin/lpusers -l
```

Removing User's Priorities

To restore a user's priority limit to the default value, enter:

```
/usr/sbin/lpusers -u user-name
```

Setting the System Priority Level

You can set the default priority that should be assigned to those print requests submitted without a priority. Use the following command:

```
/usr/sbin/lpusers -d priority-level
```

This default is applied when a user doesn't have an individual priority level. If the default priority is greater than the limit for a user, the limit is used instead.

Cleaning Out the Request Log

The directories `/var/spool/lp/tmp` and `var/spool/lp/requests` contain files that describe each request that has been submitted to LP print service. Each request has two files, one in each directory, that contain information about the request. The information is split to put more sensitive information in the `/var/spool/lp/requests` directory where it can be kept secure. The request file in the `/var/spool/lp/tmp` is safe from all except the user who submitted the request, while the file in `/var/spool/lp/requests` is safe from even the submitting user.

These files remain in their directories only as long as the request is on the queue. Once the request is finished, the information in the files is combined and appended to the file `/var/lp/logs/requests`. This file is not removed by LP print service but can be cleaned out periodically, using, for instance, the `crontab` facility. (Refer to the `crontab (1)` on-line manual page.)

The request log has a simple structure that makes it easy to extract data using common UNIX shell commands. The requests are listed in the order in which they were printed and are separated by lines that give the request ID. Each line below the separator line is marked with a single letter that identifies the kind of information contained in the line. Each letter is separated from the data by a single space. See the following table for details.

Letter	Content of Line
=	This is the separator line, containing the request ID, the user and group IDs of the user, the total number of bytes in the original (unfiltered) files, and the time when the request was queued. These items are separated by commas and are in the order just named. The user ID, group ID, and sizes are preceded by the word <code>uid</code> , <code>gid</code> , or <code>size</code> , respectively.
C	The number of copies printed.
D	The printer or class destination or the word <code>any</code> .
F	The name of the file printed. This line is repeated for each file printed, and files are printed in the order given.
f	The name of the form used.
H	The type of special handling used, spelled out (<code>resume</code> , <code>hold</code> , <code>immediate</code>). The only useful value found in this line will be <code>immediate</code> .
N	The type of alert used when the print request successfully completed. The type is the letter M if the user was notified by mail, or W if the user was notified by a message to his or her terminal.
O	The <code>-o</code> options.
P	The priority of the print request.
p	The list of pages printed.
r	This single letter line is present if the user asked for “raw” processing of the files (the <code>-r</code> option of the <code>lp</code> command.)
S	The character set or print wheel used.
s	The outcome of the request as a combination of individual bits expressed in hexadecimal form. While several bits are used internally by the Spooler, the most important bits are listed below: <code>0x0004</code> Slow filtering finished successfully. <code>0x0010</code> Printing finished successfully. <code>0x0040</code> The request was canceled. <code>0x0100</code> The request failed filtering or printing.
T	The title placed on the banner page.
t	The type of content found in the file(s).
U	The name of the user who submitted the print request.
x	The slow filter used for the request.
Y	The list of special modes to give to the filters used to print the request.
y	The fast filter used for the request.
z	The printer used for the request. This will differ from the destination (the D line) if the request was queued for any printer or a class of printers or if the request was moved to another destination by LP administrator.

PostScript Printers

PostScript is a general purpose programming language, like C or Pascal. In addition to providing the usual features of a language, however, PostScript allows a programmer to specify the appearance of both text and graphics on a page.

A PostScript printer is a printer equipped with a computer that runs an interpreter for processing PostScript language files.

NOTE

If the Enhanced Security Utilities are installed and running on your system, you—as an administrator—will not have access to PostScript filters because these filters carry MAC labels of `USER_PUBLIC`. If you need access to the PostScript filters (as you will, for example, if you are checking to make sure they are working properly), temporarily change the level of the filters (located in `/usr/lib/lp/postscript`) to `SYS_PUBLIC`.

When a PostScript printer receives a file, it runs that file through the interpreter and then prints it. Unless special provisions have been made by the manufacturer, files submitted to a PostScript printer must be written in the PostScript language.

Why would you want to use a PostScript printer? PostScript provides excellent facilities for managing text and graphics and combining them. Also, most major applications that support printed output support PostScript. Graphics operators facilitate the construction of geometric figures which can then be positioned and scaled with any orientation. The text capabilities allow the user to specify a number of different fonts that can be placed on a page in any position, size, or orientation. Because text is treated as graphics, text and graphics are readily combined. Moreover, the language is resolution and device independent, so that draft copies can be proofed on a low-resolution device and the final version printed in higher resolution on a different device.

Applications that support PostScript, including word-processing and publishing software, will create documents in the PostScript language without intervention by the user. Thus, it is not necessary to know the details of the language to take advantage of its features. However, standard files that some applications or special terminals produce cannot be printed on a PostScript printer because they are not described in the language. The LP Print Service provides optional filters to convert many of these files to PostScript so that users may take advantage of PostScript and continue to use their standard applications, such as `troff`.

NOTE

Many of these filters are included in the Advanced Commands package, which is part of the Utilities Set.

Retail Type 1 fonts can be installed for use with applications running on the Desktop. These fonts may be downloaded to PostScript printers if the application generates Post-

Script output that users them. The `lp` command handles this automatically using the `download` filter.

How to Use a PostScript Printer

When the PostScript printers and filters have been installed, LP manages PostScript files like any others. If `psfile` is a file containing a PostScript document and `psprinter` has been defined to LP as a PostScript printer, the command

```
lp -d psprinter -T ps psfile
```

will schedule the print request and manage the transmission of the request to the PostScript printer.

Support of Non-PostScript Print Requests

Because PostScript is a language and PostScript printers are expecting print requests written in that language, some applications may produce standard print requests that may not be intelligible to PostScript printers. The following are examples of print requests that may not be interpreted by some PostScript printers.

Content Type	Type of Print Request
simple	Print an ASCII (“simple”) text file.
troff	Print output from the <code>troff</code> command.
daisy	Print files intended for a Diablo 630 (“daisy-wheel”) printer.
dmd	Print the contents of a bit-mapped display from a terminal.
tek4014	Print files formatted for a Tektronix 4014 device.
plot	Print plot-formatted files.

Filters are provided with the LP Print Service to translate print requests with these formats to the PostScript language. For example, to convert a file containing ASCII text to PostScript code, the filter takes that text and writes a program around it, specifying printing parameters such as fonts and the layout of the text on a page.

Once the PostScript filters are installed, they will be invoked automatically by the LP Print Service when a user specifies a content-type for a print request with the `-T` option. For example, if a user enters the command

```
lp -d psprinter -T simple report2
```

the ASCII file `report2` (a file with an “ASCII” or “simple” format) will be converted to PostScript automatically, as long as the destination printer (`psprinter`) has been defined to the system as a PostScript printer.

Additional PostScript Capabilities Provided by Filters

The filters previously described also take advantage of PostScript capabilities to provide additional printing flexibility. Most of these features may be accessed through the “mode option” (invoked by the **-y** option) to the **lp** command. These filters allow you to use several unusual options for your print jobs. The following list describes these options and shows the option you should include on the **lp** command line for each one.

-y <i>reverse</i>	Reverse the order in which pages are printed.
-y <i>landscape</i>	Change the orientation of a physical page from portrait to landscape.
-y <i>x=number,y=number</i>	Change the default position of a logical page on a physical page by moving the origin.
-y <i>group=number</i>	Group multiple logical pages on a single physical page.
-y <i>magnify=number</i>	Change the logical size of each page in a document.
-o <i>length=number</i>	Select the number of lines in each page of the document.
-P <i>num_list</i>	Select, by page numbers, a subset of a document to be printed, where <i>num_list</i> is page numbers or page ranges separated by commas (for example, 1, 4, 6-8, 14 - prints pages 1, 4, 6, 7, 8, and 14 through the end).
-n <i>number</i>	Print multiple copies of a document.

NOTE

If these filters are to be used with an application that creates PostScript output, make sure that the format of the application conforms to the format of the PostScript file structuring comments. In particular, the beginning of each PostScript page must be marked by the comment

%%Page: *label ordinal*

where *ordinal* is a positive integer that specifies the position of the page in the sequence of pages in the document, and *label* is an arbitrary page label.

For example, say you have a file called **report2** that has a content type **simple** (meaning that the content of this file is in ASCII format). You want to print six pages of this file (pages 4-9) with two logical pages on each physical page. Because one of the printers on your system (**psprinter**) is a PostScript printer, you can do this by entering the following command:

```
lp -d psprinter -T simple -P 4-9 -y group=2 myfile
```

The filter which groups these logical pages will try to position the pages on the physical page to maximize space utilization. Thus when you specify **group=2**, the pages will be

printed side by side, so that the physical page will be landscape orientation. Landscape mode, which controls the orientation of the logical page rather than the physical page, would cause the logical pages to be positioned one on top of the other when combined with the `group=2` option.

The Administrator's Duties

Support of PostScript printers is similar to support of other printers, in that the printers must be defined to the system with the `lpadmin` command and the appropriate software must be installed to manage them. PostScript printers may require some additional effort in supporting fonts and establishing where "slow" filtering (discussed later) occurs.

Installing and Maintaining PostScript Printers

PostScript printers, like other printers, are installed with the `lpadmin` command. They must use the PS interface program, requested by specifying `-m PS` on the `lpadmin` command line.

NOTE

The printer-type and content-type of a PostScript printer must be consistent with the printer type used in PostScript filters. Therefore you should install your PostScript printers with a printer-type of PS, PS-b, PS-r, or PS-br, and a content-type of PS.

The PS printer types serve two functions. First, they cause LP to activate the correct "fast" filter to communicate with the printer. PS and PS-r are used to communicate with printers connected via a serial port; PS-b and PS-br, to communicate with printers connected via a parallel port. Second, the PS interface creates a PostScript banner page for PS printers. The banner page is printed last if the printer-type is PS-r or PS-br, and the pages of the document are printed in reverse order. The printer type is specified with the `-T` option to the `lpadmin` command.

Printer	Connection Type	Page Order
PS	serial	normal
PS-b	parallel	normal
PS-r	serial	reverse
PS-br	parallel	reverse

The `-b` specification (used when you select PS-b or PS-br) represents "batch," which is typically used for parallel connections but may also be used for serial connections if you don't want PostScript printer status messages. The PS and PS-r printer types cannot be used for parallel connections.

By specifying the `-I` option of the `lpadmin` command when configuring a PostScript printer, you can indicate which content types are handled by the printer without "slow" fil-

tering. For a printer on a server system, PS is the correct content type to enter. However, for a printer on a client system, consideration should be given to the question of where “slow” filtering is to occur, since network and system resource management may be of concern.

By specifying valid content types other than PS, you can force the “slow” filtering of input to occur on the server system. Conversely, if you specify a content type of PS, the input will be filtered locally before the print request is forwarded to the server system for “fast” filtering and printing.

To configure a printer on a server system:

```
lpadmin -p ps1 -T PS-b -I PS -m PS
```

To configure a printer on a client system without local filtering:

```
lpadmin -p ps1 -T PS-b -I simple,daisy,dmd,tek4014,plot
```

To configure a printer on a client system with local filtering:

```
lpadmin -p ps1 -T PS-b -I PS
```

As part of the installation procedure, you may want to install fonts on the printer or downloadable fonts on the computer. See “*Installing and Maintaining PostScript Fonts*” later in this chapter for details.

Installing and Maintaining PostScript Filters

PostScript filters are provided in the Advanced Commands package. The filters it provides cover the majority of situations. In certain circumstances, however, you may find it helpful to change the filter descriptions and install the filters differently. To help you do this, this section describes the location and function of these filters.

PostScript filters are contained in the directory

```
/usr/lib/lp/postscript
```

NOTE

There are two types of filters: fast filters and slow filters. For definitions of these types, see the **lpfilter(1M)** on-line manual page and “*Defining a Filter*” earlier in this chapter.

A prerequisite of communication between any system and a PostScript printer is the presence of the `postio` or the `lp.cat` filter on the system. Those programs are the only mandatory PostScript filters that communicate directly with the PostScript printer. The following filters allow other types of documents to be translated to PostScript and to be printed on a PostScript printer.

File Content Type/Filter

simple	postprint
troff	dpost
daisy	postdaisy
dmd	postdmd
tek4014	posttek
plot	postplot

The following filters perform special functions

Function	Filter
communicate with printer	postio,lp.cat
download fonts	download
reverse or select pages	postreverse
matrix gray scales	postmd

Installing and Maintaining PostScript Fonts

One of the advantages of PostScript is its ability to manage fonts. Fonts are stored in outline form in the Type 1 format, either on the printer or on a computer that communicates with a printer. When a document is printed, the PostScript interpreter generates each character as needed (in the appropriate size) from the outline description of it. If a font required for a document is not stored on the printer being used, it must be transmitted to that printer before the document can be printed. This transmission process is called “downloading fonts.”

Fonts are stored and accessed in several ways.

- Fonts may be stored permanently on a printer. These “printer resident” fonts may be installed in ROM on the printer by the manufacturer. If the printer has a disk, fonts may be installed on that disk by you (that is, by the Print Service administrator). Most PostScript printers are shipped with thirty-five standard fonts, although less expensive models have only thirteen.
- A font may be “permanently-downloaded” by being transmitted to a printer with a PostScript **exitserver** program. A font downloaded in this way will remain in the printer’s memory until the printer is turned off. Memory allocated to this font will reduce the memory available for PostScript print requests. Use of **exitserver** programs requires the printer system password and may be reserved for the printer administrator. This method is useful when there is continual use of a font by the majority of print requests serviced by that printer.
- Fonts may be prepended to a user’s print request by the user, and be transmitted as part of the user’s print request. When the user’s document has

been printed, the space allocated to the font is freed for other print requests. The font is stored in the user's directory. This method is preferable for fonts with more limited usage.

- Fonts may be stored on a system shared by many users. These fonts may be described as "host-resident." This system may be a server for the printer or may be a system connected to the printer by a network. Each user may request fonts in the document to be printed. This method is useful when there are a large number of available fonts or when there is not continual use of these fonts by all print requests. If the fonts will be used only on printers attached to a server, they should be stored on the server. If the fonts are to be used by users on one system, who may send jobs to multiple printers on a network, they may be stored on the users' system.

The LP Print Service allows you to manage fonts in any of these ways. It provides a special download filter to manage fonts using the last method described above.

The LP Print Service can utilize **troff** width tables for the thirty-five standard PostScript fonts which reside on many PostScript printers, for use by the **dpost** program.

Managing Printer-Resident Fonts

Most PostScript printers come equipped with fonts resident in the printer ROM. Some printers have a disk on which additional fonts are stored. A list of the Type 1 fonts in ROM or on disk of an attached PostScript printer can be obtained from the printer manufacturer's documentation. For PostScript printers attached via a serial port, a list of these fonts can also be generated using the **postio** command and a PostScript program, **romfonts**.

First, obtain the device that the PostScript printer is connected on:

```
lpstat -v
```

Given a system on which the PostScript printer **prlocal** is attached on a serial port, this would return output like:

```
device for prlocal: /dev/tty01
```

This shows the printer to be attached on device **/dev/tty01**.

Then, as **root**, run the commands

```
cd /usr/lib/lp/postscript
postio -L /tmp/postio.o -l /dev/tty01 -t romfonts
```

postio The **romfonts** program is a PostScript program that queries the PostScript printer for a list of resident fonts. For our sample **prlocal** printer, this will produce output in the file **/tmp/postio.o** that looks like this:

```

printer startup
%%[ status: waiting; source: serial 25 ]%%
%%[ status: endofjob ]%%
%%[ status: idle ]%%
sending file romfonts
waiting for end of job
%%[ status: busy; source: serial 25 ]%%
/AGaramond-Bold
/AGaramond-BoldItalic
/AGaramond-Italic
/AGaramond-Regular
/AvantGarde-Book
/AvantGarde-BookOblique
/AvantGarde-Demi
/AvantGarde-DemiOblique
...more PostScript font names...
/ZapfChancery-MediumItalic
/ZapfDingbats
%%[ status: endofjob ]%%
job complete

```

When a printer is installed, the list of printer-resident fonts should be added to the font-list for that printer. This font-list file can be edited to contain only the font names in the printer's memory (AGaramond-Bold through ZapfDingbats, in the example), and placed into the file `/etc/lp/printers/prlocal/residentfonts` to prevent downloading of these fonts from the host computer.

The font-lists are kept in the printer administration directories. For a particular printer, this list is contained in the file

`/etc/lp/printers/printer-name/residentfonts`

where *printer-name* is the name of the printer. The `-p` option to `download` tells it to check this file to see what Type 1 fonts are ROM-resident and disk-resident (some PostScript printers have directly attached fonts disks) in the printer so that it does not download such fonts.

This file is not automatically created when a PostScript printer is first set up on your system using `lpadmin`; you may need to create this file yourself. When fonts are permanently downloaded to the printer, the font names should be added to this file. (This will prevent fonts from being downloaded when they are already on the printer, a time-consuming procedure.)

If the printer is attached to a remote system, this list should include fonts which reside on that system and are available for downloading to the printer. This prevents fonts from being transmitted unnecessarily across a network. These files must be edited manually; that is, with the help of a text editor such as `vi`.

Installing and Maintaining Host-Resident Fonts

Some fonts will be resident on the host and transmitted to the printer as needed for particular print requests. As the administrator, it's your job to make PostScript fonts available to all the users on a system. To do so, you must know how and where to install these fonts, using the guidelines described previously. Because fonts are requested by name and stored in files, the LP Print Service keeps a map file that shows the correspondence between the

names of fonts and the names of the files containing those fonts. Both of these must be updated when fonts are installed on the host.

Retail Type 1 fonts can be installed from diskette for use with XWin and **lp**. The fonts are installed in an XWIN directory (`/usr/X/lib/fonts/type1`) and the **map** file in `/usr/share/lib/hostfontdir` is updated to make their location available to download. The actual updating of the **map** file is done with the `mkfontscale` utility.

Available Type 1 fonts are the ones listed in **map**, a map table which consists of font name-filename pairs. The font name is the name of the PostScript Type 1 font, exactly as it would appear in a `%%DocumentFonts:` comment and exactly as it appears in the literal `/FontName` in the Type 1 font program itself. The filename is the pathname of the host resident Type 1 font.

Install host-resident PostScript fonts by doing the following:

- Copy the font file to the appropriate directory.
- Add to the map table the name of the font and the name of the file in which it resides.
- If you will be using **troff**, you must create new width tables for this font in the standard **troff** font directory.

Where Are Fonts Stored?

The fonts available for use with PostScript printers reside in the directory `/usr/share/lib/hostfontdir` or other directories you may choose.

Adding an Entry to the Map Table

Also in the `hostfontdir` directory, you (the administrator) must create and maintain a map table that shows the correspondence between the name assigned to each font by the foundry (the company that created the font) and the name of the file in which that font resides. A filename that begins with a `/` is used as is; otherwise the pathname is relative to the host font directory. Comments in the map table are introduced by `%` (as in PostScript) and extend to the end of the line.

For example, to map the font called “Palatino Bold,” add the following line to the map table:

```
Palatino-Bold /usr/share/lib/hostfontdir
```

(The map table itself is in the file `/usr/share/lib/hostfontdir/map`).

Once this entry exists in the map table on your system, your users will be able to have a Palatino Bold font used in their print jobs. When they submit, for printing, a file containing a request for this font, the LP Print Service will prefix a copy of the file

```
/usr/share/lib/hostfontdir
```

to that file before sending it to the printer, as long as it is not defined in the `resident-fonts` file.

Downloading Host-Resident Fonts

The creators of the PostScript language anticipated that users would want to download fonts to printers. The *PostScript Language Reference Manual* (by Adobe Systems, Inc., Addison-Wesley Publishing Co., Inc., 2nd edition, 1990) states the following:

“. . . programs that manage previously generated PostScript page descriptions, such as `printer spooler' utilities, may require additional information about those page descriptions. For example, if a page description references special fonts, a spooler may need to transmit definitions of those fonts to the PostScript printer ahead of the page description itself.

To facilitate these and other operations, [PostScript] defines a standard set of structuring conventions for PostScript programs.”

The download filter relies on these structuring conventions to determine which fonts must be downloaded.

When the PostScript document contains a request for fonts not loaded on the printer, the download filter manages this request. This filter is invoked as a “fast filter”; it downloads fonts automatically if the fonts reside on the same system as the printer. The download filter may also be used to send fonts to a remote printer. To do this, you may create a new filter table entry which calls the download filter as a “slow” filter through the `-y` option. Alternatively, you may force selection of this filter by changing the input-type.

The download filter does five things:

- It searches the PostScript document to determine which fonts have been requested. These requests are documented with the following PostScript structuring comments:

```
%%DocumentFonts: font1 font2v. . .
```

in the header comments.

- It searches the list of fonts resident on that printer (in `/etc/lp/printers/printer-name/residentfonts`) to see if the requested font must be downloaded.
- If the font is not resident on the printer, it searches the host-resident font directory to see if the requested font is available. The only candidates for downloading are fonts listed in the map table that point download to readable files. A Type 1 font is downloaded once, at most, for a single document, even if it occurs multiple times in the `%%DocumentFonts:` comment or PostScript file. The downloading of fonts occurs only for the duration of the PostScript job; however, permanent downloading of fonts to the printer's RAM can be done with special PostScript programming techniques using the `exitserver` operator.

Requests for unlisted fonts or inaccessible files are ignored; all requests are ignored if the map table can't be read.

- If the font is available, the filter takes the file for that font and prefixes it to the file to be printed.
- The filter sends the font definition file and the PostScript source file (the file to be printed) to the PostScript printer.

Customizing the Print Service

Although LP tries to be flexible enough to handle most printers and printing needs, you may buy a printer that doesn't quite fit into the way the LP handles printers or may have a printing need that the standard features of the LP won't accommodate.

You can customize LP in a few ways. This section tells you how you can

- adjust the printer port characteristics
- adjust the Terminfo database
- write an interface program

Figure 13-1 gives an overview of the processing of a print request.

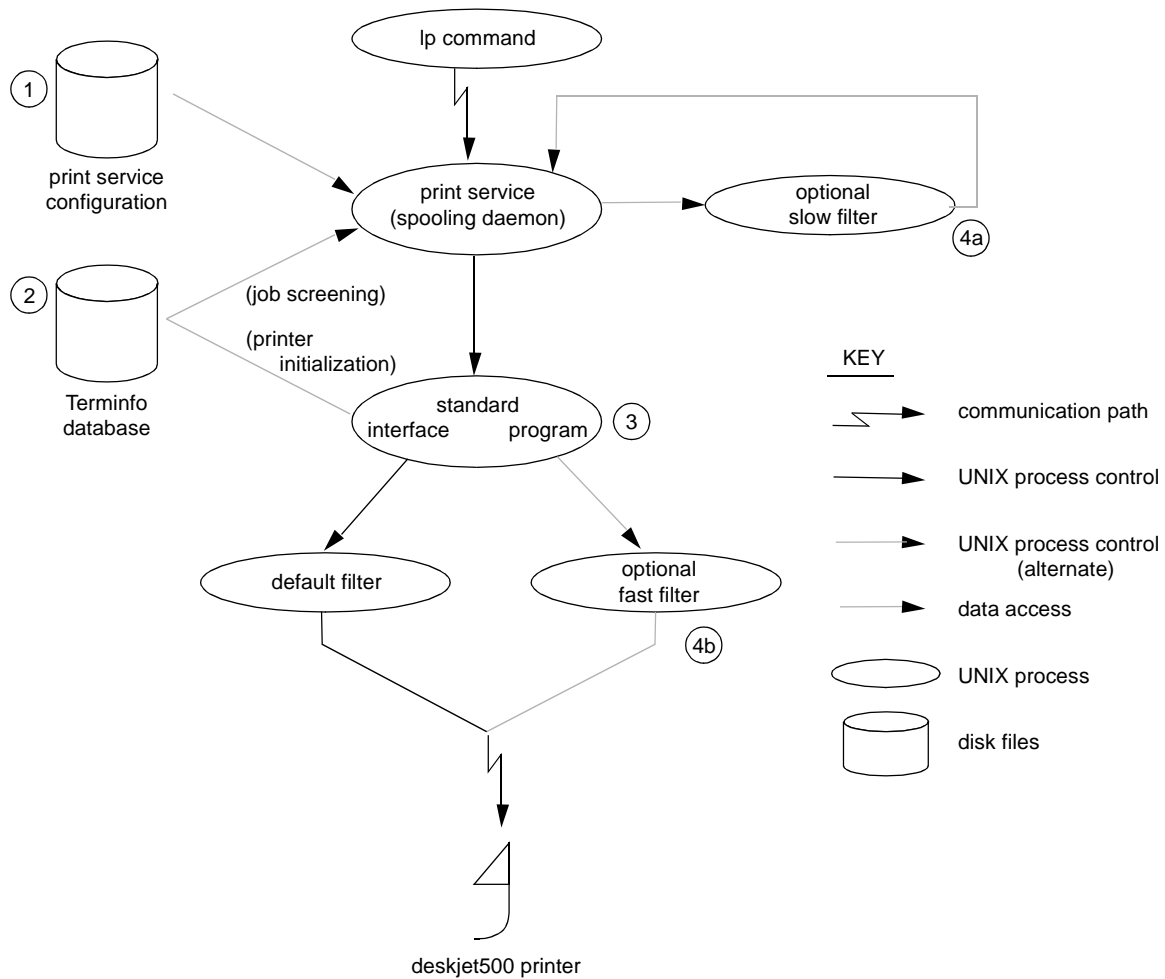
Each print request is sent to a *spooling daemon* that keeps track of all the requests. The daemon is created when you start LP print service. This operating system process is also responsible for keeping track of the status of the printers and slow filters; when a printer finishes printing a user's file, the daemon will start it printing another request, if one is queued.

You can customize the print service by adjusting or replacing some of the pieces shown in the figure. (The numbers are keyed to the diagram.)

1. For most printers, you need only change the printer configuration stored on disk. The earlier sections of this chapter have explained how to do this. Some of the more printer-dependent configuration data are the printer port characteristics: baud rate, parity, and so on.
2. For printers that are not represented in the Terminfo database, you can add a new entry that describes the capabilities of the printer. This database is used in two parallel capacities: screening print requests to ensure that those accepted can be handled by the desired printer and setting the printer in a state where it is ready to print the request.

For instance, if the Terminfo database does not show a printer capable of setting a page length requested by a user, the spooling daemon will reject the request. On the other hand, if it does show it capable, then the same information will be used by the interface program to initialize the printer.

3. For particularly difficult printers or if you want to add features not provided by the delivered LP print service, you can change the standard interface program. This program is responsible for managing the printer: it prints the banner page, initializes the printer, and invokes a filter to send copies of the user's files to the printer.



163000

Figure 13-1. How LP processes print request `lp -d deskjet500 file`

4. (both 4a. and 4b.) To provide a link between the applications used on your system and the printers, you can add slow and fast filters. Each type of filter can convert a file into another form, mapping one set of escape sequences into another, for instance, and can provide special setup by interpreting print modes requested by a user. Slow filters are run separately by the daemon to avoid tying up a printer. Fast filters are run so their output goes directly to the printer; thus, they can exert control over the printer.

Each of these items will be covered in detail in the sections which follow.

Adjusting the Printer Port Characteristics

You should make sure that the printer port characteristics set by LP match the printer communication settings. The standard printer port settings were designed to work with typical operating system files and many printers, but they won't work with all files and printers. This isn't really a customizing step, since a standard feature of LP is to allow you to specify the port settings for each printer. However, it's an important step in getting your printer to work with the LP, so it's described in more detail here.

When you add a new printer, read the documentation that comes with it so that you understand what it expects from the host (LP). Then read the manual page for the **stty(1)** command. It summarizes the various characteristics that can be set on a terminal or printer port.

Only some of the characteristics listed in the **stty(1)** manual page are important for printers. The ones likely to be of interest to you are listed below (but you should still consult the **stty(1)** manual page for others.)

stty Option	Meaning
evenp	Sends even parity in the 8th bit
oddp	Sends odd parity in the 8th bit
-parity	Doesn't generate parity, sends all 8 bits unchanged
110 - 38400	Sets the communications speed to this baud rate
ixon	Enables XON/XOFF (also known as START/STOP or DC1/DC3) flow control
-ixon	Turns off XON/XOFF flow control
-opost	Doesn't do any "output post-processing"
opost	Does "output post-processing" according to the settings listed below
onlcr	Sends a <CR> before every linefeed
-onlcr	Doesn't send a <CR> before every linefeed
ocrnl	Changes <CR> into linefeeds
-ocrnl	Doesn't change <CR> into linefeeds
-tabs	Changes tabs into an equivalent number of spaces
tabs	Doesn't change tabs into spaces

When you have a set of printer port characteristics you think should apply, adjust the printer configuration as described in the "Printer Port Characteristics" section in the "Basic Printer Service" Chapter in this manual. You may find that the default settings are sufficient for your printer.

Adjusting the Terminfo Database

LP relies on a standard interface and the Terminfo database to initialize each printer and set up a selected page size, character pitch, line pitch, and character set. Thus, it is usually sufficient to have the correct entry in the Terminfo database to add a new printer to LP. A number of entries for popular printers are delivered in Terminfo database entries with LP package.

Each printer is identified in the Terminfo database with a short name; this kind of name is identical to the kind of name used to set the TERM shell variable. If you cannot find a Terminfo entry for your printer, you should add one. If you don't, you may still be able to use the printer with the LP, but you won't be able to get automatic selection of page size, pitch, and character sets, and you may have trouble keeping the printer set in the correct modes for each print request. Another option to follow instead of updating the Terminfo entry is to customize the interface program used with the printer. See the next section for details on how to do this.

There are hundreds of items that can be defined for each terminal or printer in the Terminfo database. However, LP uses fewer than fifty of these, and most printers need even less than that. You can check items defined for a specific Terminfo entry by typing the following command:

