FX Series™
AIX Version 4.1

Configuring and Maintaining the System

FXCMSA/UM1

First Edition (January 1997)

This edition of *Configuring and Maintaining the System* applies to AIX 4.1 and to all subsequent releases of this product until otherwise indicated in new releases or technical newsletters.

The following paragraph does not apply to the United Kingdom or any country where such provisions are inconsistent with local law: THIS MANUAL IS PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions; therefore, this statement may not apply to you.

It is not warranted that the contents of this publication or the accompanying source code examples, whether individually or as one or more groups, will meet your requirements or that the publication or the accompanying source code examples are error-free.

This publication could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication.

It is possible that this publication may contain references to, or information about, products (machines and programs), programming, or services that are not announced in your country. Such references or information must not be construed to mean that such products, programming, or services will be offered in your country. Any reference to a licensed program in this publication is not intended to state or imply that you can use only that licensed program. You can use any functionally equivalent program instead.

Trademarks and Acknowledgments

AIX is a trademark of International Business Machines Corporation.

AIXwindows is a trademark of International Business Machines Corporation.

DEC VT100, VT220, VT320, and VT330 are trademarks of Digital Equipment Corporation.

Diablo is a a trademark of Xerox Corporation.

HP, HP-GL, Hewlett-Packard, HP LaserJet+, JetDirect, and LaserJet are trademarks of Hewlett-Packard Company.

IBM is a registered trademark of International Business Machines Corporation.

IMP is a trademark of Integrated Micro Products, Inc.

Impactwriter is a trademark of International Business Machines Corporation.

INed is a trademark of INTERACTIVE Systems Corporation.

InfoExplorer is a trademark of International Business Machines Corporation.

NCK is a trademark of Apollo Computer, Inc.

Network Computing System is a trademark of Apollo Computer, Inc.

NFS, Network File System, SunOS, and Sun Microsystems are trademarks of Sun Microsystems, Inc.

Open Software Foundation, OSF, OSF/1, OSF/Motif, and Motif are trademarks of Open Software Foundation, Inc.

Personal System/2 and PS/2 are trademarks of International Business Machines Corporation.

Photo CD is a trademark of Eastman Kodak Company.

POSIX is a trademark of the Institute of Electrical and Electronic Engineers (IEEE).

PostScript is a trademark of Adobe Systems Incorporated.

PRINTRONIX and PRINTRONIX P9012 are trademarks of Printronix, Inc.

Proprinter is a trademark of International Business Machines Corporation.

QMS, QMS Colorscript 100, and QMS-PS 2200 are trademarks of QMS, Inc.

QuattroPro is a trademark of Boreland International Corporation.

Quickwriter is a trademark of International Business Machines Corporation.

Quietwriter is a trademark of International Business Machines Corporation.

RISC System/6000 is a trademark of International Business Machines Corporation.

RT and RT PC are trademarks of International Business Machines Corporation.

SUN is a trademark of Sun Microsystems, Inc.

TI Omnilaser is a trademark of Texas Instruments, Inc.

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

WYSE, WYSE 30, WYSE 50, WY-50, WYSE 60, and WYSE 350 are trademarks of WYSE Corporation.

Xstation Manager is a trademark of International Business Machines Corporation.

X Window System is a trademark of The Open Group.

Note to Users

The term “network information service (NIS)” is now used to refer to the service formerly known as “Yellow Pages.” The functionality remains the same; only the name has changed. The name “Yellow Pages” is a registered trademark in the United Kingdom of British Telecommunications plc, and may not be used without permission.

Legal Notice to Users Issued by Sun Microsystems, Inc.

“Yellow Pages” is a registered trademark in the United Kingdom of British Telecommunications plc, and may also be a trademark of various telephone companies around the world. Sun will be revising future versions of software and documentation to remove references to “Yellow Pages.”

Contents

Chapter 1. Introduction

This chapter contains introductory information about this book, *Configuring and Maintaining the System*. It includes the following topics:

- the audience and organization of the book
- an overview of the contents of each chapter
- a description of the major typographical conventions
- a list of other sources of information regarding system administration

About This Book

This book contains information for understanding and performing tasks that are integral to your day-to-day life as a system administrator. The book discusses the tools AIX provides for system management and devotes chapters to the major tasks and topics you encounter as a system administrator.

Note: Information pertaining to the FX Series system, a fault–tolerant version of AIX, is interspersed within this guide. If you are not running the FX Series system, then simply ignore the text.

Who Should Use This Book

This book is intended for persons performing system management on the computer and operating system. Readers of this book are expected to know basic operating system commands.

This book assumes you are familiar with the information and concepts presented in the following publications:

- *System User's Guide: Operating System and Devices*
- *System User's Guide: Communications and Networks*
- *Operating System Installation*
- *Installation Troubleshooting*
- *Making and Using Backups*

If you are performing system management on an FX Series system, then you should also be familiar with the information and concepts presented in the following FX Series publications:

- *System Architecture for the Fault Tolerant System*
- *Administering Your Fault Tolerant System*

How to Use This Book

Since your time is in demand, this book is organized to help you quickly find the information you need. The tasks of each chapter are arranged in the following order:

- Overview of topic/task group
- Configuration tasks
- Maintenance tasks
- Troubleshooting

Note: The troubleshooting sections are helpful when you know the cause of your problem. If you encounter a problem for which you do not know the cause, refer to the *Problem Solving Guide and Reference*. If you are administering an FX Series system, then also refer to *Diagnostics and Troubleshooting on Your Fault Tolerant System.*

Overview of Contents

This book contains the following chapters and appendixes:

- Chapter 1, “About This Book,” provides an overview of this book and lists related publications.
- Chapter 2, “System Management with AIX,” introduces the major tools provided to assist you in system management and the features specific to the system.
- Chapter 3, “System Management Interface Tool,” describes the use and structure of the System Management Interface Tool, or SMIT. SMIT is a command-building user interface created to assist the system manager in constructing and re-creating many system management tasks. The interface can be used in either an ASCII or windows environment.
- Chapter 4, “Starting and Stopping the System,” contains conceptual information about starting and stopping the system as well as procedural information to guide you in performing these tasks.
- Chapter 5, “Managing Users and Groups,” discusses the features available to manage individual users and illustrates procedures used to manage groups of users.
- Chapter 6, “System Environment,” discusses basic environment components you can manage and how to work with these components. Also included are instructions that explain how to change the message of the day, broadcast messages to users, and work with profiles.
- Chapter 7, “Process Management,” introduces system processes and how to use them.
- Chapter 8, “System Resource Controller and Subsystems,” introduces this feature and discusses ways to use the controller.
- Chapter 9, “System Accounting,” introduces the conceptual background information needed to use the wide array of system accounting commands and subroutines.
- Chapter 10, “Security,” introduces the security features, including the Trusted Computing Base (TCB), the **virscan** command that detects viruses, auditing, and access control.
- Chapter 11, “Managing the InfoExplorer Program,” describes the system management considerations and tasks for InfoExplorer, the user interface to the hypertext version of the system documentation.
- Chapter 12, “Managing the AIX Common Desktop Environment,” provides detailed instructions for starting, stopping, disabling and enabling the AIX Common Desktop Environment. It also discusses system management functions for customizing display devices for AIX Common Desktop Environment.
- Chapter 13 “National Language Support,” introduces the special considerations necessary for managing a system in various languages and time zones.
- Chapter 14, “Managing Power Management,” describes the system management and general user tasks for Power Management.
- Chapter 15, “AIX for BSD System Managers,” contains information for system managers who are familiar with the 4.3 BSD UNIX or System V operating system. This chapter explains both similarities and differences.

Documentation Conventions

The following documentation conventions are used in this book:

Bold	identifies commands, subroutines, keywords, files, structures, directories, and other items whose names are predefined by the system, and identifies graphical objects such as buttons, labels, and icons that the user selects
Italics	identifies parameters whose actual names or values are to be supplied by the user
`Monospace`	identifies examples of specific data values, examples of text similar to what you might see displayed, examples of portions of program code similar to what you might write as a programmer, messages from the system, or information you should actually type

Other Key Sources of System Management Information

Publications Covering Other Aspects of System Management

In today's computing environment, it is impossible to create a single book that addresses all the needs and concerns of a system administrator. While this guide cannot address everything, we have tried to structure the rest of our library so that a few key books can provide you with direction on each major aspect of your job.

The following books cover other key topics of interest to you:

- *System Management Guide: Communications and Networks* covers network administration and maintenance.
- *Network Installation Management Guide and Reference* covers configuration and maintenance of diskless workstations on a standard AIX system. This guide is not applicable for an FX Series system.
- Problem Solving and Messages:
  - *Problem Solving Guide and Reference*
  - *Messages Guide and Reference*
  - *Diagnostics and Troubleshooting on Your Fault Tolerant System* (if applicable)
- *General Programming Concepts: Writing and Debugging Programs*
- *Communications Programming Concepts*
- *Files Reference*
- Monitoring and tuning system performance:
  - *Performance Tuning Guide* describes the performance monitoring and tuning tools available with the base operating system.
  - *Performance Toolbox for AIX: Guide and Reference* describes the additional monitoring tools available with Performance Toolbox for AIX.

Reference Information

The following publications contain information on the commands and files used in the operating system.

- *Commands Reference* is a multi-volume set that contains supported commands in alphabetical order.
- *Files Reference* contains information on the files available with the operating system.

Additional information that is *not* included in the hardcopy references manuals is available via the online **man** command. This is particularly true for additional functionality that is supported on the FX Series system.

The following books contain other information you may find useful in day-to-day managing:

- *Quick Reference* contains brief descriptions of frequently used commands, along with brief summaries of the commands used for SMIT, InfoExplorer, and each of the supported editors.
- *INed Editor User's Guide* contains information on INed.

Chapter 2. System Management with AIX

System management is the task of an individual who is usually referred to in UNIX literature as the system administrator. Unfortunately, only a few of the system administrator's activities are straightforward enough to be properly called administration. This and related guides are intended to help system administrators with their numerous duties.

This chapter describes:

- the objectives of a system administrator
- main elements and management functions of a computer system
- the user interfaces available on AIX
- the unique features of AIX

System Management Overview

The System Administrator's Objectives

The system administrator has three main objectives:

- See to it that the system does its job effectively and efficiently.
- Ensure that the information stored on the system is secure from intentional or accidental destruction.
- Administer the system owner's rules for the use of the system.

To achieve these objectives the system administrator must understand more than just the structure and interaction of the hardware and software under his or her control. He or she must also understand the interconnected environment in which almost all of today's systems exist, and the effects of that environment on the local system's function and performance.

A System Is More Than a Computer

A contemporary computer system includes a number of hardware, software, and information elements that must work cooperatively if the system is to satisfy the needs of its users. The main elements and their management functions are:

- fixed-disk drives
  - control the grouping and subdivision of disk space
  - control the location of data and programs for optimum performance
  - control the amount of space allocated for different purposes
- application programs
  - control the use of sensitive or costly programs
  - install and performance-tune major applications
- application data
  - control access to sensitive data
  - ensure that appropriate backup measures are taken
- individual computer processors and memory
  - ensure that resources are used in accordance with the organization's priorities
  - control access to the system by individuals and groups
  - tune the operating system for optimal use of the available resources
- local area networks
  - ensure that networks are tuned for optimum performance
  - control network addressing mechanisms
- local terminals
  - control the connection of terminals to processors
  - ensure that terminals and processors are set up for maximum performance

- connections to other networks
  - ensure that bridges and gateways to other networks are properly configured
  - ensure that interaction with remote networks does not degrade local systems
- access to and from remote systems
  - control the access permissions in both directions
  - monitor and performance-tune the workload imposed by remote connections
- access to remotely owned data
  - control the methods and availability of access

Unique Aspects of AIX System Management

AIX provides its own particular version of system-management support in order to promote ease of use and to improve security and integrity. This chapter presents information on these unique features:

- available interfaces
- unique features of the operating system
- InfoExplorer and the **man** command

Available Interfaces

In addition to conventional command-line system administration, AIX provides these optionally installable interfaces:

- System Management Interface Tool (SMIT), a menu-based user interface that constructs commands from the options you choose and executes them

  With SMIT, you can:

  – install, update, and maintain software

  – configure devices

  – configure disk storage units into volume groups and logical volumes

  – make and extend file systems and paging space

  – manage users and groups

  – configure networks and communication applications

  – print

  – perform problem determination

  – schedule jobs

  – manage system environments

  – monitor and maintain modules on a fault tolerant system, if applicable

  See page 3-1 for more information on managing your system with SMIT.

- Visual System Management (VSM), a graphical interface that allows you to perform system management tasks through direct manipulation of objects (icons)

  VSM has extensive online help for working in the interface. For more information about using VSM, see “Using the Visual System Management Applications” in *Getting Started*.

  Many of the tasks that can be completed in SMIT can be done in the following VSM applications:

  – Device Manager displays system objects and dialogs based on what is contained in your system’s Device Configuration database. This allows you to manage some devices not covered by SMIT such as graphic adapters, graphic display subsystems, ports, buses, expansion drawers, non-SCSI adapters, standard adapters, and memory cards. However, some SMIT functions are unavailable such as tracing devices, and managing communication applications, LFT devices, Xstations and the printer subsystem.

  – Print Manager allows you to perform basic SMIT tasks on printers and queues and makes creating queues much simpler. In addition, it provides a graphical representation of queue and printer attachments as well as print jobs waiting in the queue. Print Manager does not include the ability to start or schedule a print job.

  – Storage Manager allows you to manage physical volumes, volume groups, logical volumes, and file systems through direct manipulation of objects. In addition, the VSM interface provides an easy way to view the contents of a logical volume – something that was previously only possible from the command line.

  – Users and Groups Manager makes adding users easy by providing system-defined and user-customized templates. Simple drag-and-drop actions allow you to set user passwords, select a user’s default interface, and allow or disallow user logins.

- Distributed System Management Interface Tool (DSMIT), an ASCII-based user interface that allows you to perform system administration tasks on “clusters” of workstations, including machines running Sun/OS 4.1.3 and HP/UX 9.0

  This product is separately purchasable. For more information on this product, see the *Distributed SMIT 2.2 for AIX: Guide and Reference*.

Unique Features of the Operating System

Following are brief discussions of unique system-management features of the operating system.

Logical Volume Manager

The Logical Volume Manager (LVM) allows logical volumes to span multiple physical volumes. Data on logical volumes appears to be contiguous to the user, but can be discontiguous on the physical volume. This allows file systems, paging space, and other logical volumes to be resized or relocated, span multiple physical volumes, and have their contents replicated for greater flexibility and availability.

LVM enhancements are available on FX Series systems. These enhancements enable such features as locating fault tolerant volume groups in different I/O domains.

For more detailed information, see *Managing System Storage*.

System Resource Controller

The System Resource Controller (SRC) provides a set of commands and subroutines for creating and controlling subsystems and is designed to minimize the need for human intervention in system processing. It provides a mechanism to control subsystem processes by using a command-line C interface. This allows you to start, stop, and collect status information on subsystem processes with shell scripts, commands, or user-written programs.

For more detailed information, see the overview of the System Resource Controller on page 8-2.

Object Data Manager

The Object Data Manager (ODM) is a data manager intended for the storage of system data. Many system management functions use the ODM database. Information used in many commands and SMIT functions is stored and maintained as objects with associated characteristics. System data managed by ODM includes:

- device configuration information
- display information for SMIT (menus, selectors, and dialogs)
- vital product data for installation and update procedures
- communications configuration information
- system resource information

Software Vital Product Data

Certain information about software products and their installable options is maintained in the Software Vital Product Data (SWVPD) database. The SWVPD consists of a set of commands and Object Data Manager (ODM) object classes for the maintenance of software product information. The SWVPD commands are provided for the user to query (**lslpp**) and verify (**lppchk**) installed software products. The ODM object classes define the scope and format of the software product information that is maintained.

The **installp** command uses the ODM to maintain the following information in the SWVPD database:

- name of the installed software product
- version of the software product

- release level of the software product, which indicates changes to the external programming interface of the software product
- modification level of the software product, which indicates changes that do not affect the external programming interface of the software product
- fix level of the software product, which indicates small updates that are to be built into a regular modification level at a later time
- fix identification field
- names, checksums, and sizes of the files that make up the software product or option
- installation state of the software product: available, applying, applied, committing, committed, rejecting, or broken

InfoExplorer and the man Command

InfoExplorer

InfoExplorer gives you access to thousands of pages of online information about using, managing, and programming the system. Information is stored in multiple databases so that you can install only the using and managing information or one or more of the programming information databases. You can install these databases on a server to save disk space on clients or use the optional CD-ROM, which contains approximately 35,000 pages of documentation on one compact disc.

As a system manager, you can use the notes facility of InfoExplorer to annotate the information for your specific installation. For example, you can put your name and phone number in places where documentation refers to contacting the system manager. You can use the bookmark feature to build bookmark files to guide system users to information you determine to be important for your site.

Updates to the operating system also include InfoExplorer databases, which are installed with your code update to keep your documentation current with software and hardware updates.

See the InfoExplorer overview of system management on page 11-1 for further details about managing InfoExplorer. See “A New Way of Looking at Documentation” in *Getting Started* for details about how to access InfoExplorer and use its features.

man command

The **man** command is used mainly to access reference information on commands, subroutines, and files. For example, to view information on the **gprof** command, enter:

```
man gprof
```

Much of the information displayed is actually taken from InfoExplorer, but displayed in nroff format. Many system managers find using the **man** command more convenient than starting an InfoExplorer session when they simply need to find out about a certain flag or the syntax of a given command.

For more information on the **man** command, see *Commands Reference*. Also, on page 15-24 there is a discussion on the **man** command for BSD 4.3 system managers.

Chapter 3. System Management Interface Tool

This chapter introduces the design of the System Management Interface Tool (SMIT) panels. It includes the following information for both the ASCII and graphical interfaces:

- an overview of SMIT and description of its features
- defining SMIT fast paths and providing a list of fast paths
- the graphical SMIT window design and its window menus and button or key functions
- entering and exiting SMIT and selecting SMIT options
- completing SMIT dialogs
- the SMIT command output windows and command status screens
- duplicating system configuration
- printing SMIT screens
- getting contextual help

System Management Interface Tool (SMIT)

The System Management Interface Tool (SMIT) is the primary tool for managing the system. The SMIT facility presents a natural-language, task-oriented interface to the many commands required to manage the system, enabling you to quickly perform tasks that, otherwise, might require many typed commands. SMIT steps you through the desired task with the use of menus, selectors and dialogs, thereby freeing you from the details of complex command syntax, valid parameter values, and system command spelling. In addition, SMIT creates log files that you can use to duplicate system configuration or to learn how to use specific commands. The SMIT facility runs in two interfaces, ASCII (nongraphical) or AIXwindows (graphical).

Note: Other system management tools are available. In a graphical environment, you can use the Visual System Management (VSM) applications that are discussed in *Getting Started*. In a networked environment, you can use Distributed SMIT as discussed in the *Distributed SMIT 2.2 for AIX: Guide and Reference*.

When you enter SMIT, a main menu similar to the following displays:

```
Software Installation & Maintenance
Software License Management
Devices
Configuration Management System (for FX Series systems only)
System Storage Management (Physical & Logical Storage)
Security & Users
Communications Applications and Services
Print Spooling
Problem Determination
Performance & Resource Scheduling
System Environments
Processes & Subsystems
Applications
Using SMIT (information only)
```

Main menu selections lead into submenus, helping to narrow the scope of choice to a particular task. To skip the main menu and directly access a submenu or dialog, use the **smit** command with a *Fast Path* parameter. To learn more about a menu item or the fields in a dialog, use the help function provided in SMIT.

Using SMIT (Information Only) in SMIT

For a general overview of the purposes, functions, and conventions of SMIT, select the **Using SMIT** option.

When you access **Using SMIT** in an ASCII interface, you are presented with a table of contents similar to the following:

1. Overview
2. Understanding the Menu System
3. Understanding the Dialog Screen
4. Understanding the Command Status Panel
5. Making Selections from a List of Choices
6. Using SMIT Functions
7. Using SMIT Fastpath
8. Related Information in InfoExplorer

When you access **Using SMIT** in the Graphical Interface, you are presented with a table of contents similar to the following:

- Overview
- Understanding the Menu Window
- Understanding the Dialog Window
- Understanding the Path Window
- Understanding the Command Output Panel
- Making Selections from a List of Choices
- Using SMIT Functions
- Using SMIT Fastpath
- Related Information in InfoExplorer

Defining Fast Paths in SMIT

The SMIT facility runs in two interfaces, ASCII (nongraphical) or graphical. You can access SMIT at the main menu, or you can use a *FastPath* parameter to enter the application at a lower level (a submenu or a dialog). Using the **smit** command with a *FastPath* parameter can save you time by allowing you to go directly to the dialog for your task, bypassing the upper-level menus. There are several ways to determine the *FastPath* parameter for your chosen task:

- Command Names
- F8 Function Key (ASCII Interface)
- Show Menu or F8 Key (Graphical Interface)
- Fast Path Algorithm for Device Tasks
- Fast Path Lists

Command Names

All commands run by SMIT can be used in a fast path construction. Command names entered as a *FastPath* parameter will take you to a submenu or dialog for that command. For example, to change the characteristics of a user, at the command line enter:

```
smit chuser
```

The **smit** command plus the command **chuser**, which is the fast path construction, takes you directly to the menu, Change User Attributes, which guides you through the steps to change a user's characteristics.

F8 Function Key (ASCII Interface)

The Image key (F8 or Esc+8) in the ASCII interface displays the fast path for the current screen.

1. Display the screen whose *FastPath* parameter you want to record.
2. Press the F8 key.

   A pop-up screen displays the current *FastPath* parameter.
3. Record the *FastPath* parameter for future use. If you do not want to hand-record the *FastPath* parameter, you can check the **smit.log** file. The **smit.log** file has a record of the menu screens selected along with their titles and fast paths.
4. Press the F3 key to exit the pop-up and return to the previous screen.

Show Menu or F8 Function Key (Graphical Interface)

Select **Show –-> FastPath** or press the function key F8 to display the *FastPath* parameter for the current submenu or dialog panel.

1. Display the submenu or dialog panel whose *FastPath* parameter you want to record.
2. Press the F8 key.

   OR

   Select **Show –-> FastPath**

   A pop-up window displays the current *FastPath* parameter. Selecting the window's **Help** button evokes a help window displaying an example of how to use the *FastPath* parameter.

3. Record the *FastPath* parameter for future use. If you do not want to hand-record the *FastPath* parameter, you can check the **smit.log** file. The **smit.log** file has a record of the menu screens selected along with their titles and fast paths.
4. Press the **Cancel** button to close the window and return to the SMIT application.

FastPath Algorithm for Device Tasks

To access a specific device task menu, create a *FastPath* parameter by combining the action prefix with the device abbreviation. The formula is:

Action Prefix + Device Abbreviation = *FastPath* **Parameter**

Action	**Prefix**	**Device**	**Abbreviation**
add or make	mk	printer	prt
change	ch	tty	tty
list	ls	pty	pty
remove	rm	disk	dsk
		cdrom	cdr
		diskette	dskt
		tape	tpe
		adapter	adp

The following are examples of fast path commands for devices:

- To add a printer device, use the **smit mkprt** fast path.

  This takes you to the **Add a Printer** submenu which guides you through the steps for adding a printer device.

- To remove a tape device, use the **smit rmtpe** fast path.

  The Tape Drive selector screen appears and prompts you to select a specific tape drive to remove. After selecting a tape drive, the selector disappears and is replaced by the Remove a Tape Drive dialog which guides you through removing the tape drive.

- To change characteristics of a disk device, use the **smit lsadp** fast path.

  This takes you to the **List Communication Adapters** submenu, which leads you to a menu that lists the communication adapters configured on your system.

Fast Path Lists

Refer to the following lists for information about fast paths:

- The SMIT fast path lists on page 3-6 provide a quick reference to fast paths for installing and managing the system.
- The section “Understanding the SMIT Interface for TCP/IP” in the *System Management Guide: Communications and Networks* provides the fast paths for a list of TCP/IP system management menus and tasks.

SMIT Fast Path Lists

This section lists the **smit** command fast paths for a variety of tasks.

Installing

Task	Command
Begin the installation process.	**smit install_update**
Install an optional software program with updates on a standard system.	**smit install_selectable_all**
Install an optional software program with updates on a diskless system.	**smit dinstallp**

Managing System Access

Task	Command
Add the group.	**smit mkgroup**
Add the user account.	**smit mkuser**
Remove the user from the group.	**smit rmuser**
Change the user account characteristics.	**smit chuser**
Change the user password.	**smit passwd**
Change or show the initial user interface.	**smit chinterface**

Managing Graphic Input Devices

Task	Command
Modify the keyboard.	**smit keyboard**
Generate error or trace reports on the mouse.	**smit mouse**
Modify dials.	**smit dials**
Modify the lighted programmable function keyboard.	**smit lpfk**
Modify the tablet.	**smit tablet**

Managing the Low-Function Terminal

Task	Command
Change keyboard map.	**smit keymap**
List fonts.	**smit lsfont**
Change fonts for next reboot.	**smit chfont**
Add a font.	**smit mkfont**
Change the display.	**smit chdisp**
List available displays.	**smit lsdisp**

Managing System Environment, Performance, and Problems

Task	Command
Change the primary dump device.	**smit dumpchgp**
Report system activity.	**smit sar**
Show the status of all current processes.	**smit ps**

Managing Storage and File Systems

Task	Command
List all of the logical volumes on the system.	**smit lslv2**
Add the file system.	**smit crjfs**
Change the paging space.	**smit chps**
Back up the file system.	**smit backup**
Restore the file system.	**smit restore**

Managing the FX Series System (If Applicable)

Task	Command
Access the Configuration Management System.	**smit cms**

Managing Networks

Task	Command
List all of the TCP/IP inetd subservers.	**smit lsservices**
Show the host characteristics.	**smit chhostent**
List all of the TCP/IP nameservers.	**smit lsnamerslv**

Managing Printers, Terminals, and Tape Drives

Task	Command
List all of the defined devices.	**smit lsddev**
Add the printer to the existing system.	**smit mkprt**
Change the printer characteristics.	**smit chgprt**
List all of the defined printers.	**smit lsdprt**
Add the terminal to the existing system.	**smit mktty**
Change the tape drive characteristics.	**smit chtpe**
Diagnose the display terminal.	**smit trcstart**
Generate the error report for the printer.	**smit errpt**

Managing Print Jobs and Print Queues

Task	Command
Start the print job.	**smit qprt**
Cancel the print job.	**smit qcan**
Show the status of print jobs.	**smit qchk**
Add a print queue.	**smit mkpq**
Change the print queue characteristics.	**smit chpq**
List all of the print queues.	**smit lspq**

SMIT in the Graphical Interface

The following sections give an overview on how to use the graphical version of the System Management Interface Tool (SMIT). SMIT in the graphical interface is often referred to as a Motif interface since the interface relies on Motif-style windows.

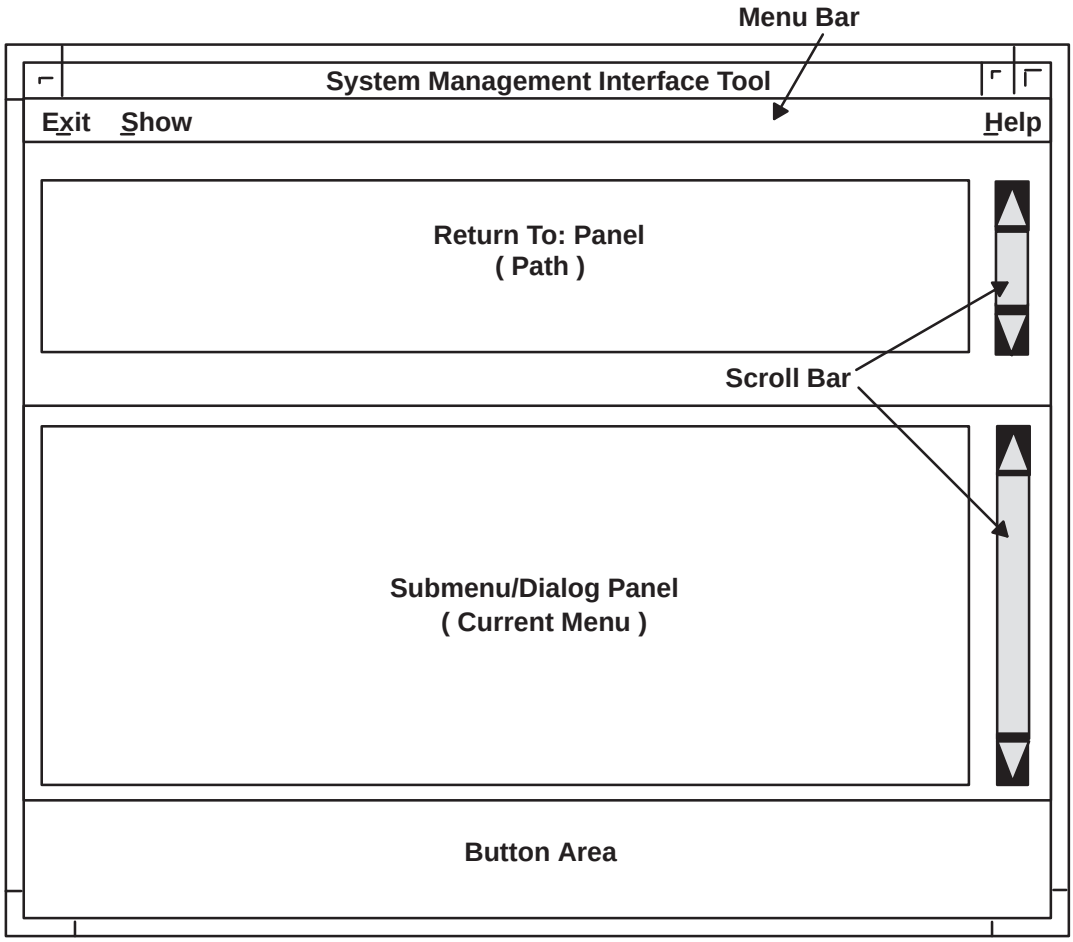
In the windowing version of SMIT, you use standard mouse-driven window conventions to manipulate the menu-based format, which guides you through system management tasks. SMIT menus, selectors, and dialogs present complex system management tasks in an organized manner, prompting you for information when necessary. As you make menu selections, SMIT builds or runs the appropriate command.