```
infocmp terminfo_name
```

The following table lists the items that need to be defined (as appropriate for the printer) to add a new printer to LP.

Terminfo Item	Meaning
Booleans:	
daisy	Printer needs operator to change character set
Numbers:	
bufsz	Number of bytes buffered before printing
* cols	Number of columns in a line
* it	Tabs initially every this many spaces
* lines	Number of lines on a page
orc	Horizontal resolution in units per character
orhi	Horizontal resolution in units per inch
orl	Vertical resolution in units per line
orvi	Vertical resolution in units per inch
cps	Average print rate in characters per second
Strings:	
* cr	Carriage return (<CR>)
cpi	Change number of characters per inch
lpi	Change number of lines per inch

Terminfo Item	Meaning
chr	Change horizontal resolution
cvr	Change vertical resolution
csnm	List of character set names
mgc	Clear all margins (top, bottom and sides)
* hpa	Horizontal position absolute
* cud1	Down one line
* cuf1	Carriage right
swidm	Enable double wide printing
rwidm	Disable double wide printing
* ff	Page eject
* is1	Printer initialization string
* is2	Printer initialization string
* is3	Printer initialization string
* if	Name of initialization file
* iprog	Path name of initializing program
* cud	Move carriage down # lines
* cuf	Move carriage right # columns
* rep	Repeat a character # times
* vpa	Vertical position absolute
scs	Select character set
smgb	Set bottom margin at current line
smgbp	Set bottom margin
* smgl	Set left margin at current column
smglp	Set left margin
* smgr	Set right margin at current column
smgrp	Set right margin
smgt	Set top margin at current line
smgtp	Set top margin
scsd	Start definition of a character set
* ht	Tab to next 8 space tab stop

Consult on-line manual page **terminfo(4)** for the file structure details on how to construct a Terminfo database entry for a new printer.

Once you've made the new entry, you need to compile it into the database using the **tic** program. Just enter the following command:

`tic filename`

filename is the name of the file containing the Termino entry you have crafted for the new printer. (This program is available in the Terminal Information Utilities.)

How to Write an Interface Program

If you have a printer that is not supported by simply adding an entry to the Termino database or if you have printing needs that are not supported by the standard interface program you can furnish your own interface program. It is a good idea to start with the standard interface program and change it to fit, rather than starting from scratch. You can find a copy of it under the name

`/usr/lib/lp/model/standard`

What Does an Interface Program Do?

Any interface program performs the following tasks:

- Initializes the printer port, if needed. The generic interface program uses the `stty` command to do this.
- Initializes the physical printer. The generic interface program uses the Termino and the `TERM` shell variable to get the control sequences to do this.
- Prints a banner page, if needed.
- Prints the correct number of copies of the request content.

An interface program is not responsible for opening the printer port. This is done by LP, which calls a “dial-up” printer if that’s how the printer is connected. The printer port connection is given to the interface program as standard output, and the printer is set as the “controlling terminal” for the interface program. A “hang-up” of the port will cause a `SIGHUP` signal to be sent to the interface program.

A customized interface program must not terminate the connection to the printer or in any fashion “uninitialize” the printer. This allows LP to use the interface program only for preparing the printer and printer port, while the printing of content is done elsewhere, by LP, for example, for pre-printed form alignment patterns.

NOTE

If a file is being processed by a non-secure system package and then shipped to a secure system machine over a network to be printed, use a dumb plotter interface so that application data is fed directly into the printer/plotter.

How Is an Interface Program Used?

When LP routes an output request to a printer, the interface program for the printer is invoked as follows:

```
/etc/lp/interfaces/P id user title copies options file1 file2...
```

Arguments for the interface program are

<i>P</i>	printer name
<i>id</i>	request id returned by lp
<i>user</i>	logname of user who made the request
<i>title</i>	optional title specified by the user
<i>copies</i>	number of copies requested by user
<i>options</i>	list of options separated by blanks, specified by user or set by the LP
<i>file</i>	full path name of a file to be printed

When the interface program is invoked, its standard input comes from `/dev/null`, its standard output is directed to the printer port, and its standard error output is directed to a file that will be given to the user who submitted the print request.

The standard interface recognizes the following values in the list in *options*:

nobanner	This option is used to skip the printing of a banner page; without it, a banner page is printed.
nofilebreak	This option is used to skip page breaks between separate data files; without it, a page break is made between each file in the content of a print request.
<i>cpi=decimal-number1</i>	
<i>lpi=decimal-number2</i>	These options say to print with <i>decimal-number1</i> columns per inch and <i>decimal-number2</i> lines per inch, respectively. The standard interface program extracts from the Terminfo database the control sequences needed to initialize the printer to handle the character and line pitches.
	The words <i>pica</i> , <i>elite</i> , and <i>compressed</i> are acceptable replacements for the <i>decimal-number1</i> and are synonyms for 10 columns per inch, 12 columns per inch, and as many columns per inch as possible.
<i>length=decimal-number1</i>	
<i>width=decimal-number2</i>	These options specify the length and width, respectively, of the pages to be printed. The standard interface program extracts from the Terminfo database the control sequences

needed to initialize the printer to handle the page length and page width.

`stty='stty-option-list'` The *stty-option-list* is applied after a default *stty-option-list* as arguments to the `stty` command. The default list is used to establish a default port configuration; the additional list given to the interface program is used to change the configuration as needed.

The above options are either specified by the user when issuing a print request or by LP from defaults given by the administrator for the printer (`cpi`, `lpi`, `length`, `width`, **`stty`**) or for the pre-printed form used in the request (`cpi`, `lpi`, `length`, `width`.)

Additional printer configuration information is passed to the interface program in shell variables:

`TERM=printer-type` This shell variable specifies the type of printer. The value is used as a key for getting printer capability information from the extended Terminfo database.

`FILTER='pipeline'` This shell variable specifies the filter to use to send the request content to the printer; the filter is given control of the printer.

`CHARSET=character-set` This shell variable specifies the character set to be used when printing the content of a print request. The standard interface program extracts from the Terminfo database the control sequences needed to select the character set.

A customized interface program should either ignore these options and shell variables or should recognize them and treat them in a consistent manner.

Customizing the Interface Program

You want to make sure that the custom interface program sets the proper **`stty`** modes (terminal characteristics such as baud rate or output options). The standard interface program does this, and you can follow suit. Look for the section that begins with the shell comment

```
## Initialize the printer port
```

Follow the code used in the standard interface program. It sets both the default modes and the adjusted modes given by LP or the user with a line like the following:

```
stty mode options 0<&1
```

This command line takes the standard input for the **`stty`** command from the printer port. An example of an **`stty`** command line that sets the baud rate at 1200 and sets some of the option modes is shown below:

```
stty -parenb -parodd 1200 cs8 cread clocal ixon 0<&1
```

One printer port characteristic not set by the standard interface program is hardware flow control. The way that this is set will vary depending on your computer hardware. The code for the standard interface program suggests where this and other printer port characteristics can be set. Look for the section that begins with the shell comment

```
# Here you may want to add other port initialization
code.
```

Because different printers have different numbers of columns, make sure the header and trailer for your interface program correspond to your printer. The standard interface program prints a banner that fits on an 80-column page (except for the user's title which may be longer). Look for the section in the code for the standard interface program that begins with the shell comment

```
## Print the banner page
```

The custom interface program should print all user-related error messages on the standard output or on the standard error. The messages sent to the standard error will be mailed to the user; the messages printed on the standard output will end up on the printed page where they can be read by the user when he or she picks up the output.

When printing is complete, your interface program should exit with a code that tells the status of the print job. Exit codes are interpreted by LP as follows:

Code	Meaning to LP Print Service
0	The print request has completed successfully. If a printer fault has occurred, it has been cleared.
1 to 127	A problem was encountered in printing this particular request (for example, too many non-printable characters or the request exceeds the printer capabilities). This problem will not affect future print requests. LP notifies the person who submitted the request that there was an error in printing it. If a printer fault has occurred, it has been cleared.
128	Reserved for internal use by LP print service. Interface programs must not exit with this code.
129	A printer fault was encountered in printing the request. This problem will affect future print requests. If the fault recovery for the printer directs LP to wait for the administrator to fix the problem, it will disable the printer. If the fault recovery is to continue printing, LP will not disable the printer but will try printing again in a few minutes.
greater than 129	These codes are reserved for internal use by LP. Interface programs must not exit with codes in this range.

As the table shows, one way of alerting the administrator to a printer fault is to exit with a code of 129. Unfortunately, if the interface program exits, LP has no choice but to reprint the request from the beginning when the fault has been cleared. A way of getting an alert to the administrator without requiring reprinting the entire request is to have the interface program send a fault message to the LP but wait for the fault to clear. When the fault clears, the interface program can resume printing the user's file. When done printing, it can give a zero exit code just as if the fault never occurred. An added advantage is that the interface program can detect when the fault is cleared automatically so that the administrator doesn't have to enable the printer.

Fault messages can be sent to LP using the `lp.tell` program. This is referenced using the `$LPTELL` shell variable in the standard interface code. The program takes its standard

input and sends it to LP where it is put into the message that alerts the administrator to the printer fault. If its standard input is empty, `lp.tell` does not initiate an alert. Examine the standard interface code immediately after these comments for an example of how the `lp.tell` (`$LPTELL`) program is used:

```
# Here's where we set up the $LPTELL program to
# capture fault messages.

# Here's where we print the file.
```

With the special exit code 129 or the `lp.tell` program, there is no longer the need for the interface program to disable the printer itself. Your interface program can disable the printer directly, but doing so will override the fault alerting mechanism. Alerts are sent only if LP detects the printer has faulted and the special exit code and the `lp.tell` program are its main detection tools.

If LP has to interrupt the printing of a file at any time, it will “kill” the interface program with a signal 15 (see the `kill(1)` and `signal(2)` on-line manual pages.). If the interface program dies from receipt of any other signal, LP assumes that future print requests won't be affected and will continue to use the printer. LP will notify the person who submitted the request that it did not finish successfully.

The signals `SIGHUP`, `SIGINT`, `SIGQUIT`, and `SIGPIPE` (trap numbers 1, 2, 3, and 13) start out being ignored when the interface is invoked. The standard interface changes this to trap these signals at appropriate times. The standard interface will consider receipt of these signals as meaning the printer has a problem and will issue a fault.

An Example: Adding a Printer with a Customized Interface

Add a new printer called `laser` on printer port `/dev/term/tty01`. It should use a customized interface program, located in the directory `/usr/doceng/laser_interface`. It can handle three file types—`i10`, `i300`, and `impress`—and it may be used only by the users `doceng` and `docpub`. (The following command line is split into multiple lines for readability.)

```
lpadmin -p laser -v /dev/term/tty01 \
-i /usr/doceng/laser_interface \
-I "i10,i300,impress" \
-u "allow:doceng,docpub"
```

Troubleshooting

Networking Problems

You may encounter several types of problems while trying to get files printed over a network: (1) requests being sent to server printers may back up in the client queue; (2) requests sent to server printers may be backed up in the server queue; or (3) a user may

receive contradictory messages about whether a server printer has accepted a print request. The rest of this section describes each of these situations and suggests how to resolve them.

Jobs Backed Up in the Client Queue?

There are a lot of jobs backing up in the client queue for a server printer. There are three possible explanations:

- The server system is down or the network between the client and server systems is down. To resolve this problem, run the **reject** command for all the server printers on your system, as follows:

```
reject printer-name
```

This will stop new requests for those printers from being added to the queue. Once the system comes up again, and jobs start being taken from your queue, type **accept printer_name** to allow new jobs to be queued.

- The server printer is disabled on the client system.
- ID mapping was not set up properly. For instructions, see “*Define Attribute Mapping*” earlier in this chapter.
- The underlying operating system network software was not set up properly. For details, see **lpssystem (1M)**.

Jobs Backed Up in the Server Queue?

The server printer has been disabled.

Conflicting Messages about the Acceptance of Jobs?

A user enters a print request and is notified that the system has accepted it. The job is sent to a server system and the user receives mail that the job has been rejected. This may be happening for one of two reasons. First, the client computer may be accepting requests while the server computer is rejecting requests.

Second, the definition of the server printer on the client computer may not match the definition of that printer on the server computer. Specifically, the definitions of print job components such as filters, character sets, print wheels, and forms are not the same on the client and server systems. Identical definitions of these job components must be registered on both the client and the server systems, if client users are to be able to access server printers.

Administering LP through OA&M Menus

The system administration menus are only available if the Operations, Administration and Maintenance (OA&M) package is installed on your system. To access the system administration menu for the LP Print Service, type **sysadm** and select the **printers** item from the main menu. The menu shown in Screen 13-1 will appear.

```

2          Line Printer Services Configuration and Operation

classes    - Manage Classes of Related Printers
filters    - Manage Filters for Special Processing
forms      - Manage Pre-Printed Forms
operations - Perform Daily Printer Service Operations
printers   - Configure Printers for the Printer Service
priorities - Assign Print Queue Priorities to Users
requests   - Examine and Manipulate Print Requests
status     - Display Status of the Print Service
    
```

Screen 13-1. Main Menu for Print Service

NOTE

If, in addition to the LP Print Service you install the Commands Networking Extension package, the following entry will also appear in the above menu:

```
systems - Configure Connections to Remote Systems
```

The following table shows how the tasks listed on the **printers** menu correspond to the shell commands discussed throughout this chapter.

Task to Be Performed	sysadm Task	Shell Command
Group printers into classes	classes	lpadmin (1M)
Provide pre-processing software for files to be printed	filters	lpfilter (1M)
Define pre-printed forms for print requests	forms	lpforms (1M)
Control (turn on/off) queuing of requests; enable & disable printers; mount forms and fonts; start & stop Print Service; and report status of printers, classes, & forms	operations	accept & reject [see accept (1M)], enable & disable [see enable (1)], lpadmin (1M) , lpsched & lpshut [see lpsched (1M)], lpstat (1)

Task to Be Performed	sysadm Task	Shell Command
Configure printers for Print Service	printers	lpadmin (1M)
Define levels of priority available to users requesting print jobs	priorities	lpusers (1M)
Identify active printers, print wheels & character sets, mounted forms, and pending requests	status	lpstat (1)
Submit and cancel print requests	requests	lp & cancel [see lp (1)], lpmove [see lpsched (1M)]
Set up communication to remote Print Service	systems	lpssystem (1M)

Quick Reference to LP Print Service Administration

LP administrative commands are found in the `/usr/sbin` and `/usr/bin` directories. (If you expect to use them frequently, you might find it convenient to include these paths in your `PATH` variable.) To use these commands, you must be an LP administrator.

You'll also need to use commands designed for users as well as administrators, such as the commands for disabling and enabling a printer.

- Defining a device special file for a printer in the Device Database:

putdev (1M)

- Allocating a device special file for use:

admalloc (1M)

- Activating a printer:

/usr/bin/enable

See **enable (1M)**.

- Canceling a request for a file to be printed:

/usr/bin/cancel

See **lp (1)**.

- Sending a file (or files) to a printer:

/usr/bin/lp

See **lp (1)**.

- Reporting the status of the LP Print Service:

/usr/bin/lpstat

See **lpstat (1)**.

- Deactivating a specified printer(s):

/usr/bin/disable

See **enable (1M)**.

- Permitting job requests to be queued for a specific destination:

/usr/sbin/accept

See **accept (1M)**.

- Preventing jobs from being queued for a specified destination:

/usr/sbin/reject

See **accept (1M)**.

- Setting up or changing printer configurations:

/usr/sbin/lpadmin

See **lpadmin (1M)**.

- Setting up or changing filter definitions:

/usr/sbin/lpfilter

See **lpfilter (1M)**.

- Setting up or changing preprinted forms:

/usr/sbin/lpforms

See **lpforms (1M)**.

- Mounting a form:

/usr/sbin/lpadmin

See **lpadmin (1M)**.

- Moving output requests from one destination to another:

/usr/sbin/lpmove

See **lpmove (1M)**.

- Starting the LP Print Service scheduler:

/usr/lib/lp/lpsched

See **lpsched (1M)**.

- Stop the LP Print Service scheduler

`/usr/sbin/lpshut`

See `lpsched(1M)`.

- Setting or changing the default priority and priority limits that can be requested by users of the LP Print Service:

`/usr/sbin/lpusers`

See `lpusers(1M)`.

The sysadm Interface

Replace with Part 5 tab for 0890430

Part 5 - The sysadm Interface

Part 5 The sysadm Interface

Chapter 14	Using the sysadm Interface	14-1
Chapter 15	Customizing the sysadm Interface	15-1
Chapter 16	Modifying the sysadm Interface	16-1

Using the sysadm Interface

Introduction	14-1
The Menu Interface Window	14-2
Types of Frames	14-3
Menus	14-4
Forms	14-4
Changing Answers on a Form	14-4
Saving the Answers on a Form	14-4
Messages	14-4
Full Window Tasks	14-5
Frame Manipulation Tools	14-5
Navigation Keys	14-5
Function Keys	14-5
Alternate Keystrokes	14-6
Automatic Function Key Downloading	14-7
Using the LOADPFK Environment Variable to Disable Downloading	14-7
Using a Shell Script to Restore Function Key Settings	14-7
Downloading FMLI Sequences More Efficiently	14-7
The Command Menu	14-8
The Command Line	14-8
Help for Forms: CHOICES	14-9
Getting Help	14-9
Using Express Mode	14-10
A Sample Session: Adding an Account for a New User	14-10
Step 1: Access the Menu Interface	14-10
Step 2: Select a Menu	14-11
Step 3: Select a Task	14-12
Step 4: Fill Out the Forms	14-12
Step 5: Exit from the Interface	14-14
System Administration Menu Map	14-14
Quick Reference to the sysadm Interface	14-20
Function Keys	14-20
Menus	14-20
Forms	14-20
Text Frames	14-20
Command Line	14-20
Exiting from sysadm	14-21

Introduction

NOTE

The System Administration Menu Interface is only available if the Operations, Administration and Maintenance (OAM) package is installed on your system.

This chapter explains how to use the system administration menu interface (accessed through the **sysadm** command) for the Operating System (OS). For a summary of commands for navigating the menu system on your terminal, see “*Quick Reference to the sysadm Interface*” at the end of this chapter.

This chapter is organized as follows:

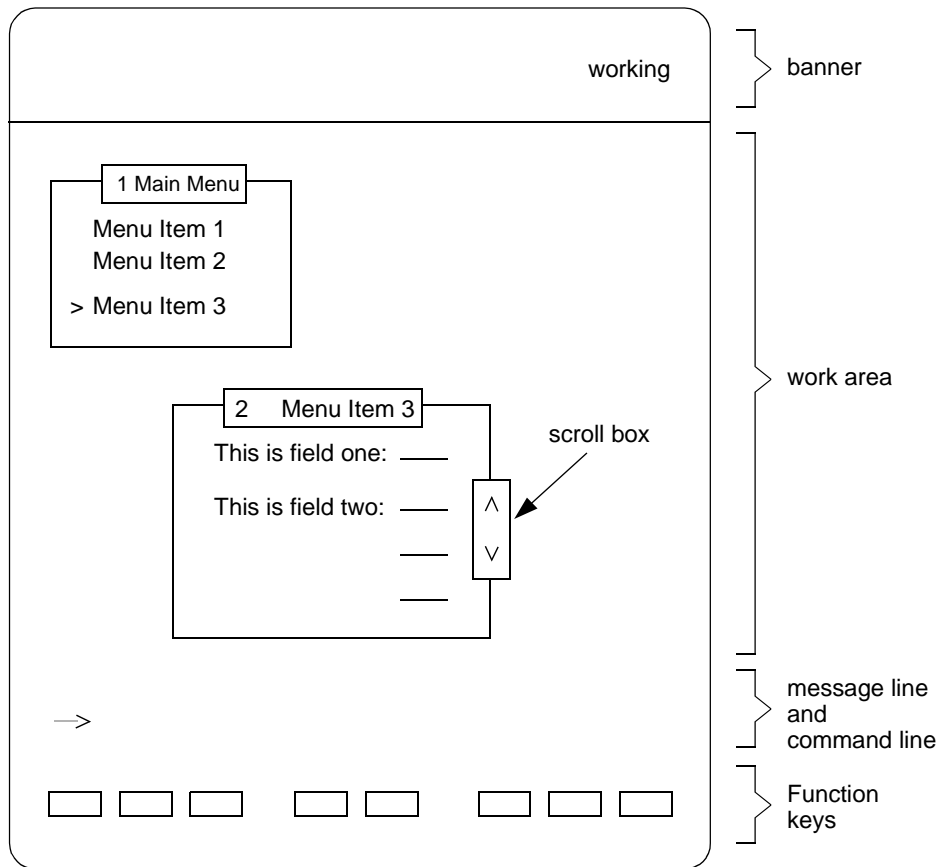
- “*The Menu Interface Window*” introduces the visual components of a **sysadm** window.
- “*Frame Manipulation Tools*” explains how to move around in a **sysadm** window.
- “*A Sample Session: Adding an Account for a New User*” explains a task all system administrators must perform: adding an account for a new user.
- “*System Administration Menu Map*” lists all the menus and tasks available through **sysadm**.
- “*Quick Reference to the sysadm Interface*” summarizes use of the interface.

The **sysadm** menu interface supports all terminals supported by the Forms and Menus Language Interface, however, the screens will not be drawn correctly if the **TERM** environment variable is not set appropriately.

For color terminals, the **/usr/sadm/sysadm/pref/.colorpref** file determines the colors of the text and borders displayed for the **sysadm** menu interface.

The Menu Interface Window

The system administration menu interface contains a series of menus and prompts that guide you through a task. Figure 14-1 shows the organization of a window during a typical **sysadm** session.



162990

Figure 14-1. The System Administration Menu Interface Window

The window is divided into the following five areas:

- | | |
|-------------|--|
| banner line | The banner line is the top line of the window, reserved for the word working . It appears when tasks are in progress and new tasks cannot be started. |
| work area | The work area is where frames are displayed. A frame is an independently scrollable region of the window, enclosed in a "box." |

There are three basic types of frames: menus, tasks, and messages. Frames can overlap one another, but the active frame is always distinguished from other frames by its highlighted title.

If a frame displays three or more lines of information a scroll box appears in its right-hand border. Up and down icons can appear in the scroll box signaling more information.

message line	The message line displays one-line messages from sysadm , which can be an instruction or an error message.
command line	The command line is available for interface commands that can be typed in addition to being selected from the command menu. It is marked by an arrow prompt (-->) .
function keys	The display of eight boxes at the bottom of the window lists the function keys available for the current frame. A different subset of the function keys is displayed for each type of frame used by the interface.

These labels correspond to the function keys on your keyboard. Many keyboards have eight function keys (some have none and some have more than eight) and they are usually located in a row across the top of the keyboard. (If your keyboard does not have function keys, see “*Alternate Keystrokes*” under “*Frame Manipulation Tools*” later in this chapter.

The following icons are used in a **sysadm** window:

Right angle bracket (>)

A right angle bracket in the left-hand margin of a menu indicates your current location on that menu. This cursor can be moved from item to item by several methods, including pressing the up or down arrow keys on your keyboard.

Up and down icons (^ and v)

An up or down icon in the right-hand margin of a frame shows that more information is available either above or below what is visible in the frame.

Right arrow (-->)

A right arrow appears as a special command prompt at the bottom of the window if you invoke it with a <CTRL>-j sequence. The special commands you can enter at this prompt include **cancel**, **cleanup**, **exit**, **help**, **refresh**, and **update**, which are also available on the Command Menu.

Types of Frames

The three types of frames that can be displayed in the **sysadm** work area are menu frames, form frames, and message frames. Figure 14-1 shows examples of a menu frame and a form frame.

Menus

A menu frame contains a list of other menus and tasks. In Figure 14-1, the menu titled “Main Menu” has three menu items from which you can select.

Forms

A form, in the context of the **sysadm** interface, is a frame containing one or more prompts for information, each followed by a line on which a default response appears (when one is available). To enter a new or different response, you can invoke a menu of choices by pressing **CHOICES**, and selecting the appropriate item.

NOTE

If your attempt to access a form fails, the cause may be corrupted, missing, or improperly installed **Menu.*** files. To rebuild these, run **/usr/sadm/install/bin/ie_build**. If you have installed and are running the Enhanced Security Utilities, you must change your system to single-user mode before you can run the **ie_build** command; see the “Security Administration” part of *System Administration, Volume 1*, for details about single-user mode.

In Figure 14-1, the frame titled **Item 3 Form** is a form. It appeared in the window when **Menu Item 3** was selected from the **Main Menu**.

Changing Answers on a Form

After you have answered all the questions on a form, you can still change your answer to a particular question. Move to the answer you want to change, and type the new answer over the original one.

Your answer will be validated for correctness. If it is invalid, an error message will appear on the message line. Press **HELP** or **CHOICES** for more information about valid answers.

Saving the Answers on a Form

When you are satisfied with all your answers on a form, press **SAVE**. If there are more forms to be filled out, the interface will display the next one. If there are no more forms, the system will perform the specified task and display a report about the results.

Messages

The third frame type in the menu interface is a message frame. Message frames usually provide help messages (background information about a task). They do not prompt for information. You can see a help message for any task by pressing the **HELP** function key.

Full Window Tasks

A full window task is an activity involving an interactive function outside the menu interface. It may be a series of forms, a prompt for information, or a piece of text to be read. The menu interface temporarily disappears. The only text from the menu interface appearing in a full window task is a prompt at the bottom of the window: Press <ENTER> to continue.

Press <ENTER> or <CR>, to reenter the menu interface, and regain access to the function keys, Command Menu, and other menu interface features.

Frame Manipulation Tools

To help you move among frames and respond to prompts within frames, the **sysadm** interface defines four tools: navigation keys, function keys, alternate keystrokes, and a menu of frame manipulation commands known as the “Command Menu.” Each of these tools is described below.

Navigation Keys

The following keys are available for navigating a **sysadm** window. These keys are located on the left-hand or right-hand side of your keyboard. (Not all keyboards have all the keys described in this section.)

Arrow Keys	Use the up and down arrow keys to scroll backward and forward, respectively, through a menu.
“Top” and “Bottom” Keys	Use these keys to skip to the first and last items, respectively, on a menu.
<HOME>	Use this key to return to the first line of a menu.
<TAB>	Use this key to move from field to field in a form.

Function Keys

NOTE

If you are using a software application package, you may see function key definitions other than those described here, displayed on your screen. Refer to the documentation delivered with the software application for definitions of these function keys.

Function keys are numbered, programmable keys that have names and functions assigned by the menu interface. These keys are usually found in a row across the top of the keyboard, or in a double column on the left-hand side of the keyboard. (If your keyboard does not have function keys, see “*Alternate Keystrokes*” later in this section.) Each key is labeled Fn; where n is a number between 1 and the number of keys (usually 8).

At the bottom of the `sysadm` window you will see a row of eight boxes. Each box shows the name of the function assigned to the corresponding function key, such as `<CANCEL>` or `<SAVE>`.

The `sysadm` interface defines a total of thirteen function keys. At any given time, a subset of functions is assigned to the function keys. This subset is determined by the task being performed.

CANCEL	Deletes the current frame and returns you to the previous frame
CHOICES	Displays possible answers to a prompt
CMD-MENU	Displays a menu of commands for controlling the interface
CONT	Resumes a task that was interrupted by a query from the system asking you to verify that you want to continue the current task. (Such interruptions are usually made to inform you of possible undesirable results from a task being performed.)
ENTER	Selects the current menu item (also done with the <code><CR></code> key)
HELP	Displays explanatory text regarding the task at hand.
MARK	Lets you select multiple items within a <code>CHOICES</code> menu
NEXTRM	Moves you to the next frame
NEXTPAGE	Displays the next page of information
PREVFRM	Moves you to the previous frame.
PREVPAGE	Displays the previous page of information
RESET	Restores the default value to the current prompt
SAVE	Saves the answers currently displayed, and either displays the next form (if there is one) or performs the task

Alternate Keystrokes

If function keys are not provided or cannot be used with your keyboard, perform these functions by entering special character sequences called “alternate keystrokes.” An alternate keystroke is entered as follows:

- press `<CTRL><f>` (hold down the key marked `<CTRL>` or `<CONTROL>` while pressing `<f>`)
- press the number corresponding to the desired function key

For example, suppose you know you can get help by pressing the F1 function key, but you do not have function keys. Enter <CTRL><f> and then press 1. (Do not try to enter an alternate keystroke by pressing all three keys at once.)

Automatic Function Key Downloading

The **sysadm** interface was built with the FMLI (Form and Menu Language Interface) tool. FMLI relies heavily on the use of a terminal's keyboard function keys F1 through F8 but these keys are not available on some terminals or are not designed so the OS **terminfo(4)** database can work with them cleanly. To help in these situations, FMLI provides alternate keystrokes (discussed above). Sometimes, however, users of **sysadm** do not know about alternate keystrokes, and are frustrated in their attempts to use the interface. A partial fix to this problem is available, however.

If a terminal, as defined in its **terminfo** entry, does not have default escape sequences for its function keys but can download strings into its function keys, then FMLI attempts to download the alternate keystrokes into the function keys. This occurs, by default, when **sysadm** is first invoked, and each time the user returns from any full-screen activity. For terminals not fitting this description, no downloading is done.

For an experienced user who may have already placed other strings into the function keys, this is inconvenient, as the downloaded alternate keystrokes replace the user's own function key definitions, and FMLI cannot restore the user's automatically.

Using the LOADPFK Environment Variable to Disable Downloading

The environment variable **LOADPFK** can be used to disable this downloading. Setting **LOADPFK=NO** in the environment before invoking **fml i** (and thus **sysadm**) prevents this downloading from occurring at any time.

Using a Shell Script to Restore Function Key Settings

Users who have entered strings in their function keys on this type of terminal can use the **tput** utility, as shown in the following shell script, to download their personal function key strings:

```
tput pfx 1 'string-for-function-key-1'
tput pfx 2 'string-for-function-key-2'
tput pfx 3 'string-for-function-key-3'
tput pfx 4 'string-for-function-key-4'
tput pfx 5 'string-for-function-key-5'
tput pfx 6 'string-for-function-key-6'
tput pfx 7 'string-for-function-key-7'
tput pfx 8 'string-for-function-key-8'
```

Downloading FMLI Sequences More Efficiently

Even when a user wants the FMLI sequences downloaded, it is more efficient to do it only once if no program run from the FMLI application changes the function key settings (most programs do not); thus, any possible delays required to download the sequences will occur just once, instead of every time a full-screen activity is run. This can be done using **tput**,

To get a command line prompt, enter <CTRL><j> or <CTRL><f><c>. When you enter the control character, an arrow prompt (-->) is displayed on the command line at the bottom of the **sysadm** window. (Both control character sequences are described in “*Alternate Keystrokes*” earlier in this chapter.)

You can also move among frames by using the command line, then typing the frame number (found in the upper left-hand corner of each frame) and pressing <CR>.

Help for Forms: CHOICES

For some questions in a form you can display the valid answers by pressing **CHOICES**. One of two things will happen:

If there are only two possible answers to the current question, the interface displays an answer after the prompt in the form. You can use that answer, or you can press **CHOICES** again, to have the second answer overwrite the first answer. You can toggle back and forth between them by pressing **CHOICES** repeatedly.

If there are three or more possible answers to the current question, the interface will display a “pop-up menu” from which you can select. Screen 14-2 shows a sample pop-up menu: the menu of valid specifications for a time zone. It is displayed when you are setting the time zone for the system.

```

4 Available Timezones
> Greenwich (GMT & GDT)
  Atlantic (AST & ADT)
  Eastern (EST & EDT)
  Central (CST & CDT)
  Mountain (MST & MDT)
  Pacific (PST & PDT)
  Yukon (YST & YDT)
  Alaska (AST & ADT)
  Bering (BST & BDT)
  Hawaii (HST)

```

Screen 14-2. Sample Pop-Up CHOICES Menu: Valid Time Zones

When you select an item from a pop-up menu, that value is entered after the current prompt in the form you are using.

Getting Help

When you do not know what to do next, press the **HELP** key. In a menu, you will get a help message for the current menu item (which is highlighted). In a form, you will get a help message for the current question.

Using Express Mode

The **sysadm** “express mode.” saves time by allowing access to a menu or task directly from the operating system shell command line. Simply type **sysadm**, followed by the name of the desired menu or task. For example, to change a user's password, and when you know the task called **password** on the **Users** menu will let you do that, type **sysadm password** to access that task. Because the name of this task (**password**) is unique within the menu interface, the form for that task will be displayed on the screen.

If, however, you specify a menu or form that does not have a unique name, the interface will display a menu of matching items, from which you can select the one you want.

If you know the task you want does not have a unique name, but you know its location in the menu system, you can request it by specifying its menu pathname. For example, to access the **add** task from the **User and Group Management** menu (abbreviated as **users**), enter the following:

```
sysadm users/add
```

If the task name cannot be located, the system displays a message saying so.

Keep in mind, when you use express mode, you cannot access any menus or task forms that appear higher on any menu in the menu tree than the menu you originally accessed. (For a chart showing the entire menu tree see “*System Administration Menu Map*” later in this chapter)

A Sample Session: Adding an Account for a New User

This section shows you how to add an account for a new user to your system.

Step 1: Access the Menu Interface

Type **sysadm**. You will be prompted to enter your terminal type and password. Then the system administration main menu will appear in a frame in your terminal window, as shown in Screen 14-3.

The type of terminal will affect the layout of the menus on your screen; more menu items will be immediately visible on some types of terminals than on others. The types of software installed affects the entries you see on the main menu. For example, if you have installed one or more application packages on your system, the main menu will include an entry called **applmgmt**. This entry will not appear on your main menu if no application packages are installed.

All the entries on the main menu represent other menus. The entries on these second-level menus (and the menus that can be accessed through them), however, include both menus and tasks.

```

1          Operating System Administration

backup_service - Backup Scheduling, Setup, and Control
file_systems  - File System Creation, Checking and Mounting
machine       - Machine Configuration, Display and Shutdown
network_services - Network Services Administration
ports        - Port Access Services and Monitors
restore_service - Restore From Backup Data
schedule_task - Schedule Automatic Task
security     - Security Management
software     - Software Installation and Removal
storage_devices - Storage Device Operations and Definitions
system_setup - System Name, Date/Time and Initial Password Setup
> users      - User Login and Group Administration

[HELP] [ ] [ENTER] [PREV-FRM] [NEXT-FRM] [CANCEL] [CMD-MENU] [ ]

```

Screen 14-3. System Administration Main Menu

NOTE

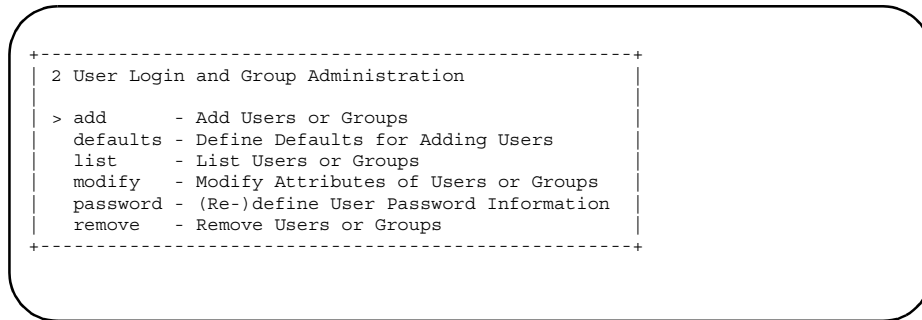
If your attempt to access a task fails, the cause may be corrupted, missing, or improperly installed **Menu.*** files. To rebuild these, run **/usr/sadm/install/bin/ie_build**. If you have installed and are running the Enhanced Security Utilities, you must be in single-user mode before you can run the **ie_build** command. (See the “Security Administration” part of *System Administration*, Volume 1, for details about single-user mode.)

Step 2: Select a Menu

To add a login account for a new user, select the User Login and Group Administration menu (abbreviated as **users**). There are two ways to get to this item:

- Use the down arrow key to position the cursor on the **users** menu item.
- Enter the first letter or letters of the **users** menu entry: **u**. This method of menu navigation requires typing as many letters as needed to uniquely identify the item. Here, only “u” is typed because that is all you need to identify **users**.

Screen 14-4 shows the display of the **users** menu.

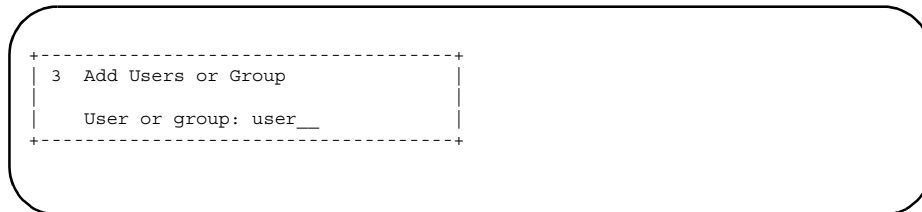


Screen 14-4. Step 2: the Menu Selected from the Main Menu

Step 3: Select a Task

To select the task that allows you to add an account for a new user, navigate to the add task the same way you selected a menu item in Step 2:

After you select this task, you will see a form in a new frame.



Step 4: Fill Out the Forms

The frame displayed when you select a task is called a task form. A task form is an electronic questionnaire that you must fill out.

When you select add a task form containing one question appears. Only two answers are possible (add a new user or add a new group). The `user` choice is already filled in, because it is the default selection. In this case, the default is the correct choice because you want to add a user to the system. Press the `<SAVE>` function key (or the alternate keystrokes `<CTRL><f3>`) so the default value will be accepted as input for this task.

The interface responds by presenting a detailed form for you to enter information about the new user. Note that some of the questions have a default response already filled in, and some do not.

```

+-----+
| 4           Add a User                               |
+-----+
| Comments: _____                               |
| Login: _____                                   |
| User ID: 100 _____                             |
| Primary group: other _____                   |
| Supplementary group(s): _____               |
| Home directory: /home/_____                    |
| Create home directory? no _____             |
| Shell: /usr/bin/sh _____                    |
| Number of days of login inactivity after which   |
|   user cannot log in: 0 _____               |
| Login expiration date: _____                |
+-----+

```

Fill out the form and then press **SAVE** to enter the information on the form. Note that three of the fields are mandatory, and the interface will not allow you to proceed without entering a value in them: **Comments**, **Login**, and **Login expiration date**.

The interface accepts this information and then presents a third form, prompting you for information used in assigning a password to the new user. Note that some of the questions have a default response already filled in, and some do not.

```

+-----+
| 5   Defines User Password Information                |
+-----+
| Password status: lock _____                  |
| Maximum number of days the password is valid: 24  |
| Minimum number of days allowed between            |
|   password changes: 0 _____                 |
| Number of days for warning message: 1 _____ |
+-----+

```

Invoke the **CHOICES** menu (by pressing the **CHOICES** key until the password choice appears) and select **password** to assign a temporary password for the new user. The word **password** will replace the word **lock** on the **Password status:** prompt line.

When you have finished filling out the password information form, the task changes to full window mode (explained in a previous section), and the following prompts appear:

```

New password:
Re-enter new password:

```

Enter a password after the prompt and press **<CR>**. The password you assign will be valid only until the first time the new user to whom it's assigned logs in. At that time, the user will be prompted to select another password.

After you re-enter the temporary password, the system will confirm the password information. Press **<CTRL>** to return to the **Add a User** form. Now, you can add another user or **CANCEL** out of the task.

Step 5: Exit from the Interface

To exit from the interface and return to the OS shell, invoke the Command Menu by pressing the CMD-MENU function key or by entering <CTRL><f7>. Select the **exit** command. The screen will be cleared and a shell prompt will appear.

System Administration Menu Map

Administrative tasks are grouped together under entries on the main menu, shown below

```
applmgmt      - Administration for Available Applications
backup_service - Backup Scheduling, Setup and Control
file_systems  - File System Creation, Checking and Mounting
machine       - Machine Configuration, Display and Shutdown
network_services - Network Services Administration
ports         - Port Access Services and Monitors
printers      - Printer Configuration and Services
restore_services - Restore from Backup Data
schedule_task - Schedule Automatic Task
security      - Security Management
software      - Software Installation And Removal
storage_devices - Storage Devices Operations and Definitions
system_setup  - System Name, Date/Time And Initial Password Setup
users         - User Login and Group Administration
```

The following table lists the menus on the system administration main menu and all the subsidiary menus and tasks available under each entry. Some entries, such as applmgmt will vary, depending on what is installed on your system.

Main Menu Item	Item or Task	Item or Task	Item or Task
network_services	attr_map	add list remove mappings	add list remove check fix
	basic_networking	devices polling systems	add list remove add list remove add list remove
	cr1	add modify remove set start stop	
	name_map	add list remove mappings	add list remove enable disable check fix
	remote_files	local_resources remote_resources setup specific_ops	list modify share unshare list modify mount unmount nfs rfs nfs rfs
network_services (continued)	selection name_to_address	display modify loopback inet	services hosts services

Main Menu Item	Item or Task	Item or Task	Item or Task
ports	port_monitors	add disable enable list	all by pmtag by type
	port_services	modify remove start stop add disable enable list modify remove	add to one add to many all by pmtag by type
printers	classes	add list modify remove	
	filters	add list modify remove restore	
	forms	add list modify remove	
	operations	accept control disable enable mount reject set_default umount	

Main Menu Item	Item or Task	Item or Task	Item or Task
printers (continued)	printers priorities requests status systems	add list modify remove default list remove system users cancel hold move release forms printers requests wheels add list modify remove	
restore_service	basic extended	pers restore system restore <i>Installation dependent</i>	Restore files under / Selective restore files under / Restore system Selective sys restore
schedule_task	add change delete display		
security	<i>Installation dependent</i>		
software	check defaults install interact list read_in remove	installed original spooled add list modify remove	

Main Menu Item	Item or Task	Item or Task	Item or Task
storage_devices	add copy devices display erase format groups remove systems	add attributes list remove reservation diskette mdens qtape mdens add list membership remove disk add list modify remove	add list modify remove free list reserve add list remove
system set-up	datetime file_maintenance nodename password setup	display set display set	
users	add defaults list modify password remove		

Quick Reference to the sysadm Interface

Function Keys

The main tool for manipulating the interface is a set of eight function keys. Labels highlighted in reverse video at the bottom of the screen show the function assigned to each; the functions assigned to some keys change for different types of frames, but F1 is always mapped to HELP.

If your function keys do not seem to work, you can simulate them using the two-character sequences <CTRL><f1> through <CTRL><f8>. The CANCEL function key dismisses the current frame (except for the main menu, which cannot be canceled). The CMD-MENU function key provides a Command Menu of other useful commands.

Menus

To move between menu items, use the down arrow \Downarrow and up arrow \Uparrow keys. To select a menu item, use the <ENTER> key or the <ENTER> function key.

Forms

To move to the next field, use the <TAB> key or arrow keys. After filling in a form, press the <SAVE> function key to process the data entered.

Text Frames

A text frame contains more than one logical page of text if the scroll bar on the right frame border contains a caret \wedge at the top or a \vee at the bottom; use the <NEXTPAGE> and <PREVPAGE> function keys to move between these pages.

Command Line

To go to the command line, use the <CTRL><j> or <CTRL><f><c> character sequence. Any command from the Command Menu can be typed directly here; press <ENTER> to process the command and return to the current frame. Use the **refresh** command to redraw a corrupted screen and the **cleanup** command to dismiss most frames from a cluttered screen.

Exiting from sysadm

To exit from the **sysadm** interface, press the <CMD-MENU> function key and select the **exit** item, or go to the command line and type **exit** <ENTER>. (The <CANCEL> function key is not equivalent to **exit**.)

Customizing the sysadm Interface

Overview of Customization	15-1
The Structure of the Menu Interface	15-1
Menu Modification Tools	15-2
Planning Interface Modifications	15-3
Selecting New Menus and Tasks	15-3
Naming New Menus and Tasks	15-4
Collecting Information for the Definition Form	15-4
Naming Requirements	15-5
How the System Handles Naming Collisions	15-5
Writing Help Messages	15-6
Item Help File Entries	15-7
The Menu Item Help Message Format	15-7
The Default Title Format	15-8
The Field Item Help Message Format	15-8
The Title Hierarchy	15-9
Setting Up for Item Help in an FMLI Object	15-10
Example Item Help Files	15-10
Creating or Changing a Menu Entry	15-10
Creating a Menu Entry	15-11
Changing a Menu Entry	15-13
Testing Your Menu Changes On-Line	15-15
The Menu Definition Form	15-15
Creating or Changing a Task Entry	15-16
Creating a Task Entry	15-16
Changing a Task Entry	15-18
The Task Definition Form	15-19
Deleting a Menu or Task Entry	15-20

Customizing the sysadm Interface

Overview of Customization

NOTE

If your system is being run in compliance with the security criteria described in Part 2, “Security Administration” of the “*System Administration*” Volume 1, no modifications should be made to the **sysadm** interface, which is part of the Trusted Computing Base. See the chapters under Part 2, “Security Administration” of Volume 1 for more information on the concept of the Trusted Computing Base.

The operating system provides a menu interface to most common administrative procedures. Because it's invoked by running the **sysadm** command, it's referred to as “the **sysadm** interface.” (Instructions for using this interface are provided in “Using the sysadm Interface,” in this manual.

This interface can be modified to suit your environment by changing or adding to the menus with the **edsysadm** and **delsysadm** commands. Instructions for using these commands are provided here.

The Structure of the Menu Interface

The **sysadm** interface is a hierarchy of menus. At the top of the hierarchy is the main menu (labeled Operating System Administration). It appears on the screen (as shown below), when you invoke **sysadm**.

NOTE

The **applmgmt** menu will not appear on the main menu unless at least one menu or task has been placed under it.

```
1          OS Administration

applmgmt   - Administration for Available Applications
backup_service - Backup Scheduling, Setup and Control
file_systems - File System Creation, Checking and Mounting
machine    - Machine Configuration, Display and Shutdown
network_services - Network Services Administration
ports      - Port Access Services and Monitors
printers   - Printer Configuration and Services
restore_service - Restore From Backup Data
software   - Software Installation and Removal
storage_devices - Storage Device Operations and Definitions
system_setup - System Name, Date/Time and Initial Password Setup
users      - User Login and Group Administration
```

The main menu is a list of function-specific menus. The left-hand column notes the menu names (such as **machine**); the right-hand column gives descriptions of these menus. Each menu offers other menus and/or names of tasks. For example, the **machine** menu, shown below, contains one menu (**configuration**) and four tasks.

```
Machine Configuration Display and Powerdown

configuration - System Configuration Display
shutdown      - Stops All Running Programs and Halts Machine
reboot        - Stops All Running Programs and Reboots Machine
whos on       - Displays List of Users Logged onto Machine
```

Choosing the entry **configuration** from this screen will cause another menu to be presented. Choosing a task entry, such as **shutdown**, will begin execution of that task.

Menu Modification Tools

You can modify the **sysadm** menu interface in three ways:

- make changes to existing menus
- add new menus (or tasks)
- remove menus (or tasks) that were previously added. (Do not remove menus or tasks included in the menu interface as delivered.)

When adding a new item—whether menu or task—you must also add a “help item file”: a set of messages that will appear on a user's screen to provide assistance with that menu or task.

When adding a new task, you must first have the executable files for it. Executable files are the programs that will be run when a user chooses this task. These programs may be commercial applications or your own programs. tasks, see the instructions in

When you have all the files needed to support the modified menu, you can start using the menu modification tools: **edsysadm** and **delsysadm**.

edsysadm allows you to make changes or additions to the interface. It functions much like the **sysadm** command itself, displaying a series of prompts for information. After you respond to all the prompts, you must either fill out a new “definition form” (for menus or tasks your are adding) or edit an existing “definition form” (for menus or tasks your are changing).

For every new menu or task you add to your interface, you must also create an “item help” file, containing one or more help messages.

delsysadm removes any menu or task added to the interface with **edsysadm**, after verifying that no packages depend on the item being removed. (A dependency exists if the menu being removed contains an entry added by an application package.) If dependencies are found, **edsysadm** prompts you to choose between quitting the deletion process and removing the menu (or task) and continuing.

Planning Interface Modifications

Before invoking **edsysadm**, plan your interface modifications by deciding if the planned changes are permissible (that is, determine if changes can be made without damaging the existing menu structure). If you are adding new tasks, decide where the tasks fit in the interface, choose names for the tasks, and determine the full menu structure needed to support them.

Selecting New Menus and Tasks

The criteria for new tasks are simple. Any type of task can be added to the **sysadm** interface with the following two restrictions:

- Tasks that can be automated should not be added to the interface. For example, do not add procedures that can run automatically as part of system booting or as part of package installation.
- Although the system allows it, it is strongly recommended that you do not add tasks to the interface that require the system to be in firmware mode.

To decide where to put modifications, you should become familiar with the organization of the **sysadm** menus. See “*Overview of Customization*” for a description of the menu structure.

Next, organize the administrative tasks you plan to add into an appropriate structure:

1. Choose the tasks to be added to the interface.

You can add any number of tasks. Create separate entries for each task to be performed. For example, if you are going to add software for adding, removing, and making changes to a log, create an entry for each of these tasks instead of offering all three under a single entry called `log administration`.

2. Select a menu where new tasks can be found.

You can create new **sysadm** menus at any level and change or add to any of the original **sysadm** menus. Be aware, however, that changing an original menu may cause problems in the execution of standard **sysadm** operations. Whenever you add a new menu, you should place new menus and/or tasks under the **applmgmt** menu (located on the main menu) or under a new menu that you added to the **main** menu.

3. Organize your tasks.

Organize your tasks under one menu or place them in submenu groups. For example, if you are adding tasks needed to support a package called `pkgA` and these tasks must be performed daily and weekly, you can create a structure such as the following:

- Under the **applmgmt** menu on the **main** menu, add an entry called `pkgAadmin`.
- Under `pkgAadmin`, add two submenus called **daily** and **weekly**.
- Under the submenu **daily**, add entries for each of the daily tasks.
- Under the submenu **weekly**, add entries for each of the weekly tasks.

It is important to plan your full administrative structure before running **edsysadm** because you must create a menu entry before placing a task or submenu under it.

After planning your structure, select names for your menus and tasks.

Naming New Menus and Tasks

Naming a new menu or task involves selecting both a name and a location for the new item, and creating some descriptive text that will appear on the menu where the new entry is listed. When you select a name, keep in mind the naming requirements listed below and the consequences, discussed below in “*How the System Handles Naming Collisions*”, of giving the same name to two menu entries.

Collecting Information for the Definition Form

As part of the interactive process begun by invoking **edsysadm**, you must fill out a menu (or task) definition form. Prepare the following information before filling out the form.

name	The name of the menu or task as it will appear in the left-hand column of the screen.
description	The one-line description of the menu or task as it will appear in the right-hand column of the screen.
location	The “pathname” of a menu (or task) in the <code>sysadm</code> menu hierarchy: that is, a list of the menus and/or tasks traversed to reach the specified item. Elements of the pathname are specified by colons. For example, the <code>shutdown</code> task is under the menu <code>machine</code> , which, in turn, is under the <code>main</code> menu. Thus, the location of the <code>shutdown</code> task is <code>main:machine:shutdown</code> .

All elements of a pathname must be existing menus or tasks. For example, when you add a task with a location of `main:applmgmt:mypkg`, you must already have created an entry for the menu `mypkg`.

Although you can make changes or additions to any of the original `sysadm` menus, you should make a new menu when you want to make a change. (If you make the change to the original menu, you may introduce errors into the execution of standard `sysadm` operations.)

Store the menus for any application packages you install in the menu location `main:applmgmt`. This menu location is empty when your system is delivered. `applmgmt` does not appear on the main menu until a menu is installed in it.

Naming Requirements

A menu or task name should be as short as possible but, it should also be descriptive. Menu and task names can contain only lowercase letters and underscores and a maximum length of 16 characters.

The description field can contain any character string with a maximum length of 58 characters. The text for a menu's description field is also used as the title for that menu when it is displayed. Follow the standard rules for capitalization in titles described in the next section.

How the System Handles Naming Collisions

If you assign a package a name that is already used for another package, the following menu will appear:

```
do not install
install
relocate
rename
```

If you are installing an updated version of an existing package, select **install**. The existing package will be overwritten.

If you are installing an entirely new package, select **relocate** or **rename** and provide any necessary information at the appropriate prompt. The renaming option adds the first available numerical suffix (beginning with 2) to the redundant name. For example, if you are trying to install a package called **menuA** and an unrelated package of the same name is already installed, select **rename** and your new package will be called **menuA2**.

Writing Help Messages

The **edsysadm** command changes the interface structure to accommodate the changes specified in the definition form. As part of this process, **edsysadm** reserves a space for a file called **Help-in** in the appropriate location—for any menu or task you are going to add. For every modification made to the interface, you must create an item help file: a file containing either or both of the following types of messages:

abstracts	messages that provide help, on request, for a user looking at a particular parent menu
form items	messages that explain how to fill out individual fields in an FMLI (Forms and Menus Language Interface) form for a particular task

If your help file is not called **Help**, **edsysadm** will rename it when you install it in the interface. Therefore, you can give an item help file any name, but it must be in your directory. This naming flexibility is useful for working on more than one item help file at the same time. You can:

- create one item help file for each task
- combine all your help messages for multiple tasks into one file
- use the same message for multiple FMLI forms
- define a title hierarchy for the help message screens.

Because an item help file is required for every menu and task, you must specify one on the definition form. If you specify a file that cannot be found, **edsysadm** helps avoid further delay in creating it—by invoking an editor on your screen for a file called **Help**.

Item Help File Entries

There are three types of entries in an item help file:

- a menu item help
- a default title (which can define both a global default and a form default)
- a field item help

The following three sections describe these types.

The Menu Item Help Message Format

A menu item help message is the text displayed on the screen when a user has placed the cursor on a particular menu (or task) item and requested help. Messages of this type must be written for both menu and task entries. For example, if you add a menu under `main:applmgmt` which menu has three tasks under it, you must deliver four menu item help messages.

Write each menu item help message in the following format:

```
[menu_name:]ABSTRACT:
TAB line 1 of message text
TAB line 2 of message text
TAB line n of message text
```

Here, *menu_name* is the name of the menu (or task) to which this help message belongs. This name appears in the left-hand column of the menu. (See “*Naming New Menus and Tasks*” for details.) *menu_name* is not optional when an item help file contains more than one menu item help definition.

Enter the text of a help message beneath the header line. A message can contain multiple lines of text with a maximum length of 69 characters per line. Each line must begin with a tab character. Blank lines may be included within the message as long as they also begin with a tab. The following is an example of a menu item help message:

```
cust_backups: ABSTRACT:
    This menu provides short-cuts to doing regular
    backups requested by specific departments in this
    building.

    For short-cuts to doing remote backups requested
    by staff in other buildings, see rem_backups.
```

The title for a menu item help message is always the one-line description of the menu or task that appears in the right-hand column of the menu display, prepended by the string `Help on`.

The Default Title Format

You can define two types of default titles:

- a global default title to be used on all the help messages defined in the item help file
- a form default title to be used on all the help messages defined for a particular form in an item help file with messages defined for numerous forms

Defaults can be overridden, see “*The Title Hierarchy*”. A default title definition is recommended, but not required.

The following is the format for the default title definition:

```
[form_ID:]TITLE:Title_Text
```

form_ID is the name of the form as defined with `lininfo` in your form definition. When a *form_ID* is supplied, this line defines a form default title. When it is not supplied, this line defines a global default title.

The title text defined after the `TITLE` keyword will have the string `HELP on` prepended to it when displayed. Keep this in mind when writing the title.

An example form default title definition is shown below.

```
task1:TITLE:Package Administration Task1
```

If `task1` was not added before `TITLE`, this example would define a global default title. The title defined by the example above will be displayed as:

```
HELP on Package Administration Task1
```

The Field Item Help Message Format

The field item help message will appear whenever a user requests help from within an FMLI form. Each field on the form must have a help message defined in the item help file.

The format for the field item help definition is as follows:

```
[form_ID:]field_ID:[Title_Text]  
TAB line 1 of message text  
TAB line 2 of message text  
TAB line n of message text
```

form_ID is the name of the form as defined with `lininfo` in your FMLI form definition. When one item help file contains messages for multiple tasks (and so multiple forms), it is used to distinguish the form with which a field belongs. It is optional if the file contains messages for only one task. **field_ID** is the name of the field as defined with `lininfo` in your FMLI form definition. *Title text* defines a title used only with the help message for this field. As with the default title, the text defined here will have the string `HELP on` prepended to it when displayed.

The message text should be entered beneath the header line. There can be multiple lines of text with a maximum length of 69 characters per line. Each line must begin with a tab

character. Blank lines may be included in the message, but they must begin with a tab character. An example field item help definition is shown below.

```
task1:fld1:the Name Field
  This is the text for a field item help for a name
  field.

  The preceding line will appear as a blank line
  when the help message is shown because it begins
  with a tab.
```

The title for this field item help message, as defined above, will be

```
HELP on the Name Field
```

The Title Hierarchy

You can define a global default title, a form default title, and a field title in an item help file. When all three are defined in the same file, use the following rules:

- The global default title is used for any message defined in an item help file that does not have a form default title or field title.
- The form default title is used for any message defined in an item help file and associated with the form, unless it has a field title.
- The field title is used only for the one help message for which it is defined.

If no field title is defined, the form default title is used. If no form default title is defined, the global default title is used. You should always have at least a global default title defined; otherwise, the string `HELP on` will be displayed with no descriptive text.

To define a global default title, add a line to the item help file using the following format:

```
TITLE:Title_Text
```

where *Title_Text* is the text for the global default title.

To define a form default title, add a line to the item help file using the following format:

```
form_ID:TITLE:Title_Text
```

where *form_ID* is the name of form as defined with `lininfo` in your FMLI form definition and *Title_Text* is the text for the form default title.

To define a field title, use the following format for the field item help header line:

```
form_ID:field_ID:Title_Text
```

where *form_ID* is the name of the form as it is defined with `lininfo` in your FMLI form definition, *field_ID* is the name of the field as it is defined with `lininfo` in your FMLI form definition and *Title_Text* is the text for the field title.

NOTE

In all cases, the text defined as *Title_Text* is prepended with HELP on when displayed to a user.

Setting Up for Item Help in an FMLI Object

To help the interface read item help file and recognize the associated forms and fields for a help message, the `help` and `lininfo` descriptors in the FMLI object definition must be defined as follows:

- The `help` descriptor must be defined exactly as shown on the line below:

```
help=OPEN TEXT $INTFBASE/Text.itemhelp $LININFO
```

- The `lininfo` descriptor for each field must be defined as

```
lininfo=[form_ID:]field_ID
```

where *form_ID* and *field_ID* are names no longer than 30 characters. The names defined here as *form_ID* and *field_ID* must match those used as *form_ID* and *field_ID* in the item help file.

NOTE

Because no FMLI form definition is created for a menu entry, no setup actions are required. Be certain that the *task_name* keyword precedes the ABSTRACT heading line for a menu entry help message.

Example Item Help Files

This section shows two example item help files. Screen 15-1 shows an item help file that defines messages for only one form. Screen 15-2 shows an example of defining messages for multiple forms in one item help file.

Creating or Changing a Menu Entry

The procedures for creating a new menu and for changing an existing one are similar and both result in the display of a menu definition form. Each procedure is described below, followed by a description of the menu definition form.

```

ABSTRACT:
  The text defined here will be shown to
  users when they request help while
  viewing the parent menu for this
  task. The task name is "adding users."

TITLE:Adding Users

field1:
  The text defined here will be shown to
  users when they request help from the
  form and the cursor is positioned at
  field1. The title for this message will
  be "HELP on Adding Users" as defined above.

field2:Field 2
  The text defined here will be shown to
  users when they request help from the
  form and the cursor is positioned at
  field2. The title for this message will
  be "HELP on Field 2".

Note: The lininfo descriptors in the form definition associated with this file should
look like this:

.
.
.

lininfo=field1

.
.
.

lininfo=field2

```

Screen 15-1. Item Help File for One Form

Creating a Menu Entry

Before creating a new menu entry:

- select a name and description for the menu
- select a location in the interface
- prepare an item help file for the menu entry; see *“Writing Help Messages”* earlier in this chapter

Complete the following procedure:

1. Type **edsysadm** and press <CR>.

NOTE

If you do not execute this command from the directory where the item help file resides, supply the full pathname when prompted for the name of the item help file.