SMIT Window Design

The following provides an overview of the major sections in the main SMIT window. The graphical interface offers a progression of menus, submenus, and dialogs in a graphical format that assists you in the construction of system management commands. Other windows, such as message windows, may appear as you use SMIT functions.

The following SMIT Window Design figure shows the major sections of the main SMIT window:

- Menu Bar
- Return To: Panel
- Submenu/Dialog Panel
- Button Area



SMIT Window Design

Menu Bar

The menu bar, located beneath the System Management Interface Tool title bar, displays the titles of available menus. When selected, each menu produces a pull-down list of options. The possible menus available at different times during a SMIT windows session are:

Menu Title	Menu Options
Exit	Exit SMIT
Help	On Context On Window On Keys Index On Help
Show	Command FastPath Find Find Next

For more information about the available menu options and how to select them, see “Menus and Menu Options” on page 3-12.

Return To: Panel

Previous SMIT menu selections are listed in the `Return To:` panel. The list of menu titles provides you with a history of your menu selections. To return to a previous menu, select the menu or submenu title. For more information, see the section on “Selecting SMIT Menu Titles (Graphical Interface)” on page 3-16.

Submenu/Dialog Panel

The largest panel in a SMIT window is the Submenu/Dialog panel.

Submenus The current submenu title appears in the left corner above the panel, and submenu options are listed in the panel. Submenu selections produce other submenu panels or dialog panels. See “Selecting SMIT Menu Titles (Graphical Interface)” on page 3-16.

Dialogs The dialog title appears in the left corner above the panel, and text entry fields are listed in the panel. Dialog panels prompt you for specific parameters needed by SMIT to build the command. When you have entered all necessary data, select the **OK** button to run the command. For more information, see “SMIT Dialogs: Overview (Graphical Interface)” on page 3-17.

Button Area

At the bottom of the submenu/dialog panel are available buttons. For more information about all the available buttons in SMIT, see the section on SMIT buttons on page 3-13.

Cancel The **Cancel** button recalls previous submenu choices in reverse order until you reach the main menu. From the main SMIT menu, the **Cancel** button exits SMIT.

SMIT Window Menu and Button Functions

This section provides a quick summary of the menus and buttons available in SMIT windows and pop-ups. Gray, or ghosting, menu items and buttons are inactive.

Menus and Menu Options

SMIT menu options are selected by mouse, mnemonics, or accelerators.

Mouse
To select a menu item by mouse, click on the menu title to display the list of options then click on the desired option. You can also point and drag with the mouse to select a menu option.

Mnemonics
Mnemonics are the underlined letters in each menu-option name. To select a menu item using the mnemonic, press and hold the Alt key and then press the key that corresponds to the underlined letter.

Accelerators
Accelerators are designated keys that accomplish menu commands without opening menus. Common accelerators are the function keys (F1 or F2, for example). Accelerators appear beside the menu items they represent in the pull-down menus.

Menus	Description
Exit Menu	contains the **Exit SMIT** option
Exit SMIT	closes all SMIT windows and ends the session, returning you to the command line (Accelerator: F12)
Show Menu	contains the **Command**, **FastPath**, **Find**, and **Find Next** options
Command	displays the command statement SMIT is building for you (Accelerator: F6) This option is active only in a dialog panel.
FastPath	displays the *FastPath* parameter that corresponds to your present position in SMIT (Accelerator: F8) The **smit** command and the *FastPath* parameter will bring you directly to the current submenu or dialog panel when entered at the command line. See the section on how to define fast paths in SMIT on page 3-4 for more information.
Find	prompts for a pattern to find in the command output window (Accelerator: /) If that pattern is found, it is highlighted in the output window. Simple, regular-expression patterns are supported. This option is active only in the command output window.
Find Next	finds the next string that matches the pattern entered when using the **Find** option (Accelerator: n) This option is active only in the command output window after you have defined a pattern using the **Find** option.
Help Menu	contains the **On Context**, **On Windows**, **On Keys**, **Index**, and **On Help** options

On Context displays help on submenus or dialogs that have contextual help (Accelerator: Ctrl + F1)

See “Getting Contextual Help in SMIT (Graphical Interface),” on page 3-37 for more information.

On Window displays help about the window from which a selection was made

On Keys displays the defined mnemonic names, function keys, and keyboard accelerators

Index displays an index of help topics

On Help displays help about the use of the help functions

SMIT Buttons

To activate a button, point to the button with the mouse cursor and click once with the left button.

Help Button within Pop-Up Windows

Help displays help on how to use the pop-up window

Buttons: Primary Window

Cancel returns to the previous menu

If no previous menu exists (no menu is listed in the `Return To:` panel), this function will exit SMIT.

Buttons: Dialog Panel

List displays a list of choices for a particular field

Selections made from this list are entered into the entry field. This button is active only in dialog panels that have a list of choices available for an entry field.

Down Arrow selects the next entry from the option ring

This button is active only in dialog panels that contain an option ring.

Up Arrow selects the previous entry from the option ring

This button is active only in dialog panels that contain an option ring.

OK commits changes, closes the dialog, and performs the task

The `Command Output` panel appears.

Reset resets all the fields to their original values

This optional is active only in a dialog panel.

Cancel cancels any changes and returns to the previous menu

? Provides help for a dialog attribute field and the button where the cusor is located.

Help opens the Online Help System Window

Buttons: Multiselect List Window

OK	commits selections made in a multiselect list to the entry field
Find	finds the first occurrence of a character string in the list of options
Find Next	finds the next occurrence of a character string in the list of options
Cancel	returns to the dialog menu
Help	displays help on how to use the pop-up window

Buttons: Command Output Window

Stop	stops the execution of a command
Done	closes the command output window
Find	finds the first occurrence of a character string in the list of options
Find Next	finds the next occurrence of a character string in the list of options

Entering and Exiting SMIT in the Graphical Interface

In the windowing version of SMIT, you use standard mouse-driven window conventions in a menu-based format to construct complex system management commands. You can enter SMIT at the main menu, or you can enter at a specific submenu or dialog.

Prerequisites

SMIT requires access to the following files:

sm_menu_opt	SMIT database
sm_name_hdr	SMIT database
sm_cmd_hdr	SMIT database
sm_cmd_opt	SMIT database
smit.log	SMIT log file
smit.script	SMIT script file
/usr/lpp/info/...	InfoExplorer
/usr/lpp/msg/.../smit.cat	Message Catalog

Note: If any of these files are damaged or exist on an NFS server and that server goes down, SMIT may hang.

Enter SMIT at the Main Menu

From the window command line, enter:

```
smit
```

The main SMIT window appears on the screen.

Enter SMIT in a Submenu or Dialog

From the window command line, use the **smit** command and a *FastPath* parameter:

```
smit FastPath
```

The *FastPath* parameter allows you to enter at a lower-level menu. For example, to add a printer device, use the **smit mkprt** fast path.

This takes you directly to the Add a Printer submenu panel, which guides you through the steps for adding a printer device. See “Defining Fast Paths in SMIT” on page 3-4 for more information.

Exit SMIT

- Select **Exit –-> Exit Menu**

  OR

- Press the F12 key.

All the SMIT windows close, the session ends, and the command line returns.

Selecting SMIT Menu Titles (Graphical Interface)

You can select a menu or submenu title from the Return To: panel or the submenu or dialog panel. Menu selections from the Return To: panel return you to a previous menu, while selections from the submenu or dialog panel lead into other submenus or dialogs, further refining the task you want to perform.

To select SMIT menu titles, complete the following procedure:

1. Point to the desired menu or submenu title by placing the mouse pointer tip on the title or on the checkbox preceding the title.
2. Select the title by clicking once with the left button.

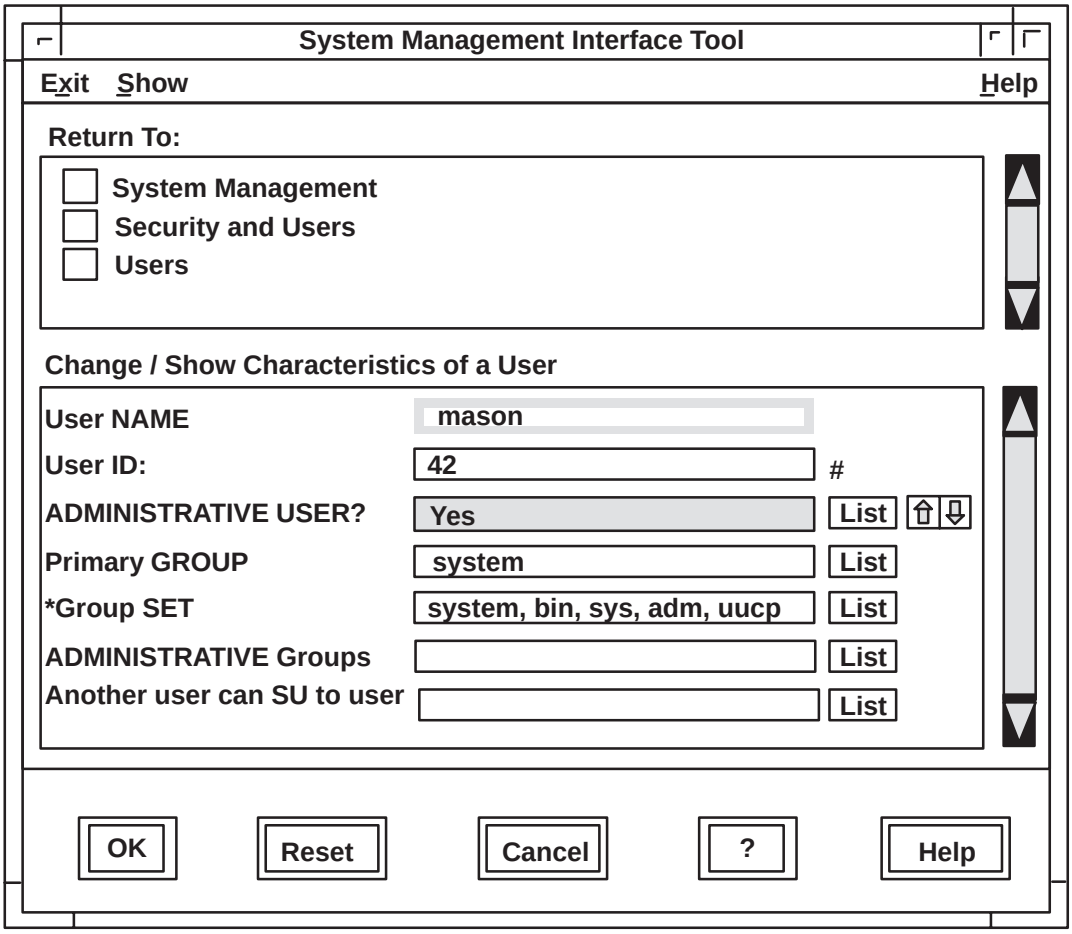
   Return To: panel selection:
   The selected menu title and any additional titles listed after it disappear from the Return To: panel. The selected menu now appears in the submenu or dialog panel.

   Submenu or dialog panel selection:
   The menu in the submenu or dialog panel disappears and is replaced by the selected menu. The previous menu title now appears as the last menu title in the Return To: panel.

SMIT Dialogs: Overview (Graphical Interface)

The following provides an overview of the main elements in a SMIT window dialog. For information on how to make entries in a dialog window, see “Completing SMIT Dialogs (Graphical Interface)” on page 3-19.

Many commands require parameters to accomplish their tasks. In SMIT windows, you supply parameters from a dialog panel. The following Sample SMIT Windows Dialog figure shows an example of a SMIT windows dialog panel:



Sample SMIT Windows Dialog

The dialog title, Change User Attributes, appears in the left corner above the dialog panel. The panel displays text-entry fields in the center, field names (labels) to the left and dialog symbols to the right. The menu bar reveals the three possible menu options: **Exit**, **Show**, and **Help**. Previous SMIT menu selections are listed in the Return To: panel. The button panel displays the **OK**, **Reset, Cancel**, **?**, and **Help** buttons.

Text-Entry Fields

There are two types of text-entry fields in a SMIT windows dialog: locked and unlocked:

Locked Locked entry fields appear two-dimensional with a gray rectangular outline as in the `User Name:` field in the dialog example. You cannot change the text in a locked entry field. If you click on a locked entry field, the flashing **`I`** cursor will not appear.

Unlocked Unlocked entry fields appear three-dimensional with a shadowed outline. You can enter and edit parameters in unlocked fields. When you select an unlocked entry field box, a flashing **`I`** cursor appears inside the field box indicating the ability to enter or edit text.

When you change a text-entry field value, the background of the field becomes gray. Entry fields containing edited parameters remain gray to indicate a change has been made. In the previous example, the gray text-entry field for `ADMINISTRATIVE User:` signifies a modified field.

Dialog Symbols

Dialog symbols designate the types of information required in each field. Dialog symbols appear to the left of field names and to the right of text-entry fields. In the previous example, note the * (asterisk) to the left of Group SET: and the # (pound sign) to the right of the User ID: text-entry field.

Symbol	Symbol Name	Meaning
*****	Asterisk	An entry is required. The symbol appears to the left of the field name or prompt.
#	Number sign	Enter a number.
X	The letter X	Enter a hexadecimal number.
Up and Down Arrow Buttons	Ring icon	This symbol offers a fixed set of options, which are displayed singly in the text-entry field.
List Button	List icon	This symbol displays a list of current, available choices generated by the system.

Completing SMIT Dialogs (Graphical Interface)

Within a dialog panel, you enter data in one of three ways:

- selecting a text-entry box and typing
- using the ring icon (represented by **Up Arrow** and **Down Arrow** buttons)
- selecting items from a list (represented by the **List** button)

When you have entered all necessary data, select the **OK** button to run the command. A command output window with the title of the command you are running appears over the dialog panel.

Prerequisites

You should be familiar with dialog panels and dialog symbols. See the section on “SMIT Dialogs: Overview (Graphical Interface)” on page 3-17.

Select a Text-Entry Box and Type Text

You can type text into an entry-field that is not locked or followed by a ring icon (**Up** and **Down Arrow** buttons).

1. Point to the text-entry box by placing the mouse pointer on the box.
2. Select the text-entry box by single clicking with the left mouse button. A flashing **I** shaped cursor appears in the box.
3. Enter the text.

   Note: The dialog symbol **#** requires that you enter a number, and the symbol **X** requires that you enter a hexadecimal number in the text-entry field.

Use the Option-Ring Buttons (Represented by Up Arrow and Down Arrow Buttons)

The option-ring buttons indicate a finite list of choices like **yes** and **no** and are represented by the **Up** and **Down Arrow** buttons to the right of the text-entry field. Option rings offer one choice at a time in the corresponding text-entry field. To change the value of the option ring, click once on the **Up Arrow** or **Down Arrow** button. The option-ring selection appears in the text-entry box.

Button	Action
Down Arrow	selects the next option from the option ring
Up Arrow	selects the previous option from the option ring

Sometimes, **List** buttons and option-ring buttons are offered together. The **List** button produces a pop-up display of all options while the option ring presents the options one at a time in the entry field.

Select Items from a List (Represented by the List Button)

List buttons appear to the right of many text-entry fields. Single-click on the **List** button to display a pop-up window listing the available options for the corresponding field. There are two types of list pop-up windows:

- single select
- multiselect

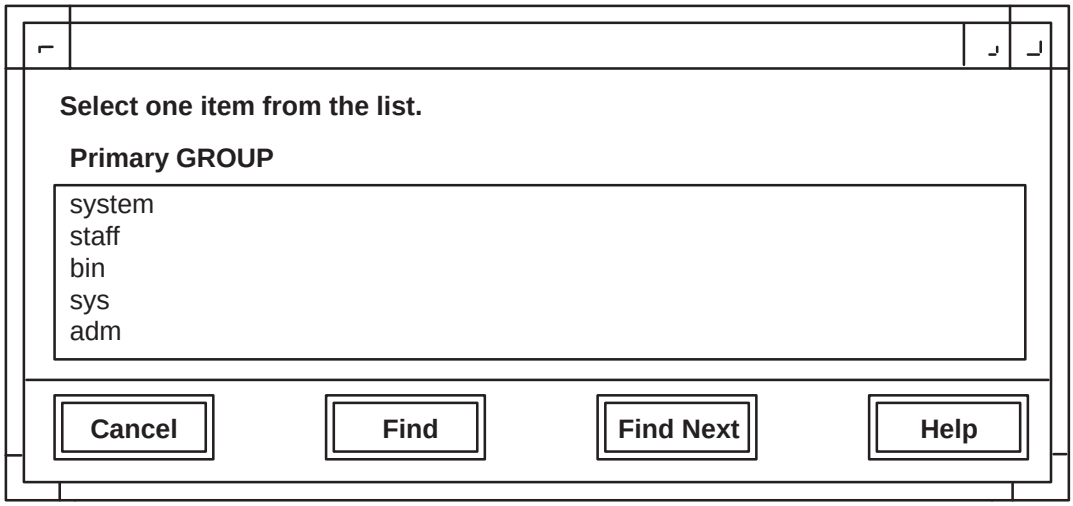
List pop-ups contain the **Cancel** button, which closes the pop-up list without changing the original field value. The **Help** button displays help messages associated with the pop-up. The **Find** button allows you to search for an occurrence of a character string in a list of items. The **Find Next** button finds the next occurrence of a string in a list.

Single-Selection List Window

The single-selection list windows enable you to select one item from the available options for the corresponding field. To make a selection in a single-selection list window, click on the desired item. The pop-up list disappears and the entry field is updated. The entry field remains gray to indicate that it has been modified.

The single-selection list window also has the **Cancel**, **Find**, **Find Next**, and **Help** buttons which give you additional assistance when using the window. For example, use the **Find** button to find a single occurrence of a character string and the **Find Next** button to find the second or several occurrences of a character string in the list of options. The **Cancel** button closes the pop-up list without changing the original field value. The **Help** button displays help messages associated with the pop-up list.

This window is an example of a typical single-selection list window:



Single Selection List Window

Multiselection List Window

Multiselect lists allow you to choose more than one item for the entry field. The multiselect list contains the **OK**, **Find**, and **Find Next** buttons as well as the **Cancel** and **Help** buttons.

You can select more than one item from a multiselection list by selecting one at a time:

Note: You cannot select more than 200 items.

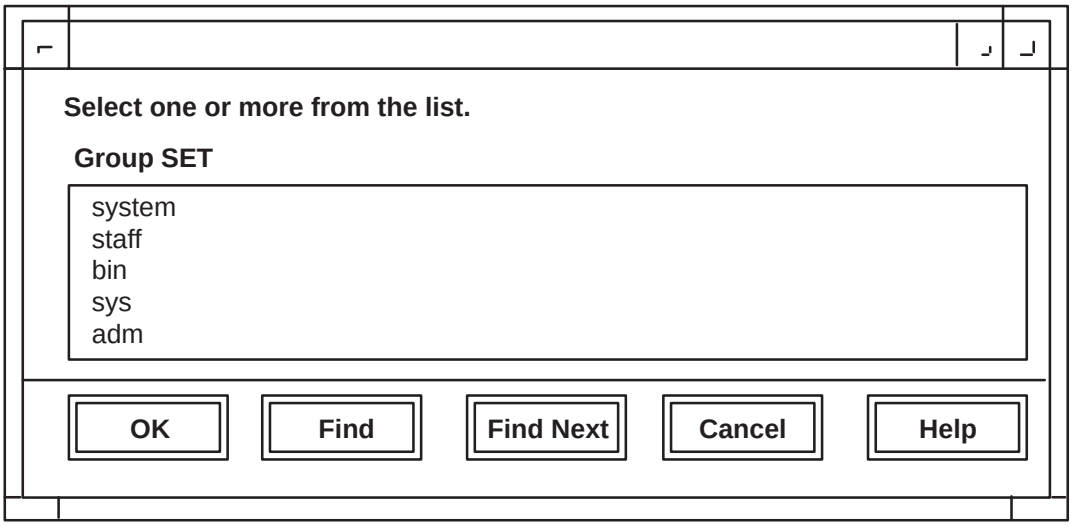
1. Click on each appropriate item.

   Selected items remain highlighted. To deselect an item, simply click on the highlighted item again.

2. To commit your choices, click on the **OK** button.

   The list pop-up disappears, and all the items are committed to the text-entry field. The entry field remains highlighted to indicate that it has been modified.

This window is an example of a typical multiselection list window:

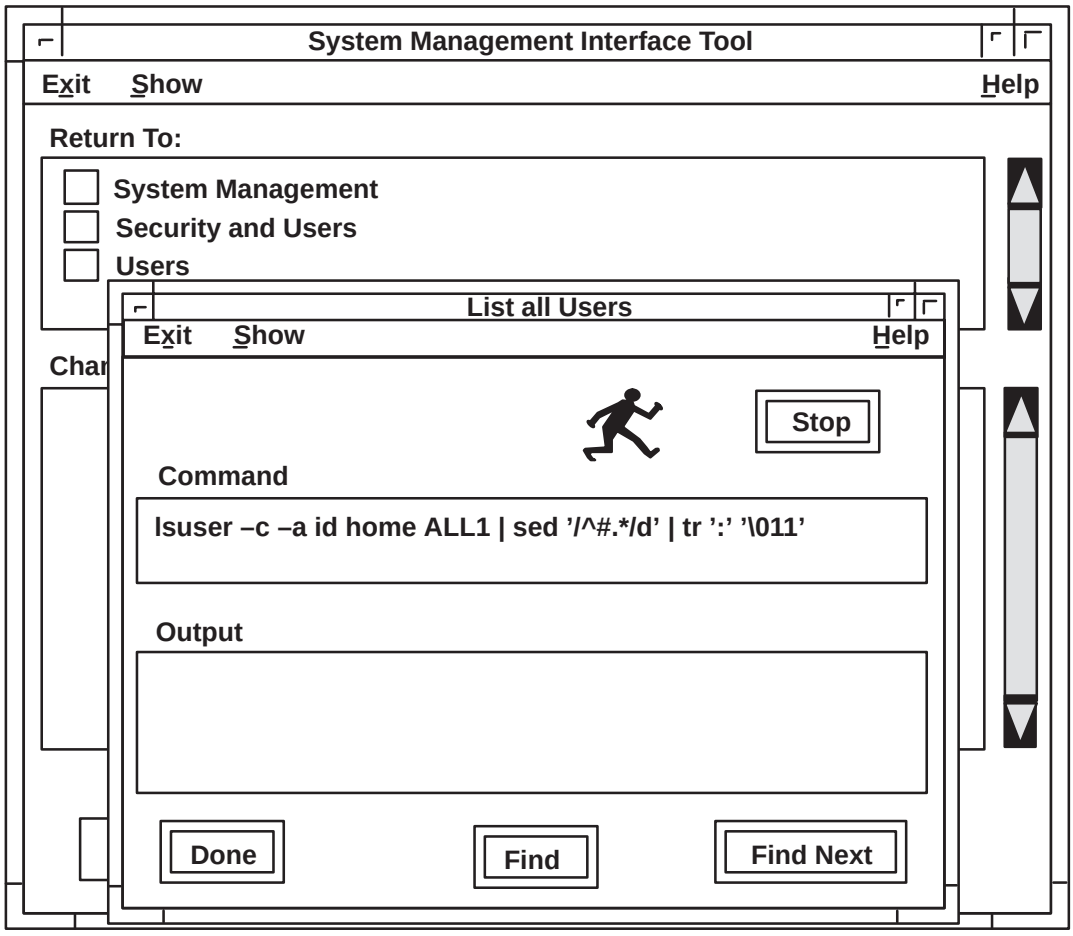


Multiselection List Window

SMIT Output Windows

SMIT Command Output Windows

When you have entered all the necessary data, select the **OK** button to run the command. A command output window with the title of the command you are running appears over the dialog panel as in the following figure. The command output window displays the status of the command that is running, the actual command statement as SMIT has written it, and the results of the command when completed.



Sample SMIT Windows Command Output Panel

While the command is running, an animated running-figure icon is displayed next to the **Stop** button. During the execution of a command, you can use the **Stop** button to cancel the process. When the command is complete, the `OK` status indicator appears and the figure raises its arms. If the command is not successful, the `Failed` status indicator appears and the figure falls down.

The Command panel displays the command statement you constructed with your menu choices. The Output panel displays the results of the executed command.

The **Done** button closes the command output window.

Alternate Command Execution Forms

The vast majority of commands that are run through SMIT produce the command output window. However, command execution can take two alternate forms:

- SMIT replaces itself with the target command. That is, rather than being run as a separate process, which SMIT normally does, the command is run in the current process. When the command finishes, the command-line prompt is displayed.
- Another alternate form temporarily gives the target command complete control of the screen. The command output is not controlled by SMIT.

SMIT in the ASCII Interface

The following sections provide an overview on how to use the ASCII (nongraphical) version of the System Management Interface Tool (SMIT). The ASCII interface is sometimes referred to as the curses interface because it uses the curses library.

The ASCII version of SMIT uses the cursor keys in a menu-based format to guide you through system management tasks. SMIT menus, selectors, and dialogs present complex system management tasks in an organized manner, prompting you for information when necessary. As you make menu selections, SMIT builds or runs the appropriate command.

Entering and Exiting SMIT in the ASCII Interface from a Terminal Environment

In the ASCII version of SMIT, you use the cursor keys and a menu-based format to help you construct system management commands. You can enter SMIT at the main menu, or you can enter a specific submenu or dialog.

Prerequisites

SMIT requires access to the following files:

sm_menu_opt	SMIT database
sm_name_hdr	SMIT database
sm_cmd_hdr	SMIT database
sm_cmd_opt	SMIT database
smit.log	SMIT log file
smit.script	SMIT script file
/usr/lpp/info/...	InfoExplorer
/usr/lpp/msg/.../smit.cat	Message Catalog

Note: If any of these files are damaged or exist on an NFS server and that server goes down, SMIT may hang.

Enter SMIT at the Main Menu

From the command line, enter:

```
smit
```

The main SMIT menu appears on the screen.

Enter SMIT in a Submenu or Dialog

From the command line, use the **smit** command and a *FastPath* parameter:

```
smit FastPath
```

The *FastPath* parameter allows you to enter at a lower-level menu. For example, to add a printer device, you can use the **smit mkprt** fast path. This takes you directly to the Add a Printer submenu, which guides you through the steps for adding a printer device.

Exit SMIT

Exit SMIT from any ASCII screen by pressing the F10 key or the Esc+0 key sequence.

If you use the Esc+0 key sequence, press and release the Esc key and immediately press the number 0.

Entering SMIT in the ASCII Interface from a Window Environment

SMIT in the ASCII interface (curses) can be accessed from a windowing environment. If you are interested in capturing and printing a screen image, use the ASCII version. It is easier to print a screen image from the ASCII version than from the AIXwindows version of SMIT. For more information, see the section on printing SMIT screens (ASCII interface) on page 3-36.

In the ASCII version of SMIT, you use the cursor keys and a menu-based format to help you construct system management commands. You can enter SMIT at the main ASCII menu screen, or you can enter a specific submenu or dialog further down in the application.

Prerequisite

SMIT requires access to the following files:

sm_menu_opt	SMIT database
sm_name_hdr	SMIT database
sm_cmd_hdr	SMIT database
sm_cmd_opt	SMIT database
smit.log	SMIT log file
smit.script	SMIT script file
/usr/lpp/info/...	InfoExplorer
/usr/lpp/msg/.../smit.cat	Message Catalog

Note: If any of these files are damaged or exist on an NFS server and that server goes down, SMIT may hang.

Enter SMIT at the Main Menu in an ASCII Interface

From the command line in a window, enter:

```
smit –C
```

The **–C** flag starts SMIT in an ASCII interface. The main SMIT menu in an ASCII interface appears on the screen.

Enter SMIT in a Submenu or Dialog in the ASCII Interface

From the command line in a window, use the **smit –C** command and a *FastPath* parameter:

```
smit –C FastPath
```

The **–C** flag starts SMIT in an ASCII interface and the *FastPath* parameter allows you to enter at a lower-level menu. For example, to add a user, you can use the **smit –C mkuser** fast path. This takes you directly to the Create User ASCII menu, which guides you through the steps in adding a user.

SMIT Function Keys (ASCII Interface)

When you are using the ASCII version of SMIT, the valid functions are displayed at the bottom of the screen. Only those functions that are valid for the specific menu, selector, or dialog are displayed. The following chart describes all functions in SMIT. If you use a key sequence (Esc+*Number*), press and release the Esc key, then immediately press the number key.