```
add:ABSTRACT:
  The text defined here will be shown to
  users when they request help while
  viewing the parent menu for the task
  named add.

add_user:TITLE:Adding Users

add_user:field1:
  The text defined here will be shown to
  users when they request help from the
  form and the cursor is positioned at
  field1. The title for this message will
  be "HELP on Adding Users" as defined above.

add_user:field2:Field 2
  The text defined here will be shown to
  users when they request help from the
  form and the cursor is positioned at
  field2. The title for this message will
  be "HELP on Field 2".

delete:ABSTRACT:
  The text defined here will be shown to
  users when they request help while
  viewing the parent menu for the task
  named delete.

delete_user:TITLE:Deleting Users

delete_user:field1:
  The text defined here will be shown to
  users when they request help from the
  form and the cursor is positioned at
  field1. The title for this message will
  be "HELP on Deleting Users" as defined above.

delete_user:field2:Field 2
  The text defined here will be shown to
  users when they request help from the
  form and the cursor is positioned at
  field2. The title for this message will
  be "HELP on Field 2".

Note: The lininfo descriptors in the form definition associated with this
file should look like this:

.
.
lininfo=add_user:field1
.
.
lininfo=add_user:field2
.
.
lininfo=delete_user:field1
.
.
lininfo=delete_user:field2
```

Screen 15-2. Item Help File for Multiple Forms

2. You are asked to choose between a menu and a task. Choose **menu** and press <CR>.
3. You are asked to choose between adding a new menu and changing an existing one. Choose **add** and press <CR>.
4. You are given an empty menu definition form. Fill it in and press <SAVE>. (See “The Menu Definition Form” for descriptions of the fields on this form.)
5. You are asked if you want to test the changes before actually making them. Answer *yes* or *no* and press <SAVE>. (If you answer *yes*, see “Testing Your Menu Changes On-Line,” below, to learn about the test.)
6. You are asked if you want to install the modifications in the interface on your machine or save them for a package. Choose **install** and press <SAVE>.
7. You will see one of the following:
 - A confirmation screen indicating success; you now have an empty menu. (To populate the menu entry just created, follow the procedure for adding a new menu entry or for adding a new task.)
 - An error message describing the reason installation was unsuccessful.
 - A statement that there is a problem with the name chosen for this menu. You will then be prompted to choose from four possible actions:

install	The new entry will be installed despite the collision of names.
rename	A numeric suffix will be added to the name of the new entry to distinguish it from an existing entry with the same name.
relocate	You will be prompted to select a new location for the new entry.
do not install	The new entry will not be installed (default).

Your changes will not be installed if you press <SAVE> without making a selection.

Changing a Menu Entry

Before changing a menu entry:

- determine the name and description of the menu entry
- determine its location in the interface
- change the associated item help file, if necessary, or create a new one. (See “Writing Help Messages”, earlier in this chapter, for instructions.)

Complete the following procedure:

1. Type **edsysadm** and press <CR>.

NOTE

If you have changed an item help file or created a new one and you do not execute this command from the directory in which the file resides, supply the full pathname when asked for the name.

2. You are asked to choose between a menu and a task. Choose **menu** and press <CR>.
3. You are asked to choose between adding a new menu and changing an existing one. Choose **change** and press <CR>.
4. You are asked if your change is for an on-line menu or a menu that has been saved for a package. Choose **on-line** and press <SAVE>.
5. You are asked to supply the name and location of the menu to be changed. Fill in the blanks and press <SAVE>.
6. You are given a menu definition form filled in with the current values for the menu named above. Make the desired changes and press <SAVE>. (See “*The Menu Definition Form*” for descriptions of the fields on this form.)
7. You are asked if you want to test the changes before actually making them. Answer **yes** or **no** and press <SAVE>. (If you answer **yes**, see “*Testing Your Menu Changes On-Line*,” below, to learn about the test.)
8. You are asked if you want to install the modifications in the interface on your machine or save them for a package. Choose **install** and press <SAVE>.
9. One of the following appears:
 - A confirmation screen indicating success; the menu has been changed.
 - An error message describing the reason installation was unsuccessful.
 - A statement that there is a problem with the name chosen for this menu. You will then be prompted to choose from four possible actions:

install	The new entry will be installed despite the collision of names.
rename	A numeric suffix will be added to the name of the new entry to distinguish it from an existing entry with the same name.
relocate	You will be prompted to select a new location for the new entry.

do not install The new entry will not be installed (default).

Your changes will not be installed if you press <SAVE> without making a selection.

Testing Your Menu Changes On-Line

Before installing your menu changes, verify that you added the entry to a menu. The **edsysadm** command gives you a chance to do this after you fill in the menu definition form. To perform your test:

1. Type yes when **edsysadm** presents the following prompt:

```
Do you want to test this modification before
continuing?
```

2. The parent menu (on which your addition or change is listed) is displayed. Make sure your modification has been made correctly.
3. Put the cursor on the new or changed menu entry and press <HELP>. The text of the help message for that menu entry is displayed so you can check it. (Press <CANCEL> to return to the menu.)
4. To exit on-line testing, press <CANCE>L.
5. You are returned to the prompt:

```
Do you want to test this modification before
continuing?
```

To continue executing the change, type no.

To make additional modifications to the menu definition form, press <CANCEL>. You are returned to the form and can make further changes. (Press <SAVE>. when you are done editing. Then, retest your changes or continue executing the change.)

The Menu Definition Form

You must provide values for the following fields on this form:

menu_name

The name of the new menu (as it should appear in the left-hand column of the screen). This field has a maximum length of 16 characters and should consist of alphanumeric characters.

menu_description

A description of the new menu (as it should appear in the right-hand column of the screen). This field has a maximum length of 58 characters and can consist of any alphanumeric character except an at sign (@), circumflex (^), tilde (~), single quote ('), back quote (`), and double quotes (").

menu_location

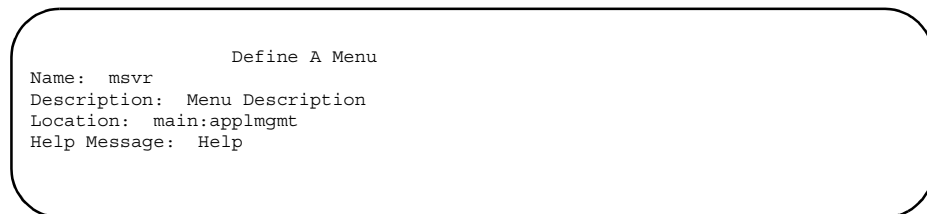
The location of the menu in the menu hierarchy expressed as a menu pathname. The pathname should begin with the main menu followed by all other menus that must be traversed (in the order they are traversed) to access this menu. Menu names must be separated by colons. For example, the menu location for a menu entry being added to the Applications menu is `main:applmgmt`. Do not include the menu name in this location definition. The complete path to this menu entry will be the menu location plus the menu name defined at the first prompt.

This is a scrollable field, showing a maximum of 50 alphanumeric characters at a time.

menu_help_file_name

Pathname to the item help file for this menu entry. If it resides in the directory from which you invoked `edsysadm`, you do not need to give a full pathname. If you name an item help file that does not exist, you are placed in an editor (as defined by `$EDITOR`) to create one. The new file is created in the current directory and named **Help**.

The following screen shows a sample menu definition form that has been filled in:



```
Define A Menu
Name: msvr
Description: Menu Description
Location: main:applmgmt
Help Message: Help
```

Creating or Changing a Task Entry

The procedures for creating a new task and for changing an existing one are similar and both result in the display of a task definition form. Each procedure is described below, followed by a description of the task definition form.

Creating a Task Entry

Before creating a task entry:

- Gather all files that will be associated with this task, such as the help files, FMLI forms, and other executables. All files should already be prepared.
- Decide on the task name and description.
- Decide on its location in the interface.

- Create an item help file. (See “Writing Help Messages” earlier in this chapter for instructions.)

Complete the following procedure.

1. Type **edsysadm** and press <CR>.

NOTE

If you do not execute this command from the same directory in which the files associated with this task reside, enter full pathnames when supplying filenames.

2. You are asked to choose between a menu and a task. Choose **task** and press <CR>.
3. You are asked to choose between adding a new task or changing an existing one. Choose **add** and press <CR>.
4. You are given an empty task definition form. Fill it in and press <SAVE>. (See “The Task Definition Form,” below, for descriptions of the fields on this form. Be aware that when you name the menu under which you want this new task to reside, that menu must already exist.)
5. You are asked if you want to install the modifications in the interface on your machine or save them for a package. Choose **install** and press <SAVE>.
6. You will see one of the following:
 - A confirmation screen indicating success; the new task is now in the menu and can be executed.
 - An error message describing the reason installation was unsuccessful.
 - A statement that there is a conflict with the name you have chosen for this task. You will then be prompted to choose from four possible actions:

install	The new entry will be installed despite the collision of names.
rename	A numeric suffix will be added to the name of the new entry to distinguish it from an existing entry with the same name.
relocate	You will be prompted to select a new location for the new entry.
do not install	The new entry will not be installed (default).

Changing a Task Entry

Before changing a task entry:

- Gather any new or changed files associated with the task. All files should already be prepared or changed.
- Know the menu name and description.
- Know its location in the interface.
- Change the associated item help file, if necessary. (See “*Writing Help Messages*” earlier in this chapter for instructions.)

Complete the following procedure.

1. Type **edsysadm** and press <CR>.

NOTE

If your change requires new files or changes to existing files and you do not execute this command from the directory in which the files reside, enter full pathnames when supplying filenames.

2. You are asked to choose between a menu and a task. Choose **task** and press <CR>.
3. You are asked to choose between adding a new task and changing an existing one. Choose **change** and press <CR>.
4. You are asked if your change is for an on-line task or for a task that has been saved for a package. Choose **on-line** and press <SAVE>.
5. You are asked to supply the name and location of the task to be changed. Fill in the blanks and press <SAVE>.
6. You are given a task definition form filled in with the current values for the task named above. Make the desired changes and press <SAVE>. (See “*The Task Definition Form*,” below, for descriptions of the fields on this form.)
7. You are asked if you want to install the modifications in the interface on your machine or save them for a package. Choose **install** and press <SAVE>.
8. You will see one of the following:
 - A confirmation screen indicating success: the change is in place.
 - An error message describing the reason installation was unsuccessful.
 - A statement that there is a conflict with the name you have chosen for this task. You will then be prompted to choose from four possible actions:

install	The new entry will be installed despite the collision of names.
rename	A numeric suffix will be added to the name of the new entry to distinguish it from an existing entry with the same name.
relocate	You will be prompted to select a new location for the new entry.
do not install	The new entry will not be installed (default).

The Task Definition Form

This form contains six fields in which you must provide the following information: a task name, a task description, a task location, the name of a help message for the task, a task action file, and the files associated with the task.

<i>task_name</i>	The name of the new task (as it should appear in the left-hand column of the screen). This field has a maximum length of 16 characters and should consist of alphanumeric characters.
<i>task_description</i>	A description of the new task (as it should appear in the right-hand column of the screen). This field has a maximum length of 58 characters and can consist of any alphanumeric character except an at sign (@), circumflex (^), tilde (~), single quote ('), back quote (`), and double quotes (").
<i>task_location</i>	The location of the task in the menu hierarchy, expressed as a pathname. The pathname should begin with the main menu followed by all other menus that must be traversed (in the order they are traversed) to access this task. Each menu name must be separated by colons. For example, the task location for a task entry being added to the Applications menu is <code>main:applmgmt</code> . Do not include the task name in this location definition. The complete path to this task entry will be the task location plus the task name defined at the first prompt. This is a scrollable field showing a maximum of 50 alphanumeric characters at a time.
<i>task_help_file_name</i>	Pathname to the item help file for this task entry. If it resides in the directory from which you invoked edsysadm , you do not need to give a full pathname. If you name an item help file that does not exist, you are placed in an editor (as defined by \$EDITOR) to create one. The new file is created in the current directory and named Help .

task_action The FMLI form name or executable that will be run when this task is selected. This is a scrollable field showing a maximum of 50 alphanumeric characters at a time. This pathname can be relative to the current directory as well as absolute.

task_files Any FMLI objects or other executables that support the task action listed above and might be called from within that action. *Do not include the item help filename or the task action in this list.* Pathnames can be relative to the current directory as well as absolute. A dot (.) implies “all files in the current directory” and includes files in subdirectories.

This is a scrollable field showing a maximum length of 50 alphanumeric characters at a time.

The following screen shows a sample task definition form that has been filled in:

```
Define A Task
Name: msvrtask
Description: Task Description
Location: main:applmgmt:msvr
Help Message: Help
Action: form.msvrtask
Task Files: form.task2, text.task2
```

Deleting a Menu or Task Entry

Modifications to the **sysadm** interface can be deleted in one of two ways: automatically, as part of the removal of a software package that contains modifications, or by request, with the **delsysadm** command.

To request the deletion of a menu (or task), enter

```
delsysadm pathname
```

where *pathname* is the location of the task (or menu) in the interface. For example, to delete a task named *mytask* from a menu called *mymenu* (found on the *applmgmt* menu, which is listed on the *main* menu), enter

```
delsysadm main:applmgmt:mymenu:mytask
```

Before an entry for a menu can be removed, that menu must be empty (that is, it may contain no other menus or tasks). If it is not, you must use the **-r** option with **delsysadm**. This option requests that, in addition to the named menu, all menus and tasks located under that menu be removed. For example, to remove *main:applmgmt:mymenu* and all menus and tasks that reside under it, enter

```
delsysadm -r main:applmgmt:mymenu
```

When you use the **-r** option, **delsysadm** checks for dependencies before removing any subentries. (A dependency exists if the menu being removed contains an entry placed there by an application package.) If a dependency is found, you are shown a list of packages that depend on the menu you want to delete and asked whether you want to continue. If you answer yes, the menu and all its menus and tasks are removed (even those shown to have dependencies). If you answer no, the menu is not deleted.

CAUTION

Use **delsysadm** to remove only those menu or task entries that you have added to the interface with **edsysadm**.

Modifying the sysadm Interface

Overview of sysadm Modification	16-1
Introduction to the Tools	16-1
The edsysadm Command	16-2
The delsysadm Command	16-2
The Data Validation Tools	16-3
Introduction to the Package Modification Files	16-3
Overview of the Interface Modification Process	16-4
Planning Your Interface Modifications	16-4
Deciding If You Should Modify the Interface	16-4
Planning the Location of Your Modifications	16-4
An Overview of the Interface Structure	16-5
Planning Your Administration Structure	16-5
Naming Your Interface Modifications	16-6
How to Name Your Modifications	16-6
Interface Naming Requirements	16-7
How the System Handles Naming Collisions	16-7
Writing Your Administration Actions	16-7
Privileged Command Execution in Task Action Files	16-8
Writing Your Help Messages	16-9
The Item Help File	16-10
The Menu Item Help Message Format	16-10
The Default Title Format	16-11
The Field Item Help Message Format	16-11
The Title Hierarchy	16-12
Setting Up for Item Help in an FMLI Object	16-13
Example Item Help Files	16-13
Packaging Your Interface Modifications	16-16
Basic Steps for Packaging Your Modifications	16-16
Creating or Changing the Packaging for a Menu Entry	16-16
Creating the Packaging for a Menu Entry	16-16
Changing the Packaging for a Menu Entry	16-17
Testing Your Menu Changes On-Line	16-19
The Menu Definition Form	16-19
Creating or Changing the Packaging for a Task Entry	16-20
Creating the Packaging for a Task Entry	16-20
Changing the Packaging for a Task Entry	16-21
The Task Definition Form	16-22
Preparing Your Package	16-24
Deleting Interface Modifications	16-25
Data Validation Tools	16-25
Types of Tools	16-26
Characteristics of the Tools	16-26
The Data Validation Tool Prompts	16-27
The Data Validation Tool Help Messages	16-27
The Data Validation Tool Error Messages	16-28
Message Formatting	16-28
The Shell Commands	16-28

The Visual Tools 16-29

Modifying the sysadm Interface

Overview of sysadm Modification

The operating system provides a menu interface to the most common administrative procedures. It is invoked by executing **sysadm** and so is referred to as the **sysadm** interface. (A complete description of this interface and instructions on how to use it can be found in Chapter 14, “*Using the sysadmin Interface*”).

You can deliver additions or changes to this interface as part of your application software package. Creating the necessary information for an interface modification is a simple process due to the tools provided by the operating system.

This chapter describes these tools, provides all of the needed background information, and details the procedures necessary to design and write your package administration and to package it so that it will become a part of the administration interface on the installation machine.

NOTE

This chapter assumes you are familiar with the material covered in the chapter entitled “*Packaging Your Software Applications*”, in the *Real-Time Programming Guide*.

Introduction to the Tools

Two commands can be used to create the files necessary to deliver modifications to the **sysadm** interface as a part of your package.

- **edsysadm** creates all of the files needed for your interface modifications to be installed along with your package
- **delsysadm** deletes menus or tasks from the interface

This chapter also provides an overview of a group of tools known as the data validation tools. You can use them when writing your system administration to simplify and standardize the programming of administrative interaction.

The edsysadm Command

edsysadm, which allows you to make changes or additions to the interface, is an interactive command that functions much like the **sysadm** command itself. It presents a series of prompts for information. (Which prompt appears depends on your response to the previous prompt.)

After you have responded to all the prompts, **edsysadm** presents a form that you must fill in with information describing the menu or task being changed or added. This form is called the menu (or task) definition form. If you are changing an existing menu or task entry, the definition form will already be filled in with the current values, which you can edit. If you are adding a new menu or task entry, the form will be empty and you will have to fill it in.

When you follow the procedures in this chapter, **edsysadm** creates all of the files and directories necessary to deliver your interface modifications as a part of your package. The section entitled “*Introduction to the Package Modification Files*” describes the three files that **edsysadm** creates.

edsysadm builds the directory structure required by the **sysadminterface**. You do not need to know this structure and you are not required to have your work directory organized in any predefined way. When you fill in a menu or task definition form, you supply the names of files that should be used by **edsysadm** in creating the packaging for your interface modifications. (For example, you should specify any files containing help messages.) **edsysadm** creates a **prototype** file and builds the interface directory format by using the path1=path2 naming convention. path2 defines the location of the files on your machine; path1, the location in which they should be placed on the installation machine.

The delsysadm Command

delsysadm removes tasks and menus from the interface. When you deliver your modifications as a part of your package, you do not need to use **delsysadm** to remove them. Any time an interface modification is delivered as a part of a package, those modifications are automatically removed at the same time as the package. This chapter describes the **delsysadm** command in case you need to use it on your own machine, for example, to remove modifications added for testing.

delsysadm checks for dependencies on the entry being removed before deleting the entry. (A dependency exists if the menu being removed contains an entry placed there by an application package.) If **delsysadm** discovers a dependency, you are asked whether you want to continue with the removal. (If a dependency is found during an automatic removal, the interface entry is not removed.)

When you delete a menu entry with **delsysadm**, that entry must already be empty (that is, it may contain no other menus or tasks). If it is not, you can run **delsysadm** with the **-r** option. This option removes a menu and all its entries at the same time.

CAUTION

Use **delsysadm** to remove only those menu or task entries that you have added to an interface.

The Data Validation Tools

The data validation routines help standardize administration interaction and also make development easier. The tools are available as shell commands and as visual modules to be used in an FMLI (Form and Menu Language Interpreter) form. The tools perform the following series of tasks:

- prompt a user for a particular type of input
- validate the response
- format and print help and error messages
- return the input if it passes validation

The type of validation performed is defined by the tool itself. For example, the shell command **ckyorn** prompts for and validates an affirmative or negative response. These tools should be used in your administration programs if they are to be added to the **sysadm** interface to maintain consistency within the interface.

Introduction to the Package Modification Files

When you execute **edsysadm** to define menus and tasks and save those definitions to be included in your application software package, it creates three files:

- the package description file
- the menu information file
- the **prototype** file

The package description file contains information **edsysadm** uses to change interface modifications already saved for packaging. When you decide to change your modifications after already creating the packaging (meaning the menu information and **prototype** files are already created), the package description file provides **edsysadm** with the information it needs to locate the other package modification files and to make the changes. Without this file, **edsysadm** cannot make such a change. You are asked to supply a name for this file during the **edsysadm** interaction and it is created in your current directory (unless you supply a full pathname to a different directory with the name).

The menu information file contains the menu or task name, where it is located in the interface structure, and, for tasks, what executable to use when the task is invoked. It tells the interface installation software how to modify the interface structures to include the new definitions. The file's name is the hour, minute, second, day-of-year, and year that the file was created, followed by a **.mi** suffix. It is created in your current directory.

The **prototype** file created by **edsysadm** contains entries for all of the interface modification components that must be packaged with your software (for example, the menu information file and, for tasks, the executables). These entries must be incorporated into your package either by reading the **edsysadm**-created file into your package **prototype** file or by using the **include** command in the main **prototype** file for your package. The **prototype** file created by **edsysadm** is created in your current directory with the name of **prototype**.

Overview of the Interface Modification Process

You must take a number of steps to add your package administration to the **sysadm** interface. This chapter explains each of these steps in detail.

- planning your package administration (with details on how to decide if you should modify the interface and where to place it in the interface structure)
- writing your administration actions (with general information on what your executables can be)
- writing your help message (with a description of the required help message file)
- packaging your interface modifications (with procedural details on executing **edsysadm** and what steps must be taken afterwards)

This chapter also includes instructions on executing **delsysadm**.

Planning Your Interface Modifications

You will need to plan your interface modifications before executing **edsysadm**. Planning begins with deciding if your administration tasks should become a part of the **sysadm** interface. If so, you must decide on where your tasks fit into the interface, what to name your tasks, and the full menu structure involved with your administrative tasks.

Deciding If You Should Modify the Interface

Any type of task can be added to the **sysadm** interface with the following two restrictions:

- Tasks that can be automated should not be added to the interface (for example, procedures that can run automatically as part of system booting or as part of your package installation).
- Tasks that require the system to be in firmware mode can be added to the interface but it is strongly recommended that they not be.

Once you have decided to add your administration tasks to the interface, you must determine where in the interface you want to locate tasks and menus.

Planning the Location of Your Modifications

To plan your modification you must first become familiar with the interface organization. Then you must decide how to organize the tasks you want to add and how to fit your modifications into the overall structure.

An Overview of the Interface Structure

The **sysadm** interface consists of a hierarchy of menus. At the top of the hierarchy is the main menu (labeled UNIX System V Administration). It appears on the screen, immediately after **sysadm** is invoked, as follows:

```

1          UNIX System V Administration

applmgmt   - Administration for Available Applications
backup_service - Backup Scheduling, Setup and Control
file_systems - File System Creation, Checking and Mounting
machine    - Machine Configuration, Display and Shutdown
network_services - Network Services Administration
ports      - Port Access Services and Monitors
preSVR4    - Peripherals Setup
printers   - Printer Configuration and Services
restore_service - Restore From Backup Data
software   - Software Installation and Removal
storage_devices - Storage Device Operations and Definitions
system_setup - System Name, Date/Time and Initial Password Setup
users      - User Login and Group Administration
    
```

NOTE

The **applmgmt** menu will not appear on the main **sysadm** menu until at least one menu or task has been placed under it.

The main menu consists of a list of function-specific menus. The left-hand column notes the menu names (such as **machine**) and the right-hand column gives descriptions of these menus. Each menu offers other menus and/or names of tasks. For example, the **MachineConfigurationDisplayandPowerdown** menu, shown below, contains one menu (**configuration**) and three tasks.

```

          Machine Configuration Display and Powerdown

configuration - System Configuration Display
shutdown      - Stops All Running Programs and Halts Machine
reboot        - Stops All Running Programs and Reboots Machine
whos on       - Displays List of Users Logged onto Machine
    
```

Choosing the entry **configuration** from this screen will cause another menu to be presented. Choosing a task entry, such as **shutdown**, will begin execution of that task.

Planning Your Administration Structure

Planning your administration structure requires three steps:

1. Deciding what tasks to add to the interface.

You can add any number of tasks. You should have separate entries for each task to be performed. For example, if your administration allowed a log to be changed, added to, and removed, you should create an entry for each task and not combine them into one entry called `log administration`.

2. Deciding under which menu the tasks should be placed.

You can create new `sysadm` menus at any level and you can change or add to any of the original `sysadm` menus. You should be aware, however, that if you make changes to original menus you might cause problems in the execution of standard `sysadm` operations. It is therefore recommended (though not mandatory) that you create new menus for your package administration by placing it under the `applmgmt` menu (located on the main menu) or by creating a new main menu entry.

3. Organizing your tasks.

You can organize your tasks under one menu or place them in submenu groups. For example, if your package has tasks to be performed daily and weekly, you might create a structure such as the following:

- Under the `applmgmt` menu on the main menu, add an entry for your package called `pkgAdmin`.
- Under `pkgAdmin`, add two submenus called `daily` and `weekly`.
- Under the submenu `daily`, add entries for each of the daily tasks.
- Under the submenu `weekly`, add entries for each of the weekly tasks.

It is important that you have your full administrative structure planned before running `edsysadm` because you must create a menu entry before placing a task or submenu under it.

After you have planned your structure, you should decide on the names for your menus and tasks.

Naming Your Interface Modifications

Naming your interface modifications requires the three pieces of information described below. This section also details the interface naming requirements and tells you how the system handles naming collisions.

How to Name Your Modifications

When naming your interface modifications, you must decide on these three pieces of information:

Name	The name of the menu or task as it will appear in the left-hand column of the screen.
------	---

Description	The description of the menu or task as it will appear in the right-hand column of the screen.
Location	<p>The location of a menu or task in the <code>sysadm</code> menu hierarchy. This location is a combination, step-by-step, of all the menu names that must be chosen to reach the menu or task. Each step must already exist when the entry is added. For example, when you add a task with a location of <code>main:applmgmt:mypkg</code>, you must already have created an entry for the menu <code>mypkg</code>.</p> <p>All locations begin with <code>main</code>. When defining a location in the procedures that follow, each step should be separated by a colon. For example, the <code>shutdown</code> task is under the menu <code>machine</code>, which, in turn, is under the <code>main</code> menu. Thus, the location of the <code>shutdown</code> task is <code>main:machine</code>.</p>

You will supply these pieces of information on the menu (or task) definition form.

Interface Naming Requirements

A menu or task name should be as short as possible in length but, at the same time, be descriptive. It can contain only lowercase letters and underscores and has a maximum length of 16 characters.

The description field can contain any character string and has a maximum length of 58 characters. This description field text for a menu is also used as the title for that menu when it is displayed. We recommend following standard rules for capitalization in titles.

How the System Handles Naming Collisions

A naming collision might occur under two circumstances:

- When the package being installed is an update to an existing version.

The administrator will be asked during installation if this is an update, in which case the existing menus and tasks will be overwritten.

- When two packages have created identical interface modifications.

The colliding menu or task will be renamed by adding the first available numerical suffix (beginning with 2). For example, if an entry for `menuA` already exists and a package attempts to add an identical entry, the one being added will be renamed to `menuA2`.

Writing Your Administration Actions

When you execute `edsysadm` to create packaging for a task entry, you will fill in a task definition form. One of the fields on that form asks for the name of the task action file. The task action file is the executable that will run when your task is selected from the

interface. Your administrative task can use more than one executable but, if it does, you must create one that is called when the task is selected and call any other executables associated with the task from within it.

The task action can be one of two types:

- Non-interactive

A non-interactive task action can be any shell executable.

- Interactive

An interactive task action must be an FMLI form.

Use the data validation tools whenever possible when writing administrator interaction.

Privileged Command Execution in Task Action Files

For a user to execute a particular privileged command, one of two criteria must be met:

- If the system is running a uid-based privilege module that imparts all privileges to a particular user id, a new process that has an effective user id equal to that privileged user id must be spawned to execute the command.
- If the system is running a file-based privilege module that requires discrete privileges for various sensitive system operations, there must be an entry in a trusted database that allows the user to execute the given command with appropriate privileges.

The first case above is the case in versions of the OS prior to Release 1.1, and in Release 1.1 and later systems running the Super User Module (SUM). Setting the set-user-id bit on an executable file allows any user that has access to the file the ability to run the file as the owner and perform sensitive operations (such as mounting a file system). Again, this requires the user to be logged in as a particular user (usually `root`).

The second case is true for OS Release 1.1 and later systems running the Least Privilege Module (LPM); this privilege mechanism does not depend on a particular user ID. (For details on different privilege mechanisms, see the chapter entitled “Administering Privileges” in Volume 1, *System Administration*.)

In order to alleviate the need to write applications twice for these two types of privilege mechanisms, the `$TFADMIN` environment variable can be used to execute the command that checks the database for appropriate entries for the user attempting to execute a particular command. If `$TFADMIN` is null or not set, then the command is executed without the prefixed value of `$TFADMIN`.

For example, a typical use of `$TFADMIN` to automatically check if the user has the privileges required to execute a command would look like the following:

```
$TFADMIN -t command [options]
```

The `-t` option specifies that only a check for the ability to execute the *command* is performed. If the requested operation succeeds (return code of 0), `tfadmin` can exe-

ecute the privileged command without error; otherwise, an error code of 1 is returned to indicate that the user cannot execute the privileged command.

The task action file should test for these return codes, and take the appropriate action. The following segment of commented shell code is an example of such a test case:

```
#
# Test to see if $TFADMIN is set
#
if [ $TFADMIN ]
then
#
# If it is, use $TFADMIN -t to see if current process has privilege(s)
# needed to execute the command (in this example, the mount command).
#
    if $TFADMIN -t mount
    then
#
# If current process has privilege(s) necessary, then execute the
# command (including options on command line); otherwise,
# print error message and exit.
#
        $TFADMIN mount options
    else
        echo "User does NOT have appropriate privileges"
        exit 1
    fi
else
#
# If $TFADMIN is not set, just execute the command.
# (Though not shown here, you will probably want to test the error
# code return of this command for errors and take appropriate action
# on error).
#
    mount options
fi
```

If you want your application to be compatible with both types of privilege mechanisms described above, use \$TFADMIN in your administrative action files as shown above.

Writing Your Help Messages

You must write help messages to be packaged with every interface modifications. They are delivered in what is called an item help file. This file has text for two types of messages:

- the help message that will be shown when the user requests help from the parent menu
- the help messages that will be shown for each field when your task action is an FMLI form

The format of the item help file allows you to create one item help file for each task, combine all of your help messages for multiple tasks into one file, use the same message for multiple FMLI forms, and to define a title hierarchy for the help message screens.

The Item Help File

There are no naming restrictions for the item help file that resides on your machine. However, within the interface structure, the item help file must always be named **Help**. You can use this name if you want to but it is not mandatory since **edsysadm** uses the *path1=path2* naming convention in the **prototype** file that it creates to define the directory structure required by the interface. Regardless of what the item help file is named on your machine, *path1* in the **prototype** file will have the name **Help**. This means you can have more than one item help file in your current directory at the same time and **edsysadm** will handle the details of giving it the correct name.

There are three types of entries in an item help file:

- the menu item help
- the default title (can define both a global default and a form default)
- the field item help

A description of each type of entry and its format follows. All of the entries use the colon (:) as the keyword delimiter.

The Menu Item Help Message Format

The menu item help message will be shown whenever a user requests help on an entry from the parent menu. Menu item help must be written for each menu and task entry being delivered as an interface modification. For example, if your package administration is adding a menu under `main:applmgmt` and that menu has three tasks under it, you will need to deliver four menu item help messages.

The format for the menu item help definition is as follows:

```
[task_name:] ABSTRACT:  
TAB Line 1 of message text  
TAB Line 2 of message text  
TAB Line n of message text
```

task_name defines the task (or menu) entry to which this help message belongs. This name must match the name that you have decided should appear in the left-hand column of the menu screen. (Refer back to “*Naming Your Interface Modifications*” for more details on this name.) *task_name* is not optional when more than one menu item help definition is defined in the same item help file. This helps to distinguish to which task or menu the message belongs.

The message text should be entered beneath the header line. There can be multiple lines of text with a maximum length of 69 characters per line. Each line must begin with a tab character. Blank lines may be included within the message as long as they also begin with a tab character. An example menu item help definition is shown below.

```
task1:ABSTRACT:  
This is line one of the menu item help message.  
This is a second line of message text.
```

The preceding line will appear as a blank line

when the help message is shown because it begins with a tab.

The title for a menu item help message is always the description text, as it appears in the left-hand column of the menu display, prepended by the string `Help on`.

The Default Title Format

You can define two types of default titles:

- a global default title to be used on all the help messages defined in the item help file
- a form default title to be used on all of the help messages defined for a particular form in an item help file with messages defined for numerous forms

Defaults can be overridden, as described in the section “*The Title Hierarchy*.” A default title definition is recommended but not required.

The format for the default title definition is as follows:

```
[form_id:] TITLE: Title Text
```

form_id is the name of the form as it is defined with `lininfo` in your FMLI form definition. When a *form_id* is supplied, this line defines a form default title. When it is not supplied, this line defines a global default title.

The title text defined after the `TITLE` keyword will have the string `HELP on` prepended to it when displayed. Keep this in mind when writing the title.

An example form default title definition is shown below.

```
task1:TITLE:Package Administration Task1
```

If `task1` had not been added before `TITLE`, this example would be defining a global default title. The title defined by the example above will be displayed as:

```
HELP on Package Administration Task1
```

The Field Item Help Message Format

The field item help message will be shown whenever a user requests help from within an FMLI form. Each field on the form must have a help message defined in the item help file.

The format for the field item help definition is as follows:

```
[form_id:] field_id: [Title Text]
TAB   Line 1 of message text
TAB   Line 2 of message text
TAB   Line n of message text
```

form_id is the name of the form as it is defined with `lininfo` in your FMLI form definition. When one item help file contains messages for multiple tasks (and so multiple forms), it is used to distinguish with which form a field belongs. It is optional if the file

contains messages for only one task. `field_id` is the name of the field as it is defined with `lininfo` in your FMLI form definition. *Title text* defines a title used only with the help message for this field. As with the default title, the text defined here will have the string `HELP` on prepended to it when displayed.

The message text should be entered beneath the header line. There can be multiple lines of text with a maximum length of 69 characters per line. Each line must begin with a tab character. Blank lines may be included within the message as long as they also begin with a tab character. An example field item help definition is shown below.

```
task1:fld1:the Name Field
    This is the text for a field item help for a name
    field.
```

```
    The preceding line will appear as a blank line
    when the help message is shown because it begins
    with a tab.
```

The title for this field item help message, as defined above, will be `HELP on the Name Field`

The Title Hierarchy

You can define a global default title, a form default title, and a field title in the item help file. When all three are defined in the same file, the following rules are followed:

- The global default title is used for any message defined in an item help file that does not have a form default title or field title.
- The form default title is used for any message defined in an item help file and that is associated with the form, unless it has a field title.
- The field title is used only for the one field item help message for which it is defined.

In summary, if no field title is defined, the form default title is used. If no form default title is defined, the global default title is used. You always want at least a global default title defined; otherwise, the string `HELP` on will be displayed with no descriptive text.

To define a global default title, add a line to your item help file in the following format:

```
TITLE: Title Text
```

where *Title Text* is the text for the global default title.

To define a form default title, add a line to your item help file in the following format:

```
form_id:TITLE: Title Text
```

where *form_id* is the name of form as it is defined with `lininfo` in your FMLI form definition and *Title Text* is the text for the form default title.

To define a field title, use the following format for the field item help header line:

form_id:field_id:Title Text

where *form_id* is the name of the form as it is defined with `lininfo` in your FMLI form definition, *field_id* is the name of the field as it is defined with `lininfo` in your FMLI form definition and *Title Text* is the text for the field title.

NOTE

In all cases, the text defined as *Title Text* is always prepended with the string `HELP on` when displayed to a user.

Setting Up for Item Help in an FMLI Object

To help the interface read your item help file and know with which forms and fields a help message is associated, you must define your `help` and `lininfo` descriptors in your FMLI object definition as follows:

- The `help` descriptor must be defined exactly as shown on the line below:

```
help=OPEN TEXT $INTFBASE/Text.itemhelp $LININFO
```

- The `lininfo` descriptor for each field must be defined as

```
lininfo=[form_id:]field_id
```

where *form_id* and *field_id* are names each no longer than 30 characters. The names defined here as *form_id* and *field_id* must match exactly those used as *form_id* and *field_id* in the item help file.

NOTE

Since you do not create an FMLI form definition for a menu entry, you do not need to take any setup actions. However, you should be certain that the *task_name* keyword precedes the `ABSTRACT` heading line for a menu entry help message.

Example Item Help Files

This section shows two example item help files. Screen 16-1 shows an item help file that defines messages for only one form. Screen 16-2 shows an example of defining messages for multiple forms in one item help file.

ABSTRACT:

The text defined here will be shown to users when they request help while viewing the parent menu for this task. The task name is "adding users."

TITLE:Adding Users

field1:

The text defined here will be shown to users when they request help from the form and the cursor is positioned at field1. The title for this message will be "HELP on Adding Users" as defined above.

field2:Field 2

The text defined here will be shown to users when they request help from the form and the cursor is positioned at field2. The title for this message will be "HELP on Field 2."

Note:

The lininfo descriptors in the form definition associated with this file should look like this:

```
.  
. .  
lininfo=field1  
  
. .  
/lininfo=field2
```

Screen 16-1. Item Help File for One Form


```

add:ABSTRACT:
  The text defined here will be shown to
  users when they request help while
  viewing the parent menu for the task
  named add.

add_user:TITLE:Adding Users

add_user:field1:
  The text defined here will be shown to
  users when they request help from the
  form and the cursor is positioned at
  field1. The title for this message will
  be "HELP on Adding Users" as defined above.

add_user:field2:Field 2
  The text defined here will be shown to
  users when they request help from the
  form and the cursor is positioned at
  field2. The title for this message will
  be "HELP on Field 2".

delete:ABSTRACT:
  The text defined here will be shown to
  users when they request help while
  viewing the parent menu for the task
  named delete.

delete_user:TITLE:Deleting Users

delete_user:field1:
  The text defined here will be shown to
  users when they request help from the
  form and the cursor is positioned at
  field1. The title for this message will
  be "HELP on Deleting Users" as defined above.

delete_user:field2:Field 2
  The text defined here will be shown to
  users when they request help from the
  form and the cursor is positioned at
  field2. The title for this message will
  be "HELP on Field 2."

Note: The lininfo descriptors in the form definition associated with this
file should look like this:
:
:
lininfo=add_user:field1
:
:
lininfo=add_user:field2
:
:
lininfo=delete_user:field1
:
:
lininfo=delete_user:field2

```

Screen 16-2. Item Help File for Multiple Forms

Packaging Your Interface Modifications

To prepare your interface modifications for installation, you must create the packaging for your menus and tasks by executing **edsysadm**. The packaging created by **edsysadm** consists of two files, a **prototype** file and a menu information file. This section describes the procedures for creating these files and what to do after they have been created. (It also describes how to change the packaging after it has been created.)

NOTE

edsysadm also creates a package description file. **edsysadm** uses this file during its execution and is not a part of the packaging.

Basic Steps for Packaging Your Modifications

The procedures described next must be repeated for each menu and task entry being added. Begin with creating the menu entry (or entries) because you cannot add tasks or submenus to a menu that does not exist. Be certain that you use the same package description file name for all of the entries belonging to a package.

After running **edsysadm**, be certain to follow the steps described in “*Preparing Your Package*” (at the end of this section) to incorporate the modifications into your software package.

For example, if your administration requires the addition of one menu and four tasks, you will need to follow the procedure for creating the packaging for a menu entry, then repeat the procedure for creating the packaging for a task entry four times. Each time, when asked for a package description file name, give the same name to ensure that the packaging created contains all the necessary entries. These procedures will create a menu information file and a **prototype** file with all of the information necessary to include your interface modifications in your package. The two remaining steps (described in “*Preparing Your Package*”) are to include the **edsysadm** created **prototype** file in your package **prototype** file and to edit the **CLASSES** parameter in the **pkginfo** file.

Creating or Changing the Packaging for a Menu Entry

The procedures for creating and changing the packaging for a new menu are similar and both result in the display of a menu definition form. Each procedure is described below, followed by a description of the menu definition form.

Creating the Packaging for a Menu Entry

Before creating the packaging for a new menu entry, you should:

- Select a name and description for the menu.

- Select a location for it in the interface.
 - Prepare a help message file for the menu entry (refer to “*Writing Your Help Messages*” presented earlier in this chapter for instructions).
 - Know the name of the package description file to which the information for this menu should be added (if you are adding multiple menus and tasks)
1. Type **edsysadm** and press RETURN.

NOTE

If you do not execute this command from the directory in which the help message file resides, supply the full pathname when prompted for the name of the help message file.

2. You are asked to choose between a menu and a task. Choose menu and press RETURN.
3. You are asked to choose between adding a new menu or changing an existing one. Choose add and press RETURN.
4. You are given an empty menu definition form. Fill it in and press SAVE. (See “*The Menu Definition Form*” for descriptions of the fields on this form.)
5. You are asked if you want to test the changes before actually making them. Answer either *yes* or *no* and press SAVE. (If you answer *yes*, refer to the “*Testing Your Menu Changes On-Line*” section to learn what the test involves.)
6. You are asked if you want to install the modifications into the interface on your machine or save them for a package. Choose *save* and press SAVE.
7. You are asked to supply a file name. Enter a name for the package description file and press SAVE.
8. If the file name given for the package description file already exists, you are asked if you want to overwrite it or add to its contents. Answer *overwrite*, *do not overwrite*, or *add* and press SAVE.
9. If the file name does not already exist (or after you have completed Step 8) you will see a message stating that the menu information file and **prototype** file have been verified and the top-level **prototype** must be edited to include the new **prototype** file. Press CANCEL to return to the menu shown in Step 3. Press CONT to return to the form shown in Step 4.

Changing the Packaging for a Menu Entry

Before changing the packaging for a menu entry, you should:

- Know the name and description of the menu entry.
- Know its location in the interface.

- Change the associated help message file, if necessary, or create a new one (refer to “*Writing Your Help Messages*” presented earlier in this chapter for instructions).
- Know the name of the package description file associated with the package being changed (and know that it is available in your current working directory).

Once this information is obtained, complete this task as follows:

1. Type **edsysadm** and press RETURN.

NOTE

If you have changed a help message file or created a new one and you do not execute this command from the directory in which the help message file resides, supply the full pathname when asked for the name of the file.

2. You are asked to choose between a menu and a task. Choose menu and press RETURN.
3. You are asked to choose between adding a new menu and changing an existing one. Choose change and press RETURN.
4. You are asked if your change is for an on-line menu or for a menu that has been saved for a package. Choose packaged and press SAVE.
5. You are asked to supply the package description file name for the package being changed. Fill in the name of a valid package description file and press SAVE.
6. You are given a menu definition form filled in with the current values for the menu named above. Make the desired changes and press SAVE. (See the “*The Menu Definition Form*” for descriptions of the fields on this form.)
7. You are asked if you want to test the changes before actually making them. Answer either *yes* or *no* and press SAVE. (If you answer *yes*, refer to the section entitled “*Testing Your Menu Changes On-Line*” to learn what the test involves.)
8. You are asked if you want to install the modifications into the interface on your machine or save them for a package. Choose *save* and press SAVE.
9. You are asked to supply a file name. Enter a name for the package description file and press SAVE. (This must be the same package description file named in Step 5.)
10. If the file name given for the package description file already exists, you are asked if you want to overwrite it or add to its contents. Answer *overwrite*, *do not overwrite*, or *add* and press SAVE.
11. If the file name does not already exist (or after you have completed Step 10) you will see a message stating that the menu information file and **pro-**

totype file have been verified and the top-level **prototype** must be edited to include the new **prototype** file. Press CANCEL to return to the menu shown in Step 4. Press CONT to return to the form shown in Step 5.

Testing Your Menu Changes On-Line

Before installing your menu changes, you may want to verify that you've added an entry to a menu. The **edsysadm** command gives you a chance to do this after you fill in the menu definition form. Follow these steps to perform your test.

1. Type yes when **edsysadm** presents the following prompt:

```
Do you want to test this modification before continuing?
```

2. The parent menu (on which your addition or change is listed) is displayed. Check to make sure your modification has been made correctly.
3. Put the cursor on the new or changed menu entry and press the HELP key. The text of the help message for that menu entry is displayed so you can check it. (Press CANCEL to return to the menu.)
4. To exit on-line testing, press the CANCEL key.

5. You are returned to the prompt:

```
Do you want to test this modification before continuing?
```

If you want to continue executing the change, type no.

If you want to make additional modifications to the menu definition form, press CANCEL. You are returned to the form and can make further changes at that time. (Press SAVE when you have finished your editing. You can then retest your changes or continue executing the change.)

The Menu Definition Form

This form contains four fields in which you must provide: a menu name, a menu description, a menu location, and the name of the help message for the menu. Below are descriptions of the information you must provide in each field.

Menu Name	The name of the new menu (as it should appear in the left-hand column of the screen). This field has a maximum length of 16 alphanumeric characters.
Menu Description	A description of the new menu (as it should appear in the right-hand column of the screen). This field has a maximum length of 58 characters and can consist of any alphanumeric character except the at sign (@), carat (^), tilde (~), back grave (`), grave (`), and double quotes (").
Menu Location	The location of the menu in the menu hierarchy, expressed as a menu pathname. The pathname should begin with the main menu followed by all other menus that must be traversed (in the order they are traversed) to access this

menu. Each menu name must be separated by colons. For example, the menu location for a menu entry being added to the Applications menu is `main:applmgmt`. Do not include the menu name in this location definition. The complete pathname to this menu entry will be the menu location plus the menu name defined at the first prompt.

This is a scrollable field, showing a maximum of 50 alphanumeric characters at a time.

Menu Help File	Pathname to the item help file for this menu entry.
Name	If it resides in the directory from which you invoked edsysadm , you do not need to give a full pathname. If you name an item help file that does not exist, you are placed in an editor (as defined by <code>\$EDITOR</code>) to create one. The new file is created in the current directory and named <code>Help</code> .

The following screen shows a filled-in sample menu definition.

```
Define A Menu
Name: msvr
Description: Menu Description
Location: main:applmgmt
Help Message: Help
```

Creating or Changing the Packaging for a Task Entry

The procedures for creating and changing the packaging for a new task are similar and both result in the display of a task definition form. Each procedure is described below, followed by a description of the task definition form.

Creating the Packaging for a Task Entry

Before creating the packaging for a task entry, you should:

- Gather all files that will be associated with this task, such as the help file, FMLI forms, or other executables. All files should already be prepared.
- Decide on the task name and description.
- Decide on its location in the interface.
- Create a help file (refer to “*Writing Your Help Messages*” presented earlier in this chapter for instructions).
- Know the name of the package description file to which the information for this task should be added (if you are adding multiple menus and tasks)

Once this information is obtained, complete this task as follows:

1. Type **edsysadm** and press RETURN.

NOTE

If you do not execute this command from the same directory in which the files associated with this task reside, enter full path-names when supplying file names.

2. You are asked to choose between a menu and a task. Choose **task** and press RETURN.
3. You are asked to choose between adding a new task or changing an existing one. Choose **add** and press RETURN.
4. You are given an empty task definition form. Fill it in and press **SAVE**. (See *"The Task Definition Form"* for descriptions of the fields on this form. Be aware that, when you name the menu under which you want this new task to reside, that menu must already be packaged.)
5. You are asked if you want to install the modifications into the interface on your machine or save them for a package. Choose **save** and press **SAVE**.
6. You are asked to supply a file name. Enter a name for the package description file and press **SAVE**.
7. If the file name given for the package description file already exists, you are asked if you want to overwrite it or add to its contents. Answer either **overwrite**, **do not overwrite**, or **add** and press **SAVE**.
8. If the file name does not already exist (or after you have completed Step 7) you see a message stating that the menu information file and **prototype** file have been verified and the top-level **prototype** must be edited to include the new **prototype** file. Press **CANCEL** to return to the menu shown in Step 3. Press **CONT** to return to the form shown in Step 4.

Changing the Packaging for a Task Entry

Before changing the packaging for a task entry, you should:

- Gather any of the files associated with this task that have been changed or are new. All files should already be prepared or changed.
- Know the menu name and description.
- Know its location in the interface.
- Change the associated help file, if necessary (refer to *"Writing Your Help Messages"* presented earlier in this chapter for instructions).
- Know the name of the package description file associated with the package being changed (and know that it is available in your current working directory).

Once this information is obtained, complete this task as follows:

1. Type **edsysadm** and press RETURN.

NOTE

If your change requires new files or changes to existing files and you do not execute this command from the directory in which the files reside, enter full pathnames when supplying file names.

2. You are asked to choose between a menu and a task. Choose **task** and press RETURN.
3. You are asked to choose between adding a new task and changing an existing one. Choose **change** and press RETURN.
4. You are asked if your change is for an on-line task or for a task that has been saved for a package. Choose **packaged** and press SAVE.
5. You are asked to supply the package description file name for the package being changed. Fill in the name of a valid package description file and press SAVE.
6. You are given a task definition form filled in with the current values for the task named above. Make the desired changes and press SAVE. (See “*The Task Definition Form*” for descriptions of the fields on this form.)
7. You are asked if you want to install the modifications into the interface on your machine or save them for a package. Choose **save** and press SAVE.
8. You are asked to supply a file name. Enter a name for the package description file and press SAVE. (This must be the same package description file named in Step 5.)
9. If the file name given for the package description file already exists, you are asked if you want to overwrite it or add to its contents. Answer either **overwrite**, **do not overwrite**, or **add** and press SAVE.
10. If the file name does not already exist (or after you have completed Step 9) you see a message stating that the menu information file and **prototype** file have been verified and the top-level **prototype** must be edited to include the new **prototype** file. Press CANCEL to return to the menu shown in Step 4. Press CONT to return to the form shown in Step 5.

The Task Definition Form

This form contains six fields in which you must provide: a task name, a task description, a task location, the name of a help message for the task, a task action file, and the files associated with the task. Below are descriptions of the information you must provide in each field.

Task Name	The name of the new task (as it should appear in the left-hand column of the screen). This field has a maximum length of 16 alphanumeric characters.
-----------	--

Task Description A description of the new task (as it should appear in the right-hand column of the screen). This field has a maximum length of 58 characters and can consist of any alphanumeric character except the at sign (@), carat (^), tilde (~), back grave (`), grave (`), and double quotes (“”).

Task Location The location of the task in the menu hierarchy, expressed as a pathname. The pathname should begin with the main menu followed by all other menus that must be traversed (in the order they are traversed) to access this task. Each menu name must be separated by colons. For example, the task location for a task entry being added to the applications menu is `main:applmgmt`. Do not include the task name in this location definition. The complete pathname to this task entry will be the task location as well as the task name defined at the first prompt.

This is a scrollable field, showing a maximum of 50 alphanumeric characters at a time.

Task Help File Pathname to the item help file for this task entry.

Name If it resides in the directory from which you invoked `edsysadm`, you do not need to give a full pathname. If you name an item help file that does not exist, you are placed in an editor (as defined by `$EDITOR`) to create one. The new file is created in the current directory and named `Help`.

Task Action The FMLI form name or executable that will be run when this task is selected. This is a scrollable field, showing a maximum of 58 alphanumeric characters at a time. This pathname can be relative to the current directory as well as absolute. (Refer to the “*Writing Your Administration Actions*” section for details.)

Task Files Any FMLI objects or other executables that support the task action listed above and might be called from within that action. *Do not include the help file name or the task action in this list.* Pathnames can be relative to the current directory as well as absolute. A dot (.) implies “all files in the current directory” and includes files in subdirectories.

This is a scrollable field, showing a maximum of 50 alphanumeric characters at a time.

The following screen shows a filled-in sample task definition form.

```
Define A Task
Name: msvrtask
Description: Task Description
Location: main:applmgmt:msvr
Help Message: Help
Action: Form.msvrtask
Task Files: Form.task2, Text.task2
```

Preparing Your Package

You must perform two steps, after executing **edsysadm**, to include your interface modification files in your application package.

1. Include the **prototype** file

The **prototype** file that **edsysadm** creates must become a part of your package **prototype** file structure. This means that you must either read it into another **prototype** file or use the **include** command in your primary **prototype** file. For example, adding

```
!include /myproject/admsrc/prototype
```

to a **prototype** file in the **/myproject** directory ensures that the **prototype** file in **/myproject/admsrc**, and all of the objects it describes, will be included when the packaging tool, **pkgmk**, creates the package.

2. Change your CLASSES parameter in the **pkginfo** file

The components defined in the **prototype** file that **edsysadm** creates are placed into the two special classes: **OAMmif** and **OAMadmin**. In addition, the package installation and removal scripts use the **OAMintf** class for the menu files. You must edit the **pkginfo** file for your package and add these classes to the **CLASSES** parameter definition. For example, a **CLASSES** definition before the change might look like this:

```
CLASSES="class1 class2"
```

If you have only modified a task, then the **CLASSES** parameter should be defined like this:

```
CLASSES="class1 class2 OAMmif OAMadmin"
```

If you have modified or created a menu, then the **CLASSES** parameter should be defined like this:

```
CLASSES="class1 class2 OAMmif OAMadmin OAMintf"
```

Your interface modifications are now ready to be included in your package when you create your package using **pkgmk**. (Details on packaging procedures are discussed in the “Application Software Packaging” chapter.)

Deleting Interface Modifications

Interface modifications can be deleted in two ways. When a package is removed, the modifications installed with the package are removed automatically. Modifications can also be removed online by executing **delsysadm**.

To delete either a menu or task entry online, execute

```
delsysadm name
```

where *name* is the location of the task or menu in the interface, followed by the menu or task name. For example, to delete a task named `mytask` with the location `main:applmgmt:mymenu`, execute

```
delsysadm main:applmgmt:mymenu:mytask
```

Before an entry for a menu can be removed, that menu must be empty (contain no submenus or tasks). If it is not, you must use the **-r** option with **delsysadm**. This option requests that, in addition to the named menu, all submenus and tasks located under that menu be removed. For example, to remove `main:applmgmt:mymenu` and all submenus and tasks that reside under it, execute

```
delsysadm -r main:applmgmt:mymenu
```

When you use the **-r** option, **delsysadm** checks for dependencies before removing any subentries. (A dependency exists if the menu being removed contains an entry placed there by an application package.) If a dependency is found, you are shown a list of packages that depend on the menu you want to delete and asked whether you want to continue. If you answer yes, the menu and all of its menus and tasks are removed (even those shown to have dependencies). If you answer no, the menu is not deleted.

CAUTION

Use **delsysadm** to remove only those menu or task entries that you have added to the interface with **edsysadm**.

Data Validation Tools

The data validation tools are a group of shell level commands that serve two purposes:

- standardize the appearance of administration interaction.
- simplify development of scripts requiring administrator input

Every tool generates a prompt, validates the answer and returns the response. There are no restrictions on when you should use them. It is recommended that you use them every time your application interacts with an administrator. Using the tools at such a time will make all administrator interaction look alike to the user, regardless of the vendor who created the package. You will see, as well, that using these tools makes writing scripts with adminis-

trator interaction much simpler, since the tools do the work based on parameters you provide.

At the very least, it is recommended that you use them in your request script (the packaging script from which you can solicit administrator input) and in the executables you deliver when your package administration will be incorporated into the **sysadm** interface. See the chapter entitled “*Application Software Packaging*” in the *Real-Time Programming Guide* for details on writing a request script.

This section introduces you to the data validation tools and discusses their characteristics. The shell commands and corresponding visual tools are provided as Section 1 manual pages.

Types of Tools

There are two types of data validation tools. Both perform the same series of tasks (described later) but are used in different environments. The two types are:

- Shell Commands

These tools are invoked from the shell level and used in shell scripts.

- Visual Tools

These tools are invoked from within the field definition in an FMLI form definition. While the shell commands perform all tasks with one command, the visual tools are broken into separate commands for defining help messages, error messages and performing validation.

Characteristics of the Tools

All of the shell commands perform the same series of tasks (the visual tools each perform a subsection of the full series). Those tasks are:

- prompt a user for input
- validate the answer
- format and print a help message when requested
- format and present an error message when validation fails
- return the input if it passes validation
- allow a user to quit the process

The tool itself defines the type of prompt shown and validation performed is defined. For example, the shell command **ckyorn** prompts for a yes or no answer and accepts only a positive or negative response. Some tools allow you to supply input during execution to help customize the validation. For example, **ckrange** prompts for and validates an

answer within a given range. The upper and lower limits of the range can be defined when executing **ckrange**.

NOTE

Leading and trailing white space is stripped from the input before validation is performed.

The Data Validation Tool Prompts

Each tool has a default prompt that you can use as is, add to, or overwrite. The manual page for each tool shows the default prompt text. You must use the **-p** option of a shell command before the default can be overwritten.

For example, executing **ckyorn** without options produces the following output:

```
Yes or No [y,n,?,q]:
```

The next example shows the use of the **-p** option and the output that is produced.

```
$ ckyorn -p "Do you want the manual page files
installed?"
Do you want the manual page files installed? [y,n,?,q]:
```

The Data Validation Tool Help Messages

Each tool has a default help message that you can use as is, add to, or completely overwrite. The manual page for each tool shows the default help message text. You must use the **-h** option of a shell command before the default can be overwritten.

For example, if you executed **ckyorn** without options and the user requested a help message by entering **?** at the prompt, the following message would be seen:

```
To respond in the affirmative, enter y, yes, Y, or YES.
To respond in the negative, enter n, no, N, or NO.
```

The next example shows the use of the **-h** option when executing **ckyorn**. The text defined after the **-h** will be shown if the user requests a help message.

```
ckyorn -h "Answer yes if you want the manual page files
\\\\\\
installed or no if you do not."
```

If you insert a tilde (~) at the beginning or end of your definition, the default text will be added at that point. For example,

```
ckyorn -h "The manual page files will be written to your
\\\\\\
system, or not, based on your answer.~"
```

will produce the help message:

```
The manual page files will be written to your system, or
not, based on your answer. To respond in the affirmative,
enter y, yes, Y, or YES. To respond in the negative,
enter n, no, N, \or NO.
```

The Data Validation Tool Error Messages

Each tool has a default error message that you can use as is, add to, or completely overwrite. The manual page for each tool shows the default error message text. You must use the `-e` option of a shell command before the default can be overwritten.

For example, if you executed `ckyorn` without options, and validation failed, the following message would be seen:

```
ERROR: Please enter yes or no.
```

The next example shows the use of the `-e` option when executing `ckyorn`. The text defined after the `-e` will be prepended with `ERROR:` and shown if validation fails.