If your keyboard has been remapped, the following functions may not be valid:

Function Key	Command	Description
F1 or Esc+1	**Help**	gives more information on the topic to which the cursor points
F2 or Esc+2	**Refresh**	redraws the screen Use if console messages overwrite the screen.
F3 or Esc+3	**Cancel**	returns to the previous screen F3 in the main menu exits SMIT.
F4 or Esc+4	**List**	presents a list of choices for the highlighted entry field A pop-up selector screen displays a scrollable list of choices.
F5 or Esc+5	**Reset**	resets the entry field to the original setting
F6 or Esc+6	**Command**	displays the command that SMIT is building
F7 or Esc+7	**Edit** Also: **Select**	presents the highlighted text-entry field in a wide, pop-up selector screen for editing, and makes individual selections on multiselect lists
F8 or Esc+8	**Image**	displays the *FastPath* parameter for the current menu or dialog screen, and saves a screen image in the **smit.log** file so that you can print it later
F9 or Esc+9	**Shell**	escapes to a shell A confirmation pop-up menu or message is displayed.
F10 or Esc+0	**Exit**	exits SMIT
Enter	**Do**	executes the command built by SMIT or commits list entries to a dialog

Scrolling through Extended SMIT Screens (ASCII Interface)

Messages

When you are in a menu, selector, dialog, or command screen that is longer than one screen, the following messages may appear:

`[TOP]`	indicates the upper boundary of the text
`[BOTTOM]`	indicates the lower boundary of the text
`[MORE...nn]`	indicates that you can scroll up or down `nn` number of lines

Procedure

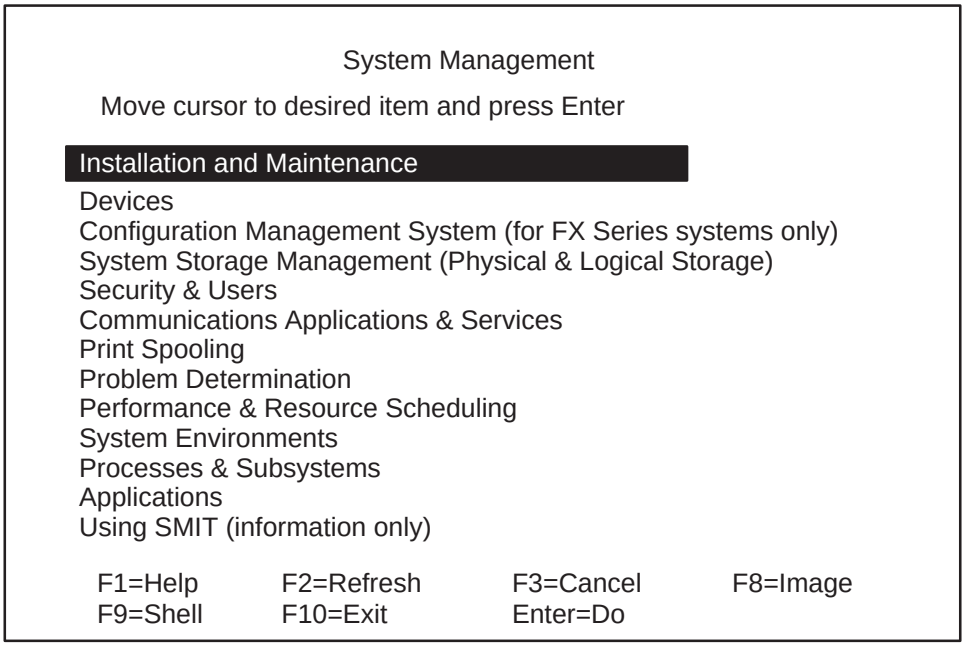
To scroll through the text you can use the following keys:

Key	**Action**
Up Arrow	scrolls up one line
Down Arrow	scrolls down one line
Page Up	scrolls up one screen
Page Down	scrolls down one screen
Home	scrolls to upper boundary of the text
End	scrolls to lower boundary of the text

Selecting SMIT Menu Options (ASCII Interface)

In SMIT menus, the list of items represents different tasks that can be performed from SMIT. From this list, select the type of task you want to accomplish. Menu selections that do not execute commands immediately lead into submenus, selector screens, or dialog screens, successively refining the task to be performed.

When you use the **smit** command to enter SMIT in the ASCII interface, you are presented with a menu similar to the following:



The first option, **Installation and Maintenance**, is highlighted, and available SMIT functions are listed at the bottom of the screen. If function keys are not enabled for your terminal, you will see key combinations instead of function keys.

To select SMIT menu options, complete the following:

1. Highlight the desired option using the following cursor keys:

Method	**Action**
Up Arrow	highlights previous option
Down Arrow	highlights next option
Home	highlights option at top of list
End	highlights option at bottom of list

2. Select the highlighted option by pressing the Enter key.

   Menu selections lead to one of the following:

   - Submenu
   - Selector screen
   - Dialog screen
   - Command output (often a result of selecting a list option from the menu)

Selecting List Options in SMIT Selectors (ASCII Interface)

SMIT selectors pop up on top of menus, dialogs, and other selectors. Selectors prompt you for a specific piece of information. There are two types of selectors:

- single-selection lists
- multiselection lists

Note: If you want a multiselection list, enter a **y** in the **multi_select** stanza when you configure your SMIT stanzas. Otherwise, a single-selection list appears when you press the F4 key.

The selector appears within a box, has a title at the top, available functions at the bottom, and the first option is highlighted.

The following selector screen appears after selecting the option **Change / Show Characteristics of a Diskette Drive** from the Diskette Drive submenu. The single-selection list prompts you to select a diskette drive.

Highlight Options in SMIT Selector Screens

The following keys can be used to highlight options:

Method	Action
Up Arrow	highlights previous option
Down Arrow	highlights next option
Home	highlights option at top of list
End	highlights option at bottom of list

Make a Selection from a Single-Selection List

A selection list takes the guesswork out of command building by removing the syntax and spelling errors commonly associated with typing commands. Single-selection lists prompt you to select a single option. In addition, use the **Find** (denoted by using a / (slash)) and the **Find Next** (denoted by using an n) options to search a list and find one or multiple occurrences of a character string in the list.

1. Highlight your choice using the cursor keys.
2. Press the Enter key.

   The entry field is updated to reflect your selection.

Make a Selection from a Multiselection List

Multiselection lists allow you to select more than one option before you press the Enter key and commit your selections. In addition, use the **Find** (denoted by using a / (slash)) and the **Find Next** (denoted by using an n) options to search a list and find one or multiple occurrences of a character string in the list.

1. Highlight the desired item and press the select key (F7 or Esc+7).

   A > (greater than sign) appears to the left of the selected item.
2. Highlight and press the select key for all other desired items.

   A > appears to the left of each of the selected items. To deselect an item, simply highlight the item you want to deselect and press the select key again. The > disappears.

3. Press the Enter key to commit the group of highlighted items.
   The entry field is updated to reflect your selections.

Scrolling and Completing SMIT Dialogs (ASCII Interface)

Dialog screens present a number of entry fields so that you can specify exactly how you want the command to run. Dialog symbols indicate the type of information required for each entry field. The information gathered in a dialog is used to build the command. After completing the dialog, press the Enter key, and SMIT builds and runs the command.

When a dialog screen appears, the first editable field name is highlighted and the cursor is waiting in the corresponding entry field. Dialog symbols are displayed to the left of the field names and to the right of the entry fields.

Prerequisites

To complete a SMIT dialog, enter the appropriate value for each field. The following dialog symbols indicate what kind of information is acceptable:

Symbol	Symbol Name	Meaning
*	Asterisk	appears to the left of the field name or prompt (required)
#	Number sign	Enter a number.
X	The letter X	Enter a hexadecimal number.
/	Slash	Enter a file name.
+	List of options or Option ring	displays a list of current, available choices generated by the system To view the list, press the List key (F4 or Esc+4). An option ring offers a fixed set of options, such as Yes or No. Press the Tab key to display ring options individually in the entry field.
[ ]	Brackets	beginning and end of an editable field
<	Less than sign	more text to the left of the visible field
>	Greater than sign	more text to the right of the visible field

Scroll in Dialog Screens

The following keys can be used to scroll in dialog screens:

Method	Action
Up Arrow	moves up to the previous field
Down Arrow	moves down to the next field
Home	moves to the top field
End	moves to the bottom field

Complete Dialogs

Within a dialog, data is entered in one of three ways:

- selection lists
- option rings
- text entries

Entry fields that you have modified remain in reverse video to indicate a change. To set an entry field to its original value, highlight the field, and press the Reset key (F5 or Esc+5).

After completing the necessary entry fields, press the Enter key. SMIT builds and runs the command.

Select an Option from a List

Making a list selection frees you from the details of complex command syntax, valid parameter values, and system command spelling. There are two types of selection lists, single-selection and multiselection, and they are noted by the dialog symbol + (addition sign) to the right of the entry field. Press the List key (F4 or Esc+4) to view the entire list of choices for the highlighted field. Use the / (slash) key to search for a single occurrence of a character string and the letter n to search for the next occurrence of a character string.

The **Group Set** screen appears when you press the / (slash) key to search for a character string.

To learn how to make a list selection, refer to “Selecting List Options in SMIT Selectors (ASCII Interface)” on page 3-30.

Select an Item from an Option Ring

Option rings offer a finite list of choices like True or False, and are noted by the dialog symbol + (plus sign) to the right of the entry field. To display an option in the entry field, choose:

Method	Action
Tab	next option appears in the entry field
Shift-Tab	previous option appears in the entry field

Note: Generally you can press the List key (F4 or Esc+4) to view the list of choices; however, some option rings do not display the choices in list form.

Make a Text Entry

Left and right brackets [ ] in the entry field indicate the beginning and end of an editable field. Enter the text between the [ ] (brackets). If you need additional room to enter text or view the entry field, press the Edit key (F7 or Esc+7). The highlighted text entry field is displayed in a wide pop-up screen for viewing or editing.

SMIT Command Status Screens (ASCII Interface)

A Command Status screen displays the status of running or executed commands. Commands run as a result of committing a dialog (pressing the Enter key while in the dialog), or sometimes as a result of selecting a list option from a menu. Output from command execution takes three forms:

- A Command Status screen displays the command currently running and the results of the command. The prompt `Command:running` is displayed at the upper left of the Command Status screen, and the function keys are not displayed. You may be prompted for any additional information necessary to execute the command.

  As output is generated by the command, it is displayed on the screen. After the command is completed, the results of the command, the function keys, and the message `Command: OK` or `Command: Failed` appears. There are also indicators regarding whether the command has written to standard output (`stdout:`) and standard error (`stderr:`).

  The following figure is a result of selecting **List all Defined Printers/Plotters** in the Printer/Plotter Devices submenu.

```
                    COMMAND STATUS

Command: OK         stdout: yes         stderr: no

Before command completion, additional instructions may
appear below.

lp0 Available 00-00-0P-00 4019 LaserPrinter




F1=Help       F2=Refresh     F3=Cancel      F6=Command
F8=Image      F9=Shell       F10=Exit       Enter=Do
```

- In the second form, SMIT is replaced with the target command. Rather than running the command as a separate process, in SMIT, the command runs in the current process. When the command finishes, the command line prompt is displayed. To reenter the SMIT application, you must type **smit** at the command line.
- The third form temporarily gives the target command complete control of the screen. The command output is not controlled by SMIT.

Duplicating System Configuration Using the smit.script File

A record of each command executed by SMIT along with a time stamp is saved in the **smit.script** file, which is located in your home directory (**$HOME/smit.script**). If you would like to save the **smit.script** file elsewhere, you can run the **smit** command with the **–s** *PathName* flag to save the **smit.script** file in the file specified by the *PathName* parameter. You must have write permission for the directory in which you have requested the **smit.script** file to be written or the file will not be created. SMIT does not overwrite the file; instead, the file is appended.

You can use the **smit.script** file as an executable shell script to duplicate system configuration, or you can use it to learn more about the command that was built from the selections made during the use of SMIT. The script is a flat text file that can be edited or modified to separate it into distinct configuration subtasks.

Prerequisites

Review or edit the **smit.script** file. The **smit.script** file can be edited to create slight variations in the configuration commands, or to use only subsets of the commands.

Procedure

1. To make the **smit.script** file executable for duplicate configuration, enter:

   ```
   chmod +x smit.script
   ```

2. To duplicate your configuration, enter:

   ```
   smit.script
   ```

   Note: SMIT runs commands under the Korn shell (**/usr/bin/ksh**). Some command strings in the **smit.script** file may require this environment to run correctly.

3. Rename or copy the **smit.script** file to prevent SMIT from modifying it.

Printing SMIT Screens (ASCII Interface)

If you want a hardcopy record of a menu, dialog, or a selector, you can print a screen image. Screen images can be captured and saved in the **smit.log** file.

To print SMIT screens, complete the following procedure:

1. Capture a screen image:
   a. Select the screen you want to capture and display it on your monitor.
   b. Press the Image key (F8 or Esc+8).
   c. Press the Enter key.

   Note: The captured image is saved to the **smit.log** file. The **smit.log** file defaults to your **$HOME** directory, unless you specified the **–l** flag to redirect it. To see how to redirect your log file, refer to the **smit** command.
2. Print the screen image from the **smit.log** file:
   a. Press the F9 + Enter key sequence to exit to a shell.
   b. Often the **smit.log** file contains more information than you want to print. To view your **smit.log** file, from the directory containing the file, enter:

      ```
      pg smit.log
      ```

      Use a text editor to remove unwanted text. For example, to use the vi editor, enter:

      ```
      vi smit.log
      ```
   c. To print the file, from the directory containing the file, enter:

      ```
      enq –P PrinterName smit.log
      ```

      The **smit.log** file is sent to the printer designated by *PrinterName.*

Getting Contextual Help in SMIT (Graphical Interface)

The Help menu offers several different options. The **On Context** option provides information about the various menu, submenu, and dialog options as well as details about dialog fields.

To get contextual help in SMIT, complete the following procedure:

1. From the menu bar, click on the **Help** menu title to display the list of options.
2. From the pull-down menu, click on the **On Context** option.

   The pointer appears as a question mark while you are in the Help mode. All labels for which SMIT contextual help is available appear in reverse video in the main SMIT window. A Help Message pop-up window appears and prompts you to select any highlighted label.
3. Click on the label for which you want help.

   The message window disappears and information on the label you selected appears in a Help Context pop-up window.

   To get information on another highlighted label, use the pointer to select the new label. The new information replaces the previous label information in the Help Context window.
4. To quit the Help mode, select either the **Keep** button or **Cancel** button on the pop-up.

   The **Keep** button exits the Help mode, yet continues to display the information in a Help Context window on your screen.

   The **Cancel** button closes the Help Context pop-up window and exits the Help mode.

Chapter 4. Starting and Stopping the System

This chapter describes the how to start and stop the system. It includes the following topics.

Note: Most of the information in this chapter pertains to both the standard AIX system and the FX Series system. Differences between these two systems are specifically noted.

Process for the System Booting Automatically

When the base operating system boots, the system initiates a complex set of tasks. Under normal conditions, these tasks are performed automatically. The following sections provide information about the process of booting the system:

Procedures for You to Boot the System

There are some situations when you want to instruct the system to boot; for example, to cause the system to recognize newly installed software, to reset peripheral devices, to perform routine maintenance tasks like checking file systems, or to recover from a system hang or crash. For information on these procedures, see the following sections:

Creating Boot Images

When the system is first installed, the **bosboot** command creates a boot image from a RAM (random access memory) disk file system image and the operating system kernel. The boot image is transferred to a particular media such as the hard disk. When the machine is rebooted, the boot image is loaded from the media into memory.

For more information, see "Creating Boot Images" on page 4-16.

Identifying and Changing the System Run Level

The system run level specifies the system state and defines which processes are started. For example, when the system run level is 3, all processes defined to operate at that run level are started. Near the end of the system boot phase of the boot process, the run level is read from the initdefault entry of the **/etc/inittab** file. The system run level can be changed with the **init** command. The **/etc/inittab** file contains a record for each process that defines run levels for that process. When the system boots, the **init** command reads the **/etc/inittab** file to determine which processes to start.

For information on these procedures, see the sections:

- “Identifying System Run Levels” on page 4-18
- “Changing System Run Levels” on page 4-19
- “Changing the /etc/inittab File” on page 4-21

Shutting Down the System

For information on stopping and shutting down the system, see “Shutting Down the System” on page 4-23.

Understanding the Boot Process

During the boot process, the system tests the hardware, loads and executes the operating system, and configures devices. To boot the operating system, the following resources are required:

- a *boot image* that can be loaded after the machine is turned on or reset
- access to the root and **/usr** file systems

There are three types of system boots:

Hard Disk Boot A machine is started for normal operations for both a standard AIX system and an FX Series system. For more information, see “Understanding System Boot Processing” on page 4-4.

Diskless Network Boot
On a standard AIX system, a diskless or dataless workstation is started remotely over a network. One or more remote file servers provides the files and programs that diskless or dataless workstations need to boot. Note that this type of system boot does not exist on an FX Series system.

Service Boot A machine is started from a tape or CD-ROM. (On a standard AIX system a machine can also be started from the network.) This condition is also called *maintenance mode*. In maintenance mode, a system administrator can perform tasks such as installing new or updated software and running diagnostic checks. For more information, see “Understanding the Service Boot Process” on page 4-9.

Understanding System Boot Processing

Most users perform a hard disk boot when starting the system for general operations. The boot image is found on a local disk created when the operating system was installed.

A hard disk boot occurs when the system is started either by:

- turning on the power switch (a cold boot)
- restarting with the **reboot** or **shutdown** commands (a warm boot)

During the boot process, the system configures all devices found in the machine and initializes other basic software required for the system to operate (such as the Logical Volume Manager). On an FX Series system, only those devices that were online prior to the last system shutdown are brought online.

On a standard AIX system, diskless network clients also require a boot image and access to the operating system file tree. Diskless network clients have no local file systems and get all of their information by way of remote access.

A number of events must occur before the system is ready for use. These events can be divided into the following phases, which are described in the following sections:

- Phase 1: ROS Kernel Init Phase
- Phase 2: Base Device Configuration Phase
- Phase 3: System Boot Phase

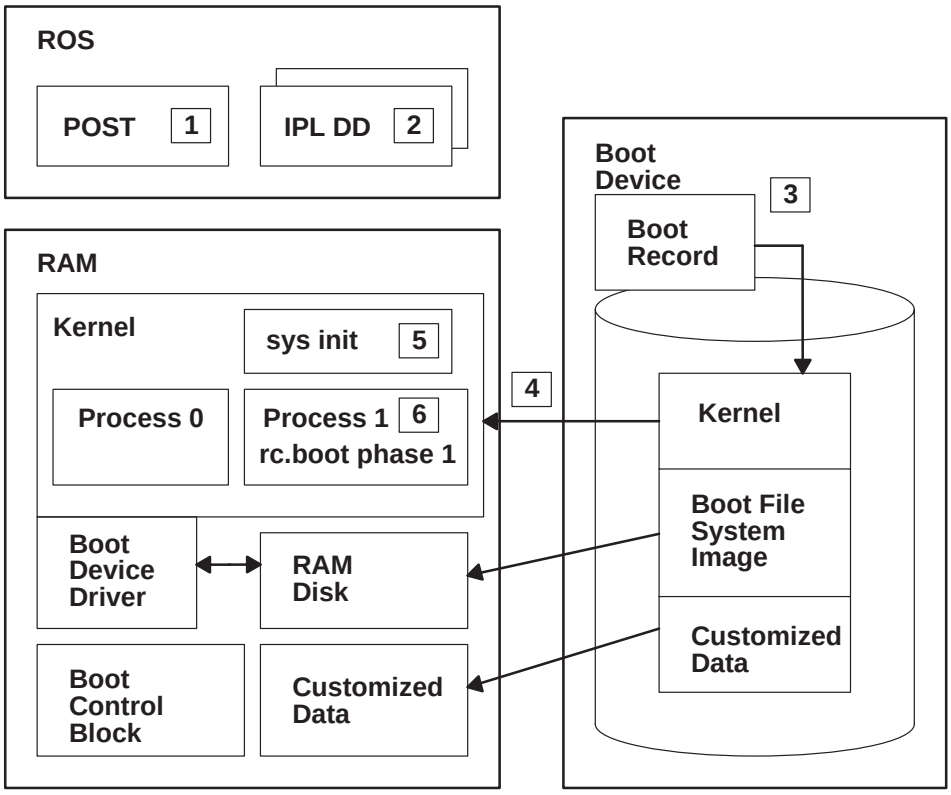
Phase 1: ROS Kernel Init Phase

The Read Only Storage ROS Kernel Init Phase diagram on page 4-5 illustrates the kernel initialization that takes place before the system boot process is started. The ROS kernel initialization phase involves the following steps:

1. The ROS initial program load (IPL) checks the user boot list, a list of available boot devices. This boot list can be altered to suit your requirements using the **bootlist** command. If the user boot list in NVRAM is not valid or if a valid boot device is not found, the default boot list is then checked. In either case, the first valid boot device found in the boot list is used for system startup. If a valid user boot list exists in NVRAM, the devices in the list are checked in order. If no user boot list exists, all adapters and devices on the bus are checked. In either case, devices are checked in a continuous loop until a valid boot device is found for system startup.

   Note: The system maintains a default boot list located in ROS and a user boot list stored in NVRAM, for a normal boot.

2. When a valid boot device is found, the first record or program sector number (PSN) is checked. If it is a valid boot record, it is read into memory and is added to the initial program load (IPL) control block in memory. Included in the key boot record data are the starting location of the boot image on the boot device, the length of the boot image, and instructions on where to load the boot image in memory.
3. The boot image is read sequentially from the boot device into memory starting at the location specified in the boot record. The disk boot image consists of the kernel, a RAM file system, and base customized device information.
4. Control is passed to the kernel, which begins system initialization.
5. Process 1 executes **init**, which executes phase 1 of the **rc.boot** script.



ROS Kernel Init Phase

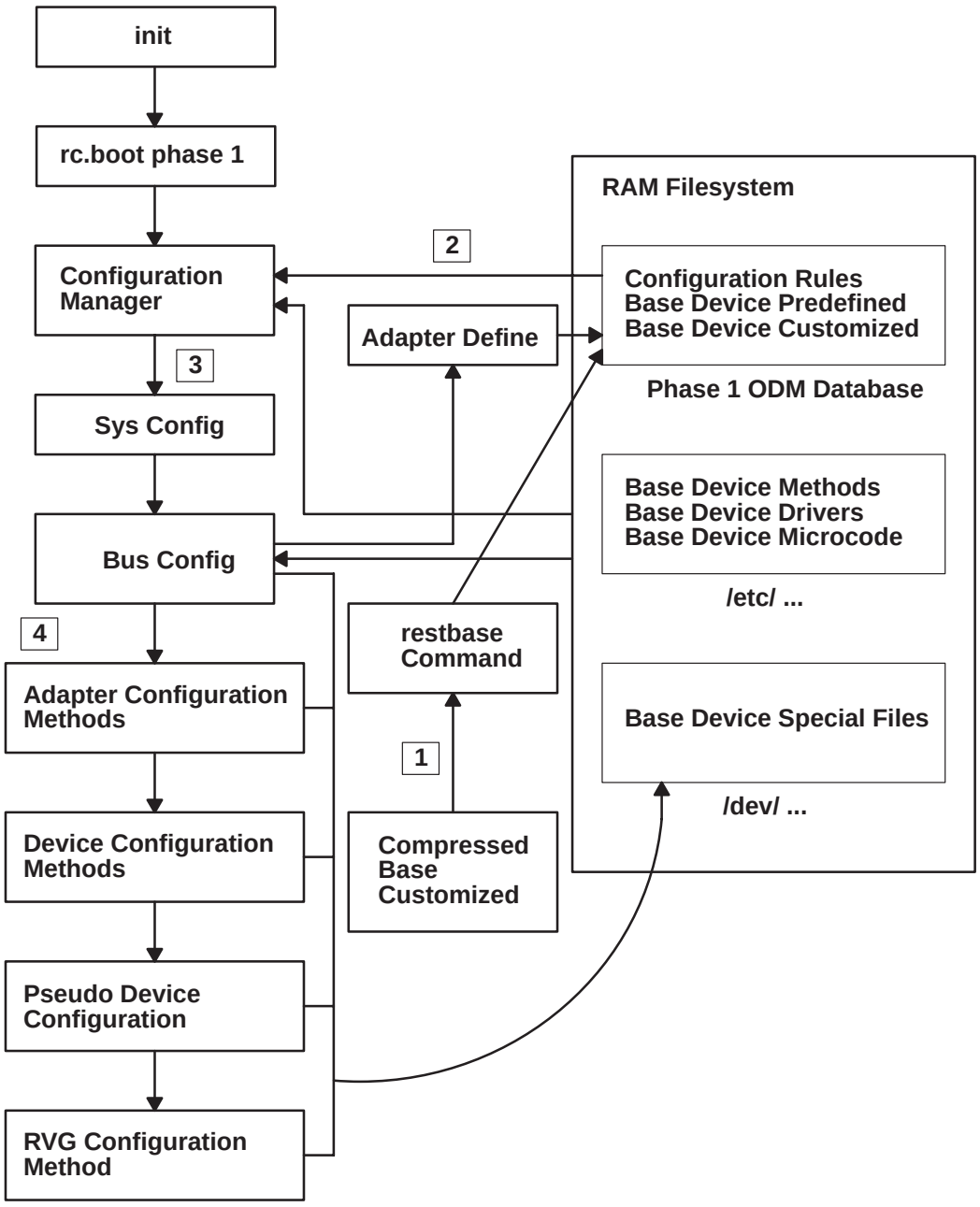
Phase 2: Base Device Configuration Phase

When the kernel initialization phase is completed, base device configuration begins. The Base Device Configuration Phase diagram on page 4-6 illustrates this part of the boot process.

The **init** process starts the **rc.boot** script. Phase 1 of the **rc.boot** script performs the base device configuration, and it includes the following steps:

1. The boot script calls the **restbase** program to build the customized Object Database Manager (ODM) database in the RAM file system from the compressed customized data.
2. The boot script starts the configuration manager, which accesses phase 1 configuration rules to configure the base devices.
3. The configuration manager starts the **sys**, **bus**, **disk**, SCSI, and the Logical Volume Manager (LVM) and **rootvg** volume group (RVG) configuration methods.
4. The configuration methods load the device drivers, create special files, and update the customized data in the ODM database.

For more detailed information, refer to the guides *Writing a Device Driver* and *Writing a Fault Tolerant Device Driver*.



Base Device Configuration Phase

Phase 3: System Boot Phase

The System Boot Phase diagram on page 4-8 illustrates the steps involved in the system boot process.

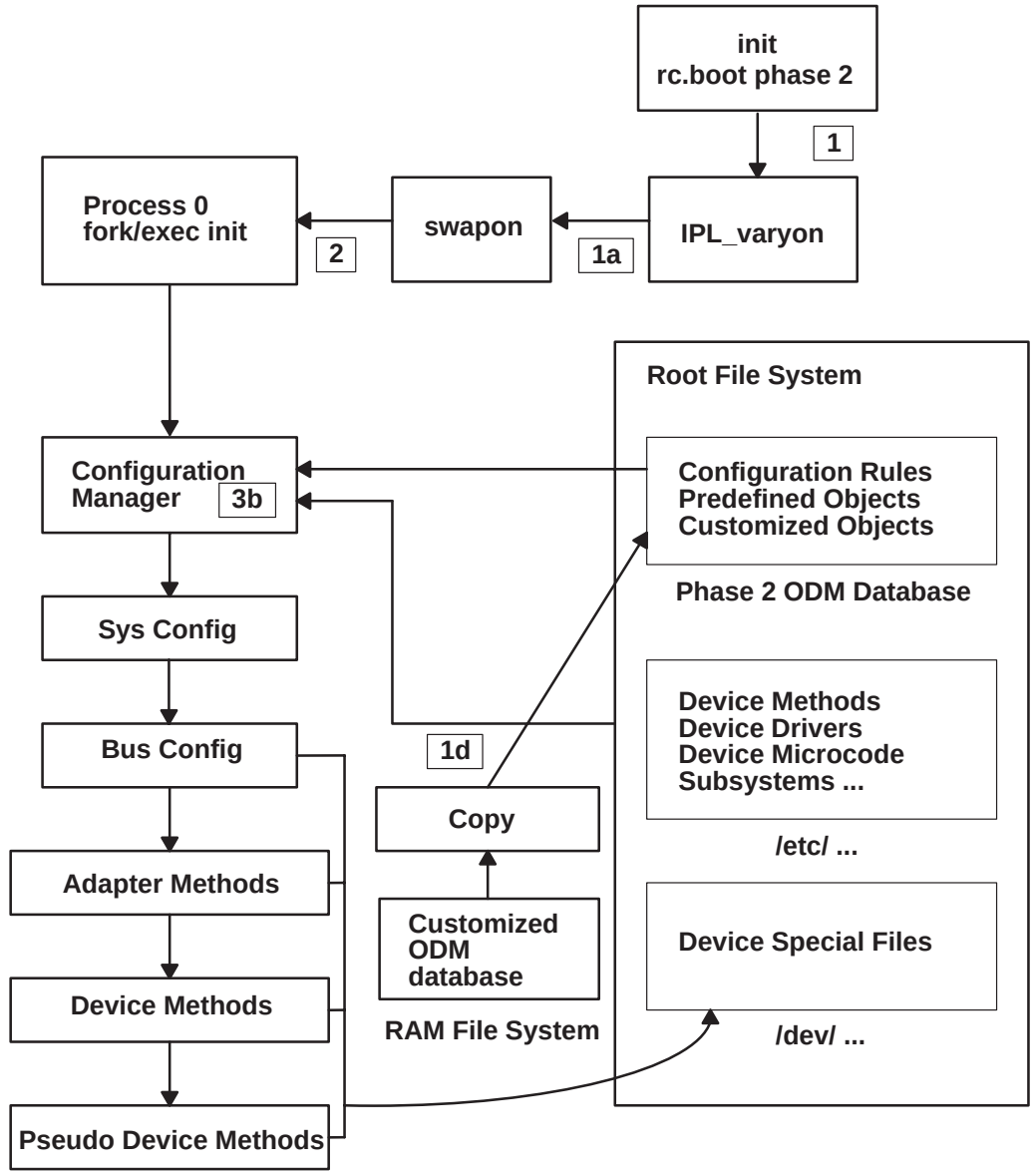
1. The **init** process starts phase 2 execution of the **rc.boot** script. Phase 2 of **rc.boot** includes the following steps:
   a. Call the **ipl_varyon** program to vary on the **rootvg** volume group (RVG).
   b. Mount the hard disk file systems onto the RAM file system.
   c. Run **swapon** to start paging.
   d. Copy the customized data from the ODM database in the RAM file system to the ODM database in the hard disk file system.
   e. Unmount temporary mounts of hard disk file systems and then perform permanent mounts of root, **/usr**, and **/var**.
   f. Exit the **rc.boot** script.
2. After phase 2 of **rc.boot**, the boot process switches from the RAM file system to the hard disk root file system.
3. Then the init process executes the processes defined by records in the **/etc/inittab** file.

   One of the instructions in the **/etc/inittab** file executes phase 3 of the **rc.boot** script, which includes the following steps:
   a. Mount **/tmp** hard disk file system.
   b. Start the configuration manager phase 2 to configure all remaining devices.

   Note: On an FX Series system, during phase 2 the change daemon is started.

   c. Use the **savebase** command to save the customized data to the boot logical volume.
   d. Exit the **rc.boot** script.

At the end of this process, the system is up and ready for use.



System Boot Phase

Understanding the Service Boot Process

Occasions may arise when a service boot is needed to perform special tasks such as installing new or updated software, performing diagnostic checks, or maintenance. In this case, the system starts from a bootable medium (CD-ROM or tape). On a standard AIX system, the system can also start from the network.