```
ckyorn -e "You did not respond with yes or no."
```

If you insert a tilde (`~`) at the beginning or end of your definition, the default text will be added at that point.

Message Formatting

All three message types (prompt, error, and help) are limited in length to 78 characters and are automatically formatted. Regardless of how you define them in your code, any white space used (including newline) is stripped during formatting.

You can use the `-w` option of a shell command (or the `ckwidth` variable of a function) to define the line length to which your messages should be formatted.

The Shell Commands

Table 16-1 lists the shell commands and what they are used for. All of the shell commands perform the same series of tasks, as described previously. The table's "Purpose" column describes the type of prompt and validation with which the command deals. Details for each command can be found on their respective manual pages.

Table 16-1. The Shell Commands

Command (and Function)	Purpose
ckdate	Prompts for and validates that the answer is a date (can define format for date).
ckgid	Prompts for and validates that the answer is a group id.
ckint	Prompts for and validates an integer value (can define base for input).
ckitem	Builds a menu, prompts for and validates a menu item (can define characteristics of the menu).
ckkeywd	Adds keywords to a prompt and validates that the return answer matches a keyword.
ckpath	Prompts for and validates a pathname (can define what type of validation to perform, such as “pathname must be readable”).
ckrange	Prompts for and validates an integer within a range (can define the upper and lower limits of the range).
ckstr	Prompts for and validates that the answer is a string (can define a regular expression, in which case the string must match the expression).
cktime	Prompts for and validates that the answer is a time (can define format for time).
ckuid	Prompts for and validates that the answer is a user id.
ckyorn	Prompts for and validates a yes/no answer. Input must be <i>y</i> , <i>yes</i> , <i>Y</i> , <i>YES</i> , <i>n</i> , <i>no</i> , <i>N</i> , or <i>NO</i> .
dispgid	Displays a list of all valid group names.
dispuid	Displays a list of all valid user names.

The Visual Tools

The visual tools are invoked from within the field definition of an FMLI form. Because of the nature of FMLI form definitions, it is necessary to divide the tasks performed by only one shell command into sets. The purpose of a visual tool set parallels the purpose of a shell command. For example, **ckdate** performs a group of tasks for a prompt whose response should be a date. The same group of tasks requires three visual tools:

- **errdate** (formats and presents an error message)
- **helpdate** (formats and presents a help message)
- **valdate** (validates the answer to be a date)

The format and description of each visual tool set is shown on the equivalent shell command manual page. For example, the equivalent shell command for the set described above is **ckdate**. Refer to the manual page **ckdate (1)** for details on the three visual tools **errdate**, **helpdate**, and **valdate**.

Table 16-2 lists the visual tool sets and their associated response type.

Table 16-2. The Visual Tools

Visual Tool Set	Response Type
erryor, helpyor, valyor	yes or no
errint, helpint, valint	integer
errrange, helprange, valrange	integer in a range
errstr, helpstr, valstr	string (potentially matching an expression)
errpath, helppath, valpath	pathname
erritem, helpitem	menu item
errgid, helpgid, valgid	existing group
errtime, helptime, valtime	time of day
errdate, helpdate, valdate	date

There are two other visual tools. **dispuid** displays a list of login ids and **dispgid** displays a list of group ids. These two tools can be used with the FMLI **rmenu** keyword to display a list of ids.

The following example shows a field definition written in FMLI using the visual tools:

```

name="Do you want to install the manual page files?"
value=y
choicemsg='helpyor'
invalidmsg='erryor -e "~Enter yes to install the manual page files"'
valid='valyor $F1'
rows=1
columns=1
    
```


Glossary

This glossary defines terms used in the documentation. If a term has multiple definitions, each definition is numbered. Some terms may describe software not loaded on your system.

Terms in *italics* are defined in the glossary. If a term with multiple definitions is used in a definition, the applicable definition number appears in brackets (for example, [1]).

access the ability of any *subject* to communicate with any *object* or any other subject.

Access Control List (ACL)

a component of *Discretionary Access Control* (DAC), consisting of one or more *user* entries, one or more *group* entries, and one other entry. ACLs permit a finer grain of discretionary *access* than the nine *access permission* bits available without the Enhanced Security Utilities package.

access permission components of *Discretionary Access Control*. These bits define the *access permissions* and can be changed by the file's owner via the **chmod** command. Running **ls -l** lists the permission bits before the file name(s).

address a number, label, or name that shows the location of information in the computer's *memory*.

administrative role any one of the roles defined in the Trusted Facilities Management database: *Auditor* (AUD), *Operator* (OP), *Security Operator* (SOP), *System Security Officer* (SSO), also known as the *Site Security Officer*, and the *Network Administrator* (NET).

a.out the default name of a compiled *object file*, pronounced "a-dot-out". **a.out** is the default name produced by the **cc** command.

allocation unit a group of consecutive blocks on a file system that contain resource summaries, free resource maps, inodes, and data blocks. The "allocation unit" is equivalent to the **ufs** "cylinder group."

archive 1. a collection of data gathered from several *files* into one file. 2. especially, collection gathered by **ar** for use as a library.

argument The element of a command line that specifies data on which a command is to operate. Arguments follow the command name and can include numbers, letters, or text strings. For instance, in the command **lp -m myfile**, **lp** is the command and **myfile** is the argument.

audit the act of recording all potentially *sensitive* and *security*-related transactions on a *trusted computer system*. The *System*

Administrators decide whether to install the auditing programs and which transactions to audit.

audit trail	the written record that reports on all potentially <i>sensitive</i> and <i>security</i> -related transactions on a <i>trusted computer system</i> . Only the <i>Trusted Computing Base</i> can write audit trail data.
Auditor (AUD)	an authorized individual entrusted with <i>secure audit</i> administrative duties on a <i>trusted computer system</i> . Auditor duties may include selecting events to be audited enabling the recording of events, analyzing the <i>audit trail</i> , and modifying or deleting auditing information.
authentication	1. verification of the <i>client</i> machine and <i>login_name</i> of an incoming request. 2. the mechanism by which the <i>Trusted Computing Base</i> verifies the identity of a <i>user</i> .
authorization	allowing or disallowing <i>user access</i> to a service.
automatic calling unit	a hardware <i>device</i> used to dial stored telephone numbers. This unit enables a system to contact another system over phone lines without manual intervention.
automatic data	data that is persistent only during the invocation of a procedure. It describes data belonging to a process. Automatic data occupies the stack segment. See <i>static data</i> .
bad block	a <i>sector</i> of a storage medium which cannot store data reliably.
bandwidth	a measurement of the amount of information that can be passed through a communication <i>channel</i> in a given amount of time. Bandwidth is usually in units of bits per second.
block	the basic unit of <i>buffering</i> in the <i>kernel</i> , 1024 bytes; see <i>indirect</i> , <i>logical</i> , and <i>physical blocks</i> .
block device	a <i>device</i> upon which a <i>file system</i> [1] can be <i>mounted</i> , typically a permanent storage device such as a disk drive, so called because data transfers to the device occur by <i>blocks</i> ; see <i>character device</i> .
boot	the process by which the operating system is started. The <i>kernel</i> must bootstrap itself from secondary storage into an empty machine. No <i>login</i> [3] or <i>process</i> persists across a boot.
boot program	loads the <i>operating system</i> into <i>core</i> .
buffer	1. a staging area for input/output where arbitrary-length transactions are collected into convenient units. The <i>file system</i> [3] uses buffers, as does <i>stdio</i> . 2. to use buffers.
buffer pool	a region of storage available to the <i>file system</i> [3] for holding <i>blocks</i> . To make read and write operations independent of <i>device</i> blocks, all but <i>raw</i> input/output for <i>block devices</i> goes through the buffer pool.

cartridge tape	a storage medium that consists of a magnetic tape wound on spools housed in a plastic container.
category	a non-hierarchical, restrictive grouping of <i>objects</i> to which a name is applied (for example, “Project Alpha,” “Project Sigma,” “Project Phi”). The system supports up to 1024 categories. The non-hierarchical category and the hierarchical <i>classification</i> together constitute the <i>security level</i> to which a <i>sensitivity label</i> is applied; see <i>security level</i> and <i>sensitivity label</i> .
certification	the technical evaluation of a <i>trusted computer system</i> , by the <i>National Computer Security Center</i> (NCSC). This evaluation is part of the accreditation process establishing the extent a computing system's design and implementation meet the security requirements of the NCSC. From least secure to most secure, the levels of accreditation are D (untrusted), C1, C2, B1, B2, B3, and A1.
certify	the action which produces <i>certification</i> .
channel	a path or mechanism by which information is transferred within a computer system. On a <i>trusted computer system</i> , a channel offers a potential avenue of <i>compromise</i> to the <i>Trusted Computing Base</i> and to <i>sensitive information</i> , and it must be <i>trusted</i> or <i>audited</i> .
character device	a <i>device</i> on which a <i>file system</i> [1] cannot be <i>mounted</i> , such as a terminal or the <i>null device</i> .
child process	See <i>fork</i> .
classification	one of 256 levels of a hierarchy for grouping <i>objects</i> of like <i>sensitivity</i> , where 0 is the lowest level and 255 is the highest. The levels are usually known by the names associated with them (“Classified,” “Secret,” “Top Secret”). The hierarchical classification and the non-hierarchical <i>category</i> together constitute a <i>security level</i> ; see <i>security level</i> .
client	a <i>host</i> that has <i>mounted</i> an <i>advertised resource</i> from another host in a <i>Network File Sharing</i> environment.
command	1. an instruction to the <i>shell</i> , usually to run a <i>program</i> [1] as a <i>child process</i> . 2. by extension, any <i>executable file</i> , especially a <i>utility program</i> .
command file	See <i>shell script</i> .
compromise	a violation of the <i>security policy</i> that causes potential or actual <i>unauthorized</i> disclosure, modification, or destruction of <i>sensitive information</i> .
configuration	the arrangement of the software or hardware of a system, peripheral, or network as defined by the nature, number, and chief characteristics of its functional units.
configuration management	the identification of a system's hardware, software, firmware,

	documentation, test fixtures, and test documentation and changes made to them throughout the development and operational life of the system. The current configuration and revision level of all components should be known so these components can be effectively and productively integrated, managed, and used.
console terminal	the directly connected terminal used for communication between the operator and the computer.
controller	a <i>device</i> that directs the transmission of data over the data links of a <i>network</i> .
core	Core is a name commonly associated with primary memory, although very little memory is still “core”.A better term might be “primary memory”.
core file	a <i>core image</i> of a terminated <i>process</i> saved for debugging. A core file is created under the name core in the current directory of the process.
core image	a copy of all the <i>segments</i> of a running or terminated program. The copy may exist in main storage, the <i>swap area</i> , or a <i>core file</i> .
covert channel	a communications <i>channel</i> that allows a <i>process</i> to transfer or deduce <i>sensitive information</i> in violation of the <i>security policy</i> of a <i>trusted computer system</i> .
covert storage channel	a <i>covert channel</i> that involves the direct or indirect writing of a storage location by one <i>process</i> and the direct or indirect reading of that storage location by another process. Covert storage channels typically involve a finite resource that is shared by two <i>subjects</i> at different <i>security levels</i> .
covert timing channel	a <i>covert channel</i> in which one <i>process</i> signals information to another by modulating its own use of system resources in a way that affects the real response time observed by the second process.
crash	occurs when a hardware or software <i>error</i> condition occurs that causes the system to take itself out of service. For example, such conditions may occur when the system cannot allocate resources, manage <i>processes</i> , or respond to requests for system functions, or when the electrical power is unstable.
cron	a command that creates a <i>daemon</i> that invokes <i>commands</i> at specified dates and times; see <i>daemon</i> .
current directory	The directory in which you are presently working. You have direct access to all files and subdirectories contained in your current directory. The shorthand notation for the current directory is a dot (.).
cylinder	the set of all <i>tracks</i> on a <i>disk</i> that are the same distance from the axis about which the disk rotates.

daemon	a background process, often perpetual, that performs a system-wide public function. cron is an example.
data integrity	the consistency between stored data that has not been exposed to alteration or destruction (either accidental or malicious) and the source data.
destination	the remote system that will ultimately receive a <i>file</i> transferred over a <i>network</i> .
device	1. a <i>file</i> [2] that is not a plain <i>file</i> or a <i>directory</i> , such as a tape drive, or the <i>null device</i> ; a <i>special file</i> . 2. a physical input/output unit.
device activation	the means by which a device is <i>accessed</i> . A device can be accessed only by an activated <i>Device Special File</i> that maps to it.
device allocation	<i>device access</i> by any one of several <i>kernel</i> mechanisms that support access to devices by <i>trusted</i> processes. The permission, usage, and protection strategies for accessing a device depend on the specific device, its intended use, and the programs that do the allocation.
Device Control Information (DCI)	information associated with a <i>device</i> when it is not in the <i>device disabled state</i> . DCI includes the <i>device level range</i> , the <i>multilevel</i> or <i>single-level</i> mode, and the activated <i>device special file</i> list.
Device Database (DDB)	the database that contains critical security attributes for <i>devices</i> , and information defining which <i>users</i> can use the device. (Administered via the admalloc command.)
device disabled state	the state in which a <i>device</i> is not accessible. The only operations allowed on a device in this state are the transitions to <i>device setup state</i> or <i>device enabled state</i> using the <code>devalloc</code> system call.
device driver	a module of the operating system that controls a specific input/output <i>device</i> in response to a request by a <i>subject</i> for input or output.
device enabled state	the state in which a <i>device</i> is fully operational at a specific <i>security level</i> , which cannot be changed even by a <i>privileged</i> process. From the enabled state, a device can be taken to the <i>device setup state</i> or the <i>device disabled state</i> by the <code>devstat</code> system call.
device level range	the range of <i>security levels</i> that a device can store or process.
device readiness state	one of three states, <i>device disabled</i> , <i>device setup</i> , or <i>device enabled</i> , that each <i>configured device</i> must be in. These states are used to assure proper allocation by <i>device allocation</i> mechanisms. A device can make the transition from any state to either of the others, but some restrictions apply.

device setup state the state used by a *device allocation* program to prepare a *device* for use.

Device Special File (DSF)

the mechanism through which *processes* address *devices*. The DSF has the structure of a regular file but it lacks a data area and has only an *inode*; data in the *inode* indicates a path through various kernel tables that eventually maps to a single, logical device; although this logical device is conceptually a single data object, more than one DSF can map onto a single device; see *device allocation*.

diagnostic a message printed at your terminal that identifies and isolates *program* errors.

directory a *file* that comprises a catalog of *filenames* [2]. The organizing principle of the *file system* [2], a directory consists of *directory entries* which specify further *files* [2] (including directories), and constitutes a node of the *directory tree*.

directory entry 1. an association of a name with an *inode number* appearing as an element of a *directory*. 2. the name part of such an association.

directory hierarchy the tree of all *directories*, in which each is reachable from the *root* via a chain of subdirectories.

directory tree See *directory hierarchy*.

Discretionary Access Control (DAC)

one of the mechanisms controlling sharing *objects* among *subjects*. The DAC mechanism uses the object *owner*, the object *group*, the nine *access permission* bits. It also uses the *Access Control List* of an object to determine the discretionary access to the object; contrasts with *Mandatory Access Control*. see *access permissions*.

disk A magnetic data storage device consisting of several round plates similar to phonograph records. Disks store large amounts of data and allow quick access to any piece of data.

diskette a magnetic storage medium which is smaller and more flexible than a hard *disk*.

domain a logical grouping of *hosts* in a *Remote File Sharing* environment. Each host in a domain relies on the same *domain name server(s)* for certain *resource* sharing and security services. Each domain has one *primary* and zero or more *secondary domain name servers*.

domain name server a computer that creates and maintains the following information for *hosts* in a *Network File Sharing* domain: *advertised resources*, *host* names and passwords, names and addresses for name servers of other domains (optional), *host* user and group information used for *ID mapping* (optional).

dominate	a relationship between <i>security levels</i> ; security level S1 is said to dominate security level S2 if the hierarchical <i>classification</i> of S1 is greater than or equal to that of S2, and if the non-hierarchical <i>categories</i> of S1 include those of S2 as a subset.
drive	the hardware device that holds magnetic disks, diskettes, and tapes while they are in use.
dump	a copy of the <i>core image</i> of the operating system.
encryption	the process of transforming data or text, usually by means of a cipher or substitution code, from its ordinary readable state to a superficially nonsense state. Changing the state of the information makes it much harder to <i>access</i> understandably and thus increases the confidentiality of the information.
environment	1. a set of strings, distinct from arguments, made available to a <i>process</i> when it <i>executes</i> [2] a <i>file</i> ; the environment is usually inherited across exec (2) operations; see <i>exec</i> . 2. a specific environment maintained by the <i>shell</i> .
error	occurs when a hardware or software condition prevents the successful <i>execution</i> of a system or a user <i>process</i> .
error message	a message sent from the system when an <i>error</i> occurs.
exclusive access	a feature used mainly by <i>device allocation</i> programs allowing a <i>process</i> , or multiple processes, to exclude <i>access</i> to a <i>device</i> by other processes.
exec	a system call which allows the user to request the execution of another program.
executable file	1. an <i>object file</i> that is ready to be copied into the <i>address</i> space of a <i>process</i> to run as the code of that process. 2. a file that has <i>execute permission</i> , either an <i>executable file</i> [1] or a <i>shell script</i> .
execute	1. informally, to run a <i>program</i> . 2. to replace the text <i>segment</i> and data <i>segments</i> of a <i>process</i> with a given <i>program</i> [1].
expired password	a <i>password</i> that is invalid because it is older than the mandatory retirement age allotted it by the <i>system administrator</i> .
exploitable channel	a <i>channel</i> that is usable or detectable by <i>subjects</i> external to the <i>Trusted Computing Base</i> .
exportation	the extraction of information from one computer system for transportation to another computer system; a potential source of <i>trojan horses</i> and <i>viruses</i> ; see <i>importation</i> .
FIFO	a named permanent <i>pipe</i> which allows two unrelated <i>processes</i> to exchange information using a pipe connection.
file	1. in general, a potential source of input or destination for output. 2. an <i>inode</i> and/or its associated contents, that is, a plain or

	ordinary file, <i>special file</i> , or <i>directory</i> . 3. a <i>directory entry</i> ; several directory entries may name the same file [2]. 4. a plain file.
file descriptor	a conventional integer quantity that designates an <i>open file</i> [1].
filename	1. a <i>pathname</i> . 2. the last component name in a <i>pathname</i> .
file system	1. a collection of <i>files</i> that can be <i>mounted</i> on a block <i>special file</i> . Each file of a file system appears exactly once in the <i>i-list</i> of the file system and is accessible via some <i>path</i> from the <i>root</i> directory of the file system. 2. the collection of all <i>files</i> on a computer. 3. the part of the kernel that deals with file systems [1].
filter	a <i>program</i> [1] that reads from the <i>standard input</i> and writes on the <i>standard output</i> , so called because it can be used as a data-transformer in a <i>pipeline</i> .
fixed privilege	a set of <i>privileges</i> that are always given to <i>processes</i> created from the executable file, independent of the <i>process privileges</i> from the creating, parent process. See <i>inheritable privilege</i> .
flaw	an error of commission, omission, or oversight in a <i>trusted computer system</i> that allows protection mechanisms to be bypassed.
floppy key	a copy of the default <i>firmware password</i> for an AT&T 3B2 Computer on a <i>diskette</i> . It may be used to reset the password to its original value.
flush	to empty a <i>buffer</i> , for example to throw away unwanted input/output on <i>interrupt</i> or to release output from <i>stdio</i> .
fork	to split one <i>process</i> into two, the <i>parent process</i> and <i>child process</i> , with separate, but initially identical, <i>text</i> , data, and <i>stack segments</i> .
formatting	the process of imposing an addressing scheme on a <i>disk</i> . This includes the establishment of a <i>VTOC</i> , and the mapping of both sides of the disk into <i>tracks</i> and <i>sectors</i> .
free list	in a <i>file system</i> [1], the list of <i>blocks</i> that are not occupied by data.
functional channel	a <i>channel</i> whose existence can be determined from the system call interface, independent of the internal implementation. Such <i>channels</i> include inode information, file system directories containing elements whose level is different from that of the directory, signals, FIFO pipes, IPC mechanisms, I/O device control mechanisms, file and record locking, and text locking.
getty	one of a series of <i>processes</i> that connect the user to the UNIX system. <i>getty</i> is invoked by init , and in turn invokes login . See <i>init</i> and <i>login</i> .
group	1. a set of <i>permissions</i> for access to a <i>file</i> ; see <i>owner</i> . 2. a set of <i>user IDs</i> that may assume the privileges of a <i>group</i> [1]. 3. the <i>group ID</i> of a file. 4. a set of <i>users</i> who share a common project or purpose, for example, all users performing basic systems

operations may belong to group `operator` and all users working on the “Alpha Project” may belong to group `alpha`. The *login names* of group members are listed together in `/etc/group`; the members of a group share the same group *access permissions*.

group ID (GID)	an integer value, usually associated with one or more <i>login names</i> ; as the <i>user ID</i> of a process becomes the <i>owner</i> of files created by a process, so the group ID of a process becomes the <i>group</i> [3] of such files. The <i>Trusted Computing Base</i> uses the group ID along with the <i>user ID</i> and <i>Access Control Lists</i> to enforce <i>Discretionary Access Control</i> .
hole	a gap in a plain file caused by seeking while writing [see <code>lseek(2)</code>]; <code>read(2)</code> takes data in holes to be zero; a <i>block</i> in a hole occupies no space in its <i>file system</i> .
host	a computer that is configured to share <i>resources</i> in a <i>Remote File Sharing</i> environment.
ID mapping	a means of setting the permissions each remote user and group will have for a <i>host's advertised resources</i> in a <i>Remote File Sharing</i> environment.
identification	via the <i>login name</i> , the mechanism by which the <i>Trusted Computing Base</i> recognizes a <i>user</i> as legitimate.
i-list	the index to a <i>file system</i> [1] listing all the <i>inodes</i> of the file system; see <i>inode number</i> .
importation	the inclusion of information on one computer system after transportation from another computer system; a potential source of <i>trojan horses</i> and <i>viruses</i> ; see <i>exportation</i> .
indirect blocks	data blocks that are not directly referenced by a <i>inode</i> . The <i>inode</i> has three <i>addresses</i> that indirectly reference data blocks.
inheritable privilege	<i>privileges</i> which a <i>parent process</i> can pass to a process it creates.
init	a general <i>process</i> spawner which is invoked as the last step in the <i>boot</i> procedure. It regularly checks a table that defines which processes should run at what <i>run level</i> .
inode	an element of a <i>file system</i> [1]. An <i>inode</i> specifies all properties of a particular <i>file</i> [2] and locates the file's contents, if any.
inode number, i-number	the position of an <i>inode</i> in the <i>i-list</i> of a <i>file system</i> [1].
instruction	See <i>address</i> .
integrity	in a <i>file system</i> , the quality of being without errors due to <i>bad blocks</i> . 1. in a <i>file system</i> , the quality of being without errors due to <i>bad blocks</i> . 2. the logical completeness of the hardware and software that implement the <i>security</i> mechanisms; the consistency of the data structures and the accuracy of the stored data.

interface programs	<i>shell scripts</i> furnished with spooler software (for the LP print service) which processes data sent by a user to a printer.
interrupt	1. a <i>signal</i> that normally terminates a <i>process</i> , caused by a break or an interrupt character. 2. a signal generated by a hardware condition or a peripheral <i>device</i> . 3. any <i>signal</i> .
IPC	an acronym for interprocess communication.
kernel	resident code that implements the <i>system calls</i> .
kernel address space	a portion of memory used for data and code addressable only by the <i>kernel</i> .
label	See <i>sensitivity label</i> .
least privilege	the principle in the <i>security policy</i> that restricts a <i>process</i> to the minimum <i>privileges</i> necessary to perform a given function at a given time; see <i>privilege</i> , <i>process privileges</i> , <i>maximum set of process privileges</i> , and <i>working set of process privileges</i> .
level	See <i>security level</i> .
level alias	See <i>security level alias</i> .
line discipline	a module to handle protocol or data conversion for a <i>stream</i> . A line discipline, unlike a <i>filter</i> , is part of the <i>kernel</i> .
link	1. to add an entry for an existing <i>file</i> to a <i>directory</i> ; converse of <i>unlink</i> . 2. by extension, a <i>directory entry</i> . 3. any <i>link[2]</i> or <i>symbolic link</i> for a given <i>inode</i> .
link count	the number of <i>directory entries</i> that pertain to an <i>inode</i> . A <i>file</i> ceases to exist when its link count becomes zero and it is not <i>open</i> .
load device	designates the physical <i>device</i> from which a program will be loaded into main <i>memory</i> .
log files	contain records of transactions that occur on the system. For example, software that <i>spools</i> generates various log files.
logical block	a unit of data as it is handled by the software.
login	1. the act of <i>logging on</i> : a <i>user</i> identifies himself or herself to the operating system by supplying a <i>login name</i> (and the correct <i>password</i>) in response to prompts; the system then creates a <i>process</i> that runs on the user's behalf. 2. an abbreviated form of <i>login name</i> . 3. the <i>program</i> that controls logging on. 4. by extension, the computing session that follows a login [1].
login name	a unique character string that identifies a <i>user</i> to the operating system.
machine alias	an abbreviated notation for a collection of remote machines; machine aliases can be used on command lines to simplify the specification of destination machines.

- maintenance mode the state of a *secure system* when only a *trusted administrator* has access to it, usually for the purpose of running diagnostics or performing maintenance on the system or the software.
- Mandatory Access Control (MAC)
the mechanism controlling *access* of *objects* among *subjects*. Unlike *Discretionary Access Control*, MAC is controlled by the *Trusted Computing Base* and is independent of the object owner's ability to grant discretionary access. The TCB enforces MAC by the assignment of a *security level* (recorded in or on a *sensitivity label*) to the information contained in the object.
- maximum set of process privileges
the set of all *process privileges* a process is allowed to use. A process may delete privileges from its maximum set, but it may not add them. If the process does not have a process privilege, it will not be allowed to perform the privileged function; see *process privileges* and *working set of process privileges*.
- memory 1. See *core image*. 2. physical memory representing the available space in main memory; *programs* are either *swapped* or *paged* into physical memory for *execution*. 3. virtual memory management techniques permitting *programs* to treat *disk* storage as an extension of main memory.
- mode, file mode the *permissions* of a *file*; referred to by a 3-digit octal number, for example "a 755 file"; see **chmod (1)**.
- mount extends the *directory hierarchy* by associating the *root* of a *file system* [1] with a *directory entry* in an already mounted file system; the converse is to unmount using the `umount` system call.
- mounted file system range
the range of the *security levels* of the information that can be stored on a file system.
- multilevel device a device which separately and simultaneously processes, displays, or stores data of different *security levels* without risk of *compromise*. To accomplish this separation of data, *sensitivity labels* are normally stored on the same physical medium and in the same form (that is, machine-readable or human-readable) as the data. These devices include disks that contain *multilevel directories*, archival and storage media, and administrative devices such as `/dev/kmem` and `/dev/mem`.
- Multilevel Directory (MLD)
a directory, such as `/tmp`, which allows *untrusted* processes to create files at different *security levels* within that same directory.
- namelist See *symbol table*.
- National Computer Security Center (NCSC)
the agency of the government of the United States of America which *certifies* computing systems as being *trusted*.

network	the hardware and software that connect computer systems, permitting electronic communication between systems and associated peripherals.
Network Administrator (NET)	the administrator responsible for <i>networking</i> when the Enhanced Security Utilities are installed.
networking	for computer systems, means sending data from one system to another over some communications medium (coaxial cable, phone lines, and so on). Common networking services include <i>file transfer</i> , remote <i>login</i> , and remote <i>execution</i> .
node name	a character string that identifies a single computer in a <i>network</i> . The node name may contain up to eight characters; it resides in the NODE parameter.
normal operations	The usual, multi-user state of a <i>trusted computer system</i> , when <i>users</i> can <i>login</i> .
null device	a <i>device</i> [1] that always yields end-of-file on reading and discards all data on writing.
object	anything that contains or receives information. <i>access</i> to an object implies access to the information it contains. Examples of objects are: <i>processes</i> , <i>blocks</i> , <i>files</i> , <i>directories</i> , <i>pipes</i> , <i>programs</i> , bits, bytes, processors, keyboards, clocks, and printers.
object file	a <i>file</i> of machine language code and data. Object files are produced from source programs by compilers and from other object files and libraries by the link editor. An object file that is ready to run is an <i>executable file</i> [1].
Object Reuse (OR)	the reassignment, from one process to another, of a storage <i>object</i> (such as a segment of memory, disk block, or magnetic tape) that contains data owned by the original “owner process.” In order to be securely reassigned, such media must contain no residual data from the previous owner.
open file	1. the destination for input or output obtained by <i>opening a file</i> or creating a <i>pipe</i> . Open files are shared across <i>forks</i> and persist across <i>executes</i> [2]; see <i>file descriptor</i> . 2. a file that has been opened; however, an <i>open file</i> [1] need not exist in a <i>file system</i> [1], and a <i>file</i> [2] may be the destination of several <i>open files</i> simultaneously.
Operator (OP)	An <i>untrusted</i> administrator. The operator performs regular, non- <i>secure</i> activities, such as starting and stopping the system, and generating raw accounting data, but not changing <i>passwords</i> , <i>login levels</i> , or other <i>security</i> -related login parameters.
operating system	the <i>program</i> for managing the resources of the computer. It provides tools for handling basic operations such as input/output procedures and process scheduling, thus rendering unnecessary <i>user programs</i> for maintaining these functions.

other	1. a set of <i>permissions</i> regulating <i>access</i> to a <i>file</i> by <i>processes</i> with a <i>user ID</i> different from that of the <i>owner</i> , and a <i>group ID</i> different from that of the <i>group</i> for that file. 2. the customary name of the default <i>group</i> [2] assigned upon <i>login</i> .
output	information that has been <i>exported</i> by the <i>Trusted Computing Base</i> .
overt channel	a communications path within a network that is designed for the authorized transfer of data.
owner	the <i>user ID</i> of the <i>process</i> that created a <i>file</i> .
page	a fixed length, 4096-byte block that has a virtual <i>address</i> , and that can be transferred between main and secondary storage.
paging	a memory-management technique in which <i>programs</i> are organized into <i>pages</i> that can be transferred between main and secondary storage by the virtual handler (or paging <i>daemon</i>).
parent process	See <i>fork</i> .
partitions	units of storage space on disk.
password	a character string typed by a <i>user</i> during the login process. The user is prompted to type a password after entering his or her <i>login name</i> . By correctly supplying the password, a user proves he or she has both authorization to <i>access</i> the account and the <i>privileges</i> associated with the <i>login name</i> . By keeping the password a secret, users help protect a system from unauthorized access.
path, pathname	a chain of <i>directory</i> names ending in a <i>filename</i> , that shows the location of a file in the <i>file system</i> . The location is expressed as a list of the <i>directories</i> that must be traversed to access the specified file. There are two types of paths: relative and full. Relative paths show how to reach the location of a file from the directory in which you're currently working; full paths, from the / (root) directory. Thus, for example, to designate a file called file_x while you're working in a directory called B , use the following relative path: B/C/file_x . Because the first element of a full path is the / (root) directory, the full path for this example is /A/B/C/file_x .
permission	1. a right to access a <i>file</i> for one, two, or three purposes: to <i>read</i> , <i>write</i> , and/or <i>execute</i> the file. Because permission for each function is granted separately to <i>owner</i> , <i>group</i> , and <i>other</i> , nine permissions are associated with every file. 2. an abbreviated form of <i>permission bit</i> .
permission bit	a permission, so called because each permission is encoded into one bit in an <i>inode</i> .
physical block	a unit of data as actually stored and manipulated.
physical memory	See <i>memory</i> .

pipe	a direct stream connection between <i>processes</i> , whereby data written on an <i>open file</i> in one process becomes available for reading in another.
pipeline	a sequence of <i>programs</i> [1] connected by <i>pipes</i> .
plain file	A <i>file</i> , containing text or data, that is not executable. See <i>executable file</i> .
polling	the interrogation of <i>devices</i> by the <i>operating system</i> to avoid contention, determine operation status, or ascertain readiness to send or receive data.
ports	the point of physical connection between a peripheral <i>device</i> (such as a terminal or a printer) and the device <i>controller</i> (ports board), which is part of the computer hardware.
primary name server	the computer on which the <i>Remote File Sharing Utilities</i> are installed and maintained.
privilege	the ability to override system restrictions and thus <i>access</i> or use a system call, function, resource, or program; see <i>least privilege</i> and <i>process privileges</i> .
privileged process	a <i>process</i> that has at least one of the <i>process privileges</i> .
process	a sequence of computations characterized by a <i>core image</i> with instruction location counter, current directory, <i>open files</i> , control terminal, <i>user ID</i> , and <i>group ID</i> ; a program in execution. A process is characterized by a single current execution point and address space. A process is a <i>subject</i> when it requests some action; an <i>object</i> , when it receives data or a signal.
process ID	an integer that identifies a <i>process</i> .
process number	See <i>process ID</i> .
process privilege	authorization for a process to perform sensitive operations.
profile	1. an optional <i>shell script</i> , .profile , used by the <i>shell</i> on <i>logging in</i> to establish an <i>environment</i> [2] and other working conditions for a particular <i>user</i> . 2. to collect an instruction location counter value histogram of a <i>process</i> .
program	1. an <i>executable file</i> . 2. a <i>process</i> .
programmer	a <i>user</i> who writes code. A programmer needs to know which system commands, system calls, and libraries are affected by the <i>Trusted Computing Base</i> and how they are affected.
pseudo-device	a special <i>object</i> in the system used to perform special-purpose system functions; examples of pseudo-devices are memory pseudo-devices and pseudo-terminals. Pseudo-devices have associated <i>Device Special Files</i> and the same access semantics as physical devices, but they may not have associated device hardware.

queue	a line (or list) of items waiting for service in a system.
RAID	Redundant Arrays of Independent Disks.
raw device	a <i>block device</i> for which read and write operations are synchronized to natural records of the physical <i>device</i> (not <i>buffered</i>).
read	a fundamental operation that results only in the flow of information from an <i>object</i> to a <i>subject</i> .
read access	authorization for a <i>subject</i> to <i>read</i> the information in an <i>object</i> .
reboot	See <i>boot</i> .
reference monitor concept	at levels B3 and above (see <i>certification</i>), equivalent to the <i>Trusted Computing Base</i> ; the smallest section of code that needs to be <i>trusted</i> and that mediates all security-relevant decisions.
region	a group of machine <i>addresses</i> that refer to a base address.
release	one of multiple, sequentially produced versions of a software product, each of which contains improvements on the last. a distribution of fixes or new functions for an existing software product.
Remote File Sharing	a software utilities package that enables the users of multiple computers to share <i>resources</i> across a <i>network</i> .
resource	a directory that is <i>advertised</i> in a <i>Remote File Sharing</i> environment. When a <i>resource</i> is <i>mounted</i> on a <i>client</i> , its contents (files, devices, named pipes, and subdirectories) are potentially available to users on the <i>client</i> .
resource channels	are <i>channels</i> inherent in the design of the operating system kernel, consisting mainly of internal data structures shared by <i>processes</i> of different <i>security levels</i> ; resource channels include the system file table, the system lock table, disk blocks, STREAMS buffers, proc and u structures, file system inodes, and IPC structures.
re-tension	the process of rewinding a tape in a <i>cartridge tape device</i> to get the amount of tautness necessary for accurate recording of data.
role	a named description of a given administrative function regardless of the individual identity of the administrators to which the function is assigned. It is a convenient way of assigning privileges to a group of users who will be doing the identical type of work (also see “administrative role”).
root	1. a distinguished directory that constitutes the origin of the <i>directory hierarchy</i> in a <i>file system</i> [1]. 2. specifically, the origin for the <i>file system</i> [2], with the conventional <i>pathname</i> “/”. 3. the origin of the directory hierarchy in a <i>file system</i> [1].

rotational gap	the gap between the actual <i>disk</i> locations of blocks of data belonging to the same <i>file</i> . The rotational gap compensates for the continuous, high-speed rotation of the disk so that when the controller is ready to reference the next physical block the read-write head is positioned correctly at the beginning of that block.
run level	See <i>system</i> state.
schedule	to assign resources (main store and CPU time) to <i>processes</i> .
scheduler	a permanent <i>process</i> (with <i>process number</i> 0 and associated <i>kernel</i> facilities) that determines the order in which various processes are executed.
search path	in the <i>shell</i> , a list of <i>pathnames</i> that determine the directories in which a desired file will be sought and the order in which they will be investigated. The command name is prefixed with members of the search path in turn until a pathname of an <i>executable file</i> [2] results; the search path is given by the shell variable <code>PATH</code> .
secondary name server	a <i>host</i> that is configured to take over <i>domain name server</i> responsibilities temporarily in case the <i>primary name server</i> goes down.
sector	A 512-byte portion of a <i>track</i> that can be accessed by magnetic disk heads in the course of a predetermined <i>rotational</i> displacement of the storage device.
secure	describes the assurance that the hardware and software comprising a computer system are <i>trusted</i> , and that the mechanisms, policies, and procedures governing the use of that computer system are enforced to protect against unauthorized <i>access</i> to or modification or destruction of the <i>Trusted Computing Base</i> or <i>sensitive information</i> .
Secure Attention Key (SAK)	a character or asynchronous line condition (for example, break, line drop) that a <i>user</i> must enter to invoke the <i>Trusted Path</i> ; note that several keys can be required to enter one character, as with a control character. The <i>system administrators</i> define the default SAK for each <i>terminal</i> on the system; the active SAK is the one in effect for a given terminal.
security	1. the mechanisms and techniques that control <i>access</i> to a <i>trusted computer system</i> . 2. the assurance that these mechanisms and techniques are functioning correctly. 3. the state of being <i>secure</i> .
Security Administrator	an authorized administrator responsible for the security of computer system; see <i>System Security Officer</i> .
security level	the combination of a hierarchical <i>classification</i> and a set of non-hierarchical <i>categories</i> that represents the <i>sensitivity</i> of information.

security level alias	a name applied to a <i>security level</i> for easy reference.
Security Operator (SOP)	a <i>trusted</i> administrator. The Security Operator performs routine, daily activities similar to those performed by the <i>Operator</i> ; however, certain of these activities are <i>security</i> -related and thus restricted to the Security Operator. Because these activities are routine and would require the efforts of more than one person, there can be several Security Operators. The Security Operator can perform all of the activities of the <i>Operator</i> .
security policy	the set of rules, concepts, and practices that regulate the organization, management, protection, and distribution of <i>sensitive information</i> . With the Enhanced Security Utilities, the security policy states that a <i>subject</i> can read only those <i>objects</i> that are <i>dominated</i> by the subject's <i>level</i> , and can write only to those objects at the same level.
segment	a contiguous address space range of a <i>process</i> with consistent <i>read</i> , <i>write</i> , and <i>execute</i> capabilities. For example, three common segments are (i) the text segment, containing read-only instructions and data; (I) the data segment, containing static data that is explicitly initialized; (iii) the bss segment, containing static data that is initialized to zero.
semaphore	an IPC facility which allows two or more processes to be synchronized.
sensitive	describes any information, any action, or any part of a <i>trusted computer system</i> which involves <i>security</i> and to which <i>access</i> is restricted.
sensitive information	information that must be protected because its unauthorized disclosure, alteration, loss, or destruction will cause harm or damage.
sensitivity	the degree to which information is restricted and requires special authorization in order to be <i>accessed</i> .
sensitivity label	data that represents the <i>security level</i> of an <i>object</i> , that describes the <i>sensitivity</i> of the data in the object, and that can be changed only by a <i>privileged process</i> via <code>chlvl</code> . The <i>Trusted Computing Base</i> enforces <i>Mandatory Access Control</i> by comparing the sensitivity label of an object with the security level of a <i>subject</i> trying to <i>access</i> the object. Often mistakenly referred to as "security label."
server	a <i>host</i> that actively shares one of its <i>advertised resources</i> with another <i>host</i> in a <i>Remote File Sharing</i> environment.
set user ID	a special <i>permission</i> for an <i>executable file</i> [1] that causes a <i>process</i> executing it to have the access rights of the <i>owner</i> of the file. The owner's <i>user ID</i> becomes the effective user ID of the process, distinguished from the real user ID under which the process began.

set user ID bit	the associated <i>permission bit</i> .
shared memory	an IPC facility that allows two or more processes to share the same data space.
shared text	Shared text is a text segment, one copy of which may be used simultaneously by more than one process.
shell	1. the program sh(1) , which causes other programs to be executed on command; the shell is usually started on a user's behalf when the user <i>logs in</i> . 2. by analogy, any program started upon logging in.
shell script	an executable <i>file of commands</i> taken as input to the <i>shell</i> .
signal	an exceptional occurrence that causes a <i>process</i> to terminate or divert from the normal flow of control; see <i>interrupt</i> , <i>trap</i> .
single-level device	a device that processes, display, or stores data of a single <i>security level</i> at any one time. Because the device need not separate data of different security levels, <i>sensitivity labels</i> do not need to be stored with the data. These devices include <i>terminals</i> , storage devices such as magnetic tape, and administrative devices such as /dev/console .
single-user	a <i>system state</i> in which only one user is supported.
Site Security Officer	See <i>System Security Officer</i> .