The service boot sequence of events is similar to the sequence of a normal boot. The events can be outlined as follows:

1. Control is passed to ROS, which performs a power-on self-test (POST).
2. ROS checks the user boot list, which can be altered to suit your requirements using the **bootlist** command. If the user boot list in NVRAM is not valid or if no valid boot device is found, the default boot list is checked. In either case, the first valid boot device found in the boot list is used for system startup.

   Note: The system maintains a default boot list, located in ROS, and a user boot list, stored in NVRAM, for a normal boot.
3. When a valid boot device is found, the first record or Program Sector Number (PSN) is checked. If it is a valid boot record, it is read into memory and is added to the Initial Program Load (IPL) control block in memory. Included in the key boot record data are the starting location of the boot image on the boot device, the length of the boot image, and the offset to the entry point to start execution when the boot image is in memory.
4. The boot image is read sequentially from the boot device into memory, starting at the location specified in the boot record.
5. Control is passed to the kernel, which begins executing programs in the RAM file system.
6. The Object Data Manager (ODM) database contents determine which devices are present, and the **cfgmgr** command dynamically configures all devices found, including all disks which are to contain the root file system.
7. If CD-ROM, tape, or the network is used to boot the system, the rootvg volume group (RVG) is not varied on, since the RVG may not exist (as is the case when installing the operating system on a new system). Network configuration may occur at this time. No paging occurs when a service boot is performed.

At the end of this process, the system is ready for installation, maintenance, or diagnostics.

Understanding the RAM File System

The RAM file system, part of the boot image, is totally memory-resident and contains all programs that allow the boot process to continue. The files in the RAM file system determine the type of boot.

A service boot RAM file system might not have the logical volume routines, since the **rootvg** volume group may not need to be varied on. During a hard disk boot, however, it is desirable that the **rootvg** volume group be varied on and paging activated as soon as possible. Although there are differences in these two boot scenarios, the structure of the RAM file system does not vary to a great extent. The following steps are performed:

1. The **init** command on the RAM file system used during boot is actually the **ssh** (simple shell) program. The **ssh** program controls the boot process by calling the **rc.boot** script.
2. The first step for **rc.boot** is to determine from what device the machine was booted. The boot device determines which devices should be configured on the RAM file system.
3. On a standard AIX system, if the machine is booted over the network, the network devices need to be configured so that the client's file systems can be remotely mounted.
4. In the case of a tape or CD-ROM boot, the console is configured to display the BOS install menus.
5. After the **rc.boot** script finds the boot device, then the appropriate configuration routines are called from the RAM file system.
6. The **rc.boot** script itself is called twice by the **ssh** program to match the two configuration phases during boot.
7. A third call to **rc.boot** occurs during a disk or a network boot when the real **init** command is called.
8. A **rc.boot** stanza in the **inittab** file does the final configuration of the machine.

The RAM file system for each boot device is also unique due to the various types of devices to be configured. There is a prototype file associated with each type of boot device. The prototype file is a template of files making up the RAM file system. The **mkfs** command is used by the **bosboot** command to create the RAM file system using the various prototype files. See the **bosboot** command for more details.

Booting an Uninstalled System

The procedure for booting a new or uninstalled system is part of the installation process. For information on how to boot an uninstalled system, see *Installation Troubleshooting*.

Rebooting a Running System

There are two methods for shutting down and rebooting your system, depending upon whether multiple users are logged on to the system:

- If multiple users are logged in to the system, use the **shutdown** command.
- If you are the only user logged in to the system, use the **reboot** command.

Rebooting a Multiuser System

You can use either SMIT or the **shutdown** command to stop and reboot the operating system when it is being accessed by multiple users.

Using SMIT

1. Use the **smit shutdown** fast path to access the **Stop the System** menu.
2. Specify **yes** in the field:

   `RESTART the system after shutdown`

   On this menu you can also:

   - send a message to other users notifying them that the system is shutting down
   - tell the system whether or not you want to receive status messages during shutdown
   - enter the time that the shutdown is to occur
3. Confirm your choice to shut down and reboot the system.

Using the shutdown Command

The **shutdown** command is the safest and most thorough way to halt the operating system. This command notifies users that the system is about to go down, kills all existing processes, unmounts file systems, and halts the system. When the **–r** flag is specified with the **shutdown** command, the system reboots after it completes the shutdown.

Type `shutdown –r` and press the Enter key. The system shuts down and reboots.

Rebooting a Single-User System

Use the **reboot** command to reboot the operating system when it is only being accessed by one user. The **reboot** command synchronizes the hard disks and performs some other shutdown activities without halting the system.

Type `reboot` and press the Enter key. The system shuts down and reboots.

Booting a System That Crashed

In some instances, you may have to boot a system that has stopped (crashed) without being properly shut down. This procedure covers the basics of how to boot if your system was unable to recover from the crash.

Prerequisites

- Your system crashed and was not properly shut down due to unusual conditions.
- Your system is turned off.

Procedure

1. Ensure that all hardware and peripheral devices are properly connected.
2. On a standard AIX system, turn on the peripheral devices.
3. Turn on the system.
4. Watch the screen for information about automatic hardware diagnostics.
   - If any hardware diagnostics tests are unsuccessful, refer to the hardware documentation.
   - If all hardware diagnostics tests are successful, go to the next step.

Diagnosing Boot Problems

A variety of factors can cause a system to be unable to boot:

- hardware problems
- defective boot tapes or CD-ROMs
- damaged file systems
- errors in scripts such as **/etc/rc.boot**

For information on accessing a system that will not boot from the disk drive, see “Accessing a System That Will Not Boot” on page 4-15.

For other diagnostic information, refer to the *Problem Solving Guide and Reference*.

Accessing a System That Will Not Boot

If you have a system that will not boot from the hard disk, see the procedure on how to access a system that will not boot in the *Installation Troubleshooting* guide.

This procedure basically describes how to boot off the CDROM or tape and follow the service boot process. It enables you to get a system prompt so that you can attempt to recover data from the system or perform corrective action that will enable the system to boot from the hard disk.

Notes:

1. This procedure is intended only for experienced system managers who have knowledge of how to boot or recover data from a system that is unable to boot from the hard disk. Most users should not attempt this procedure but should instead contact their service representative.
2. This procedure is not intended for system managers who have just completed a new installation, since in this case the system will not contain data that needs to be recovered. If you are administering a standard AIX system and are unable to boot from the hard disk after completing a new installation, then contact your service representative. If you are administering an FX Series system, then first refer to the guide *Diagnostics and Troubleshooting on a Fault Tolerant System*.

Creating Boot Images

To install the base operating system or to access a system that will not boot from the system hard drive, you need a boot image. This procedure describes how to create boot images. The boot image varies for each type of device. The associated RAM disk file system contains device configuration routines for the following devices:

- disk
- tape
- CD-ROM
- Network Token-Ring, Ethernet, or FDDI device (for a standard AIX system only)

Prerequisites

- You must have root user authority to use the **bosboot** command.
- The physical disk must contain the boot logical volume.

  1. To determine which disk device to specify, enter:

     ```
     lsvg –l rootvg
     ```

     The **lsvg –l** command lists the logical volumes on the root volume group (**rootvg**). From this list you can find the name of the boot logical volume.

  2. Then use the following command:

     ```
     lslv –l boot_volume
     ```

     where *boot_volume* is the name of the boot logical volume that you just located. From this list you can find the name of the physical disk containing the boot logical volume.

Creating a Boot Image on a Boot Logical Volume

If the base operating system is being installed (either a new installation or an update), the **bosboot** command is called to place the boot image on the boot logical volume. The boot logical volume is a physically contiguous area on the disk created through the Logical Volume Manager (LVM) during installation.

The **bosboot** command does the following:

1. Checks the file system to see if there is enough room to create the boot image.
2. Creates a RAM file system using the **mkfs** command and a prototype file.
3. Calls the **mkboot** command, which merges the kernel and the RAM file system into a boot image.
4. Writes the boot image to the boot logical volume.

To create a boot image on the default boot logical volume on the fixed disk **/dev/hdisk0**, enter:

```
bosboot –a –d /dev/hdisk0
```

Note: Do not reboot the machine if the **bosboot** command fails while creating a boot image. The problem should be resolved and the **bosboot** command run to successful completion. For information about solving boot problems, see *Problem Solving Guide and Reference*.

You must reboot the system for the new boot image to be available for use.

Creating a Boot Image Containing an Uncompressed RAM File System Boot Image

To create an uncompressed RAM file system boot image for the fixed disk **/dev/hdisk0**, enter:

```
bosboot –a –U –d /dev/hdisk0
```

Creating a Boot Image Containing a Compressed RAM File System for a Network on a Standard AIX System

To create a compressed RAM file system boot image for an Ethernet boot, enter:

```
bosboot –ad /dev/ent
```

For a Token-Ring boot:

```
bosboot –ad /dev/tok
```

Identifying System Run Levels

Before performing maintenance on the operating system or changing the system run level, you may need to examine the various run levels. This procedure describes how to identify the run level at which the system is operating and how to display a history of previous run levels. The **init** command determines the system run level.

Identifying the Current Run Level

At the command line, type `cat /etc/.init.state` and press the Enter key. The system displays one digit; that is the current run level. See the **init** command or the **/etc/inittab** file for more information about run levels.

Displaying a History of Previous Run Levels

You can display a history of previous run levels using the **fwtmp** command.

Note: The **bosext2.acct.obj** code must be installed on your system to use this command.

1. Log in as root user.
2. Type `/usr/lib/acct/fwtmp </var/adm/wtmp |grep run-level` and press the Enter key.

   The system displays information similar to the following:

```
run-level 2  0 1 0062 0123 697081013 Sun Feb  2 19:36:53 CST 1995
run-level 2  0 1 0062 0123 697092441 Sun Feb  2 22:47:21 CST 1995
run-level 4  0 1 0062 0123 698180044 Sat Feb 15 12:54:04 CST 1995
run-level 2  0 1 0062 0123 698959131 Sun Feb 16 10:52:11 CST 1995
run-level 5  0 1 0062 0123 698967773 Mon Feb 24 15:42:53 CST 1995
```

Changing System Run Levels

This procedure describes two methods for changing system run levels for multi-user or single-user systems.

When the system starts the first time, it enters the default run level defined by the initdefault entry in the **/etc/inittab** file. The system operates at that run level until it receives a signal to change it.

Currently Defined Run Levels

The following are the currently defined run levels:

0–9	When the **init** command changes to run levels 0–9, it kills all processes at the current run levels, and then restarts any processes associated with the new run levels.
0–1	reserved for the future use of the operating system
2	default run level
3–9	can be defined according to the user's preferences.
a, b, c	When the **init** command requests a change to run levels **a**, **b**, or **c**, it does not kill processes at the current run levels; it simply starts any processes assigned with the new run levels.
Q, q	tells the **init** command to reexamine the **/etc/inittab** file

Changing Run Levels on Multiuser Systems

1. Check the **/etc/inittab** file to confirm that the run level to which you are changing supports the processes that you are running. The getty process is particularly important, since it controls the terminal line access for the system console and other logins. Ensure that the getty process is enabled at all run levels.
2. Use the **wall** command to inform all users that you intend to change the run level and request that users log off.
3. Use the **smit telinit** fast path to access the **Set System Run Level** menu.
4. Enter the new run level in the field:

   `System RUN LEVEL`
5. Confirm your choice to set the new run level.

   The system responds by telling you which processes are terminating or starting as a result of the change in run level and by displaying the message:

   `INIT: New run level:` *n*

   where *n* is the new run-level number.

Changing Run Levels on Single-User Systems

1. Check the **/etc/inittab** file to confirm that the run level to which you are changing supports the processes that you are running. The getty process is particularly important, since it controls the terminal line access for the system console and other logins. Ensure that the getty process is enabled at all run levels.
2. Use the **smit telinit** fast path to access the **Set System Run Level** menu.
3. Enter the new system run level in the field:

   `System RUN LEVEL`
4. Confirm your choice to set the new run level.

   The system responds by telling you which processes are terminating or starting as a result of the change in run level and by displaying the message:

   `INIT: New run level: n`

   where *n* is the new run-level number.

Changing the /etc/inittab File

In some cases, you may need to add, change, list, and remove records in the **/etc/inittab** file. The **/etc/inittab** file defines which processes to start at each run level. When you run the **init** command, it reads the records in the **/etc/inittab** file. Each record defines a run level for a specific process and contains four parameters:

Identifier	identifies unique objects in one to fourteen characters
Run Level	defines the run levels in which the object can be processed in one to twenty characters
Action	defines what action the **init** command should take for this process The following actions can be specified: respawn, wait, once, boot, bootwait, powerfail, powerwait, off, hold, ondemand, initdefault, and sysinit.
Command	contains the shell command to be executed

The following commands are the only supported method for modifying the records in the **/etc/inittab** file:

chitab	changes records in the **/etc/inittab** file
lsitab	lists records in the **/etc/inittab** file
mkitab	adds records to the **/etc/inittab** file
rmitab	removes records from the **/etc/inittab** file

Adding Records

To add a record to the **/etc/inittab** file, type `mkitab` *`Identifier`*`:`*`Run Level`*`:`*`Action`*`:`*`Command`* and press the Enter key.

For example, to add a record for tty2, type:

```
mkitab tty002:2:respawn:/usr/sbin/getty /dev/tty2
```

In the above example:

`tty002`	identifies the object whose run level you are defining
`2`	specifies the run level at which this process should run
`respawn`	specifies the action that the **init** command should take for this process
`/usr/sbin/getty /dev/tty2`	specifies the shell command to be executed

Changing Records

To change a record to the **/etc/inittab** file, type `chitab` *`Identifier`*`:`*`Run Level`*`:`*`Action`*`:`*`Command`* and press the Enter key.

For example, to change a record for tty2 so that this process runs at run levels 2 and 3, type:

```
chitab tty002:23:respawn:/usr/sbin/getty /dev/tty2
```

In the above example:

`tty002` identifies the object whose run level you are defining

`23` specifies the run levels at which this process should run

`respawn` specifies the action that the **init** command should take for this process

`/usr/sbin/getty /dev/tty2`
specifies the shell command to be executed

Listing Records

To list records in the **/etc/inittab** file:

- To list all records, type `lsitab –a` and press the Enter key.
- To list a specific record, type `lsitab` *`Identifier`* and press the Enter key.

For example, to list the record for tty2, type `lsitab tty2` and press the Enter key.

Removing Records

To remove a record from the **/etc/inittab** file, type `rmitab` *`Identifier`* and press the Enter key. For example, to remove the record for tty2, type `rmitab tty2` and press the Enter key.

Shutting Down the System

From the user's viewpoint, stopping (shutting down) a system is simple. However, from the system's viewpoint, the shutdown process actually involves a series of events designed to preserve file integrity.

The **shutdown** command is the safest and most thorough way to halt the operating system. When you designate the appropriate flags, this command notifies users that the system is about to go down, kills all existing processes, unmounts file systems, and halts the system. See the following information for more about shutting down your system:

You can shut down the system without rebooting by using either SMIT or the **shutdown** command. The following sections describe how to shut down the system.

Understanding the Shutdown Process

There are several controlled situations in which you may want to shut down your system:

- after installing new software or changing the configuration for existing software
- when a hardware problem exists
- when the system is irrevocably hung
- when system performance is degraded
- when the file system is possibly corrupt

Prerequisites

You must be a root user to shut down the system.

Using SMIT

1. Use the **smit shutdown** fast path to access the **Stop the System** menu.

   On this menu you can:

   - send a message to other users notifying them that the system is shutting down
   - tell the system whether or not you want to receive status messages during shutdown
   - tell the system to restart after shutdown
   - enter the time that the shutdown is to occur

2. When you have finished making changes, confirm your choice. The system will shut down.

Using the shutdown Command

Type `shutdown` and press the Enter key. The system shuts down; the system waits one minute before stopping the user processes and the **init** process.

Shutting Down the System to Single-User Mode

In some cases, you may need to shut down the system and enter single-user mode to perform software maintenance and diagnostics.

Note: When an FX Series system is shut down to single–user mode, the change daemon is no longer running. Consequently, many fault tolerant operations are compromised, and the system cannot guarantee fault–tolerant operation.

1. Type `cd /` and press the Enter key to change to the root directory. You must be in the root directory to shut down the system to single-user mode to ensure that file systems are unmounted cleanly.
2. Type `shutdown -m` and press the Enter key. The system shuts down to single-user mode. A system prompt displays and you can perform maintenance activities.

Shutting Down the System in an Emergency

You can also use the **shutdown** command to shut down the system under emergency conditions. Use this procedure to stop the system quickly without notifying other users.

Type `shutdown -F` and press the Enter key. The **–F** flag instructs the **shutdown** command to bypass sending messages to other users and shuts down the system as quickly as possible.

Chapter 5. Managing Users and Groups

This chapter contains information on managing users and groups. Some topics discussed are:

- adding new users
- showing and changing user attributes
- managing authentication methods
- establishing default attributes for users
- adding new groups and showing and changing group attributes

Also included in this chapter is information on disk quotas and on setting up the environment for authenticating a user. For suggestions on how to improve the efficiency of managing users, see the information on CPU–efficient UserID administration in the *Performance Tuning Guide*.

Completing Basic User Tasks

This section describes basic system administration tasks for adding a user to the system and configuring the user's environment.

Adding a User

This procedure describes how to use SMIT to add a user to your network. The system supplies values for fields marked as optional. You must supply values for the required fields. If a field is not marked as required or optional, no value is required, and no value is supplied by the system.

1. Use the **smit mkuser** fast path to access the **Create User** menu.
2. On this menu, enter the information for the new user.
3. When you have finished making entries for the new user, confirm your choice to add the user to the system.

Setting Initial Login Shell for a User Environment

You can change the login shell for a particular user with the **chsh** command. You can run the **chsh** command either from the command line or from within SMIT.

Prerequisites

- To change the login shell for another user, you must have root user authority.
- The shell you specify must be defined in the usw stanza of the **/etc/security/login.cfg** file.

Using the Command Line

To set the initial login shell for a particular user, enter:

```
chsh UserName
```

The *UserName* parameter is not necessary if you are changing your own login shell. This command displays the list of defined shells and shows which is your current login shell. It also asks you if you want to change your login shell.

If you want to change your login shell, type y and press the Enter key. You are then asked to type in the full path of the shell you want. When you have typed the path, press Enter to make the change take effect.

Using SMIT

1. Use the **smit chuser** fast path to access the **Change User Attributes** menu.
2. Enter the name of the user whose login shell you want to change in the field:

   ```
   User NAME
   ```

   This will display the second **Change User Attributes** panel that contains the entry fields for user attributes.
3. Enter the full path of the shell you want to use in the field:

   ```
   Initial PROGRAM
   ```
4. Confirm your choice to make the changes on the system.

Setting Login Attributes for a User

As a system administrator, you can change the attributes that control how and when a user can log in to the system.

1. Use the **smit login_user** fast path to access the **Change/Show Login Attributes for a User** menu.
2. Enter the name of the user whose login attributes you want to change in the field:

   `User NAME`

   and press Enter.
3. In the displayed dialog fields, you can add or change a user's login attributes.
4. When you have finished making entries for the user, confirm your choice.

Changing/Showing Login Attributes for a Port

You can display or change the attributes that control a port when a user attempts to log in.

1. Use the **smit login_port** fast path to access the **Change/Show Login Attributes for a Port** dialog.
2. Enter the name of the port whose login attributes you want to change in the field:

   `Port NAME`

   and press Enter.
3. In the displayed dialog fields, add or change information for the port.
4. When you have finished, confirm your choice.

Assigning or Changing a User's Password

This procedure describes how to use SMIT to assign and change user passwords.

1. Use the **smit passwd** fast path to access the **Change a User Password** menu.
2. Enter the name of the user whose password you want to assign or change in the field:

   `User NAME`
3. When the system prompts you, enter the old password.
4. When the system prompts you, enter the new password. The system requests that you enter the new password twice.

Changing User Password Attributes

This procedure describes how to use SMIT to change user password attributes.

1. Use the **smit passwdattrs** fast path to access the **Change/Show Password Attributes for a User** menu.
2. Enter the name of the user whose password attributes you want to change in the field:

   `User NAME`

   and press Enter.
3. Add or change the user password information.

   Note: See "Recommended, Default, and Maximum Password Attribute Values" on page 10-5 for more information on field values.
4. When you have finished, confirm your choice.

Establishing Default Attributes for New Users

The default attributes for new users are stored in the **/usr/lib/security/mkuser.default** file. These values are read by the **mkuser** command unless you use the **mkuser** command to override them. To establish or change the default attributes, you must use the **chsec** command to edit this file.

Changing User Attributes

This procedure describes how to use SMIT to change a user's attributes.

1. Use the **smit chuser** fast path to access the **Change User Attributes** menu.
2. Enter the user name of the user whose attributes you want to change in the field:

   `User NAME`

   A second **Change User Attributes** menu displays.
3. Change the desired attributes.
4. When you have finished, confirm your choice.

Locking/Unlocking a User's Account

Using SMIT

1. Use the **smit chuser** fast path to access the **Change User Attributes** menu.
2. Enter the name of the user account you want to lock or unlock in the field:

   `User NAME`
3. If you want to lock the user's account, select **true** in the field:

   `Is this user ACCOUNT LOCKED?`

   If you want to unlock the user's account, select **false**.
4. When you have finished, confirm your choice.

Using the Command Line

To lock a user's account, enter:

```
chuser account_locked=true AccountName
```

To unlock a user's account, enter:

```
chuser account_locked=false AccountName
```

Managing Authentication Methods

The system administrator can specify which authentication paths a user is required to pass by specifying a token for the **SYSTEM** attribute in the **/etc/security/users** file or by using the SMIT fast path.

1. To use the SMIT fast path, enter:

   `smit mkuser` for a new user

   Or

   `smit chuser` for an existing user

2. Select **Login AUTHENTICATION GRAMMAR**. The authentication grammar allows the system administrator to specify combinations of the following methods, operators, and results:

```
"SYSTEM"      ::=  EXPRESSION
EXPRESSION    ::=  PRIMITIVE |
                   "("EXPRESSION")" |
                   EXPRESSION OPERATOR EXPRESSION
PRIMITIVE     ::=  METHOD |
                   METHOD "["RESULT"]"
RESULT        ::=  "SUCCESS" | "FAILURE" | "NOTFOUND" | "UNAVAIL" | "*"
OPERATOR      ::=  "AND" | "OR"
METHOD        ::=  "compat" | "files" | "NONE" | [a-z,A-Z,0-9]*
```

For normal system authentication, the method **compat** is specified.

```
SYSTEM = "compat"
```

Another method may be specified; for example, the Distributed Computing Environment (DCE).

```
SYSTEM = "DCE"
```

With the application of operators, the system administrator can provide alternative authentication paths. For example, the following allows the user to log in if DCE authentication is successful (`DCE` implies `DCE[SUCCESS]`):

```
SYSTEM = "DCE OR DCE[UNAVAIL] AND compat"
```

If DCE is unavailable and the user can pass normal system authentication, the user will be allowed on the system. You can use SMIT or the command line to lock or unlock a user's account.

Listing User Attributes

This procedure describes different ways to list attributes for specific users or all users.

Listing All Users

Use the **smit lsuser** fast path to display a scrollable list of all the users on the system, including user IDs and home directories.

Listing All Attributes for a Specific User with SMIT

1. Use the **smit chuser** fast path to access the **Change User Attributes** menu.
2. Enter the user name of the user whose attributes you want to list in the field:

   `User NAME`

   and press the Enter key. A scrollable list of all attributes for that user will be displayed.

Listing All Attributes for a Specific User from the Command Line

There are two styles for listing all of the attributes for a specific user:

- You can list each attribute in the form `Attribute=Value` separated by a blank space. This is the default style. For example, to list all attributes for the user `kirk`, enter:

  ```
  lsuser kirk
  ```

  A list similar to the following displays:

  ```
  kirk id=516 pgrp=system groups=system home=/u/kirk shell=/bin/ksh
  gecos=Kerry Kirk – Contractor login=true su=true rlogin=true
  daemon=true admin=false sugroups=ALL tpath=nosak ttys=ALL
  ```

- You can also list the information in stanza format. For example, to list all attributes for the user `kirk` in stanza format, enter:

  ```
  lsuser –f kirk
  ```

  A list similar to the following is displayed:

  ```
  kirk:
          id=516
          pgrp=system
          groups=system
          home=/u/kirk
          shell=/bin/ksh
          gecos=Kerry Kirk – Contractor
          login=true
          su=true
          rlogin=true
          daemon=true
          admin=false
          sugroups=ALL
          tpath=nosak
          ttys=ALL
  ```

Listing Specific Attributes for a Specific User

To list specific attributes for a specific user, enter:

```
lsuser –a Attributes User
```

For example, to list the ID and groups for user `kirk`, enter:

```
lsuser –a id groups kirk
```

A list similar to the following displays:

```
kirk id=516 groups=system
```

Listing Specific Attributes for All Users

There are two styles for listing specific attributes for all users:

- You can list attributes in the form `Attribute=Value` separated by a blank space. This is the default style. For example, to list the ID and groups for all of the users on the system, enter:

  ```
  lsuser –a id groups ALL
  ```

  A list similar to the following displays:

  ```
  root id=0
  groups=system,bin,sys,adm,uucp,mail,security,cron,printq,
  audit,uucp,mail,dba
  su id=0 groups=system,sys,adm,mail
  ```

- You can also list the information in stanza format. For example, to list the ID and groups for all of the users on the system in stanza format, enter:

  ```
  lsuser –a –f id groups ALL
  ```

  A list similar to the following displays:

  ```
  root:
          id=0
          groups=system,bin,sys,adm,uucp,mail,security,cron
  smitroot:
          id=0
          groups=system
  smittest:
          id=0
  su:
          id=0
          groups=system,sys,adm,mail
  ```

Removing a User

With SMIT, you can remove a user from your system. The system removes all attributes defined for the user. However, the system does not remove the user's home directory and files owned by the user.

You must remove information in other subsystems before removing a user, because the **cron** and **at** facilities both allow users to request programs to be run at a future date. Use the **crontab** command to remove a user's **cron** jobs. You can examine a user's **at** jobs with the **atq** command, then remove the jobs with the **atrm** command.

1. Use the **smit rmuser** fast path to access the **Remove a User From the System** menu.
2. Enter the user name of the user you want to remove from the system in the field:

   `User NAME`
3. If you want to remove the user password and other authentication information from the **/etc/security/passwd** file, select **yes** in the Remove AUTHENTICATION Information field.
4. If you want to leave the **/etc/security/passwd** file unchanged, select **no** in the field:

   `Remove AUTHENTICATION Information`
5. Confirm your choice to remove the user.

Turning Off and On Login Access for Users

System administrators can turn off a user's access to certain commands that are used to log into the system. Two kinds of login access can be controlled:

- local login access through the **login** command
- remote login access through the **rlogin**, **rsh** and **telnet** commands

Note: In general, this procedure is not suggested for systems using NIS. This procedure will not work at all for NIS clients and it will work on NIS master servers only for users logging into the master server.

Using SMIT

1. Use the **smit chuser** fast path to access the **Change User Attributes** menu.
2. Enter the name of the user you want to enable or prevent from logging in.
3. At the field:

   `LOGIN User?`

   set the value to **true** to enable a user to log in with the **login** command. Otherwise, set the value to **false** to prevent a user from logging in with the **login** command.
4. At the field:

   `User CAN RLOGIN?`

   set the value to **true** to enable a user to log in remotely with the **rlogin** or **telnet** command. Otherwise, set the value to **false** to prevent a user from logging in remotely with the **rlogin** or **telnet** command.
5. Confirm your choice to make the changes.

Using the Command Line

1. To enable a user to log in with the **login** command, enter:

   `chuser login=yes` *`UserName`*
2. To prevent a user from logging in with the **login** command, enter:

   `chuser login=no` *`UserName`*
3. To enable a user to log in remotely with the **rlogin** or **telnet** command, enter:

   `chuser rlogin=yes` *`UserName`*
4. To prevent a user from logging in remotely with the **rlogin** or **telnet** command, enter:

   `chuser rlogin=no` *`UserName`*

Adding a Group

The following procedure describes how to use SMIT to create a new group on your network.

1. Use the **smit mkgroup** fast path to access the **Add Group** menu.
2. On this menu, you can enter the information for the following fields:

   ```
   Group NAME
   Group ID
   ADMINISTRATIVE Group?
   USER List
   ADMINISTRATOR List
   ```

3. Confirm your choice to add the new group.

Changing Group Attributes

The following procedure describes how to use SMIT to change the following group attributes: administrative status of this group, group members, and users who can administer this group.

1. Use the **smit chgroup** fast path to access the **Change Group Attributes** menu.
2. Enter the group name of the group whose attributes you want to change in the field:

   `Group NAME`

   A second **Change Group Attributes** menu displays.
3. Change the information for the following fields:

   `Group ID`

   `ADMINISTRATIVE Group?`

   `USER List`

   `ADMINISTRATOR List`
4. Confirm your choice to change the group attributes.

Listing Groups

This procedure describes how to list all groups, specific attributes for all groups, all attributes for a specific group, and specific attributes for a specific group.

Listing All Groups

To list all of the groups on the system, use the **smit lsgroup** fast path.

The system displays each group, group ID, and all of the users in the group in a list similar to the following:

```
system  0       arne,pubs,ctw,geo,root,chucka,noer,su,dea,
backup,build,janice,denise
staff   1       john,ryan,flynn,daveb,jzitt,glover,maple,ken,jan,
books,smiller,frank,mary,nita,brian,marg,marj,jeanne,kaye,sarah,
leah,dewayne,solis,christin,joe,jim,dale,carl,dee,joy,jchen,
gordon,mbrady
bin     2       root,bin
sys     3       root,su,bin,sys
```

Listing Specific Attributes for All Groups

There are two styles for listing specific attributes for all groups:

- You can list attributes in the form `Attribute=Value` separated by a blank space. This is the default style. For example, to list the ID and users for all of the groups on the system, enter:

  ```
  lsgroup –a id users ALL | pg
  ```

  A list similar to the following displays:

  ```
  system id=0
  users=arne,pubs,ctw,geo,root,chucka,noer,su,dea,backup,build,janice,denise
  staff id=1
  users=john,ryan,flynn,daveb,jzitt,glover,maple,ken,jan,books,smiller,frank,mary,nita,brian,marg,marj,jeanne,kaye,sarah,leah,dewayne,solis,christin,joe,jim,dale,carl,dee,joy,jchen,gordon,mbrady
  ```

- You can also list the information in stanza format. For example, to list the ID and users for all of the groups on the system in stanza format, enter:

  ```
  lsgroup –a –f id users ALL | pg
  ```

A list similar to the following displays:

```
system:
        id=0
        users=arne,pubs,ctw,geo,root,chucka,noer,su,dea,backup,bu
ild,janice,denise

staff:
        id=1
        users=john,ryan,flynn,daveb,jzitt,glover,maple,ken,jan,bo
oks,smiller,frank,mary,nita,brian,marg,marj,jeanne,kaye,sarah,lea
h,dewayne,solis,christin,joe,jim,dale,carl,dee,joy,jchen,gordon,m
brady

bin:
        id=2
        users=root,bin

sys:
        id=3
        users=root,su,bin,sys
```

Listing All Attributes for a Specific Group

There are two styles for listing all of the attributes for a specific group:

- You can list each attribute in the form `Attribute=Value` separated by a blank space. This is the default style. For example, to list all attributes for the group system, enter:

  ```
  lsgroup system
  ```

  A list similar to the following displays:

  ```
  system id=0 users=arne,pubs,ctw,geo,root,chucka,noer,su,dea,
  backup,build,janice,denise
  ```

- You can also list the information in stanza format. For example, to list all attributes for the group `bin` in stanza format, enter:

  ```
  lsgroup –f system
  ```

  A list similar to the following displays:

  ```
  system:
          id=0
          users=arne,pubs,ctw,geo,root,chucka,noer,su,dea,
  backup,build,janice,denise
  ```

Listing Specific Attributes for a Specific Group

To list specific attributes for a specific group, enter:

```
lsgroup –a Attributes Group
```

For example, to list the ID and users for group `bin`, enter:

```
lsgroup –a id users bin
```

A list similar to the following displays:

```
bin id=2 users=root,bin
```

Removing a Group

This procedure describes how to remove a group and all of its attributes from your network. However, this procedure does not remove all of the users in the group from the system. See “Removing a User” on page 5-8.

If the group you want to remove is the primary group for any user, you must reassign the user to another primary group; then the user’s original primary group can be removed. See “Changing User Attributes” on page 5-4.

1. Use the **smit rmgroup** fast path to access the **Remove a Group From the System** menu.
2. Enter the name of the group you want to remove in the field:

   `Group NAME`

   and confirm your choice.
3. The message:

   `Are You Sure?`

   prompts you to make sure you want to remove the group. Confirm your choice to remove the group.

Disk Quota System Overview

The disk quota system allows system administrators to control the number of files and data blocks that can be allocated to users or groups. The following sections provide further information about the disk quota system, its implementation, and use.

Understanding the Disk Quota System

The disk quota system, based on the Berkeley Disk Quota System, provides an effective way to control the use of disk space. The quota system can be defined for individual users or groups, and is maintained for each journaled file system.

The disk quota system establishes limits based on three parameters that can be changed with the **edquota** command:

- user's or group's soft limits
- user's or group's hard limits
- quota grace period

The *soft limit* defines the number of 1KB disk blocks or files below which the user should remain. The *hard limit* defines the maximum amount of disk blocks or files the user can accumulate under the established disk quotas. The *quota grace period* allows the user to exceed the soft limit for a short period of time (the default value is one week). If the user fails to reduce usage below the soft limit during the specified time, the system will interpret the soft limit as the maximum allocation allowed, and no further storage will be allocated to the user. The user can reset this condition by removing enough files to reduce usage below the soft limit.

The disk quota system tracks user and group quotas in the **quota.user** and **quota.group** files that reside in the root directories of file systems enabled with quotas. These files are created with the **quotacheck** and **edquota** commands and are readable with the quota commands.

Recovering from Over-Quota Conditions

There are several methods available to reduce file system usage when you have exceeded quota limits:

- Abort the current process that caused the file system to reach its limit, remove surplus files to bring the limit below quota, and retry the failed program.
- If you are running an editor such as vi, use the shell escape sequence to check your file space, remove surplus files, and return without losing your edited file. Alternatively, if you are using the C or Korn shells, you can suspend the editor with the Ctrl-Z key sequence, issue the file system commands, and then return with the **fg** (foreground) command.
- Temporarily write the file to a file system where quota limits have not been exceeded, delete surplus files, and then return the file to the correct file system.

Implementing the Disk Quota System

You should consider implementing the disk quota system under the following conditions:

- Your system has limited disk space.
- You require more file system security.
- Your disk-usage levels are large, such as at many universities.

If these conditions do not apply to your environment, you may not want to create disk-usage limits by implementing the disk quota system.

Typically, only those file systems that contain user home directories and files require disk quotas. The disk quota system works only with the journaled file system.

Note: It is recommended that disk quotas not be established for the **/tmp** file system.

Setting Up the Disk Quota System

Prerequisites

You must have root user authority.

Procedure

1. Determine which file systems require quotas. Normally, you need to establish quotas only on those file systems that house users' home directories or other user files. The disk quota system can be used only with the journaled file system.

   Note: Because many editors and system utilities create temporary files in the **/tmp** file system, it should be free of quotas.

2. Use the **chfs** command to include the **userquota** and **groupquota** quota configuration attributes in the **/etc/filesystems** file. The following sample **chfs** command enables user quotas on the `/home` file system:

   ```
   chfs -a "quota = userquota" /home
   ```

   To enable both user and group quotas on the `/home` file system, enter:

   ```
   chfs -a "quota = userquota,groupquota" /home
   ```

   The corresponding entry in the **/etc/filesystems** would appear as follows:

   ```
   /home:
   dev         = /dev/hd1
   vfs         = jfs
   log         = /dev/hd8
   mount       = true
   check       = true
   quota       = userquota,groupquota
   options     = rw
   ```

3. Optionally, specify alternate disk quota file names. The file names **quota.user** and **quota.group** are the default names located at the root directories of the file systems enabled with quotas. You can specify alternate names or directories for these quota files with the **userquota** and **groupquota** attributes in the **/etc/filesystems** file.

   The following sample **chfs** command establishes user and group quotas for the `/home` file system, and names the quota files `myquota.user` and `myquota.group`:

   ```
   chfs -a "userquota = /home/myquota.user" -a "groupquota = /home
          /myquota.group" /home
   ```

The corresponding entry in **/etc/filesystems** would appear as follows:

```
/home:
dev         = /dev/hd1
vfs         = jfs
log         = /dev/hd8
mount       = true
check       = true
quota       = userquota,groupquota
userquota   = /home/myquota.user
groupquota  = /home/myquota.group
options     = rw
```

4. Mount the specified file systems, if not previously mounted.

5. Set the desired quota limits for each user or group. Use the **edquota** command to create each user or group's soft and hard limits for allowable disk space and maximum number of files.

   The following sample entry shows quota limits for user `davec`:

   ```
   Quotas for user davec:
   /home: blocks in use: 30, limits (soft = 100, hard = 150)
          inodes in use: 73, limits (soft = 200, hard = 250)
   ```

   This user has used 30KB of the maximum 100KB of disk space. Of the maximum 200 files, `davec` has created 73. This user has buffers of 50KB of disk space and 50 files that can be allocated to temporary storage.

   When establishing disk quotas for multiple users, use the **–p** flag with the **edquota** command to duplicate a user's quotas for another user.

   To duplicate the quotas established for user `davec` for user `nanc`, enter:

   ```
   edquota –p davec nanc
   ```

6. Enable the quota system with the **quotaon** command. The **quotaon** command enables quotas for a specified file system, or for all file systems with quotas (as indicated in the **/etc/filesystems** file) when used with the **–a** flag.

7. Use the **quotacheck** command to check the consistency of the quota files against actual disk usage.

   Note: It is recommended that you do this each time you first enable quotas on a file system and after you reboot the system.

   To enable this check and to turn on quotas during system startup, add the following lines at the end of the **/etc/rc** file:

   ```
   echo ″ Enabling filesystem quotas ″
   /usr/sbin/quotacheck –a
   /usr/sbin/quotaon –a
   ```

Chapter 6. System Environment

The system environment is primarily the set of variables that define or control certain aspects of process execution. They are set or reset each time a shell is started. From the system-management point of view, it is important to ensure the user is set up with the correct values at log in. Most of these variables are set during system initialization. Their definitions are read from the **/etc/profile** file or set by default.

This chapter includes the following information:

- a description of the system **/etc/profile** file and the user's **.profile** file
- procedures on setting the system time, date, and message of the day
- a list of time date manipulation services

Profiles Overview

The shell uses two types of profile files when you log in to the operating system. It evaluates the commands contained in the files and then executes the commands to set up your system environment. The files have similar functions except that the **/etc/profile** file controls profile variables for all users on a system whereas the **.profile** file allows you to customize your own environment.

/etc/profile File

The first file that the operating system uses at login time is the **/etc/profile** file. This file controls system-wide default variables such as:

- export variables
- file creation mask (umask)
- terminal types
- mail messages to indicate when new mail has arrived

The system administrator configures the **profile** file for all users on the system. Only the system administrator can change this file.

.profile File

The second file that the operating system uses at login time is the **.profile** file. The **.profile** file is present in your home (**$HOME**) directory and enables you to customize your individual working environment. The **.profile** file also overrides commands and variables set in the **/etc/profile** file. Since the **.profile** file is hidden, use the **li –a** command to list it. Use the **.profile** file to control the following defaults:

- shells to open
- prompt appearance
- environment variables (for example, search path variables)
- keyboard sound

The following example shows a typical **.profile** file:

```
PATH=/usr/bin:/etc:/home/bin1:/usr/lpp/tps4.0/user:/home/gsc/bin::
epath=/home/gsc/e3:
export PATH epath
csh
```

This example has defined two paths (`PATH` and `epath`), exported them, and opened a C shell (`csh`).

You can also use the **.profile** file (or if it is not present, the **profile** file) to determine login shell variables. You can also customize other shell environments. For example, use the **.chsrc** and **.kshrc** files to tailor a C shell and a Korn shell, respectively, when each type shell is started.

Changing the System Date, Time, and Message of the Day

Changing the System Date and Time

The system date and time is set with the **date** command.

To change the system date or time, first log in as root and take the machine down to single–user mode. Then, execute the **date** command.

The **date** command allows the date or time to be specified in one of several different formats. One form of the **date** command is:

```
date mmddHHMM.SSyy
```

where *mm* is the month, *dd* is the day of the month, *HH* is the hour, *MM* is the minutes, *SS* is the seconds, and *yy* is the last two digits of the year.

Changing the Message of the Day

The message of the day is displayed every time a user logs in to the system. It is a convenient way to communicate information to all users, such as installed software version numbers or current system news. The message of the day is contained in the **/etc/motd** file. To change the message of the day, simply edit that file.

List of Time Data Manipulation Services

The time functions access and reformat the current system date and time. You do not need to specify any special flag to the compiler to use the time functions.

Include the header file for these functions in the program. To include a header file, use the following statement:

```
#include <time.h>
```

The time services are the following:

adjtime	corrects the time to allow synchronization of the system clock
ctime, **localtime**, **gmtime**, **mktime, difftime, asctime, tzset**	converts date and time to string representation
getinterval, **incinterval**, **absinterval, resinc, resabs, alarm, ualarm, getitimer, setitimer**	manipulates the expiration time of interval timers
gettimer, **settimer**, **restimer, stime, time**	gets or sets the current value for the specified system–wide timer
gettimerid	allocates a per-process interval timer
gettimeofday, **settimeofday, ftime**	gets and sets date and time
nsleep, **usleep**, **sleep**	suspends a current process from execution
reltimerid	releases a previously allocated interval timer

Chapter 7. Process Management

Process is the entity that the operating system uses to control the use of system resources. AIX Version 4 introduces the use of *threads* to control processor-time consumption, but most of the system management tools still require the administrator to refer to the process in which a thread is running, rather than to the thread itself.

See the *System User's Guide: Operating System and Devices* for basic information on managing your own processes; for example, restarting or stopping a process that you started or scheduling a process for a later time. The *System User's Guide* also defines terms that describe processes, such as daemons and zombies.

This chapter describes processes and the tools that the operating system provides to manage processes from the perspective of the system administrator rather than the general user.

Process Tools

AIX contains tools to:

- observe the creation, cancellation, identity, and resource consumption of processes
  - **ps** is used to report process IDs, users, CPU-time consumption, and other attributes.
  - **who –u** reports the shell process ID of logged-on users.
  - **svmon** is used to report process real-memory consumption. (See *Performance Toolbox 1.2 and 2.1 for AIX: Guide and Reference* for information on the **svmon** command.)
  - **acct** mechanism writes records at process termination summarizing the process's resource use. (See how to set up an accounting system in "Accounting Overview" on page 9-2.)
- control the priority level at which a process contends for the CPU
  - **nice** causes a command to be run with a specified process priority. (See the *System User's Guide: Operating System and Devices.)*
  - **renice** changes the priority of a given process.
- terminate processes that are out of control
  - **kill** sends a termination signal to one or more processes.
- tune the operating system's process-management mechanisms
  - **schedtune** permits changes to the process scheduler parameters. (See the *Performance Tuning Guide* for information on the **schedtune** command.)

Process Monitoring

The **ps** command is the primary tool for observing the processes in the system. Most of the flags of the **ps** command fall into one of two categories:

- flags that specify which types of processes to include in the output
- flags that specify which attributes of those processes are to be displayed

The most widely useful variants of **ps** for system-management purposes are:

ps –ef Lists all non–kernel processes, with the userid, process ID, recent CPU usage, total CPU usage, and the command that started the process (including its parameters).

ps –fu *UserID* Lists all of the processes owned by *UserID*, with the process ID, recent CPU usage, total CPU usage, and the command that started the process (including its parameters).

To identify the current heaviest users of CPU time, you could enter:

```
ps –ef | egrep –v "STIME|$LOGNAME" | sort +3 –r | head –n 15
```

This will list, in descending order, the 15 most CPU-intensive processes other than those owned by you.

For more specialized uses, the following two tables are intended to simplify the task of choosing **ps** flags by summarizing the effects of the flags.

	Process-Specifying Flags:												
					–G				**–U**				
Processes Listed are:	**–A**	**–a**	**–d**	**–e**	**–g**	**–k**	**–p**	**–t**	**–u**	**a**	**g**	**t**	**x**
All processes	Y	-	-	-	-	-	-	-	-	-	Y	-	-
Not process group leaders and not associated with a terminal	-	Y	-	-	-	-	-	-	-	-	-	-	-
Not process group leaders	-	-	Y	-	-	-	-	-	-	-	-	-	-
Not kernel processes	-	-	-	Y	-	-	-	-	-	-	-	-	-
Members of specified process groups	-	-	-	-	Y	-	-	-	-	-	-	-	-
Kernel processes	-	-	-	-	-	Y	-	-	-	-	-	-	-
Those specified in process number list	-	-	-	-	-	-	Y	-	-	-	-	-	-
Those associated with tty(s) in the list	-	-	-	-	-	-	- (n ttys)	Y	-	-	-	Y (one tty)	-
Specified user processes	-	-	-	-	-	-	-	-	Y	-	-	-	-
Processes with terminals	-	-	-	-	-	-	-	-	-	Y	-	-	-
Not associated with a tty	-	-	-	-	-	-	-	-	-	-	-	-	Y

Column-Selecting Flags:

Column:	Default1	–f	–l	–U –u	Default2	e	l	s	u	v
PID	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
TTY	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
TIME	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
CMD	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
USER	-	Y	-	-	-	-	-	-	Y	-
UID	-	-	Y	Y	-	-	Y	-	-	-
PPID	-	Y	Y	-	-	-	Y	-	-	-
C	-	Y	Y	-	-	-	Y	-	-	-
STIME	-	Y	-	-	-	-	-	-	Y	-
F	-	-	Y	-	-	-	-	-	-	-
S/STAT	-	-	Y	-	Y	Y	Y	Y	Y	Y
PRI	-	-	Y	-	-	-	Y	-	-	-
NI/NICE	-	-	Y	-	-	-	Y	-	-	-
ADDR	-	-	Y	-	-	-	Y	-	-	-
SZ/SIZE	-	-	Y	-	-	-	Y	-	Y	Y
WCHAN	-	-	Y	-	-	-	Y	-	-	-
RSS	-	-	-	-	-	-	Y	-	Y	Y
SSIZ	-	-	-	-	-	-	-	Y	-	-
%CPU	-	-	-	-	-	-	-	-	Y	Y
%MEM	-	-	-	-	-	-	-	-	Y	Y
PGIN	-	-	-	-	-	-	-	-	-	Y
LIM	-	-	-	-	-	-	-	-	-	Y
TSIZ	-	-	-	-	-	-	-	-	-	Y
TRS	-	-	-	-	-	-	-	-	-	Y
environment (following the command)	-	-	-	-	-	Y	-	-	-	-

If **ps** is given with no flags or with a process-specifying flag that begins with a minus sign, the columns displayed are those shown for Default1. If the command is given with a process-specifying flag that does not begin with minus, Default2 columns are displayed. The **–u** or **–U** flag is both a process-specifying and column-selecting flag.

The following are brief descriptions of the contents of the columns:

PID	process ID
TTY	terminal or pseudo-terminal associated with the process
TIME	cumulative CPU time consumed, in minutes and seconds
CMD	command the process is running
USER	login name of the user to whom the process belongs

UID	numeric user ID of the user to whom the process belongs
PPID	ID of this process's parent process
C	recently used CPU time
STIME	time the process started, if today; otherwise, the date the process started
F	eight-character hexadecimal value describing the flags associated with the process (see the detailed description of the **ps** command)
S/STAT	status of the process (see the detailed description of the **ps** command)
PRI	current priority value of the process
NI/NICE	nice value for the process
ADDR	segment number of the process stack
SZ/SIZE	number of working-segment pages that have been touched times 4
WCHAN	event on which the process is waiting
RSS	sum of the numbers of working-segment and code-segment pages in memory times 4
SSIZ	size of the kernel stack
%CPU	percentage of time since the process started that it was using the CPU
%MEM	nominally, the percentage of real memory being used by the process (this measure does not correlate with any other memory statistics)
PGIN	number of page ins caused by page faults (Since all AIX I/O is classified as page faults, this is basically a measure of I/O volume.)
LIM	always "xx"
TSIZ	size of the text section of the executable file
TRS	number of code-segment pages times 4
Environment	value of all the environment variables for the process

Process Alteration or Termination

Process-Priority Alteration

For a detailed discussion of process-priority alteration, see “Controlling Contention for the CPU” in the *Performance Tuning Guide*. Basically, if you have identified a process that is using too much CPU time, you can reduce its effective priority by increasing its nice value with **renice**. For example:

renice +5 *ProcID*

The nice value of the *ProcID*s would increase process from the normal 20 of a foreground process to 25. To reset process *ProcID*s nice value to 20, you would have to be root and enter:

renice –5 *ProcID*

Process Termination

Use the **kill** command to end a process. The **kill** command sends a signal to the designated process. Depending on the type of signal and the nature of the program that is running in the process, the process may end or may keep running. The signals you would send are:

SIGTERM	(signal 15) is a request to the program to terminate. If the program has a signal handler for SIGTERM that does not actually terminate the application, this **kill** may have no effect. This is the default signal sent by **kill**.
SIGKILL	(signal 9) is a directive to kill the process immediately. This signal cannot be caught or ignored.

Normally, it is desirable to issue SIGTERM rather than SIGKILL. If the program has a handler for SIGTERM, it can clean up and terminate in an orderly fashion. You would issue:

kill –term *ProcessID*

(The **–term** could be omitted.) If the process does not respond to the SIGTERM, enter:

kill –kill *ProcessID*

Binding or Unbinding a Process

On multiprocessor systems, you can bind a process to a processor or unbind a previously bound process by using the **bindprocessor** command. Binding a process forces it to always run on a specific processor. You can run the **bindprocessor** command one of two ways:

- from the command line
- with SMIT

Note: While binding a process to a processor may lead to improved performance for the bound process (by decreasing hardware-cache misses), overuse of this facility could cause individual processors to become overloaded while other processors are underused. The resulting bottlenecks could reduce overall throughput and performance. During normal operations, it is better to let the operating system assign processes to processors automatically, distributing system load across all processors. Bind only those processes that you know will benefit from being run on a single processor.

Prerequisite

You must have root user authority to bind or unbind a process you do not own.

Binding a Process

Using the Command Line

Bind a processor by entering:

```
bindprocessor ProcessID ProcessorNum
```

where *ProcessID* is the process identifier of the process, and *ProcessorNum* specifies the logical number of the processor to which the process is to be bound. If *ProcessorNum* is omitted, a processor is selected at random.

To see which processors are available (possible *ProcessorNum* values*)*, enter:

```
bindprocessor –q
```

Using SMIT

1. Use the **smit bindproc** fast path to access the **Bind a Process to a Processor** menu.
2. Enter the process ID of the process you want to bind in the field:

   `PROCESS ID`
3. Enter the processor ID of the processor to which the process is to be bound in the field:

   `PROCESSOR ID`
4. Confirm your choice to bind the process to the processor.

Unbinding a Process

Using the Command Line

Unbind a bound process by entering:

```
bindprocessor –u ProcessID
```

Using SMIT

1. Use the **smit ubindproc** fast path to access the **Unbind a Process** menu.
2. Enter the process ID of the process you want to unbind in the field:

   `PROCESS ID`
3. Confirm your choice to unbind the process.

Chapter 8. System Resource Controller and Subsystems

This chapter contains information about the System Resource Controller (SRC) and the various subsystems it controls. Topics include:

- an overview of the system resource controller
- starting the system resource controller
- starting and stopping a subsystem, subsystem group, or subserver
- listing subsystems and displaying the status of a subsystem
- displaying the status of a subsystem
- refreshing a subsystem or subsystem group
- turning on and off tracing for a subsystem, subsystem group, or subserver

System Resource Controller Overview

The System Resource Controller (SRC) provides a set of commands and subroutines to make it easier for the system manager and programmer to create and control subsystems. A *subsystem* is any program or process or set of programs or processes that is usually capable of operating independently or with a controlling system. A subsystem is designed as a unit to provide a designated function.

The SRC was designed to minimize the need for operator intervention. It provides a mechanism to control subsystem processes using a common command line and the C interface. This mechanism includes the following:

- consistent user interface for start, stop, and status inquiries
- logging of the abnormal termination of subsystems
- notification program called at the abnormal system termination of related processes
- tracing of a subsystem, a group of subsystems, or a subserver
- support for control of operations on a remote system
- refreshing of a subsystem (such as after a configuration data change)

The SRC is useful if you want a common way to start, stop, and collect status information on processes.

Subsystem Components

A subsystem can have one or more of the following properties:

- is known to the system by name
- requires a more complex execution environment than a subroutine or nonprivileged program
- includes application programs and libraries as well as subsystem code
- controls resources that can be started and stopped by name
- requires notification if a related process is unsuccessful to perform cleanup or to recover resources
- requires more operational control than a simple daemon process
- needs to be controlled by a remote operator
- implements subservers to manage specific resources
- does not put itself in the background

A few subsystem examples are ypserv, ntsd, qdaemon, inetd, syslogd, and sendmail.

Note: Refer to each specific subsystem for details of its SRC capabilities.

Use the **lssrc –a** command to list active and inactive subsystems on your system.

Subsystem Group

A *subsystem group* is a group of any specified subsystems. Grouping subsystems together allows the control of several subsystems at one time. A few subsystem group examples are TCP/IP, SNA Services, Network Information System (NIS), and Network File Systems (NFS).

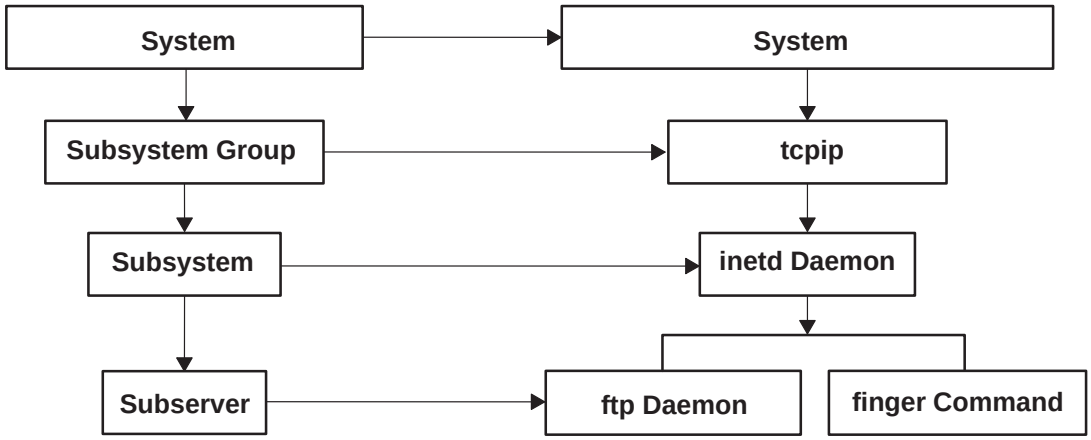
Subserver

A *subserver* is a program or process that belongs to a subsystem. A subsystem can have multiple subservers and is responsible for starting, stopping, and providing status of subservers. Subservers can be defined only for a subsystem with a communication type of IPC message queues and sockets. Subsystems using signal communications do not support subservers.

Subservers are started when their parent subsystems are started. If you try to start a subserver and its parent subsystem is not active, the **startsrc** command starts the subsystem as well.

SRC Hierarchy

The System Resource Controller has a hierarchical structure. The hierarchy begins with the operating system followed by a subsystem group (such as **tcpip**), which contains a subsystem (such as the **inetd** daemon), which in turn can own several subservers (such as the **ftp** daemon and the **finger** command).



SRC Hierarchical Structure

List of SRC Administration Commands

srcmstr daemon	starts the System Resource Controller
startsrc command	starts a subsystem, subsystem group, or subserver
stopsrc command	stops a subsystem, subsystem group, or subserver
refresh command	refreshes a subsystem
traceson command	turns on tracing of a subsystem, a group of subsystems, or a subserver
tracesoff command	turns off tracing of a subsystem, a group of subsystems, or a subserver
lssrc command	gets status on a subsystem

Starting the System Resource Controller

The System Resource Controller (SRC) is started during system initialization with a record for the **/etc/srcmstr** daemon in the **/etc/inittab** file. The default **/etc/inittab** file already contains such a record, so this procedure may be unnecessary. You can also start the SRC from the command line, a profile, or a shell script, but there are several reasons for starting it during initialization:

- Starting the SRC from the **/etc/inittab** file allows the **init** command to restart the SRC should it stop for any reason.
- The SRC is designed to simplify and reduce the amount of operator intervention required to control subsystems. Starting the SRC from any source other than the **/etc/inittab** file would be counterproductive to that goal.
- The default **/etc/inittab** file contains a record for starting the print scheduling subsystem (**qdaemon**) with the **startsrc** command. Typical installations have other subsystems started with **startsrc** commands in the **/etc/inittab** file as well. Since the **srcmstr** command requires the SRC to be running, removing the **srcmstr** daemon from the **/etc/inittab** file would cause these **startsrc** commands to fail.

Prerequisites

- Reading and writing the **/etc/inittab** file requires root user authority.
- The **mkitab** command requires root user authority.
- The **srcmstr** daemon record must exist in the **/etc/inittab** file.

Procedure

Note: This procedure is necessary only if the **/etc/inittab** file does not already contain a record for the **srcmstr** daemon.

1. Make a record for the **srcmstr** daemon in the **/etc/inittab** file using the **mkitab** command. For example, to make a record identical to the one that appears in the default **/etc/inittab** file, enter:

   ```
   mkitab –i fbcheck srcmstr:2:respawn:/etc/srcmstr
   ```

   The **–i fbcheck** flag ensures that the record will be inserted before all subsystems records.

2. Tell the **init** command to reprocess the **/etc/inittab** file by entering:

   ```
   telinit q
   ```

   When **init** revisits the **/etc/inittab** file, it will process the newly entered record for the **srcmstr** daemon and start the SRC.

Starting a Subsystem, Subsystem Group, or Subserver

Use the **startsrc** command to start a System Resource Controller (SRC) resource such as a subsystem, a group of subsystems, or a subserver. You can run the **startsrc** command in one of three ways:

- from the **/etc/inittab** file so the resource is started during system initialization
- from the command line
- with SMIT

When you start a subsystem group, all of its subsystems are also started. When you start a subsystem, all of its subservers are also started. When you start a subserver, its parent subsystem is also started if it is not already running.

Prerequisites

- The SRC must be running. The SRC is normally started during system initialization. The default **/etc/inittab** file, which determines what processes are started during initialization, contains a record for the **srcmstr** daemon (the SRC). To see if the SRC is running, enter **ps –A** and look for a process named **srcmstr**.
- The user or process starting an SRC resource must have root user authority. The process that initializes the system (**init** command) has root user authority.

Using the /etc/inittab File

1. Add a **startsrc** record to the **/etc/inittab** file using the **mkitab** command. For example, to add a record for starting a subsystem named `testsub`, enter:

   ```
   mkitab –i srcmstr "src001:2:once:/bin/startsrc –s testsub"
   ```

   The `–i srcmstr` flag in this example tells the **mkitab** command to place this record after the record identified as `srcmstr`, which is the default identifier of the record containing the **srcmstr** daemon. This placement is necessary so that the SRC will be started before the **startsrc** command is issued. The string within double quotation marks (" ") specifies, in order, the unique identifier of the record (`src001`), the run level or run levels at which the record should be processed (run level 2), the respawning instructions (once), and the full path name, flags, and parameters of the **startsrc** command. See the **mkitab** command for a complete description of the syntax.

2. Tell the **init** command to reprocess the **/etc/inittab** file by entering:

   ```
   telinit q
   ```

   When **init** revisits the **/etc/inittab** file, it will process the newly entered record for the **startsrc** command and start the subsystem.

Using the Command Line

Start the subsystem by entering:

```
/bin/startsrc –s SubsystemName
```

The syntax of the **startsrc** command includes flags and parameters for specifying such things as the subsystem environment, arguments to be passed to the subsystem, the host name for a remote subsystem, a subserver type or name, and so on. See the **startsrc** command for the exact syntax.

Using SMIT

1. Use the **smit startssys** fast path to access the **Start a Subsystem** menu.
2. Enter the name of the subsystem you want to start in the field:

   `Subsystem Name`
3. Confirm your choice to start the subsystem.