source file	1. the uncompiled version of a <i>program</i> . 2. generally, the unprocessed version of a <i>file</i> .
special file	an <i>inode</i> that designates a <i>device</i> , further categorized as either (i) a block special file describing a <i>block device</i> , or (ii) a character special file describing a <i>character device</i> .
spoofing	the act of creating a hoax on a computer system, usually with malicious intent; generally, a program that masquerades as part of the <i>Trusted Computing Base</i> in order to trick a <i>user</i> or other <i>subject</i> into revealing <i>sensitive information</i> . A typical instance of spoofing is a program that appears to be the <i>login</i> program and tricks the user into supplying a <i>password</i> .
spool	to collect and serialize output from multiple <i>processes</i> competing for a single output service.
spool area	a <i>directory</i> in which a spooler collects work.
spooler	a <i>daemon</i> that spools.
stack	a <i>segment</i> of the <i>address</i> space into which data and subroutine linkage information is allocated in last-in-first-out fashion.
standard error	one of three files described below under <i>standard output</i> .
standard input	the second of three files described below under <i>standard output</i> .

standard output	<i>open files</i> with <i>file descriptors</i> 0, 1, and 2, and <i>stdio</i> names <code>stdin</code> , <code>stdout</code> , and <code>stderr</code> , respectively. Where possible, utilities read from the standard input, write on the standard output, and place error comments on the standard error file.
static data	static represents a condition persistent throughout a process. Said of data. Static data occupies the data segment and the bss segment.
startup	See <i>boot</i>
<code>stdio</code>	(standard I/O) a library of efficient and portable I/O routines; the header file <code>/usr/include/stdio.h</code> contains definitions and declarations; see <code>stdio(3S)</code> .
sticky bit	a <i>permission</i> flag that identifies a file as a <i>sticky file</i> .
sticky file	a special <i>permission</i> for a shared text file that causes a copy of the text <i>segment</i> to be retained in the <i>swap area</i> to improve system response.
storage object	an <i>object</i> that supports both <i>read</i> and <i>write accesses</i> .
stream	A kernel aggregate created by connecting STREAMS components, resulting from an application of the STREAMS mechanism. The primary components are the Stream head, the driver, and zero or more pushable modules between the Stream head and driver.
striping	A technique that spreads data across several physical disks or using stripes. The data is allocated alternately to the stripes within the subdisks of each mirror.
striped VP	Striped VP (virtual partition) divides the contiguous data of the VP into slices.
subdirectory	A <i>directory</i> pointed to by a directory one level above it in the file system organization; also called a child directory.
subject	a <i>process</i> ; anything that causes information to flow among <i>objects</i> or that changes the system state. The initial process for a <i>user</i> is the shell invoked by <i>login</i> . A process is assigned a <i>security level</i> when it is created. The shell is established with the user's <i>security level</i> , as determined by the Identification & Authentication mechanism. Additional processes inherit the security level of the invoking process.
super-block	the second <i>block</i> in a <i>file system</i> [1], which describes the allocation of space in the file system; see <i>boot block</i> .
super user	<i>user ID</i> 0, which can access any <i>file</i> regardless of <i>permissions</i> and can perform certain privileged <i>system calls</i> , for example, setting the clock.
Super User Module	Super User Module (SUM). This privilege mechanism functions exactly the same as the superuser, a UID-based mechanism common to earlier releases of the OS. It also provides additional

flexibility by allowing the association of fixed privileges with executable files. This mechanism works by using: 1) a list of system privileges, 2) a working and maximum set of privileges for each process on the system, and 3) a fixed set of privileges for files.

swap	to move the <i>core image</i> of an executing program between main and secondary storage to make room for other <i>processes</i> .
swap area	the part of secondary store to which <i>core images</i> are <i>swapped</i> ; the swap area is disjointed from the <i>file system</i> .
symbolic link	an <i>inode</i> that contains the <i>pathname</i> of another <i>inode</i> . References to the symbolic link become references to the named <i>inode</i> .
symbol table	information in an <i>object file</i> about the names of data and functions in that file; the symbol table and <i>address</i> relocation information are used by the link editor to compile <i>object files</i> and by debuggers.
System Administration	when capitalized, refers to the menu interface for administrative tasks that's invoked through the sysadm(1) command.
System Administrator	a person who supervises the running of a computer system (or part of one). With Enhanced Security Utilities, the secure system administrators include the <i>Site Security Officer</i> , the <i>Auditor</i> , the <i>Security Operator</i> , and the <i>Network Administrator</i> .
system calls	1. the set of system primitive functions through which all system operations are allocated, initiated, monitored, manipulated, and terminated. 2. the system primitives invoked by user <i>processes</i> for system-dependent functions, such as I/O, process creation, and so on.
system console	the directly connected terminal used for communication between the operator and the computer.
system high	1. the highest <i>security level</i> supported by a <i>trusted computer system</i> at a given time; 2. a system where every <i>object</i> is assigned the highest possible security level.
system low	the lowest <i>security level</i> supported by a <i>trusted computer system</i> at a given time.
system name	an up-to-eight character name for the system; resides in the SYS parameter.
System Security Officer (SSO)	an <i>authorized</i> and <i>trusted system administrator</i> responsible for the security of a <i>trusted computer system</i> . The System Security Officer is also known as the <i>Security Administrator</i> . With the exception of security <i>audit</i> , The SSO performs the most critical <i>security-related</i> functions, excluding security <i>audit</i> .
system state	one of five configurations of the operating system for which a predetermined set of processes can be executed. The operating

	system is running in one of these states at any given time. (Also known as “run level” and “init state.”)
table	an array of data in which each item may be uniquely identified by one or more arguments.
terminal	an I/O device connected to a computer system, usually consisting of a keyboard with a video display or printer, allowing a <i>user</i> to give the computer instructions and to receive information in response. On <i>trusted computer systems</i> , we assume that only directly connected “dumb” terminals are used, because “smart” terminals may <i>compromise security</i> .
text file	(or ASCII file) a <i>file</i> which contains ASCII code.
text segment	part of a contiguous range of the address space of a process, a text segment is occupied by executable code. A text segment is a contiguous range of the address space of a process with consistent store access capabilities.
track	an addressable ring of <i>sectors</i> on a <i>disk</i> or <i>diskette</i> ; each disk or diskette has a predefined number of concentric tracks, which allows the disk head to properly access <i>sectors</i> of data.
trap	a method of detecting and interpreting certain hardware and software conditions via software. A trap is set to catch a <i>signal</i> (or <i>interrupt</i>), and determine what course of action to take.
trap door	a hidden software or hardware mechanism that permits the protection mechanisms of the <i>Trusted Computing Base</i> to be circumvented.
trojan horse	software with apparent or actual useful functions that contains additional, often hidden functions that surreptitiously violate the protection mechanisms of the <i>Trusted Computing Base</i> by exploiting the legitimate <i>authorizations</i> and <i>privileges</i> of invoking <i>processes</i> .
trusted	describes a component of the computer system that can be relied upon to enforce the system's <i>security policy</i> .
trusted computer system	a computer system on which a <i>security policy</i> is enforced by a <i>Trusted Computing Base</i> ; also known as a <i>trusted system</i> .
Trusted Computing Base (TCB)	the Trusted Computing Base is the totality of the software, firmware, and hardware that enforces a <i>security policy</i> ; the ability of the TCB to enforce a security policy correctly depends solely on the mechanisms within the TCB and on the correct input by the <i>system administrators</i> of parameters related to <i>security policy</i> .
Trusted Path (TP)	the link between the <i>user</i> and the <i>trusted computer system</i> during <i>login</i> processing; the Trusted Path is <i>trusted</i> because every step in establishing the link between the user and the TCB is trusted; the

Trusted Path is intended to help prevent *spoofing*; see *Secure Attention Key*, *identification*, and *authentication*.

trusted process channels	<i>channels</i> of low <i>bandwidth</i> that occur when <i>trusted processes</i> , usually those associated with <i>privileged</i> , non-administrative commands, are unwittingly used as <i>covert channels</i> .
trusted software	the software portion of the <i>Trusted Computing Base</i> .
trusted system	See <i>trusted computer system</i> .
tunable parameters	variables used to set the sizes and thresholds of the various control structures of the <i>operating system</i> .
tuning	<ol style="list-style-type: none">1. modifying the <i>tunable parameters</i> to improve system performance.2. reconfiguring an <i>operating system</i> so that modifications are incorporated into an <i>executable</i> version of the system.
untrusted	not <i>trusted</i> ; a component of the computer system whose incorrect or malicious execution cannot affect the <i>security</i> of the system, because it has no control over security-related actions.
user	any person who interacts directly with a computer system. With Enhanced Security Utilities, a user needs to know what system commands are affected by the <i>Trusted Computing Base</i> and how they are affected. The user's access to the TCB is determined by the <i>System Security Officer</i> .
user ID (UID)	an integer value, usually associated with a <i>login name</i> . The user ID of a <i>process</i> becomes the <i>owner</i> of files created by the process and descendent (<i>forked</i>) processes. The <i>Trusted Computing Base</i> uses the user ID along with the <i>group ID</i> and <i>Access Control Lists</i> to enforce <i>Discretionary Access Control</i> .
utility, utility program	a standard, generally useful, permanently available <i>program</i> .
verification	the process by which an authority, such as the <i>National Computer Security Center</i> , compares two levels of system specification for proper correspondence. For example, the security policy model is compared with the design documentation, the design documentation with the source code, and the source code with the object code.
version	a separate <i>program</i> product, based on an existing one, but containing significant new code or new functions.
virus	a particularly malicious form of <i>trojan horse</i> that reproduces itself or other executable code.
virtual memory	See <i>memory</i> .
VTOC	the Volume Table Of Contents is a table that shows how the <i>partitions</i> on the <i>disk</i> are allocated.

working set of process privileges	a subset of the <i>maximum set of process privileges</i> , the working set consists of those <i>privileges</i> that a <i>process</i> has while performing a given function at a given time; the process can freely alter its working set by using the <code>procpriv</code> system call, but each working set must be a subset of the <i>maximum set</i> ; see <i>maximum set of process privileges</i> .
write	a fundamental operation that results only in the flow of information from a <i>subject</i> to an <i>object</i> .
write access	authorization for a <i>subject</i> to <i>write</i> information to an <i>object</i> .

Volume 2 Index

Symbols

.colorpref file 14-1
/etc/conf/mod.d 8-3
/etc/conf/modnew.d 8-3
/etc/conf/sadapters.d/kernel 8-2
/etc/conf/sdevice.d 8-2
/etc/inittab 8-3
/stand/unix 8-1
/unix 8-1
/usr
 lib/lpsched 12-38

A

a.out(4) Glossary-1
accept(1) 12-33, 12-36
access Glossary-1
 Access Control List Glossary-1
 exclusive Glossary-7
 permission Glossary-1
 read Glossary-15
 write Glossary-23
adapters configuration file 8-2
adding
 CD-ROM Drive 2-79
 disk drive 2-76
 ethernet controller device 2-82
 FDDI Controller Device 2-82
 tape drive 2-80
address Glossary-1
administrative role Glossary-1
alias
 level Glossary-10
 machine Glossary-10
allocation units Glossary-1
archive Glossary-1
attribute mapping for printers 13-3, 13-5
attributes
 device 2-11, 2-45 to 2-55
audit Glossary-1
audit trail Glossary-2

Auditor Glossary-2
authentication Glossary-2
authorization Glossary-2
auto-load 6-6
automatic calling unit Glossary-2
auto-unload 6-8

B

backup 1-5
backup commands
 quick reference 11-44 to 11-47
 task and command summary 11-43 to 11-44
backup exception list 11-14 to 11-20
 convert from earlier backups 11-18 to 11-20
 create a customized list 11-14 to 11-18
 ignore during backup 11-14
Backup file 11-2
backup menu
 sysadm 11-43 to 11-44
backup methods
 migration 11-21 to 11-22
backup(1M) 11-2
backups
 add or change entries in tables 11-11 to 11-12
 background 11-27 to 11-30
 Backup file 11-2
 bkexcept.tab backup table 11-14 to 11-20
 bkhist.tab backup table 11-39 to 11-42
 bkreg.tab backup table 11-8 to 11-10
 bkstatus(1M) 11-36 to 11-38
 bkstatus.tab backup table 11-36 to 11-38
 checking status of 11-36 to 11-38
 complete method 11-3
 control of 11-38 to 11-39
 core file system 11-22
 creating tables of contents 11-26 to 11-27
 customizing backup tables 11-10 to 11-12
 demand 11-25, 11-32 to 11-33
 dependencies and priorities 11-25 to 11-26
 display backup tables 11-10
 examples of 11-3
 full data partition method 11-20

- full disk method 11-20
- full file method 11-14
- full image method 11-20
- history log of 11-39 to 11-42
- Ignore file 11-2
- importance of 10-1, 11-1
- incremental file method 11-14 to 11-20
- interactive 11-27 to 11-30
- label checking override 11-35
- limited 11-32 to 11-33
- log 11-39 to 11-42
- media preparation and handling 11-3
- mode 11-27 to 11-30
- monitoring 11-33 to 11-36
- OA&M menus and 11-42 to 11-44
- of user home directories 11-4
- overview of 10-1, 11-1, 11-4
- partial method 11-3
- planning 11-4 to 11-8
- previewing operations for 11-31 to 11-32
- quick reference to 11-44 to 11-47
- rotation period 11-24
- selected files method 11-3
- selecting an operator mode 11-28
- specifying destination devices 11-23 to 11-24
- specifying originating objects 11-22 to 11-23
- specifying the rotation period 11-24 to 11-25
- status 11-36 to 11-38
- validating tables 11-8 to 11-10
- backups, suspend, resume
 - and cancel 11-38
- bad blocks 2-41
- bandwidth Glossary-2
- Basic Networking Utilities (BNU) 1-11
- Bit Map 2-22
- bkexcept(1M) 11-14 to 11-20
- bkexcept.tab backup table 11-14 to 11-20
- bkhist.tab backup table 11-39 to 11-42
- bkhistory(1M) 11-39, 11-40, 11-41, 11-42
- bkoper(1M) 11-33 to 11-36
- bkreg(1M) 11-8 to 11-10
- bkreg.tab backup table 11-8 to 11-10
- bkstatus(1M) 11-36 to 11-38
- bkstatus.tab backup table 11-36 to 11-38
- block Glossary-2
 - bad Glossary-2
 - bad block numbers 4-6
 - device Glossary-2
 - directory data 4-7
 - duplicate 4-6
 - free 4-4 to 4-5
 - indirect Glossary-9
 - logical Glossary-10
 - physical Glossary-13

- size limitation 3-4
- size limitations 3-3
- super 4-4, Glossary-19
- ufs
 - free 3-9
 - storage 3-9
 - super 3-6
- xfs
 - size 3-14
- block devices 2-2, 2-10, 2-40
 - path of node for 2-47
- block size
 - choosing logical 3-32
- Blocks 2-17
- Bookkeeping Information 2-22
- boot Glossary-2, Glossary-15
 - program Glossary-2
 - startup Glossary-19
- booting the system 1-9
- buffer Glossary-2
 - pool Glossary-2
- building the kernel 8-3

C

- cartridge tape Glossary-3
- cartridge tape devices 2-47
- category Glossary-3
- CD-ROM
 - how to add 2-79
- CD-ROM devices 2-1
- CD-ROM drives 1-11
- certify Glossary-3
- channel Glossary-3
 - covert Glossary-4
 - exploitable Glossary-7
 - functional Glossary-8
 - overt Glossary-13
 - resource Glossary-15
- Chapter 1 1-1
- character device Glossary-3
- character devices 2-39 to 2-40
- character sets 12-16 to 12-18
- Checking The Integrity Of The File Systems 2-27
- class
 - scheduler (see scheduler class) 9-2
- Classes Of Devices 2-26
- clients Glossary-3
- command
 - file Glossary-3
- communication

- interprocess (IPC) Glossary-10
- Components Of The Device 2-18
- compromise Glossary-3
- con fig(1M) 7-1
- config utility, access requirements 8-7
- config utility, current configuration 8-6
- config utility, invoking config 8-6
- config utility, kernel hardware adapters menu 8-13
- config utility, kernel module menu 8-11
- config utility, kernel options menu 8-17
- config utility, kernel rebuild menu 8-16
- config utility, main menu functions 8-9
- config utility, menu operations 8-7
- config utility, object directory 8-6
- config utility, other menu functions 8-18
- config utility, output navigation 8-8
- config utility, system tunable menu 8-9
- config utility, terminal attributes 8-5
- config(1M) 8-1, 9-10
- configuration
 - management Glossary-3
- Configuration the Kernel
 - Optional Modules 8-19
- configuration util ity 7-1
- configuration utility 8-1
- configuring kernel modules 8-2
- configuring network printers 13-2 to 13-7
- Configuring the Kernel 8-18
 - Required Modules 8-18
- console
 - system Glossary-20
 - terminal Glossary-4
- controller (network) Glossary-4
- copying a file system 3-34, 3-35, 3-36
- copying files
 - disk to disk 2-43
- core
 - file Glossary-4
 - file system backup 11-22
 - image Glossary-4
- covert
 - channel Glossary-4
 - storage channel Glossary-4
 - timing channel Glossary-4
- cpio(1) 3-33, 5-3, 5-7
- crash Glossary-4
- crash(1M) 3-31
- Create a ufs File System 3-33
- Create an sfs File System 3-33
- creating
 - xfsd file system 3-24
- Creating an xfs FS 3-19
- Creating New File Systems 2-26
- Creating The Empty File System 2-26

- cron Glossary-4
- cron(1M) 5-9
- current directory Glossary-4
- Customizing the sysadm Interface 15-1
- cylinder (disk) Glossary-4
- Cylinder Group 2-20
- Cylinder Groups 2-16

D

- daemon Glossary-5
 - spooling 13-43
- Daily And Weekly Backups 10-9
- Data Blocks 2-23
- data integrity Glossary-5
- data validation tools
 - characteristics 16-26
 - error messages 16-28
 - formatting 16-28
 - help messages 16-27
 - list of 16-29 to 16-30
 - prompts 16-27
 - purpose 16-25
 - types 16-26
- date
 - setting 1-8
- dcopy(1M) 3-28
- dd(1M) 3-33, 3-34
- DDB (see Device Database) 2-3
- ddb (see Device Database) 2-1
- ddb_dsfmap (see Device Database) 2-1
- ddb_index (see Device Database) 2-1
- delsysadm(1M) 15-1, 15-3, 15-3 to 15-7, 15-20 to 15-21, 16-1, 16-2
- demand backups 11-25, 11-32 to 11-33
- destination Glossary-5
- destination devices
 - specifying for backups 11-23 to 11-24
- devattr(1M) 2-53 to 2-54
- devfree(1M) 2-59, 2-60
- device Glossary-5
 - activation Glossary-5
 - allocation Glossary-5
 - block Glossary-2
 - character Glossary-3
 - configuration Glossary-3
 - Device Special File (DSF) Glossary-6
 - disk Glossary-6
 - diskette Glossary-6
 - drive Glossary-7
 - driver Glossary-5
 - level range Glossary-5

- load Glossary-10
- multilevel Glossary-11
- null Glossary-12
- pseudo Glossary-14
- raw Glossary-15
- readiness state Glossary-5
- setup Glossary-6
- single-level Glossary-18
- state Glossary-5
- Device Control Information Glossary-5
- Device Database 2-3, 2-45 to 2-55, Glossary-5
 - adding entries to 2-35, 2-46 to 2-49
 - removing entries from 2-38, 2-55
- device drivers 6-1
- device files
 - block 2-39 to 2-41
 - character 2-39 to 2-41
- device management menu 2-69
- device number
 - major 2-2
 - minor 2-2
- device.tab (see Device Database) 2-1
- Devices 2-14
- devices
 - adding 2-35, 2-46 to 2-49
 - alias 2-46 to 2-47
 - attributes 2-11, 2-45 to 2-55
 - block 2-2, 2-10, 2-41
 - character 2-1 to 2-10, 2-39 to 2-41
 - checking available devices and media 2-42
 - checking reservations for 2-59 to 2-60
 - commands for 2-70
 - database (see Device Database) 2-45
 - displaying information about DLMS 6-4
 - drivers 2-2
 - formatting 3-3
 - groups of 2-3 to 2-55
 - identifying special 2-39 to 2-41
 - listing 2-51, 2-53
 - menu 2-69
 - partitions
 - creating 3-3
 - quick reference to 2-69 to 2-75
 - removing 2-35 to 2-39
 - reserving 2-3, 2-58 to 2-60
 - suggestions for managing 2-11 to 2-12
 - types of 2-1
- devreserv(1M) 2-58, 2-59, 2-60
- devstat(1M) 2-70
- df(1M) 3-28, 3-30, 3-39, 5-10
- diagnostics Glossary-6
- dial-up printers 13-7 to 13-9
- directory 3-1, Glossary-6
 - compressing 5-5 to 5-6
 - controlling size 5-5 to 5-8
 - data blocks 4-7
 - disconnected 4-8
 - entry Glossary-6
 - restoring 11-52
 - sub Glossary-19
 - tree Glossary-6
- Directory Links 2-25
- disable(1) 12-34
- Discretionary Access Control (DAC) Glossary-6
- disk Glossary-6
 - (see disk devices) 2-1
 - hard (see hard disk devices) 2-1
- Disk Capacity 2-14
- disk devices 2-1
 - formatting 2-41
 - naming 2-61
 - partitioning 2-61
- Disk Drive, how to add 2-76
- Disk Partitions HN5000 Systems 2-15
- disk slowdowns
 - checking for 5-42
- disk space
 - monitoring 3-39
- disk striping 2-62
- diskette Glossary-6
 - floppy (see floppy diskette devices) 2-1
- Dismounting 2-24
- dispadm(1M) 9-3, 9-13 to 9-15
- dispatcher (see scheduler) 9-8
- Dispatcher Parameter Table, Scheduling Class 8-15
- Distributed xfs FS 3-20
- DLM 6-1
- DMD
 - 630 14-7
- domain Glossary-6
- domain name server Glossary-6
- dominate Glossary-7
- DPTs 8-15
- driver
 - device Glossary-5
- drivers
 - device 2-2
- du(1M) 3-41, 5-10
- dump Glossary-7
- duplicate blocks 4-6
- dynamic kernel
 - advantages 5-43
- dynamic kernel support 8-4
- dynamic loading 8-3
- dynamic vs. static kernels 5-43
- dynamically loadable modules 6-1
 - automatic loading 6-6
 - automatic unloading 6-9

demand loading 6-7
demand unloading 6-9

E

edsysadm(1M) 15-1, 15-2, 15-3, 15-4, 15-6, 15-11,
15-14, 15-15, 15-17, 15-18, 16-1, 16-2
enable(1) 12-33
encryption Glossary-7
environment Glossary-7
Ethernet Controller
 how to add 2-82
exameoblock 2-30
Example
 create an xfsd file system 3-24
exclusive access Glossary-7
exec(2) Glossary-7
executable file Glossary-7
exportation Glossary-7

F

fc_dptbl(4) 9-9
FDDI Controller
 how to add 2-82
fdisk backup method 11-20, 11-20
fdp backup method 11-20
features of dynamic kernel 8-4
features of static kernel 8-4
ff(1M) 3-28
ffile backup method 11-14
field item help message
 description 16-11
 example 16-12
 format 16-11
FIFO Glossary-7
File 10-20
file
 command Glossary-3
 executable Glossary-7
 mode Glossary-11
 object Glossary-12
 open Glossary-12
 plain Glossary-14
 source Glossary-18
 special Glossary-18
 sticky Glossary-19
 system Glossary-8
 text Glossary-21
file descriptor Glossary-8

File System 3-33
 create sfs 3-33
 create ufs 3-33
file system
 copying a 3-34, 3-35, 3-36
 creating 3-31
 creating a 3-31
 maintenance 3-39 to 3-41
 menu for managing 3-42
 mounting 3-36 to 3-37
 moving a 3-34 to 3-36
 unmounting 3-37
File System Administration, xfs 3-18
File System Restore Utility 10-20
file system types
 commands for administering 3-28, 3-29, 3-42
 identifying unmounted 3-31
 listing installed 3-31
 memfs 3-27
 sfs 3-26 to 3-27, 3-33 to 3-34
 ufs 3-5 to 3-12, 3-33
file systems
 corrupt 2-43
 mounting 3-37
filename Glossary-8
files
 copying to disk 2-43
 monitoring growth of 3-39 to 3-40
 removing inactive 3-40 to 3-41
 restoring 11-52
filter Glossary-8
filters
 adding 13-28
 defining 13-17, 13-20 to 13-23
 defining with templates 13-23 to 13-28
 evaluating programs for use with 13-19 to 13-20
 examining 13-28
 options 13-22 to 13-23
 PostScript and 13-35
 print requests and 13-29
 removing 13-28
 restoring factory defaults 13-29
 tasks performed by 13-18 to 13-19
 type 13-22
fimage backup method 11-20
find(1) 3-36, 3-40, 3-41, 5-3, 5-5
fixed class
 scheduler class 9-4
 scheduler parameter table 9-5, 9-9
 scheduler policy 9-9
fixed priority
 process priority 9-8, 9-9
 scheduler class 9-5
 scheduler parameter table 9-5, 9-8 to 9-9

- scheduler policy 9-8 to 9-9
- fixed priority class 9-5, 9-8 to 9-9
- flaw Glossary-8
- floppy key Glossary-8
- flush Glossary-8
- fonts
 - downloading host-resident 13-42 to 13-43
 - installing and maintaining host-resident 13-40 to 13-41
 - installing and maintaining Post Script 13-38 to 13-39
 - managing PostScript resident 13-39 to 13-40
 - map table 13-41
 - specifying 12-16 to 12-18
 - storage of 13-41
- fork(2) Glossary-8
- format (disk) Glossary-8
- Formatting 2-27
- formatting Glossary-8
 - storage devices 3-3
- Formatting An SCSI Disk 2-27, 2-28
- Formatting HSA Disks with the Console Processor (HN5000 Systems Only) 2-32
- forms (printer) 12-20 to 12-21, 13-9 to 13-17
 - alerting to mount 13-13 to 13-15
 - defining 13-10 to 13-12
 - examining 13-16
 - mounting 13-15
 - removing 13-12
 - restricting user access 13-12 to 13-13
 - setting default destination 13-16
 - unmounting 13-15
- fp_dptbl(4) 9-5, 9-8 to 9-9, 9-13
- Fragments 2-17, 2-24
- fragments (ufs) 3-4
- free blocks
 - ufs 3-9
- free list Glossary-8
- free-blocks 4-4 to 4-5
- fsck(1M) 2-43, 3-28 to 3-30, 3-37, 4-1
 - description of 4-2 to 4-3
 - sfs file system 4-29 to 4-31
 - ufs file system 4-3 to 4-29
- fsdb(1M) 3-28 to 3-30
- fstyp(1M) 3-28, 3-31
- FSTypes (file system types) 3-2
- full data partition backup 11-20
- full disk backup 11-20
- full file backup 11-14
- full image backup 11-20
- function keys
 - problems 14-7
 - use with sysadm(1M) 14-6
- fuser(1M) 3-37

G

- gap
 - rotational Glossary-16
- Geometry Block 2-18
- getdev(1M) 2-51, 2-53
- getdgrp(1M) 2-56 to 2-58
- getty(1M) Glossary-8
- global priorities 9-6 to 9-7
- group Glossary-8
 - adding 1-10
 - ID (GID) Glossary-9

H

- hard disk drives 1-11
- hardware adapters 8-2
- Hardware Connections
 - xfsd network 3-25
- help messages
 - default title example 16-11
 - setting up in an FMLI object 16-13
 - title hierarchy 16-12
 - writing 15-6 to 15-11, 16-9
- history log
 - backup 11-39 to 11-42
- hole (in a file) Glossary-9
- host Glossary-9

I

- ID
 - process Glossary-14
 - set user Glossary-17, Glossary-18
- ID mapping Glossary-9
- id tune 7-2
- id tune(1M) 7-1, 8-1
- Ignore file 11-2
- i-list Glossary-9
- importation Glossary-9
- incfile backup method 11-14 to 11-20
- incremental file backup 11-14 to 11-20
- indirect block Glossary-9
- infocmp 13-46
- init(1M) 1-9, Glossary-9
- INITCLASS 9-7
- inittab File 1-9
- inode
 - 4-8

- bad inode number 4-7
 - data associated with 4-7
 - format and type 4-5
 - link count 4-6
 - size 4-7
- inodes Glossary-9
 - i-list Glossary-9
 - i-number Glossary-9
 - link count Glossary-10
 - sfs 3-26 to 3-27
 - table 5-43
 - ufs 3-6 to 3-9
- installing the kernel 8-3
- interactive backup mode 11-29 to 11-30
- interface modifications
 - naming 15-4 to 15-6, 16-6
 - naming requirements 16-7
 - planning 16-4
 - planning the location of 15-3, 16-4
 - planning the structure 15-3, 16-5
 - writing 16-7
- interface program Glossary-10
 - customizing 13-50 to 13-52
 - how to use 13-49 to 13-50
 - how to write 13-48
 - options 13-49 to 13-50
 - purpose of 13-48
 - shell variables 13-50
- interprocess communication (see IPC) Glossary-10
- interrupts Glossary-10
- i-number Glossary-9
- IPC (interprocess communication) Glossary-10
- item help file 16-10

K

- kernel Glossary-10
 - address space Glossary-10
 - profiling 5-12 to 5-15
- Kernel Config Utility Operations 8-9
- Kernel Configuration Utility 8-4
- kernel configuration utility 7-1, 8-1
- Kernel Hardware Adapters Menu 8-13
- Kernel Module Menu 8-11
- Kernel Options Menu 8-17
- Kernel Rebuild Menu 8-16
- kernel symbol table 8-3
- Kernel Symbols 8-27
 - Loading Symbols 8-28
 - Removing Symbols from Kernel Object File 8-28
- kernel-mode scheduler parameter table 9-9 to 9-10
- kill(1) 5-9

L

- labelit(1M) 3-28, 3-30, 3-33 to 3-34
- least privilege Glossary-10
- level Glossary-10
 - run Glossary-16
 - security Glossary-16
- level alias Glossary-10
- limitations
 - client access to xfsd FS 3-22
- line discipline module Glossary-10
- Line Printer services 1-5
- link Glossary-10
 - count 4-6, Glossary-10
 - symbolic Glossary-20
- Linking Directories 2-24
- Linking Files 2-24
- listdgrp(1M) 2-58
- listen(1M) 1-9
- load device Glossary-10
- load process for DLM 6-6
- loadable modules 6-1, 6-2, 6-6, 6-8
- log file Glossary-10
- Logical Block Size
 - xfs 5-4
- logical blocks 3-4, Glossary-10
- Logical Disk Address 2-13
- login Glossary-10
 - name Glossary-10
- logs
 - backup history 11-39 to 11-42
- LP Print Service
 - accepting print requests 12-33, 12-36
 - adding standard interface printer 12-30
 - administration summary 12-50 to 12-52, 13-55 to 13-57
 - advanced printer configurations 12-12 to 12-30
 - banner page in output 12-26
 - canceling print requests 12-40
 - character sets 12-16 to 12-18
 - classes 12-29 to 12-30, 12-39 to 12-40
 - configuring an alert 12-30
 - configuring network printers 13-2 to 13-7
 - control user access 12-25 to 12-26
 - controlling copy of files 12-27
 - customizing 13-43 to 13-52
 - default destination 12-29
 - default printing attributes 12-28
 - device path to 12-9
 - dial-up printers 13-7 to 13-9
 - direct connection 12-9
 - disable printer 12-32 to 12-35
 - display status of 12-32

- distributed 12-3
- enable printer 12-32 to 12-35
- fault detection and recovery 12-22 to 12-24
- filters 12-1, 13-17 to 13-29
- fonts 12-16 to 12-18
- forms 12-2, 12-20 to 12-21, 13-9 to 13-16
- installation of 12-2 to 12-5
- interface program 12-10 to 12-11, 12-16, 13-48 to 13-52, Glossary-10
- making printers available 12-32 to 12-35
- managing print loads 12-35 to 12-38
- moving print requests 12-36 to 12-37, 12-41 to 12-42
- network configuration 12-3 to 12-4
- non-direct connection 13-5 to 13-8
- OA&M menus and 12-49, 13-54
- overview of 12-1 to 12-2
- PostScript printers 12-30, 13-33 to 13-43
- print queue 13-30 to 13-32
- print style 12-28
- print wheels 12-16 to 12-20
- printable file types 12-9 to 12-10
- printer content types 12-14 to 12-16
- printer description 12-27
- printer interface 12-16
- printer port characteristics 12-12 to 12-14, 13-45
- printer types 12-9, 12-14
- putting a request on hold 12-40
- queue_name 13-5
- quick reference to 12-50 to 12-52, 13-55 to 13-57
- rejecting printing requests 12-36
- releasing held print requests 12-41
- server 13-2
- server configuration 12-3
- setting up 12-2 to 12-5
- sharing printers 13-8 to 13-9
- shell interface 12-1
- sites 12-2 to 12-4
- specify printer name 12-8
- starting 12-38 to 12-39
- stopping 12-38
- sysadm(1M) interface 12-49, 13-54
- system default destination 12-29
- Terminfo database 13-46 to 13-48
- troubleshooting 12-42 to 12-48, 13-52 to 13-53
- using forms with 12-20 to 12-21
- lp(1) 12-36, 12-40 to 12-42, 13-34, 13-35
- lpadmin(1M) 12-6, 12-8, 12-11, 12-13, 12-31, 12-44, 12-45 to 12-46, 12-47, 12-49, 12-51, 13-5 to 13-6, 13-8, 13-36, 13-37, 13-54, 13-56
- lpfilter(1M) 13-23
- lpforms(1M) 13-12
- lpmove(1M) 12-35, 12-36
- lpshut 12-38

- lpstat(1) 12-27, 12-31, 12-32, 12-40
- lpstat(1M) 13-9, 13-16
- lpssystem(1M) 13-3
- ls(1) 12-47
- LWP priority 9-3, 9-4

M

- machine alias Glossary-10
- Main Menu Functions 8-9
- maintenance mode Glossary-11
- major device number 2-2
- Management of Media Flaws for SCSI Disks (HN5000 Systems Only) 2-33
- Mandatory Access Control (MAC) Glossary-11
- map table font 13-41
- MAXCLSYPRI 9-7
- Media Flaws for SCSI Disks 2-33
- memfs 5-9
- memfs file system type 3-27
- memory Glossary-11
 - physical Glossary-13
 - shared Glossary-18
 - virtual Glossary-22
- memory-based file system(memfs) 5-9
- menus (see sysadm menus) 2-69
- menus (sysadm)
 - changing entries 16-17
 - creating entries 16-16
 - definition form 16-19
 - deleting entries 16-25
 - information file 16-3
 - item help message description 16-10
 - item help message example 16-10
 - item help message format 16-10
 - locating entries 16-7
 - testing changes 16-19
- mesg(1) 1-12
- Minimizing Kernel Size 8-18
- Minimum Bootable Configuration
 - Multi User Mode (init 2) 8-27
 - Single User Mode (init S) 8-26
- minor device number 2-2
- Mirrored VP 2-63
 - example setup 2-67
 - restoring faulty member 2-69
- mkfs(1M) 5-7
 - for sfs file system 3-33 to 3-34
 - for ufs file system 3-33
- mknod(1M) 2-70
- mode (file) (see also permissions) Glossary-11
- modem interrupts 5-43

mount memfs 5-10
 mount point 3-36
 mount(1M) 3-36 to 3-37, Glossary-11
 mountall(1M) 3-28
 Mounting 2-24
 xfsd file system 3-21
 mounting file systems 3-36 to 3-37
 moving a file system 3-34 to 3-36
 mtune 7-1

N

name
 login Glossary-10
 namelist (see symbol table) Glossary-11
 National Computer Security Center Glossary-11
 ncheck(1M) 3-28, 3-31
 NCSC Glossary-11
 network Glossary-12
 Network Connection
 xfsd file system 3-21
 Network File System (NFS) 1-11
 network printers 13-2 to 13-7
 troubleshooting 13-52 to 13-53
 news(1) 1-12
 nlist(3E) 8-17
 node name 1-8
 nroff(1) 13-19

O

object Glossary-12
 storage Glossary-19
 object file Glossary-12
 object modules 8-2
 object reuse Glossary-12
 operating system Glossary-12
 operation
 normal Glossary-12
 Operations Administration and Maintenance (OA&M)
 menus
 administering LP Print Service through 12-49,
 13-54
 backing up data through 11-42 to 11-44
 restore service through 11-60 to 11-61
 operator mode for backup
 selecting an 11-28 to 11-31
 originating objects
 specifying in backups 11-22 to 11-23
 other Glossary-13

Other Menu Functions 8-18
 output Glossary-13
 Overview
 of xfsd file system 3-20
 owner Glossary-13

P

package
 administration 16-7
 description file 16-3
 modification file 16-3
 page (memory) Glossary-13
 paging Glossary-13
 checking for excess 5-42
 parallel port printer setup 12-8
 Parallel Port Printer Setup (Power Hawk Only) 12-8
 parameters
 see tunable parameters 7-1
 Partitioning 2-27
 partitioning
 devices 2-53
 disk devices 2-61
 Partitioning An SCSI Disk 2-27, 2-28
 partitions Glossary-13
 device 3-3
 password Glossary-13
 expiration Glossary-7
 path Glossary-13
 search Glossary-16
 PATH environment variable 5-8 to 5-9
 pathname Glossary-13
 performance 5-1 to 5-43
 analysis tools list 5-10 to 5-12
 command summary 5-44 to 5-46
 file system 5-2 to 5-8
 improving and controlling 5-1 to 5-9
 kernel profiling 5-12 to 5-15
 monitoring 5-10 to 5-12
 quick reference to 5-44 to 5-46
 system activity reporting 5-15
 troubleshooting 5-41 to 5-42
 permission bit Glossary-13
 permissions Glossary-13
 Physical File Systems 2-14
 pipe(2) Glossary-14
 pipeline Glossary-14
 pmadm(1M) 13-2
 polling Glossary-14
 PostScript printers
 configuring 12-30
 downloading host-resident fonts 13-42 to 13-43

- filters and 13-35
 - how to use 13-34
 - installing and maintaining 13-36 to 13-37
 - installing and maintaining filters 13-37
 - installing and maintaining fonts 13-38 to 13-39
 - installing and maintaining host-resident fonts 13-40 to 13-41
 - purpose of 13-33
- PostScript printers, support of non-requests 13-34
- Preparing The Special File 2-26
- prf profiler commands 5-12 to 5-15
- prfdc 5-12 to 5-13, 5-14
- prfld 5-12, 5-13
- prfpr 5-12 to 5-13
- prfsnap 5-12 to 5-13, 5-14
- prfstat 5-12 to 5-13
- print wheels 12-16 to 12-20
- printer(s) 1-11
- printer(s) (see LP Print Service) 12-1
- priocntl(1) 9-3, 9-5
- priocntl(2) 9-5
- priority (see process priority) 9-2
- privilege Glossary-14
 - fixed Glossary-8
 - least Glossary-10
 - process Glossary-14
 - working set Glossary-23
- Procedure For Identifying Flaws 2-34
- Procedures on Adding Devices 2-76
- process Glossary-14
 - child Glossary-3
 - controlling a runaway 5-9
 - number Glossary-14
 - parent Glossary-13
 - privilege Glossary-14
 - privileges Glossary-14
- process ID Glossary-14
- process priority 9-2 to 9-4
 - fixed priority 9-8 to 9-9
 - global 9-3, 9-6
 - of a sleeping process 9-9
- process scheduler (see scheduler) 9-2
- profile Glossary-14
- profiler(1M) 5-12
- profiling
 - kernel 5-12 to 5-15
- program Glossary-14
 - boot Glossary-2
- prototype(4) 16-3
- ps(1) 5-9
- putdev(1M) 2-35, 2-46, 2-54 to 2-55
- putdgrp(1M) 2-55, 2-56

Q

- queue Glossary-15
- queue_name
 - printers 13-5
- quotas 3-10 to 3-12, 3-41

R

- RAID 2-62
- range
 - file system Glossary-11
- read Glossary-15
- read access Glossary-15
- reference monitor Glossary-15
- Reformatting SCSI Disk 2-34
- region Glossary-15
- Related Documentations 1-16
- Remote File Sharing Glossary-15
- Remote File Sharing (RFS) 13-8
- reserving devices 2-3, 2-58 to 2-60
- resource Glossary-15
- restore 1-5
- restore service
 - assigning an operator 11-50
 - basic 11-48 to 11-50
 - canceling 11-57
 - checking status of requests 11-54 to 11-56
 - directories/files 11-52
 - disk objects 11-52, 11-53
 - examples of 11-49 to 11-50
 - extended 11-50 to 11-58
 - media handling 11-49
 - OA&M menus and 11-60 to 11-61
 - object to new location 11-53
 - observing the progress 11-54
 - overview 11-50 to 11-51
 - pattern matching 11-48
 - pending requests 11-56 to 11-58
 - quick reference to 11-62 to 11-64
 - removing 11-57
 - restricting 11-57
 - rsnotify(1M) 11-50
 - servicing pending jobs 11-56 to 11-57
 - shell interface 11-61
 - specific versions 11-53
 - sysadm(1M) interface 11-60 to 11-61
 - tables 11-54, 11-55 to 11-56
- restore(1M) 11-48, 11-51, 11-52, 11-54
- Restoring Files 10-13
- re-tension Glossary-15

rm(1) 3-36
 role Glossary-15
 role (in Trusted Facilities Management) Glossary-1
 romfonts 13-39
 root Glossary-15
 directories 3-1
 rsnotify(1M) 11-50
 rsoper(1M) 11-56 to 11-58
 rsstatus(1M) 11-55 to 11-56
 rsstatus.tab restore table 11-54, 11-55 to 11-56
 run level Glossary-16

S

sacadm(1M) 13-3
 sadc(1M) 5-16
 sar(1M) 5-16, 5-17 to 5-39
 scheduler 9-1, Glossary-16
 changing configuration 9-10 to 9-13
 changing parameters 9-13 to 9-15
 configuring the 9-5 to 9-13
 fixed class 9-9
 fixed class policy 9-9
 fixed priority class 9-5, 9-8 to 9-9
 fixed priority policy 9-5, 9-8 to 9-9
 global priorities 9-6 to 9-7
 kernel-mode parameter table 9-9 to 9-10
 overview of 9-2 to 9-5
 parameter table 9-8 to 9-10
 system class 9-5
 system policy 9-5
 time-sharing class 9-4 to 9-5
 time-sharing policy 9-4
 tunable parameters 9-7 to 9-8
 scheduler class 9-2 to 9-5
 default 9-6
 fixed priority 9-5
 installing 9-11 to 9-13
 removing 9-11
 system 9-5
 time-sharing 9-4
 Scheduling Class DPT 8-15
 SCSI Disk Flaw Management 2-33
 search path Glossary-16
 sector (disk) Glossary-16
 secure Glossary-16
 Secure Attention Key Glossary-16
 security
 alias level Glossary-17
 level Glossary-16
 policy Glossary-17
 Security Operator Glossary-17

file system
 free space in 5-10
 used space in 5-10
 consistency 4-1
 repair 4-1
 segment Glossary-17
 select dynamic kernel 8-4
 select kernel type 8-4
 select static kernel 8-4
 semaphore Glossary-17
 sensitivity Glossary-17
 set user ID Glossary-17
 Setup
 software configuration, xfsd FS 3-25
 sfs file system
 checking 4-29
 fsck error messages for check blocks and sizes 4-30
 fsck error messages for check pathnames 4-30 to 4-31
 fsck error messages for check reference counts 4-31
 fsck error messages for rescan for duplicate block number 4-30
 sfs file system type 3-26 to 3-27, 3-33 to 3-34
 logical block size 5-4
 SFS I-nodes 2-23
 shared memory Glossary-18
 shell Glossary-18
 script Glossary-18
 shutdown 1-9
 signal Glossary-18
 single-user Glossary-18
 source file Glossary-18
 special file Glossary-18
 spoof Glossary-18
 spool Glossary-18
 spool area Glossary-18
 spooling
 daemon 13-43
 stack Glossary-18
 standard error Glossary-18
 standard input Glossary-18
 standard output Glossary-19
 start-of-day counts 5-2
 static kernel
 advantages 5-43
 static kernel support 8-4
 static modules 6-1
 stdio Glossary-19
 sticky bit Glossary-19
 sticky file Glossary-19
 storage blocks
 ufs 3-9
 storage devices (see devices) 2-1, 3-3
 storage media

- erasing 2-44
- formatting 2-41
- verifying usability of 2-72
- stream Glossary-19
- strip(1) 8-17
- Striped VP 2-62
- striping Glossary-19
- stty(1) 13-45, 13-48
- stune 7-2
- subject Glossary-19
- SUM Glossary-19
- Summary Information 2-20
- Super Block 2-19, 2-21
- super user Glossary-19
- Super User Module Glossary-19
- super-block Glossary-19
 - ufs 3-6
- super-blocks 4-4
- swap Glossary-20
- swap space
 - increasing 2-45
- swapping
 - checking for excess 5-42
- symbol table Glossary-20
- sysadm command 14-20
- sysadm menus
 - device management 2-69
 - for managing file systems 3-42
- sysadm(1M) 2-69, 14-1
 - creating and changing menus 15-10 to 15-16
 - customizing 15-1 to 15-3
 - express mode 14-10
 - forms 14-4, 14-9, 15-15 to 15-16
 - frames 14-3
 - interface hierarchy 16-5
 - main menu 15-1 to 15-2
 - messages 14-4
 - tasks 14-4, 15-16 to 15-21
 - testing menus 15-15
- sysadm(1M) interface
 - frame manipulation tools 14-5
 - menu interface window 14-1, 14-2 to 14-5
 - menu map 14-14 to 14-15
 - quick reference to 14-20
 - sample session 14-10 to 14-14
- sysadm(1M), customizing
 - creating or changing menu entries 15-3 to 15-4, 15-10 to 15-16
 - creating or changing task entries 15-3 to 15-4, 15-16 to 15-20
 - deleting entries 15-20 to 15-21
 - naming menus and tasks 15-4 to 15-6
 - overview of 15-1 to 15-3
 - writing help messages 15-6 to 15-11

- system
 - name 1-8
 - operating Glossary-12
 - states 1-9
- system activity reporting
 - application turnaround 5-39
 - buffer activity 5-19 to 5-20
 - collecting data 5-17
 - CPU utilization 5-29 to 5-30
 - disk activity 5-21 to 5-22
 - file access 5-19
 - file system usage 5-28 to 5-29
 - inode table overflows 5-43
 - interprocess communication 5-24 to 5-25
 - kernel memory allocation 5-23 to 5-24
 - memory freeing 5-22 to 5-23
 - overall system performance 5-32 to 5-39
 - page-in 5-25 to 5-26
 - page-out 5-22 to 5-23
 - process table overflows 5-43
 - queue 5-26 to 5-27
 - swapping and switching 5-31
 - system calls 5-20 to 5-21
 - system table status 5-30 to 5-31
 - terminal 5-31 to 5-32
 - unused memory 5-27 to 5-28
- System Administration Glossary-20
- System Administrator Glossary-20
- System Block 2-18
- system calls Glossary-20
- System Failures
 - xfsd file system 3-23
- system name Glossary-20
- system performance analysis tools 5-10 to 5-12
- system scheduler class 9-5
- System Security Officer Glossary-18
- system state Glossary-20
- System Tunable Menu 8-9

T

- table Glossary-21
- tables of contents
 - creating for backups 11-26 to 11-27
- tape
 - cartridge Glossary-3
- tape devices 2-47, 2-62
- Tape Drive, how to add 2-80
- tape drives 1-11
- tar(1) 5-7
- task
 - action file 16-7

- change entry 16-21
- create entry 16-20
- definition form 16-22
- delete entry 16-25
- tcpio(1) 2-70
- tcpio(1M) 3-34
- tension Glossary-15
- terminal Glossary-21
- terminals 1-11
- Terminfo database 13-46 to 13-48
- text segment Glossary-21
- The fsdump Command 10-10
- tic(1M) 13-48
- time
 - setting 1-8
- time-sharing
 - scheduler class 9-4
 - scheduler parameter table 9-5
- time-sharing class 9-4 to 9-5
- timex(1) 5-39
- timezone
 - setting 1-8
- tput(1) 12-47
- track (disk) Glossary-21
- trap Glossary-21
- trap door Glossary-21
- troff(1) 13-19
- trouble report 1-13
- Trusted Computing Base (TCB) Glossary-21
- ts_dptbl(4) 9-5, 9-14
- ts_kmdpris (kernel-mode scheduler parameter table) 9-9 to 9-10
- TSMAXUPRI 9-8
- ttymon(1M) 1-9
- tunable parameters 5-2, Glossary-22
 - scheduler 9-7 to 9-8
- tunable system parameters 7-1

U

- ufs file system
 - check pathnames 4-16 to 4-22
 - check reference counts 4-24 to 4-27
 - checking 4-3
 - cleanup phase 4-28
 - components checked by fsck 4-4 to 4-7
 - fsck error messages for check cylinder groups 4-27 to 4-28
 - initialization phase 4-8 to 4-12
 - rescan for duplicate block number 4-30
 - running fsck on 4-8 to 4-31
- ufs file system type 3-5 to 3-12, 3-33

- cylinder group map 3-5
- disk block addresses 3-6
- indirect addressing 3-8
- logical block size 5-4
- summary information block 3-5
- umount(1M) 3-28, 3-37
- umountall(1M) 3-28
- unload process for DLM 6-8
- unmounting file systems 3-37
- urestore(1M) 11-52, 11-54
- ursstatus(1M) 11-54
- user
 - communication 1-12
 - default environment 1-10
 - ID Glossary-22
 - requests 1-13
- Using fsdump To Save root 10-13
- utility program Glossary-22
- Utility, File System Restore 10-20
- Utility, Kernel Configuration 8-4

V

- verification Glossary-22
- vfstab(4) 3-29 to 3-31
- Virtual Partition (VP) 2-62
- virus Glossary-22
- volcopy(1M) 3-28, 3-31, 3-33, 3-34
- VP
 - Configuration 2-63
 - Driver Configuration 2-63
 - example setup 2-66
 - geometry 2-64
 - vpinit 2-64
 - Vpstat Command 2-65
- VP Driver 2-63
- vpstat 2-65
- VTOC (Volume Table of Contents) 11-20, Glossary-22

W

- wall(1M) 1-12
- write access Glossary-23

X

- xfs File System Administration 3-18
- xfs file system type 3-13 to 3-20

- block 0 3-16
- contiguous files 3-14
- creating 3-19
- cyclic log 3-16
- data blocks 3-17
- fast file system recovery 3-13
- features 3-13
- fragment support 3-14
- free space list 3-16
- Inode list 3-16
- large file systems 3-15
- prioritized I/O 3-15
- root directory 3-16
- space allocation 3-17
 - large files 3-14
- structure 3-15
- superblock 3-16
- xfs Logical Block Size 5-4
- xfsd file system type 3-20 to 3-26
- xfsd Network Connection 3-21
- xfsd overview 3-20
- xfsd Setup 3-24

Spine for 2" Binder

**Product Name: 0.5" from
top of spine, Helvetica,
36 pt, Bold**

**Volume Number (if any):
Helvetica, 24 pt, Bold**

**Volume Name (if any):
Helvetica, 18 pt, Bold**

**Manual Title(s):
Helvetica, 10 pt, Bold,
centered vertically
within space above bar,
double space between
each title**

**Bar: 1" x 1/8" beginning
1/4" in from either side**

**Part Number: Helvetica,
6 pt, centered, 1/8" up**

PowerMAX OS

User/Administrator

System Administration

0890430

Volume 2