Stopping a Subsystem, Subsystem Group, or Subserver

Use the **stopsrc** command to stop a System Resource Controller (SRC) resource such as a subsystem, a group of subsystems, or a subserver. You can run the **stopsrc** command from the command line or with SMIT.

Prerequisites

- The SRC must be running. See “Starting the System Resource Controller” on page 8-4 for details.
- The user or process stopping an SRC resource must have root user authority.

Using the Command Line

Stop the subsystem by entering:

```
/bin/stopsrc –s SubsystemName
```

The syntax of the **stopsrc** command includes flags and parameters for specifying such things as the host name for a remote subsystem, a subserver type or name, and so on. See the *Commands Reference* for the exact syntax.

Using SMIT

1. Use the **smit stopssys** fast path to access the Stop a Subsystem menu.
2. Select either the **Stop a Single Subsystem** option or the **Stop All Subsystems** option.
3. If you selected the **Stop a Single Subsystem** option:
   a. Enter the process ID of the subsystem you want to stop in the field:

      ```
      Subsystem PROCESS ID
      ```

   b. Specify the type of stop you desire for the subsystem in the field:

      ```
      Stop TYPE
      ```

   c. Confirm your choice to stop the subsystem.
4. If you selected the **Stop All Subsystems** option:
   a. Specify the type of stop you desire for the subsystem in the field:

      ```
      Stop TYPE
      ```

   b. Confirm your choice to stop the subsystem.

Listing Subsystems

To list all of the subsystems on a particular host, use the **lssrc** command. The resulting list includes the short status report for each subsystem. You can issue the **lssrc** command from the command line or with SMIT.

Prerequisites

To list the subsystems on a remote host, you must have root authority.

Using the Command Line

List all of the subsystems on a particular host by entering:

```
lssrc –h HostName –a
```

To list all of the subsystems on the local system, omit the **–h** flag and host name.

Using SMIT

Use the **smit lsssys** fast path to display a scrollable list of subsystems.

Note: SMIT cannot list the subsystems on a remote host.

Displaying the Status of a Subsystem

Use the **lssrc** command to display the status of a System Resource Controller (SRC) resource such as a subsystem, a group of subsystems, or a subserver. You can run the **lssrc** command from the command line or with SMIT.

All subsystems can return a short status report that includes which group the subsystem belongs to, whether the subsystem is active, and what its process ID (PID) is. If a subsystem does not use the signals communication method, it can be programmed to return a long status report containing additional status information.

The **lssrc** command provides flags and parameters for specifying the subsystem by name or PID, for listing all subsystems, for requesting a short or long status report, and for requesting the status of SRC resources either locally or on remote hosts.

Using the Command Line

To display the short status of a subsystem from the command line, enter:

```
lssrc –s SubsystemName
```

If the subsystem is defined, the **lssrc** command returns the status of the subsystem. Otherwise, **lssrc** returns an error message saying the subsystem could not be found.

See the **lssrc** command for examples of how to use the other available flags and parameters.

Using SMIT

1. Use the **smit qssys** fast path to access the **Query a Subsystem** menu.
2. Enter the PID of the subsystem you want to display in the field:

   ```
   Subsystem PROCESS ID
   ```

3. Confirm your choice to display the status of the subsystem.

Note: SMIT attempts to display the long status of a subsystem, so you can use SMIT to display the status of a subsystem only if it does not use the signals communication method and has been programmed to return a long status.

Refreshing a Subsystem or Subsystem Group

Use the **refresh** command to tell a System Resource Controller (SRC) resource such as a subsystem or a group of subsystems to refresh itself. You can run the **refresh** command from the command line or with SMIT.

Prerequisites

- The SRC must be running. See “Starting the System Resource Controller” on page 8-4 for details.
- The resource you want to refresh must not use the signals communications method.
- The resource you want to refresh must be programmed to respond to the refresh request.

Using the Command Line

Refresh a subsystem by entering:

```
refresh –s Subsystem
```

You can specify the subsystem by its process ID (PID) by entering:

```
refresh –p SubsystemPID
```

The syntax of the **refresh** command also includes flags for specifying the host name for a remote subsystem and a subsystem group. See the *Commands Reference* for the exact syntax.

Using SMIT

1. Use the **smit refresh** fast path to access the **Refresh a Subsystem** menu.
2. Enter the PID of the subsystem you want to refresh in the field:

   ```
   Subsystem PROCESS ID
   ```
3. Confirm your choice to refresh the subsystem.

Turning On Subsystem, Subsystem Group, or Subserver Tracing

Use the **traceson** command to turn on tracing of a System Resource Controller (SRC) resource such as a subsystem, a group of subsystems, or a subserver. You can run the **traceson** command from the command line or with SMIT.

Prerequisites

- The SRC must be running. See “Starting the System Resource Controller” on page 8-4 for details.
- The resource you want to trace must not use the signals communications method.
- The resource you want to trace must be programmed to respond to the trace request.

Using the Command Line

Turn on subsystem tracing by entering:

```
traceson –h Host –s Subsystem
```

The syntax of the **traceson** command includes flags and parameters for specifying such parameters as the host name for a remote subsystem, and a subserver type or name. See the description of the **traceson** command for more information.

Using SMIT

1. Use the **smit tracessyson** fast path to access the **Start Trace** menu.
2. Enter the process ID of the subsystem you want to trace in the field:

   `Subsystem PROCESS ID`
3. Select the type of trace you want for the subsystem in the field:

   `Trace TYPE`
4. Confirm your choice to start the subsystem trace.

Turning Off Subsystem, Subsystem Group, or Subserver Tracing

Use the **tracesoff** command to turn off tracing of a System Resource Controller (SRC) resource such as a subsystem, a group of subsystems, or a subserver. You can run the **tracesoff** command from the command line or with SMIT.

Prerequisites

- The SRC must be running. See “Starting the System Resource Controller” on page 8-4 for details.
- The resource you are tracing must not use the signals communications method.
- The resource you are tracing must be programmed to respond to the **tracesoff** request.

Using the Command Line

Turn off subsystem tracing by entering:

```
tracesoff –h Host –s Subsystem
```

The syntax of the **tracesoff** command includes flags and parameters for specifying parameters such as the host name for a remote subsystem and a subserver type or name. See the **tracesoff** command description for more information.

Using SMIT

1. Use the **smit tracessysoff** accesses the **Stop Trace** menu.
2. Enter the process ID of the subsystem for which you want to turn off the trace in the field:

   ```
   Subsystem PROCESS ID
   ```

3. Confirm your choice to turn off the trace.

Chapter 9. System Accounting

The system accounting utility allows you to collect and report on individual and group use of various system resources. This chapter includes the following topics:

- a description of the accounting system, including a list of accounting commands and files
- setting up an accounting system
- generating reports on system activity and summarizing accounting records
- starting and restarting the **runacct** command
- showing accounting information, such as system activity, process time, CPU and connect time usage, and disk space and printer usage
- fixing errors

Accounting Overview

This accounting information can be used to bill users for the system resources they utilize, and to monitor selected aspects of the system's operation. To assist with billing, the accounting system provides the resource-usage totals defined by members of the adm group, and, if the **chargefee** command is included, factors in the billing fee.

The accounting system also provides data to assess the adequacy of current resource assignments, set resource limits and quotas, forecast future needs, and order supplies for printers and other devices.

The following sections should help you understand how to implement the accounting utility in your system:

Collecting and Reporting System Data

For data to be collected automatically, a member of the adm group needs to follow the procedures described in "Setting Up an Accounting System" on page 9-11. These procedures enable the **cron** daemon to run the commands that generate data on:

- amount of time each user spends logged in to the system
- usage of the processing unit, memory, and I/O resources
- amount of disk space occupied by each user's files
- usage of printers and plotters
- number of times a specific command is given

The system writes a record of each session and process after they are completed. These records are converted into total accounting (**tacct**) records arranged by user and merged into a daily report. Periodically, the daily reports are combined to produce totals for the defined fiscal period. Methods for collecting and reporting the data and the various accounting commands and files are discussed in the following sections.

Although most of the accounting data is collected and processed automatically, a member of the adm group can enter certain commands from the keyboard to obtain specific information. These commands are discussed in "Keyboard Commands" on page 9-7.

Collecting Accounting Data

There are several types of accounting data: connect-time data, process data, disk-usage data, printer-usage data, and fee data. Each is described in the following paragraphs.

Connect-Time Accounting

Connect-time data is collected by the **init** command and the **login** command. When you log in, the **login** program writes a record in the **/etc/utmp** file. This record includes your user name, the date and time of the login, and the login port. Commands, such as **who**, use this file to find out which users are logged into the various display stations. If the **/var/adm/wtmp** connect-time accounting file exists, the **login** command adds a copy of this login record to it.

When your login program ends (normally when you log out), the **init** command records the end of the session by writing another record in the **/var/adm/wtmp** file. Logout records differ from login records in that they have a blank user name. Both the login and logout records have the form described in the **utmp.h** file.

The **acctwtmp** command also writes special entries in the **/var/adm/wtmp** file concerning system shutdowns and startups.

For more information, see “Connect-Time Reports” on page 9-4.

Process Accounting

The system collects data on resource usage for each process as it runs. These data include:

- user and group numbers under which the process runs
- first eight characters of the name of the command
- elapsed time and processor time used by the process
- memory use
- number of characters transferred
- number of disk blocks read or written on behalf of the process

The **accton** command records these data in a specified file, usually the **/var/adm/pacct** file.

Related commands are the **startup** command, the **shutacct** command, the **dodisk** command, the **ckpacct** command, and the **turnacct** command.

For more information, see “Reporting Accounting Data” on page 9-4.

Disk-Usage Accounting

Much accounting information is collected as the resources are consumed. The **dodisk** command, run as specified by the **cron** daemon, periodically writes disk-usage records for each user to the **/var/adm/acct/nite/dacct** file. To accomplish this, the **dodisk** command calls other commands. Depending upon the thoroughness of the accounting search, the **diskusg** command or the **acctdusg** command can be used to collect data. The **acctdisk** command is used to write a total accounting record. The total accounting record, in turn, is used by the **acctmerg** command to prepare the daily accounting report.

The **dodisk** command charges a user for the links to files found in the user’s login directory and evenly divides the charge for each file between the links. This spreads the cost of using a file over all who use it and removes the charges from users when they relinquish access to a file.

For more information, see “Disk-Usage Accounting Report” on page 9-5.

Printer-Usage Accounting

The collection of printer-usage data is a cooperative effort between the **enq** command and the queuing daemon. The **enq** command enqueues the user name, job number, and the name of the file to be printed. After the file is printed, the **qdaemon** command writes an ASCII record to a file, usually the **/var/adm/qacct** file, containing the user name, user

number, and the number of pages printed. You can sort these records and convert them to total accounting records.

For more information, see “Printer-Usage Accounting Report” on page 9-5.

Fee Accounting

You can enter the **chargefee** command to produce an ASCII total accounting record in the **/var/adm/fee** file. This file will be added to daily reports by the **acctmerg** command.

For more information, see “Fee Accounting Report” on page 9-5.

Reporting Accounting Data

After the various types of accounting data are collected, the records are processed and converted into reports.

Accounting commands automatically convert records into scientific notation when numbers become large. A number is represented in scientific notation in the following format:

*Base***e+***Exp*

OR

*Base***e–***Exp*

which is the number equal to the *Base* number multiplied by 10 to the **+***Exp* or **–***Exp* power. For example, the scientific notation 1.345e+9 is equal to 1.345×10^{9}, or 1,345,000,000. And the scientific notation 1.345e–9 is equal to 1.345×10^{-9} or, .000000001345.

Connect-Time Reports

The **runacct** command calls two commands, **acctcon1** and **acctcon2**, to process the login, logout, and system-shutdown records that collect in the **/var/adm/wtmp** file. The **acctcon1** command converts these records into session records and writes them to the **/var/adm/acct/nite/lineuse** file. The **acctcon2** command then converts the session records into a total accounting record, **/var/adm/logacct**, that the **acctmerg** command adds to daily reports.

If you run the **acctcon1** command from the command line, you must include the **–l** flag to produce the line-use report, **/var/adm/acct/nite/lineuse**. To produce an overall session report for the accounting period, **/var/adm/acct/nite/reboots**, use the **acctcon1** command with the **–o** flag.

The **lastlogin** command produces a report that gives the last date on which each user logged in.

Process Accounting Reports

Two commands process the billing-related data that was collected in the **/var/adm/pacct** or other specified file. The **acctprc1** command translates the user ID into a user name and writes ASCII records containing the chargeable items (prime and non-prime CPU time, mean memory size, and I/O data). The **acctprc2** command transforms these records into total accounting records that are added to daily reports by the **acctmerg** command.

Process accounting data also provides information that you can use to monitor system resource usage. The **acctcms** command summarizes resource use by command name. This provides information on how many times each command was run, how much processor time and memory was used, and how intensely the resources were used (also known as the *hog factor*). The **acctcms** command produces long-term statistics on system utilization, providing information on total system usage and the frequency with which commands are used.

The **acctcom** command handles the same data as the **acctcms** command, but provides detailed information about each process. You can display all process accounting records or select records of particular interest. Selection criteria include the load imposed by the process, the time period when the process ended, the name of the command, the user or group that invoked the process, and the port at which the process ran. Unlike other accounting commands, **acctcom** can be run by all users.

Disk-Usage Accounting Report

The disk-usage records collected in the **/var/adm/acct/nite/dacct** file are merged into the daily accounting reports by the **acctmerg** command.

Printer-Usage Accounting Report

The ASCII record in the **/var/adm/qacct** file can be converted to a total accounting record to be added to the daily report by the **acctmerg** command.

Fee Accounting Report

If you used the **chargefee** command to charge users for services such as file restores, consulting, or materials, an ASCII total accounting record is written in the **/var/adm/fee** file. This file is added to the daily reports by the **acctmerg** command.

Daily Reports

Raw accounting data on connect-time, processes, disk usage, printer usage, and fees to charge are merged into daily reports by the **acctmerg** command. Called by the **runacct** command as part of its daily operation, the **acctmerg** command produces the following:

/var/adm/acct/nite/dacct
: an intermediate report that is produced when one of the input files is full

/var/adm/acct/sum/tacct
: a cumulative total report in **tacct** format. This file is used by the **monacct** command to produce the ASCII monthly summary

The **acctmerg** command can convert records between ASCII and binary formats and merge records from different sources into a single record for each user.

Monthly Report

Called by the **cron** daemon, the **monacct** command produces the following:

/var/adm/acct/fiscal
: a periodic summary report produced from the **/var/adm/acct/sum/tacct** report by the **monacct** command

  The **monacct** command can be configured to run monthly or at the end of a fiscal period.

Accounting Commands

The accounting commands function several different ways. Some commands:

- collect data or produce reports for a specific type of accounting: connect-time, process, disk usage, printer usage, or command usage
- call other commands

  For example, the **runacct** command, which is usually run automatically by the **cron** daemon, calls many of the commands that collect and process accounting data and prepare reports. To obtain automatic accounting, you must first configure the **cron**

daemon to run the **runacct** command. See the **crontab** command for more information about how to configure the **cron** daemon to submit commands at regularly scheduled intervals.

- perform maintenance functions and ensure the integrity of active data file
- enable members of the adm group to perform occasional tasks, such as displaying specific records, by entering a command at the keyboard
- enable a user to display specific information

  There is only one user command, the **acctcom** command, which displays process accounting summaries.

Commands That Run Automatically

Several commands usually run by the **cron** daemon automatically collect accounting data.

runacct handles the main daily accounting procedure

Normally initiated by the **cron** daemon during non-prime hours, the **runacct** command calls several other accounting commands to process the active data files and produce command and resource usage summaries, sorted by user name. It also calls the **acctmerg** command to produce daily summary report files, and the **ckpacct** command to maintain the integrity of the active data files.

ckpacct handles **pacct** file size

It is advantageous to have several smaller **pacct** files if you must restart the **runacct** procedure after a failure in processing these records. The **ckpacct** command checks the size of the **/var/adm/pacct** active data file, and if the file is larger than 500 blocks, the command invokes the **turnacct switch** command to turn off process accounting temporarily. The data is transferred to a new **pacct** file, **/var/adm/pacct***x*. (*x* is an integer that increases each time a new **pacct** file is created.) If the number of free disk blocks falls below 500, the **ckpacct** command calls the **turnacct off** command to turn off process accounting.

dodisk calls the **acctdisk** command and either the **diskusg** command or the **acctdusg** command to write disk-usage records to the file **/var/adm/acct/nite/dacct**

This data is later merged into the daily reports.

monacct produces a periodic summary from daily reports

sa1 collects and stores binary data in the **/var/adm/sa/sa***dd* file, where *dd* is the day of the month

sa2 writes a daily report in the **/var/adm/sa/sa***dd* file, where *dd* is the day of the month

The command removes reports from the **/var/adm/sa/sa***dd* file that have been there longer than one week.

Other commands are run automatically by procedures other than the **cron** daemon:

startup initiates startup procedures for the accounting system when added to the **/etc/rc** file

shutacct records the time accounting was turned off by calling the **acctwtmp** command to write a line to **/var/adm/wtmp** file, and then calls the **turnacct off** command to turn off process accounting

Keyboard Commands

A member of the adm group can enter the following commands from the keyboard:

***ac**	prints connect-time records
acctcom	displays process accounting summaries This command is also available to users.
acctcon1	displays connect-time summaries Either the **–l** flag or the **–o** flag must be used.
accton	turns process accounting on and off
chargefee	charges the user a predetermined fee for units of work performed The charges are added to the daily report by the **acctmerg** command.
fwtmp	converts files between binary and ASCII formats
***last**	displays information about previous logins
***lastcomm**	displays information about the last commands that were executed
lastlogin	displays the time each user last logged in
***pac**	prepares printer/plotter accounting records
prctmp	displays a session record
prtacct	displays total accounting files
***sa**	summarizes raw accounting information to help manage large volumes of accounting information
sadc	reports on various local system actions, such as buffer usage, disk and tape I/O activity, TTY device activity counters, and file access counters
time	prints real time, user time, and system time required to execute a command
timex	reports in seconds the elapsed time, user time, and execution time
sar	writes to standard output the contents of selected cumulative activity counters on local activities in the operating system

Note: Commands above that are flagged with an asterisk are provided for compatibility with Berkeley Software Distribution (BSD) systems.

Accounting Files

There are two main accounting directories: the **/usr/sbin/acct** directory, where all the C language programs and shell procedures needed to run the accounting system are stored, and the **/var/adm** directory, which contains the data, report and summary files.

The accounting data files belong to members of the adm group, and all active data files (such as **wtmp** and **pacct**) reside in the adm home directory **/var/adm**.

Data Files

Files in the **/var/adm** directory are:

/var/adm/diskdiag	diagnostic output during the execution of disk accounting programs
/var/adm/dtmp	output from the **acctdusg** command
/var/adm/fee	output from the **chargefee** command, in ASCII **tacct** records
/var/adm/pacct	active process accounting file

/var/adm/wtmp active process accounting file

/var/adm/Spacct*?.mmdd*
process accounting files for *mmdd* during the execution of the **runacct** command

Report and Summary Files

Report and summary files reside in a **/var/adm/acct** subdirectory. You must create the following subdirectories before the accounting system is enabled. See “Setting Up an Accounting System” on page 9-11 for more information.

/var/adm/acct/nite
contains files that the **runacct** command reuses daily

/var/adm/acct/sum
contains the cumulative summary files that the **runacct** command updates daily

/var/adm/acct/fiscal
contains the monthly summary files that the **monacct** command creates

runacct Command Files

The following report and summary files, produced by the **runacct** command, are of particular interest:

/var/adm/acct/nite/lineuse
contains usage statistics for each terminal line on the system

This report is especially useful for detecting bad lines. If the ratio between the number of logouts and logins exceeds about 3 to 1, there is a good possibility that a line is failing.

/var/adm/acct/nite/daytacct
contains the total accounting file for the previous day

/var/adm/acct/sum/tacct
contains the accumulation of each day’s **nite/daytacct** file and can be used for billing purposes

The **monacct** command restarts the file each month or fiscal period.

/var/adm/acct/sum/cms
contains the accumulation of each day’s command summaries

The **monacct** command reads this binary version of the file and purges it. The ASCII version is **nite/cms**.

/var/adm/acct/sum/daycms
contains the daily command summary

An ASCII version is stored in **nite/daycms**.

/var/adm/acct/sum/loginlog
contains a record of the last time each user ID was used

/var/adm/acct/sum/rprt*mmdd*
contains a copy of the daily report saved by the **runacct** command

Files in the /var/adm/acct/nite Directory

active	used by the **runacct** command to record progress and print warning and error messages
	The file **active.***mmdd* is a copy of the **active** file made by the **runacct** program after it detects an error.
cms	ASCII total command summary used by the **prdaily** command
ctacct.*mmdd*	connect total accounting record
ctmp	connect session record
daycms	ASCII daily command summary used by the **prdaily** command
daytacct	total accounting records for one day
dacct	disk total accounting records, created by the **dodisk** command
accterr	diagnostic output produced during the execution of the **runacct** command
lastdate	last day the **runacct** executed, in **date +%m%d** format
lock1	used to control serial use of the **runacct** command
lineuse	tty line usage report used by the **prdaily** comman
log	diagnostic output from the **acctcon1** command
log*mmdd*	same as **log** after the **runacct** command detects an error
reboots	contains beginning and ending dates from **wtmp**, and a listing of system restarts
statefile	used to record the current state during execution of the **runacct** command
tmpwtmp	**wtmp** file corrected by the **wtmpfix** command
wtmperror	contains **wtmpfix** error messages
wtmperr*mmdd*	same as **wtmperror** after the **runacct** command detects an error
wtmp.*mmdd*	previous day's **wtmp** file

Files in the /var/adm/acct/sum Directory

cms	total command summary file for the current fiscal period, in binary format
cmsprev	command summary file without the latest update
daycms	command summary file for the previous day, in binary format
lastlogin	file created by the **lastlogin** command
pacct.*mmdd*	concatenated version of all **pacct** files for *mmdd,* which is removed after system startup by the **remove** command
rprt*mmdd*	saved output of the **prdaily** command
tacct	cumulative total accounting file for the current fiscal period
tacctprev	same as **tacct** without the latest update
tacct*mmdd*	total accounting file for *mmdd*
wtmp.*mmdd*	saved copy of the **wtmp** file for *mmdd,* which is removed after system startup by the **remove** command

Files in the /var/adm/acct/fiscal Directory

cms*?*	total command summary file for the fiscal period, specified by *?*, in binary format
fiscrpt*?*	report similar to that of the **prdaily** command for fiscal period, specified by *?*, in binary format
tacct*?*	total accounting file for fiscal period, specified by *?*, in binary format

Accounting File Formats

Accounting file output and formats are summarized in the following.

wtmp	produces the active process accounting file
	The format of the **wtmp** file is defined in the **utmp.h** file.
ctmp	produces connect session records
	The format is described in the **ctmp.h** file.
pacct*	produces active process accounting records
	The format of the output is defined in the **/usr/include/sys/acct.h** file.
Spacct*	produces process accounting files for *mmdd* during the execution of the **runacct** command
	The format of these files is defined in the **sys/acct.h** file.
daytacct	produces total accounting records for one day
	The format of the file is defined in the **tacct** file format.
sum/tacct	produces binary file that accumulates each day's command summaries
	The format of this file is defined in the **/usr/include/sys/acct.h** header file.
ptacct	produces concatenated versions of **pacct** files
	The format of these files are defined in the **tacct** file.
ctacct	produces connect total accounting records
	The output of this file is defined in the **tacct** file.
cms	produces total accounting command summary used by the **prdaily** command, in binary format, with the ASCII version **nite/cms**
daycms	daily command summary used by the **prdaily** command, in binary format, with the ASCII version is **nite/daycms**

Setting Up an Accounting System

Prerequisites

You must have root authority to complete this procedure.

Procedure

The following is an overview of the steps you must take to set up an accounting system. Refer to the commands and files noted in these steps for more specific information.

1. Enter the **nulladm** command to ensure that each file has the proper access permission: read (r) and write (w) permission for the file owner and group and read (r) permission for others:

   ```
   /usr/sbin/acct/nulladm wtmp pacct
   ```

   This provides access to the **pacct** and **wtmp** files.

2. Update the **/etc/acct/holidays** file to include the hours you designate as prime time and to reflect your holiday schedule for the year.

   Note: Comment lines can appear anywhere in the file as long as the first character in the line is an * (asterisk).

   a. To define prime time, fill in the fields on the first data line (the first line that is not a comment), using a 24-hour clock. This line consists of three 4-digit fields, in the following order:

      – current year

      – beginning of prime time (*hhmm*)

      – end of prime time (*hhmm*)

      Leading blanks are ignored. You can enter midnight as either 0000 or 2400.

      For example, to specify the year 1984, with prime time beginning at 8:00 a.m. and ending at 5:00 p.m., enter:

      ```
      1984  0800  1700
      ```

   b. To define the company holidays for the year on the next data line. Each line contains four fields, in the following order:

      – day of the year

      – month

      – day of the month

      – description of holiday

      The day-of-the-year field contains the number of the day on which the holiday falls and must be a number from 1 through 365 (366 on leap year). For example, February 1st is day 32. The other three fields are for information only and are treated as comments.

      A two-line example follows:

      ```
        1  Jan  1  New Year's Day
      332  Nov 28  Thanksgiving Day
      ```

3. Turn on process accounting by adding the following line to the **/etc/rc** file or by deleting the comment symbol (#) in front of the line if it exists:

   ```
   /usr/bin/su – adm –c /usr/sbin/acct/startup
   ```

   The **startup** procedure records the time that accounting was turned on and cleans up the previous day's accounting files.

4. Identify each file system you want included in disk accounting by adding the following line to the stanza for the file system in the **/etc/filesystems** file:

   ```
   account = true
   ```

5. Specify the data file to use for printer data by adding the following line to the queue stanza in the **/etc/qconfig** file:

   ```
   acctfile = /var/adm/qacct
   ```

6. As the adm user, create a **/var/adm/acct/nite**, **/var/adm/acct/fiscal**, and **/var/adm/acct/sum** directory to collect daily and fiscal period records:

   ```
   su – adm
   cd /var/adm/acct
   mkdir nite fiscal sum
   exit
   ```

7. Set daily accounting procedures to run automatically by editing the **/var/spool/cron/crontabs/root** file to include the **dodisk**, **ckpacct**, and **runacct** commands. For example:

   ```
   0 2 * * 4 /usr/sbin/acct/dodisk
   5 * * * * /usr/sbin/acct/ckpacct
   0 4 * * 1–6 /usr/sbin/acct/runacct
                2>/var/adm/acct/nite/accterr
   ```

   The first line starts disk accounting at 2:00 a.m. (`0 2`) each Thursday (4). The second line starts a check of the integrity of the active data files at 5 minutes past each hour (`5 *`) every day (*). The third line runs most accounting procedures and processes active data files at 4:00 a.m. (`0 4`) every Monday through Saturday (1–6). If these times do not fit the hours your system operates, adjust your entries.

   Note: You must have root user authority to edit the **/var/spool/cron/crontabs/root** file.

8. Set the monthly accounting summary to run automatically by including the **monacct** command in the **/var/spool/cron/crontabs/root** file. For example:

   ```
   15 5 1 * * /usr/sbin/acct/monacct
   ```

   Be sure to schedule this procedure early enough to finish the report. This example starts the procedure at 5:15 a.m. on the first day of each month.

9. To submit the edited **cron** file, enter:

   ```
   crontab /var/spool/cron/crontabs/root
   ```

Generating Reports on System Activity

To generate a report on system activity, use the **prtacct** command. This command reads the information in a total accounting file (**tacct** file format) and produces formatted output. Total accounting files include the daily reports on connect time, process time, disk usage, and printer usage.

Prerequisites

The **prtacct** command requires an input file in the **tacct** file format. This implies that you have an accounting system set up and running or that you have run the accounting system in the past. See “Setting Up an Accounting System” on page 9-11 for guidelines.

Procedure

Generate a report on system activity by entering:

```
prtacct –f Specification –v "Heading" File
```

Specification is a comma-separated list of field numbers or ranges used by the **acctmerg** command. The optional **–v** flag produces verbose output where floating-point numbers are displayed in higher precision notation. *Heading* is the title you want to appear on the report and is optional. *File* is the full path name of the total accounting file to use for input. You can specify more than one file.

Summarizing Accounting Records

To summarize raw accounting data, use the **sa** command. This command reads the raw accounting data, usually collected in the **/var/adm/pacct** file, and the current usage summary data in the **/var/adm/savacct** file, if summary data exists. It combines this information into a new usage summary report and purges the raw data file to make room for further data collection.

Prerequisites

The **sa** command requires an input file of raw accounting data such as the **pacct** file (process accounting file). To collect raw accounting data, you must have an accounting system set up and running. See “Setting Up an Accounting System” on page 9-11 for guidelines.

Procedure

The purpose of the **sa** command is to summarize process accounting information and to display or store that information. The simplest use of the command displays a list of statistics about every process that has run during the life of the **pacct** file being read. To produce such a list, enter:

```
/usr/sbin/sa
```

To summarize the accounting information and merge it into the summary file, enter:

```
/usr/sbin/sa –s
```

The **sa** command offers many additional flags that specify how the accounting information is processed and displayed. See the **sa** command description for more information.

Starting the runacct Command

Prerequisites

- You must have the accounting system installed.
- You must have root user or adm group authority.

Note:

1. If you call the **runacct** command with no parameters, the command assumes that this is the first time that the command has been run today. Therefore, you need to include the *mmdd* parameter when you restart the **runacct** program, so that the month and day are correct. If you do not specify a state, the **runacct** program reads the **/var/adm/acct/nite/statefile** file to determine the entry point for processing. To override the **/var/adm/acct/nite/statefile** file, specify the desired state on the command line.
2. When you perform the following task, you may need to use the full path name **/usr/sbin/acct/runacct** rather than the simple command name, **runacct**.

Procedure

To start the **runacct** command, enter the following:

```
nohup runacct 2> \
/var/adm/acct/nite/accterr &
```

This entry causes the command to ignore all **INTR** and **QUIT** signals while it performs background processing. It redirects all standard error output to the file **/var/adm/acct/nite/accterr**.

Restarting the runacct Command

Prerequisites

- You must have the accounting system installed.
- You must have root user or adm group authority.

Note: The **runacct** command can fail for a variety of reasons, most commonly because the system goes down, the **/usr** file system runs out of space, or the **/var/adm/wtmp** file has records with inconsistent date stamps.

Procedure

If the **runacct** command is unsuccessful, do the following:

1. Check the **/var/adm/acct/nite/active***mmdd* file for error messages.
2. If both the active file and lock files exist in **acct/nite**, check the **accterr** file, where error messages are redirected when the **cron** daemon calls the **runacct** command.
3. Perform any actions needed to eliminate errors.
4. Restart the **runacct** command.
5. To restart the **runacct** command for a specific date, enter the following:

   ```
   nohup runacct 0601 2>> \
   /var/adm/acct/nite/accterr &
   ```

   This restarts the **runacct** program for June 1 (`0601`). The **runacct** program reads the **/var/adm/acct/nite/statefile** file to find out with which state to begin. All standard error output is appended to the **/var/adm/acct/nite/accterr** file.
6. To restart the **runacct** program at a specified state, for example, the MERGE state, enter the following:

   ```
   nohup runacct 0601 MERGE 2>> \
   /var/adm/acct/nite/accterr &
   ```

Showing System Activity

You can display formatted information about system activity with the **sar** command.

Prerequisites

To display system activity statistics, the **sadc** command must be running.

Note: The typical method of running the **sadc** command is to place an entry for the **sa1** command in the root **crontab** file. The **sa1** command is a shell-procedure variant of the **sadc** command designed to work with the **cron** daemon.

Procedure

To display basic system-activity information, enter:

```
sar 2 6
```

where the first number is the number of seconds between sampling intervals and the second number is the number of intervals to display. The output of this command would look something like this:

```
     arthurd 2 3 000166021000    05/28/92

14:03:40    %usr    %sys    %wio   %idle
14:03:42       4       9       0      88
14:03:43       1      10       0      89
14:03:44       1      11       0      88
14:03:45       1      11       0      88
14:03:46       3       9       0      88
14:03:47       2      10       0      88

Average        2      10       0      88
```

The **sar** command also offers a number of flags for displaying an extensive array of system statistics. To see all available statistics, use the **–A** flag. For a list of the available statistics and the flags for displaying them, see the **sar** command.

Note: To have a daily system activity report written to **/var/adm/sa/sa***dd*, include an entry in the root **crontab** file for the **sa2** command. The **sa2** command is a shell procedure variant for the **sar** command designed to work with the **cron** daemon.

Showing System Activity While Running a Command

You can use the **time** and **timex** commands to display formatted information about system activity while a particular command is running.

Prerequisites

The **–o** and **–p** flags of the **timex** command require that system accounting be turned on.

Procedure

- To display the elapsed time, user time, and system execution time for a particular command, enter:

  `time` *`CommandName`*

  OR

  `timex` *`CommandName`*

- To display the total system activity (all the data items reported by the **sar** command) during the execution of a particular command, enter:

  `timex –s` *`CommandName`*

The **timex** command has two additional flags. The **–o** flag reports the total number of blocks read or written by the command and all of its children. The **–p** flag lists all of the process accounting records for a command and all of its children.

Showing Process Time

You can display formatted reports about the process time of active processes with the **ps** command or of finished processes with the **acctcom** command.

Prerequisites

The **acctcom** command reads input in the total accounting record form (**acct** file format). This implies that you have process accounting turned on or that you have run process accounting in the past. See “Setting Up an Accounting System” on page 9-11 for guidelines.

Display the Process Time of Active Processes

The **ps** command offers a number of flags to tailor the information displayed. To produce a full list of all active processes except kernel processes, enter:

```
ps –ef
```

Another useful variation displays a list of all processes associated with terminals:

```
ps –al
```

Both of these usages display a number of columns for each process, including the current CPU time for the process in minutes and seconds.

Display the Process Time of Finished Processes

The process accounting functions are turned on with the **startup** command, which is typically started at system initialization with a call in the **/etc/rc** file. When the process accounting functions are running, a record is written to **/var/adm/pacct** (a total accounting record file) for every finished process that includes the start and stop time for the process. You can display the process time information from a **pacct** file with the **acctcom** command. This command has a number of flags that allow flexibility in specifying which processes to display.

For example, to see all processes that ran for a minimum number of CPU seconds or longer, use the **–O** flag:

```
acctcom –O 2
```

This displays records for every process that ran for at least 2 seconds. If you do not specify an input file, the **acctcom** command reads input from the **/var/adm/pacct** directory.

Showing CPU Usage

You can display formatted reports about the CPU usage by process or by user with a combination of the **acctprc1**, **acctprc2**, and **prtacct** commands.

Prerequisites

The **acctprc1** command requires input in the total accounting record form (**acct** file format). This implies that you have process accounting turned on or that you have run process accounting in the past. See “Setting Up an Accounting System” on page 9-11 for guidelines.

Show CPU Usage for Each Process

To produce a formatted report of CPU usage by process, enter:

```
acctprc1 </var/adm/pacct
```

This information will be useful in some situations, but you will probably also want to summarize the CPU usage by user. The output from this command is used in the next procedure to produce that summary.

Show CPU Usage for Each User

1. Produce an output file of CPU usage by process by entering:

   ```
   acctprc1 </var/adm/pacct >out.file
   ```

   The **/var/adm/pacct** file is the default output for process accounting records. You may want to specify an archive **pacct** file instead.

2. Produce a binary total accounting record file from the output of the previous step by entering:

   ```
   acctprc2 <out.file >/var/adm/acct/nite/daytacct
   ```

   Note: The **daytacct** file is merged with other total accounting records by the **acctmerg** command to produce the daily summary record, **/var/adm/acct/sum/tacct**.

3. Display a formatted report of CPU usage summarized by user by entering:

   ```
   prtacct </var/adm/acct/nite/daytacct
   ```

Showing Connect Time Usage

You can display the connect time of all users, of individual users, and by individual login with the **ac** command.

Prerequisites

The **ac** command extracts login information from the **/var/adm/wtmp** file, so this file must exist. If the file has not been created, the following error message is returned:

```
No /var/adm/wtmp
```

If the file becomes too full, additional **wtmp** files are created; you can display connect-time information from these files by specifying them with the **–w** flag.

Procedure

- To display the total connect time for all users, enter:

  ```
  /usr/sbin/acct/ac
  ```

  This command displays a single decimal number that is the sum total connect time, in minutes, for all users who have logged in during the life of the current **wtmp** file.

- To display the total connect time for one or more particular users, enter:

  ```
  /usr/sbin/acct/ac User1 User2 ...
  ```

  This command displays a single decimal number that is the sum total connect time, in minutes, for the user or users you specified for any logins during the life of the current **wtmp** file.

- To display the connect time by individual user plus the total connect time, enter:

  ```
  /usr/sbin/acct/ac –p User1 User2 ...
  ```

  This command displays as a decimal number for each user specified equal to the total connect time, in minutes, for that user during the life of the current **wtmp** file. It also displays a decimal number that is the sum total connect time for all the users specified. If no user is specified in the command, the list includes all users who have logged in during the life of the **wtmp** file.

Showing Disk Space Utilization

You can display disk space utilization information with the **acctmerg** command.

Prerequisites

To display disk space utilization information, the **acctmerg** command requires input from a **dacct** file (disk accounting). The collection of disk-usage accounting records is performed by the **dodisk** command. Placing an entry for the **dodisk** command in a **crontabs** file is part of the procedure described in “Setting Up an Accounting System” on page 9-11.

Procedure

To display disk space utilization information, enter:

```
acctmerg –a1 –2,13 –h </var/adm/acct/nite/dacct
```

This command displays disk accounting records, which include the number of 1KB blocks utilized by each user.

Note: The **acctmerg** command always reads from standard input and can read up to nine additional files. If you are not piping input to the command, you must redirect input from one file; the rest of the files can be specified without redirection.

Showing Printer Usage

You can display printer or plotter usage accounting records with the **pac** command.

Prerequisites

- To collect printer usage information, you must have an accounting system set up and running. See “Setting Up an Accounting System” on page 9-11 for guidelines.
- The printer or plotter for which you want accounting records must have an `acctfile=` clause in the printer’s stanza of the **/etc/qconfig** file. The file specified in the `acctfile=` clause must grant read and write permissions to the root user or printq group.
- If the **–s** flag of the **pac** command is specified, the command rewrites the summary file name by appending **_sum** to the path name specified by the `acctfile=` clause in the **/etc/qconfig** file. This file must exist and grant read and write permissions to the root user or printq group.

Procedure

- To display printer usage information for all users of a particular printer, enter:

  ```
  /usr/sbin/pac –PPrinter
  ```

  If you do not specify a printer, the default printer is named by the **PRINTER** environment variable. If the **PRINTER** variable is not defined, the default is **lp0**.

- To display printer usage information for particular users of a particular printer, enter:

  ```
  /usr/sbin/pac –PPrinter User1 User2 ...
  ```

The **pac** command offers a number of other flags for controlling what information gets displayed. See the *Commands Reference* for details.

Fixing tacct Errors

If you are using the accounting system to charge users for system resources, the integrity of the **/var/adm/acct/sum/tacct** file is quite important. Occasionally, mysterious **tacct** records appear that contain negative numbers, duplicate user numbers, or a user number of 65,535.

Prerequisites

You must have root user or adm group authority.

Procedure

To patch a **tacct** file, complete this procedure:

1. Move to the **/var/adm/acct/sum** directory:

   ```
   cd /var/adm/acct/sum
   ```

2. Use the **prtacct** command to check the total accounting file, **tacctprev**:

   ```
   prtacct tacctprev
   ```

   The **prtacct** command formats and displays the **tacctprev** file so that you can check connect time, process time, disk usage, and printer usage.

3. If the **tacctprev** file looks all right, change the latest **tacct.***mmdd* file from a binary file to an ASCII file. In the following example, the **acctmerg** command converts the **tacct.***mmdd* file to an ASCII file named `tacct.new`:

   ```
   acctmerg –v < tacct.mmdd > tacct.new
   ```

   Note: The **acctmerg** command with the **–a** flag also produces ASCII output. The **–v** flag produces more precise notation for floating-point numbers.

   The **acctmerg** command is used to merge the intermediate accounting record reports into a cumulative total report (**tacct**). This cumulative total is the source from which the **monacct** command produces the ASCII monthly summary report. Since the **monacct** command procedure removes all the **tacct.***mmdd* files, you recreate the **tacct** file by merging these files.

4. Edit the **tacct.new** file to remove the bad records and write duplicate user number records to another file:

   ```
   acctmerg –i < tacct.new > tacct.mmdd
   ```

5. Create the **tacct** file again:

   ```
   acctmerg tacctprev < tacct.mmdd > tacct
   ```

Fixing wtmp Errors

The **/var/adm/wtmp**, or “who temp” file, may cause problems in the day-to-day operation of the accounting system. When the date is changed and the system is in multiuser mode, date change records are written to the **/var/adm/wtmp** file. When a date change is encountered, the **wtmpfix** command adjusts the time stamps in the **wtmp** records. Some combinations of date changes and system restarts may slip past the **wtmpfix** command and cause the **acctcon1** command to fail and the **runacct** command to send mail to the **root** and **adm** accounts complaining of bad times.

Prerequisites

You must have root user or adm group authority.

Procedure

1. Move to the **/var/adm/acct/nite** directory:

   ```
   cd /var/adm/acct/nite
   ```

2. Convert the binary **wtmp** file to an ASCII file that you can edit:

   ```
   fwtmp < wtmp.mmdd > wtmp.new
   ```

   The **fwtmp** command converts **wtmp** from binary to ASCII.

3. Edit the ASCII **wtmp.new** file to delete damaged records or all records from the beginning of the file up to the needed date change:

   ```
   vi wtmp.new
   ```

4. Convert the ASCII **wtmp.new** file back to binary format:

   ```
   fwtmp –ic < wtmp.new > wtmp.mmdd
   ```

5. If the **wtmp** file is beyond repair, use the **nulladm** command to create an empty **wtmp** file. This prevents any charges in the connect time.

   ```
   nulladm wtmp
   ```

   The **nulladm** command creates the file specified with read and write permissions for the file owner and group, and read permissions for other users. It ensures that the file owner and group are **adm**.

Fixing General Accounting Problems

You may encounter several different problems when using the accounting system. You may need to resolve file ownership and permissions problems.

This section describes how to fix general accounting problems:

- to fix incorrect file permissions
- to fix “bad times” errors
- to fix errors encountered when running the **runacct** command
- to update an out-of-date holidays file

Prerequisites

You must have root user or adm group authority.

Fixing Incorrect File Permissions

To use the accounting system, file ownership and permissions must be correct. The **adm** administrative account owns the accounting command and scripts, except for **/var/adm/acct/accton** which is owned by root.

1. To check file permissions using the **ls** command, enter:

   ```
   ls –l /var/adm/acct
   ```

   ```
   –rws––x––– 1 adm adm 14628 Mar 19 08:11 /var/adm/acct/fiscal
   –rws––x––– 1 adm adm 14628 Mar 19 08:11 /var/adm/acct/nite
   –rws––x––– 1 adm adm 14628 Mar 19 08:11 /var/adm/acct/sum
   ```

2. Adjust file permissions with the **chown** command, if necessary. The permissions should be 755 (all permissions for owner and read and execute permissions for all others). Also, the directory itself should be write-protected from others. For example:

   a. Move to the **/var/adm/acct** directory using the following command:

      ```
      cd /var/adm/acct
      ```

   b. Change the ownership for the **sum**, **nite**, and **fiscal** directories to **adm** group authority using the following command:

      ```
      chown adm sum/* nite/* fiscal/*
      ```

      To prevent tampering by users trying to avoid charges, deny write permission for others on these files. Change the **accton** command’s group owner to **adm**, and permissions to 710, that is, no permissions for others. (Processes owned by **adm** will be able to execute the **accton** command, but ordinary users will not.)

3. The **/var/adm/wtmp** file must also be owned by **adm**. If **/var/adm/wtmp** is owned by root, you will see the following message during startup:

   ```
   /var/adm/acct/startup: /var/adm/wtmp: Permission denied
   ```

   To correct the ownership of **/var/adm/wtmp**, change ownership to the **adm** group by using the following command:

   ```
   chown adm /var/adm/wtmp
   ```

Fixing Errors

Processing the **/var/adm/wtmp** file may produce some warnings mailed to root. The **wtmp** file contains information collected by **/etc/init** and **/bin/login** and is used by accounting scripts primarily for calculating connect time (the length of time a user is logged in). Unfortunately, date changes confuse the program that processes the **wtmp** file. As a result, the **runacct** command will send mail to root and adm complaining of any errors after a date change since the last time accounting was run.

1. Determine if you received any errors.

   The **acctcon1** command outputs error messages that are mailed to adm and root by the **runacct** command. For example, if the **acctcon1** command stumbles after a date change and fails to collect connect times, adm might get mail like the following mail message:

   ```
   Mon Jan 6 11:58:40 CST 1992
   acctcon1: bad times: old: Tue Jan 7 00:57:14 1992
   new: Mon Jan 6 11:57:59 1992
   acctcon1: bad times: old: Tue Jan 7 00:57:14 1992
   new: Mon Jan 6 11:57:59 1992
   acctcon1: bad times: old: Tue Jan 7 00:57:14 1992
   new: Mon Jan 6 11:57:59 1992
   ```

2. Adjust the **wtmp** file.

   ```
   /usr/sbin/acct/wtmpfix wtmp
   ```

   The **wtmpfix** command examines the **wtmp** file for date and time-stamp inconsistencies and corrects problems that could make **acctcon1** fail. However, some date changes slip by **wtmpfix**. See “Fixing wtmp Errors” on page 9-25.

3. Run accounting right before shutdown or immediately after startup.

   Using the **runacct** command at these times minimizes the number of entries with bad times. The **runacct** command will continue to send mail to the root and adm accounts, until you edit the **runacct** script, find the `WTMPFIX` section, and comment out the line where the file log gets mailed to the **root** and **adm** accounts.

Fixing Errors Encountered When Running the runacct Command

The **runacct** command processes files that are often very large. The procedure involves several passes through certain files and consumes considerable system resources while it is taking place. That’s why the **runacct** command is normally run early in the morning when it can take over the machine and not disturb anyone.

The **runacct** command is a script divided into different stages. The stages allow you to restart the command where it stopped, without having to rerun the entire script.

When the **runacct** encounters problems, it sends error messages to different destinations depending on where the error occurred. Usually it sends a date and a message to the console directing you to look in the **active***MMDD* file (such as **active0621** for June 21st) which is in the **/usr/adm/acct/nite** directory. When the **runacct** command aborts, it moves the entire **active** file to **active***MMDD* and appends a message describing the problem.

Review the following error message tables for errors you have encountered when running the **runacct** command.

Preliminary State and Error Messages from the runnacct Command

State	Command	Fatal?	Error Message	Destinations
pre	**runacct**	yes	`* 2 CRONS or ACCT PROBLEMS * ERROR: locks found, run aborted`	console, mail, active
pre	**runacct**	yes	`runacct: Insufficient space in /usr ( nnn blks); Terminating procedure`	console, mail, active
pre	**runacct**	yes	`SE message; ERROR: acctg already run for 'date': check lastdate`	console, mail, active*MMDD*
pre	**runacct**	no	`* SYSTEM ACCOUNTING STARTED *`	console
pre	**runacct**	no	`restarting acctg for 'date' at STATE`	console active, console
pre	**runacct**	no	`restarting acctg for 'date' at state (argument $2) previous state was STATE`	active
pre	**runacct**	yes	`SE message; Error: runacct called with invalid arguments`	console, mail, active*MMDD*

States and Error Messages from the runacct Command

State	Command	Fatal?	Error Message	Destinations
SETUP	**runacct**	no	`ls –l fee pacct* /var/adm/wtmp`	active
SETUP	**runacct**	yes	`SE message; ERROR: turnacct switch returned rc=error`	console, mail, active*MMDD*
SETUP	**runacct**	yes	`SE message; ERROR: SpacctMMDD already exists file setups probably already run`	active*MMDD*
SETUP	**runacct**	yes	`SE message; ERROR: wtmpMMDD already exists: run setup manually`	console, mail, active*MMDD*
WTMPFIX	**wtmpfix**	no	`SE message; ERROR: wtmpfix errors see xtmperrorMMDD`	active*MMDD*, wtmperror*MMDD*
WTMPFIX	**wtmpfix**	no	`wtmp processing complete`	active

State	Command	Fatal?	Error Message	Destinations
CONNECT1	**acctcon1**	no	`SE message; (errors from acctcon1 log)`	console, mail, active*MMDD*
CONNECT2	**acctcon2**	no	`connect acctg complete`	active
PROCESS	**runacct**	no	`WARNING: accounting already run for pacct`*N*	active
PROCESS	**acctprc1 acctprc2**	no	`process acctg complete for SpacctNMMDD`	active
PROCESS	**runacct**	no	`all process actg complete for date`	active
MERGE	**acctmerg**	no	`tacct merge to create dayacct complete`	active
FEES	**acctmerg**	no	`merged fees OR no fees`	active
DISK	**acctmerg**	no	`merged disk records OR no disk records`	active
MERGEACCT	**acctmerg**	no	`WARNING: recreating sum/tacct`	active
MERGEACCT	**acctmerg**	no	`updated sum/tacct`	active
CMS	**runacct**	no	`WARNING: recreating sum/cms`	active
CMS	**acctcms**	no	`command summaries complete`	active
CLEANUP	**runacct**	no	`system accounting completed at 'date'`	active
CLEANUP	**runacct**	no	`*SYSTEM ACCOUNTING COMPLETED*`	console
<wrong>	**runacct**	yes	`SE message; ERROR: invalid state, check STATE`	console, mail, active*MMDD*

Note: The label <wrong> in the previous table does not represent a state, but rather a state other than the correct state that was written in the state file **/usr/adm/acct/nite/statefile**.

Summary of Message Destinations

Destination	Description
console	The **/dev/console** device
mail	Message mailed to **root** and **adm** accounts
active	The **/usr/adm/acct/nite/active** file
activeMMDD	The **/usr/adm/acct/nite/active***MMDD* file
wtmperrMMDD	The **/usr/adm/acct/nite/wtmperror***MMDD* file
STATE	Current state in **/usr/adm/acct/nite/statefile** file
fd2log	Any other error messages

The abbreviation *MMDD* stands for the month and day, such as 0102 for January 2. For example, a fatal error during the CONNECT1 process on January 2 would create the file **active0102** containing the error message.

The abbreviation “SE message” stands for the standard error message such as:

```
********* ACCT ERRORS : see active0102 *********
```

Updating an Out-of-Date Holidays File

The **acctcon1** command (started from the **runacct** command) sends mail to the **root** and **adm** accounts when the **/usr/lib/acct/holidays** file gets out of date. The holidays file is out of date after the last holiday listed has passed or the year has changed.

Update the out-of-date holidays file by editing the **/var/adm/acct/holidays** file to differentiate between prime and nonprime time.

Prime time is assumed to be the period when your system is most active, such as workdays. Saturdays and Sundays are always nonprime times for the accounting system, as are any holidays that you list.

The holidays file contains three types of entries: comments, the year and prime-time period, and a list of holidays as in the following example:

```
* Prime/Non-Prime Time Table for Accounting System
*
*   Curr        Prime           Non-Prime
*   Year        Start           Start
    1992        0830            1700
*
*   Day of      Calendar        Company
*   Year        Date            Holiday
*
*   1           Jan 1           New Year’s Day
*   20          Jan 20          Martin Luther King Day
*   46          Feb 15          President’s Day
*   143         May 28          Memorial Day
*   186         Jul 3           4th of July
*   248         Sep 7           Labor Day
*   329         Nov 24          Thanksgiving
*   330         Nov 25          Friday after
*   359         Dec 24          Christmas Eve
*   360         Dec 25          Christmas Day
*   361         Dec 26          Day after Christmas
```

The first noncomment line must specify the current year (as four digits) and the beginning and end of prime time, also as four digits each. The concept of prime and nonprime time only affects the way that the accounting programs process the accounting records.

If the list of holidays is too long, the **acctcon1** command will generate an error, and you will need to shorten your list. You are safe with 20 or fewer holidays. If you want to add more holidays, just edit the holidays file each month.

Displaying Locking Activity

You can display system locking activity with the **lockstat** command. You can run this command from the command line or with SMIT.

Using the Command Line

Show locking activity by entering:

```
lockstat 2 6
```

where the first number specifies the number of seconds between sampling intervals, and the second number is the number of samples to display. If no parameters are given, a single report covering a one second period is displayed. The report's output is similar to:

```
Subsys   Name                Ocn   Ref/s   %Ref   %Block   %Sleep
------------------------------------------------------------------
PROC     PROC_LOCK_CLASS      2    1442    3.06    6.98     0.75
PROC     PROC_INT_CLASS       1    1408    2.98    5.86     1.77
IOS      IOS_LOCK_CLASS       4     679    1.44    5.19     2.29
```

The **lockstat** command can filter its output depending on a number of conditions. This allows you to limit the reports to the most active locks, or to those locks that are causing the most contention. Limiting the number of locks that are analyzed, reduces the system resources required to generate the locking reports. For more information, see the **lockstat** command.

Using SMIT

1. Use the **smit lockstat** fast path to access the **SHOW Lock Statistics** menu.
2. Select whether you wish to generate a default or a user-defined report.
3. Regardless of the report type selected, follow these steps:
   a. Specify the interval between reports in the field:

      ```
      SECONDS between samples
      ```

   b. Specify the number of reports required in the field:

      ```
      NUMBER of samples
      ```

   c. To print a list of the most active locks, specify **yes** in the field:

      ```
      Print the most ACTIVE locks
      ```

4. If you selected a user-defined report, follow these steps:
   a. Specify the maximum number of locks to be considered in the field:

      ```
      MAX. NUMBER of Locks to analyze
      ```

   This and subsequent selections can be used to limit the analysis to a small number of the most active locks, reducing the system resources required to generate each lock activity report.

   b. Specify the reference rate in the field:

      ```
      REFERENCE Rate in percent
      ```

   This limits the analysis to those locks which are at least *reference-rate* percent as active as the most active lock in the system.

c. Specify the block ratio in the field:

```
BLOCK Ratio in percent
```

This limits the analysis to those locks whose proportion of blocking to non-blocking lock requests is at least *block-ratio* percent.

d. Specify the minimum number of times a lock must be taken during each interval in the field:

```
MIN. NUMBER of times the lock has been taken
```

This limits the analysis to those locks which are heavily used in a given interval.

5. Confirm your choice to display the locking activity reports.

Chapter 10. Advanced System Security

This chapter discusses advanced system security, including auditing and the Trusted Computer Base (TCB). System security is discussed in the *System User's Guide: Operating System and Devices*. Topics include the following:

- security administration, such as user administration and identification and authentication
- system security guidelines
- setting up and maintaining system security
- the Trusted Computing Base (TCB)
- managing protected resources with access control
- a description of auditing and how to set it up

Security Administration

Proper system administration is vital to maintaining the security of information resources on a computer system. AIX security is based on establishing and maintaining proper access control and accountability policies. It is an administrator's responsibility to configure the following aspects of security:

Managing Protected Resources with Access Control
: addresses the privacy, integrity, and availability of information on your system

Identification and Authentication
: determines how users are identified and how their identities are authenticated

Trusted Computing Base (TCB)
: enforces the information security policies of the system

Auditing
: records and analyzes events that take place on the system

The primary goal of security is the detection and prevention of security violations on a system. Computer security includes the fundamental aspects that are discussed in the following sections.

User Administration

User administration consists of creating users and groups and defining their attributes. A major attribute of users is how they are authenticated. Users are the primary agents on the system. Their attributes control their access rights; environment; how they are authenticated; and how, when, and where their accounts can be accessed.

Groups are collections of users who can share access permissions for protected resources. A group has an ID and is composed of members and administrators. The creator of the group is usually the first administrator.

The operating system supports the standard user attributes usually found in the **/etc/passwd** and **/etc/group** files, such as:

Authentication Information
: specifies the password

Credentials
: specifies the user identifier, principal group, and the supplementary group ID

Environment
: specifies the home or shell environment

The operating system allows for greater control, if desired, with extended attributes. Security information can also be separately protected from public access.

Some users and groups can be defined as administrative. These users and groups can be created and modified only by the root user.

User Account Control

Each user account has a set of associated attributes. These attributes are created from default values when a user is created using the **mkuser** command. They can be altered by using the **chuser** command. The following are examples of user attributes:

ttys limits certain accounts to physically secure areas

expires manages student or guest accounts; also can be used to turn off accounts temporarily

logintimes restricts when a user can log in. For example, a user may be restricted to accessing the system only during normal business hours

The complete set of user attributes is defined in the **/usr/lib/security/mkuser.default**, **/etc/security/user**, **/etc/security/limits, /etc/security/lastlog** files. Several of these attributes control how a user can log in, and these attributes can be configured to lock the user account (prevent further logins) under specified conditions.

Once the user's account is locked, the user will not be able to log in until the system administrator resets the user's **unsuccessful_login_count** attribute in the file **/etc/security/lastlog** to be less than the value of login retries. This can be done using the following **chsec** command:

```
chsec –f /etc/security/lastlog –s username -a
     unsuccessful_login_count=0
```

The defaults can be changed by using the **chsec** command to edit the default stanza in the appropriate security file, such as the **/etc/security/user**, **/usr/lib/security/mkuser.default**, or **/etc/security/limits** files. Many of the defaults are defined to be the standard behavior.

Identification and Authentication

Identification and authentication establish a user's identity. Each user is required to log into the system. The user supplies the user name of an account and a password, if the account has one (in a secure system, all accounts should either have passwords or be invalidated). If the password is correct, the user is logged in to that account; the user acquires the access rights and privileges of the account. The **/etc/passwd** and **/etc/security/passwd** files maintain user passwords.

Alternative methods of authentication are integrated into the system by means of the **SYSTEM** attribute that appears in **/etc/security/user**. For instance, the Distributed Computing Environment (DCE) requires password authentication but validates these passwords in a manner different from the encryption model used in /**etc/passwd** and **/etc/security/passwd**. Users who authenticate by means of DCE may have their stanza in **/etc/security/user** set to `SYSTEM=DCE.`

Other **SYSTEM** attribute values are **compat**, **files**, and **NONE**. The **compat** token is used when name resolution (and subsequent authentication) follows the local database, and if no resolution is found, the Network Information Services (NIS) database is tried. The **files** token specifies that only local files are to be used during authentication. Finally, the **NONE** token turns off method authentication. To turn off all authentication, the **NONE** token must appear in the **SYSTEM** and **auth1** lines of the user's stanza.

Other acceptable tokens for the **SYSTEM** attribute may be defined in **/etc/security/login.cfg**.

Note: The root user should always be authenticated by means of the local system security file. The **SYSTEM** attribute entry for the root user is specifically set to **SYSTEM = "compat"** in **/etc/security/user**.

See the *System User's Guide: Operating System and Devices* for more information on protecting passwords.

Configuring Password Restrictions

Proper password management can only be accomplished through user education. But to provide some additional security, AIX provides configurable password restrictions. These allow the administrator to constrain the passwords chosen by users and to force passwords to be changed regularly. These restrictions are recorded in the **/etc/security/user** attribute file and are enforced whenever a new password is defined for a user. All password restrictions are defined per user. By keeping restrictions in the default stanza of the **/etc/security/user** file, the same restrictions are enforced on all users. To maintain proper password security, all passwords should be similarly protected.

The operating system also provides a method for administrators to extend the password restrictions. Using the **pwdchecks** attribute of the **/etc/security/user** file, an administrator can add new subroutines (known as methods) to the password restrictions code. Thus, local site policies can be added to and enforced by the operating system. See "Extending Password Restrictions" on page 10-6 for more information.

Restrictions should be applied sensibly. Attempts to be too restrictive, such as limiting the password space (making guessing easier) or forcing the user to select difficult-to-remember passwords (which are then written down) can jeopardize password security. Ultimately, password security rests with the user. Simple password restrictions, coupled with proper guidelines and an occasional audit (checking current passwords to see if they are unique), are the best policy.

The restrictions that can be applied are:

minage minimum number of weeks that must pass before a password can be changed

maxage maximum number of weeks that can pass before a password must be changed

maxexpired maximum number of weeks beyond **maxage** that a password can be changed before administrative action is required to change the password

Root is exempt.

minalpha minimum number of alphabetic characters the new password must contain

minother minimum number of non–alphabetic characters the new password must contain

Other characters are any ASCII printable characters that are non–alphabetic and are not national language code points.

minlen minimum number of characters the new password must contain

Note: The minimum length of a password on the system is **minlen** or **minalpha** plus **minother**, whichever is greater. The maximum length of a password is eight characters. **minalpha** plus **minother** should never be greater than eight. If **minalpha** plus **minother** is greater than eight, then **minother** is reduced to eight minus **minalpha**.

maxrepeats maximum number of times a character can appear in the new password

mindiff minimum number of characters in the new password that must be different from the characters in the old password

histexpire number of weeks that a user will not be able to reuse a password

histsize number of previous passwords that cannot be reused

Note: If both **histexpire** and **histsize** are set, the system retains the number of passwords required to satisfy both conditions up to the system limit of 50 passwords per user. Null passwords are not retained.

dictionlist list of dictionary files checked when a password is changed

Dictionary files contain passwords that are not allowable.

pwdchecks list of external password restriction methods that are used when a password is changed

Recommended, Default, and Maximum Password Attribute Values

Restriction Values	Advised Values	Default Values	Maximum Values
minage	0	0	52
maxage	8	0	52
maxexpired	4	–1	52
minalpha	4	0	8
minother	1	0	8
minlen	6	0	8
mindiff	3	0	8
maxrepeats	1	8	8
histexpire	26	0	260*
histsize	0	0	50
dictionlist	NA	NA	NA
pwdchecks	NA	NA	NA
*A maximum of 50 passwords are retained. NA = not applicable			

Restrictions should be set so that passwords are hard to guess, yet not hard to remember. Passwords that are hard to remember are often written down somewhere, which compromises system security.

If text processing is installed on the system, the administrator can use the **/usr/share/dict/words** file as a **dictionlist** dictionary file. In such a case, the administrator should set **minother** to 0. Since most words in this dictionary file do not contain characters that fall into the **minother** category, setting **minother** to 1 or more would eliminate the need for the vast majority of words in this dictionary file.

Extending Password Restrictions

The rules used by the password program to accept or reject passwords (the password composition restrictions) can be extended by system administrators to provide site-specific restrictions. Restrictions are extended by adding subroutines, known as methods, which are called during a password change. The **pwdchecks** attribute in the **/etc/security/user** file specifies the methods called.

The *Technical Reference* contains a description of the **pwdrestrict_method**, the subroutine interface that specified password restriction methods must conform to. To properly extend the password composition restrictions, the system administrator must program this interface when writing a password restriction method. Caution is advised in extending the password composition restrictions. These extensions directly affect the **login** command, the **passwd** command, the **su** command, and other programs. The security of the system could easily be subverted by malicious or defective code. Only use code that you trust.

Login User IDs

All audit events recorded for this user are labeled with this ID and should be examined when you generate audit records. Refer to the *System User's Guide: Operating System and Devices* for more information about login user IDs.

System Security Guidelines

The following guidelines are for system administrators who need to implement and maintain system security.

This information does not provide security guidelines for all operational environments. It is impossible to create a single set of guidelines for all security requirements. These guidelines are not intended to represent the only requirements for achieving a secure system.

It is helpful to plan and implement your security policies before you begin using the system. Security policies are very time consuming to change later, so a little planning now can save a lot of time later.

Note: Networks and communications security are described fully in the *System Management Guide: Communications and Networks*.

Warning: Any operating environment may have unique security requirements that are not addressed in these guidelines. To ensure a secure system, system administrators may need to implement additional security measures not discussed here.

Basic Security for User Accounts

Every system should maintain the level of security represented by these basic security policies.

Many attributes can be set for each user account, including password and login attributes. (For a list of configurable attributes, see “Adding a User” on page 5-2.) The following are recommended:

- Each user should have a user ID that is not shared with any other user. All of the security safeguards and accountability tools only work if each user has a unique ID.
- Give user names that are meaningful to the users on the system. Actual names are best, since most electronic mail systems use the user ID to label incoming mail.
- Add, change, and delete users using the SMIT interface. Although you can perform all these tasks from the command line, SMIT helps reduce small errors.
- Do not give a user account an initial password until the user is ready to log on to the system. If the password field is defined as an * (asterisk) in the **/etc/passwd** file, account information is kept, but no one can log in to that account.
- Do not change the system-defined user IDs that are needed by the system to function properly. The system-defined user IDs are listed in the **/etc/passwd** file.
- In general, do not set the **admin** parameter to **true** for any user IDs. Only the root user can change attributes for users with **admin=true** set in the **/etc/security/user** file.

File Ownership and User Groups

When a file is created, the operating system assigns the user ID of the new file the effective user ID of the process that created it. The group ID of the file is either the effective group ID of the process or the group ID of the directory that contains the file, based on the set group ID (SUID) bit of that directory.

File ownership can be changed with the **chown** command.

The **id** command shows your user ID (UID), group ID (GID), and the names of all groups you belong to.

In file listings (such as the listings shown by the **li** or **ls** command), the three groups of users are always represented in the following order: user, group, and others. If you need to find out your group name, the **groups** command shows all the groups for a user ID.

The “File Ownership and User Groups” in *System User’s Guide: Operating System and Devices* contains more information about file and directory access modes.

Groups

Groups are collections of users who can share access permissions for protected resources. Plan your system groups before you begin creating them. Groups can make administration easier, but once you start using the system, it is harder to change your group organization. There are three types of groups: user, system administrator, and system-defined.

User Groups

In general, create as few user groups as possible.

Groups should be made for people who need to share files on the system, such as people who work in the same department, or people who are working on the same project.

For example, consider a small engineering office with three sets of people in the office: office support personnel, system administrators, and engineers. Only two user groups, one for each function in the office, are needed: OFFICE (for the office management staff), and ENGINEER (for the engineers). Later, for example, if a small group of engineers begins work on a special project, a new group called PROJECT can be created and those engineer user IDs can be added to the PROJECT group. Though users can be in more than one group at a time, as in this case, they can only have one primary group at a time. Users can change their primary group with the **newgrp** command.

It is also recommended for simple systems that you do not set the **admin** characteristic when creating groups. If a group has **admin=true** set in the **/etc/security/group** file, only the root user can administer that group.

System Administrator Groups

System administrators should be members of the SYSTEM group. SYSTEM group membership allows an administrator to perform some system maintenance tasks without having to operate with root authority.

System-Defined Groups

There are several system-defined groups. The STAFF group is the default group for all nonadministrative users created in the system. You can change the default group by using the **chsec** command to edit the **/usr/lib/security/mkuser.default** file.

The SECURITY group is a system-defined group having limited privileges for performing security administration. SECURITY group members have access to programs and files in **/etc/security** directory. SECURITY group members can change most attributes for nonadministrative users and groups, such as the user’s login shell or the membership of a nonadministrative group.

Most systems do not need to use this group; only multiuser systems with many users should consider using this group. Otherwise, system administrators can perform the same tasks as SECURITY group members by using the **su** command to gain root privilege.

The other system-defined groups are used to control certain subsystems. Consult the subsystem information to see if certain users should be defined as a member of those groups. The system-defined groups and users appear in the **/etc/group** file.

File System Security

All file system objects (including files, directories, special files, link files, symbolic link files, and pipes) have security mechanisms associated with them. The most commonly used is the access control list (ACL), but the following additional ways of controlling file security can also be used:

Base ACLs These specify the permissions for the owner, group, and others. These permissions are controlled through the **chmod** command. For more information, see the section on "File Directory and Access Modes" in the *System User's Guide: Operating System and Devices*.

Extended ACLs
These provide finer access control than the base ACLs. For more information about the extended ACLs, see the section on "Access Control Lists" in the *System User's Guide: Operating System and Devices*.

Status of Extended ACLs
The extended ACL must be enabled for a file system object; otherwise, the extended ACL is ignored.

Owner ID This is the user ID of the owner of the file system object. Only this user ID has the permissions granted for the owner of the object.

Group ID This is the ID of the group associated with the object. Only members of this group have the permissions granted for the group associated with the object.

Sticky bit If the sticky bit is set for a directory, only the owner of the directory or the owner of a file can delete or rename a file within that directory, even if others have write permission to the directory. This can be set with the **t** flag with the **chmod** command.

TCB bit If the TCB bit is set for a file system object, it identifies that object as part of the Trusted Computing Base (TCB).

umask The **umask** environment parameter specifies the default permissions for any file or directory created.

Status of the file system
A file system can be mounted with read/write or read-only permissions.

Follow these guidelines when dealing with file system objects:

- In general, do not use extended ACLs. For most systems, base ACLs are sufficient for administration. If you do need the extra security control provided by extended ACLs, use them only when necessary and in an organized manner. Maintaining relevant entries in many extended ACLs can become very time-consuming. Also, do not use extended ACLs at all if you are in a heterogeneous network because they are only recognized by AIX systems.
- Use the sticky bit on directories where everyone has write permissions.
- Protect user **.profile** files with 740 permissions.
- Do not let users have write access to system directories.
- Do not change permissions on any files or directories installed as part of the system. Changing those permissions affects system integrity.

Root Access

The system administrator should have the root password to gain root authority by using the **su** command. Immediately after the system is installed, the root account should be given a password.

The root account should always be authenticated by means of the local security files.

Note: The root account should always have a password, and it should never be shared. Only one person, the system administrator, should know the root password. System administrators should only operate as root to perform system administration functions that require root privileges, and then return to a normal user account. Routinely operating as root can result in damage to the system as root overrides many safeguards in the system.

PATH Environment Variable

The **PATH** environment variable is an important security control. It specifies the directories to be searched to find a command. The default systemwide **PATH** value is specified in the **/etc/profile** file, and each user normally has a **PATH** value in the user's **$HOME/.profile** file. The **PATH** value in the **.profile** file either overrides the systemwide **PATH** value or adds extra directories to it.

Unauthorized changes to the **PATH** environment variable can enable a user on the system to "spoof" other users (including root users). Spoofing programs (also called Trojan Horse programs) replace system commands and then capture information meant for that command, such as user passwords.

For example, suppose a user changes the **PATH** value so that the system searches the **/tmp** directory first when a command is executed. Then the user places in the **/tmp** directory a program called **su** that asks for the root password just like the **su** command. Then the **/tmp/su** program mails the root password to the user and calls the real **su** command before exiting. In this scenario, any root user who used the **su** command would give away the root password and not even be aware of it. This is just one of many scenarios for gaining confidential information by altering **PATH** values.

However, following a few simple steps will prevent any problems with the **PATH** environment variable for system administrators and users:

- When in doubt, specify full path names. If a full path name is specified, the **PATH** environment variable is ignored.
- Never put the current directory (specified by **.** (period)) in the **PATH** value specified for the root user. Never allow the current directory to be specified in **/etc/profile**.
- The **PATH** value in the **/etc/profile** file is used by the root user. Only specify directories that are secure, that is, that only root can write to. It is recommended also that you do not create a **.profile** file in the **/** (root) directory. The **.profile** files should only be in users' **$HOME** directories.
- Warn other users not to change their **.profile** files without consulting the system administrator. Otherwise, an unsuspecting user could make changes that allow unintended access. A user's **.profile** file should have permissions set to 740.
- System administrators should not use the **su** command to gain root privilege from a user session, because the user's **PATH** value specified in the **.profile** file is in effect. User's can set their **.profile** files to whatever they please. System administrators should log on to the user's machine as root, or should use the following command:

```
su – root
```

This will ensure that root's environment is used during the session. If a system administrator does operate as root in another user's session, then the system administrator should specify full path names throughout the session.

- Protect the **IFS** (input field separator) environment variable from being changed in the **/etc/profile** file. And beware of any user who changes the **IFS** variable in the **.profile** file. It too can be used to alter the **PATH** value.

Advanced Security

These security policies provide a greater level of security, but also require more work to maintain. Consequently, many system administrators use these security features only in a limited way, or not at all.

Accounting

System accounting is not a direct security function, but the information it gathers is important for detecting security problems. You should activate basic accounting on your system, as explained in "Setting Up an Accounting System" on page 9-11, although you may want to consider not activating disk accounting and printing accounting as specified in the procedure. Both of these accounting functions produce a large amount of data, and they are not vital to system security.

Auditing

For smaller systems, auditing is generally not necessary. However, on large multiuser systems, it can provide useful information about system activity. For more information, see "Auditing Overview" on page 10-26.

Trusted Computing Base (TCB)

The Trusted Computing Base (TCB) allows administrators to closely monitor trusted programs and enhance the security of the system.

Most systems should use only two parts of the TCB: the **tcbck** command and the default **/etc/security/sysck.cfg** configuration file. The **tcbck** command uses information in the **/etc/security/sysck.cfg** file to compare the security status of key elements of the system against the base data stored in the **sysck.cfg** file. All the administrator must do is protect the **sysck.cfg** file and run the **tcbck** command regularly.

For larger systems, the TCB can monitor the system more extensively and provide a secure set of system components. But this extra security requires extra administrative effort. For more information, see "Trusted Computing Base Overview" on page 10-15.

Setting Up and Maintaining System Security

The guidelines in the following sections are for system administrators who need to implement and to maintain basic system security:

- “Setting Up Security at Installation” on page 10-12
- “Periodic Tasks for Maintaining System Security” on page 10-13
- “Security Tasks for Adding Users” on page 10-14
- “Security Tasks for Removing Users” on page 10-14

Warning: Any operating environment may have unique security requirements that are not addressed in these guidelines. To establish a secure system, system administrators may need to implement additional security measures not discussed here.

These guidelines do *not* include the following security subjects:

- extended accounting
- auditing
- trusted Computing Base (TCB)
- extended access control list functions

See “Auditing Overview” on page 10-26, and “Trusted Computing Base Overview” on page 10-15 for information on these security subjects.

Setting Up Security at Installation

When installing the system, set the **Install Trusted Computing Base** option to **yes** on the Installation and Settings menu. Leaving the value at **no** during installation will require you to reinstall if you later decide that you want a more secure system. Selecting **yes** enables trusted path, trusted shell, and system integrity checking. After you have installed the operating system and any major software packages, perform the following actions:

1. If your system is running TCP/IP, see “TCP/IP Security” in *System Management Guide: Communications and Networks* for recommendations.
2. Change the root password as soon as you log into the new system.
3. Activate minimal accounting by using the procedure in “Setting Up an Accounting System” on page 9-11. However, you should consider not activating disk accounting and printing accounting as specified in the procedure. Both of these functions produce a large amount of data, and neither is vital to system security.
4. If necessary, change the default user attributes by using the **chsec** command to edit the **/usr/lib/security/mkuser.default** file. If you are not going to use the STAFF group as the system default, set the **pgrp** variable to the name of the default group for your system. You should set your default to the group with the least privileges to sensitive data on your system.
5. Set the minimum password criteria by using the **chsec** command to edit the default stanza of the **/etc/security/user** file, or by using the **chuser** command to set password restrictions on specific users in the **/etc/security/user** file. Set the password criteria to the ones specified in the table of Recommended, Default, and Maximum Password Attribute Values on page 10-5.
6. Define the **TMOUT** and **TIMEOUT** values in the **/etc/profile** file.

7. Run the **tcbck** command to establish a baseline of the Trusted Computing Base (TCB). Print the **/etc/security/sysck.cfg** configuration file. Fix any problems now, and store the printout of the configuration file in a secure place.
8. Run the **errpt** command now. The **errpt** command reports software and hardware errors logged by the system.
9. If you are going to configure the **skulker** command, modify the default **cron** job in the **/usr/spool/cron/crontabs/root** file to send the output of the **skulker** command to a file for review.

   Note: Unless you have special system requirements, it is not generally recommended that you configure the **skulker** command.
10. Create a list of all directories and files in the system at this point. Change to the **/** (root) directory with the **cd** command, and then use the **su** command to gain root privilege. Enter the following command:

    ```
    li –Ra –l –a > listofallfiles
    ```

    If possible, you should print the `listofallfiles` file (it will be several thousand lines long). Store the printout in a secure place to refer to later if your system develops problems.
11. Turn the system key to the Normal position. Remove the key, and store it in a secure location. In the Normal position the system can be rebooted, but not into Service mode, thus preventing anyone from resetting the root password. Single-user systems can leave the key in the Normal position.

    If you also want to prevent users from rebooting the machine at all, set the key to the Secure position. This is recommended for multiuser systems.
12. Create the initial user IDs for the system.
13. Decide if your system will run continuously or be shut down every evening.

    Most multiuser systems should be left running continuously, although display terminals should be shut off when not in use.

    If the system will be shut down in the evenings, you should reschedule those **cron** jobs that the system sets to run at 3 a.m. every morning. These jobs include tasks such as daily accounting and the removal of unnecessary files, both of which have an impact on system security. Use the **at** command to check the **cron** jobs schedule for when your machine will be off, and reschedule them for other times.

    If your system is going to run 24 hours a day, consider disabling all remote or dial-in terminals at the end of the day (or whenever no authorized users would be using them). You may want to set a **cron** job to do this automatically.

    You should also ensure that all the system-scheduled **cron** jobs, such as accounting and auditing report generation, do not start at the same time. If you have directed the output of these operations to a single file, the output for these reports could be interleaved, making them hard to read.

Periodic Tasks for Maintaining System Security

The following tasks should be performed periodically.

- Perform system backups and check the backup tapes, probably weekly.
- Use the **tcbck** command daily or weekly.
- Run the **grpck**, **pwdck**, and **usrck** commands daily, or at least weekly.

- Update the **/etc/security/sysck.cfg** file whenever important files or **suid** programs are added to the system.
- Check the accounting output weekly.
- Run the **errpt** command periodically, at least weekly.

  The error logging system should always be active. This system is active as long as the **errdemon** is running; the **errdemon** is started automatically when the system is booted. For more information about error logging, see the “Error Logging Overview” in the *Problem Solving Guide and Reference*.
- If you are using auditing, check the output at least weekly and back up the auditing output periodically. Auditing output grows quickly, and the files should be reduced in size periodically.

Security Tasks for Adding Users

You should perform the following tasks when adding users:

1. Assign users to appropriate groups.
2. Set initial passwords.
3. Explain to users how to create acceptable passwords. Ensure that users change their initial passwords when they first log in, and ensure they follow the password guidelines.
4. Give a written statement of your security policies to new users. The statement should include:
   - the policy on unattended terminals
   - the password policy
   - directories users can safely use to store their own data

Security Tasks for Removing Users

When a user is removed from the system, perform the following tasks:

1. If the user is only being removed temporarily, consider just removing the ability of the user ID to log in to the system. For more information, see “Turning Off and On Login Access for Users” on page 5-9.
2. If the user is being removed permanently, remove all the user information. See “Removing a User” on page 5-8 for more information.
3. Recover the system key from the user.
4. Remove or reassign all the user’s files on the system. You can use the **find** command to produce a list of all files owned by a user.
5. Remove any **at** jobs the user has scheduled. A user can schedule potentially damaging programs to run long after the user is removed from the system by using the **at** command.

Trusted Computing Base

The Trusted Computing Base (TCB) is the part of the system that is responsible for enforcing the information security policies of the system. All of the computer's hardware is included in the TCB, but a person administering the system should be concerned primarily with the software components of the TCB.

TCB Overview

Many of the TCB functions are now optionally enabled at installation time. Selecting **yes** for the **Install Trusted Computing Base** option on the Installation and Settings menu enables the trusted path, trusted shell, and system integrity checking (**tcbck** command). Selecting **no** disables these features. These features can only be enabled at installation time.

The TCB software consists of:

- the kernel (operating system)
- the configuration files that control system operation
- any program that is run with the privilege or access rights to alter the kernel or the configuration files

Most system files are accessible only by the root user; however, some can also be accessed by members of an administrative group. Only the root user can alter the operating system kernel. The TCB contains the following trusted programs:

- all **setuid** root programs
- all **setgid** programs to administrative groups
- any program that is exclusively run by the root user or by a member of the system group
- any program that must be run by the administrator while on the trusted communication path (for example, the **ls** command)

In the operating system, the person who administers the system can mark trusted files as part of the Trusted Computing Base (the **chtcb** command), so that they can be clearly distinguished.

The person who administers the system must be careful to add only software that can be fully trusted to the TCB. Consider trusting software if, for example:

- You have fully tested the program.
- You have examined the program's code.
- The program is from a trusted source that has tested or examined the program.

The system administrator must determine how much trust can be given to a particular program. This determination should include considering the value of the information resources on the system in deciding how much trust is required for a program to be installed with privilege.

Checking the Trusted Computing Base

The **tcbck** command audits the security state of the Trusted Computing Base. The security of the operating system is jeopardized when the TCB files are not properly protected or when configuration files have unsafe values. The **tcbck** command audits this information by reading the **/etc/security/sysck.cfg** file. This file includes a description of all TCB files, configuration files, and trusted commands.

Note: If the **Install Trusted Computing Base** option was not selected during the initial installation, the **tcbck** command will be disabled. The command can be properly enabled only by reinstalling the system.

Using the tcbck Command

The **tcbck** command is normally used to:

- assure the proper installation of security-relevant files
- assure that the file system tree contains no files that clearly violate system security
- update, add, or delete trusted files

The **tcbck** command can be used in three ways:

- normal use
  - non–interactive at system initialization
  - with the **cron** command
- interactive use
  - useful for checking out individual files and classes of files
- paranoid use
  - store the **sysck.cfg** file offline and restore it periodically to check out the machine

Checking Trusted Files

You should run the **tcbck** command to check the installation of trusted files at system initialization. To perform this automatically and to produce a log of what was in error, add the following command to the **/etc/rc** file:

```
tcbck –y ALL
```

This causes the **tcbck** command to check the installation of each file described by the **/etc/security/sysck.cfg** file.

Checking the File System

You should run the **tcbck** command to check the file system any time you suspect the integrity of the system may have been compromised. This is done by issuing the following command:

```
tcbck –t tree
```

When the **tcbck** command is used with the *tree* parameter, all files on the system are checked for correct installation (this could take a long time). If the **tcbck** command discovers any files that are potential threats to system security, you can alter the suspected file to remove the offending attributes. In addition, the following checks are performed on all other files in the file system:

- If the file owner is **root** and the file has the **setuid** bit set, the **setuid** bit is cleared.
- If the file group is an administrative group, the file is executable, and the file has the **setgid** bit set, the **setgid** bit is cleared.
- If the file has the **tcb** attribute set, this attribute is cleared.
- If the file is a device (character or block special file), it is removed.
- If the file is an additional link to a path name described in the **/etc/security/sysck.cfg** file, the link is removed.
- If the file is an additional symbolic link to a path name described in the **/etc/security/sysck.cfg** file, the symbolic link is removed.

Note: All device entries must have been added to the **/etc/security/sysck.cfg** file prior to execution of the **tcbck** command or the system is rendered unusable. Use the **-l** option to add trusted devices to **/etc/security/sysck.cfg**.

Adding a Trusted Program

To add a specific program to the **/etc/security/sysck.cfg** file, use the following command:

```
tcbck –a PathName [attribute=value]
```

Only attributes whose values can or should not be deduced from the current state of the file need be specified on the command line. All attribute names appear in the **/etc/security/sysck.cfg** file.

For example, the following command registers a new setuid-root program named **/usr/bin/setgroups**, which has a link named **/usr/bin/getgroups**:

```
tcbck –a /usr/bin/setgroups links=/usr/bin/getgroups
```

After installing a program, you may not know which new files should be registered in the **/etc/security/sysck.cfg** file. These can be found and added with the following command:

```
tcbck –t tree
```

This command displays the name of any file that should be registered in the **/etc/security/sysck.cfg** file.

Deleting a Trusted Program

If you remove a file described in the **/etc/security/sysck.cfg** file, you should also remove the description of this file. For example, if you have deleted the **/etc/cvid** program, the following command will cause an error message to be shown:

```
tcbck –t ALL
```

The error message shown is:

```
3001–020 The file /etc/cvid was not found.
```

The description of this program can be removed with the following command:

```
tcbck –d /etc/cvid
```

Configuring the tcbck Program

The **tcbck** command reads the **/etc/security/sysck.cfg** file to determine which files to check. Each trusted program on the system should be described by a stanza in the **/etc/security/sysck.cfg** file.

Each stanza has the following attributes:

class name of a group of files

This attribute allows several files with the same class name to be checked by specifying a single argument to the **tcbck** command. More than one class can be specified, with each class being separated by a comma.

owner user ID or name of the file owner

If this does not match the file owner, the **tcbck** command sets the owner ID of the file to this value.

group group ID or name of the file's group

If this does not match the file owner, the **tcbck** command sets the owner ID of the file to this value.

mode comma-separated list of values

The allowed values are **SUID**, **SGID**, **SVTX**, and **TCB**. The file permissions must be the last value and can be specified either as an octal value or as a 9-character string. For example, either **755** or **rwxr–xr–x** are valid file permissions. If this does not match the actual file mode, the **tcbck** command applies the correct value.

links comma-separated list of path names linked to this file

If any path name in this list is not linked to the file, the **tcbck** command creates the link. If used without the *tree* parameter, the **tcbck** command prints a message that there are extra links but does not determine their names. If used with the *tree* parameter, the **tcbck** command also prints any additional path names linked to this file.

symlinks comma-separated list of path names symbolically linked to this file

If any path name in this list is not a symbolic link to the file, the **tcbck** command creates the symbolic link. If used with the *tree* argument, the **tcbck** command also prints any additional path names that are symbolic links to this file.

program comma-separated list of values

The first value is the path name of a checking program. Additional values are passed as arguments to the program when it is executed.

Note: The first argument is always one of **–y**, **–n**, **–p**, or **–t**, depending on which flag the **tcbck** command was used with.

acl text string representing the access control list for the file

It must be of the same format as the output of the **aclget** command. If this does not match the actual file ACL, the **sysck** command applies this value using the **aclput** command.

Note: The attributes **SUID**, **SGID**, and **SVTX** must match those specified for the mode, if present.

source name of a file this source file is to be copied from prior to checking

If the value is blank, and this is either a regular file, directory, or a named pipe, a new empty version of this file is created if it does not already exist. For device files, a new special file is created for the same type device.

If a stanza in the **/etc/security/sysck.cfg** file does not specify an attribute, the corresponding check is not performed.

The **tcbck** command provides a way to define and maintain a secure software configuration. The **tcbck** command also ensures that all files maintained by its database are installed correctly and have not been modified.

tcbck Checking Programs

An important aspect of the **tcbck** program is the **program** attribute, located in the **/etc/security/sysck.cfg** file. The **program** attribute lists an associated program that can check additional status. This attribute allows for more thorough and flexible checking than other attributes provide.

You can use these checking programs to check the integrity and consistency of a file's contents and its relationship with other files. Checking programs need not be bound to a particular file.

For example, assume you wrote a program, **/etc/profile**, which verifies that each user's **.profile** file is writable only by that user. The program should have the following aspects:

- owned by **root**
- member of **system** group
- has a mode of 750
- tagged as part of the TCB

You can add your program, **/etc/profile**, to the system security checker by entering the following:

```
tcbck –a /etc/profile "program=/etc/profile" class=profiles \
owner group mode
```

This command creates the following entry in the database:

```
/etc/profile:

class = profiles

owner = root

group = system

mode = TCB,rwxr–x–––

program = "/etc/profile"
```

The following **tcbck** command verifies the installation of the **/etc/profile** program and runs the program:

```
tcbck –t profiles
```

There are several requirements for **tcbck** checking programs:

- **tcbck** must accept the **–n**, **–y**, **–p**, and **–t** flags and handle these similarly to the **sysck** command.
- **tcbck** must return 0 to indicate that no errors were found and write all error messages to standard error.
- It is important to note that these programs are run with an effective user ID of 0; therefore, they are fully privileged. They should be written and inspected as setuid-root programs.

TCB Checking Programs

The operating system supplies the following TCB checking programs:

pwdck	checks the **/etc/passwd** and **/etc/security/passwd** files for internal and mutual consistency
grpck	checks the **/etc/group** and **/etc/security/group** files for internal and mutual consistency
usrck	verifies the accuracy of the user definitions in the user database files by checking the definitions for all the users or for specific users

Secure System Installation and Update

Installing or updating a program consists of importing files into the system, usually creating new directories for the program, and occasionally reconfiguring the system itself. From a security standpoint, the program may need to add user accounts, define new audit events, and assign privileges to one of the program files.

In the simplest mode, a program installation consists of installing a new subdirectory tree (from **/usr/lpp**) and possibly adding new symbolic links in the **/usr/bin** directory. However, there are two problems:

- Most real programs need to alter the system configuration and contain commands that need to be installed with some degree of administrative privilege.
- Each installation should be done under a separate access domain so that the installation procedure of one program cannot interfere with that of another.

To provide for secure program installation and update, two strategies are employed. First, privilege and access rights are delineated and limited during installation and update. This minimizes the potential for damage by untrustworthy installation packages. Second, the entire process is auditable. The auditing can be done by examining the system audit trail after installation or update of the program is complete, or auditing can be interactive. The **watch** command is provided for interactive use. The **watch** command can be used to execute a specified program and display the stream of audit records (if any) that are generated during the execution of that program.

This approach provides a great deal of flexibility in installation, while still providing a high degree of security. Even though the security is detection rather than prevention, this is still effective. Since the process is interactive, the installation of malicious programs can be halted quickly.

Guidelines for Ownership and Modes for Files

Normal User Commands

Normal user commands do not need a real owner or group since they can be executed by anyone and are not set user ID (SUID) or set group ID (SGID). If the command is expected to be run on the trusted path (for example, using the **vi**, **grep**, and **cat** commands), its TCB bit should be set. The following is an example of ownership and modes:

```
owner:  bin     r-x
group:  bin     r-x
others:         r-x
```

Administrative User Commands

Administrative user commands are executable only by root, members of an administrative group, and members who are specified in the extended access control list (ACL) entries. Since they usually perform privileged operations, some of these commands may need to be run with privilege (set user ID, or SUID). The following is an example of ownership and modes:

```
owner:  root    r-x  (possibly SUID)
group:  system  r-x  (possibly SGID)
others: (possibly extended Access Control List entries)
```

For an example of a typical administrative user command scenario, consider network files containing critical network configuration information:

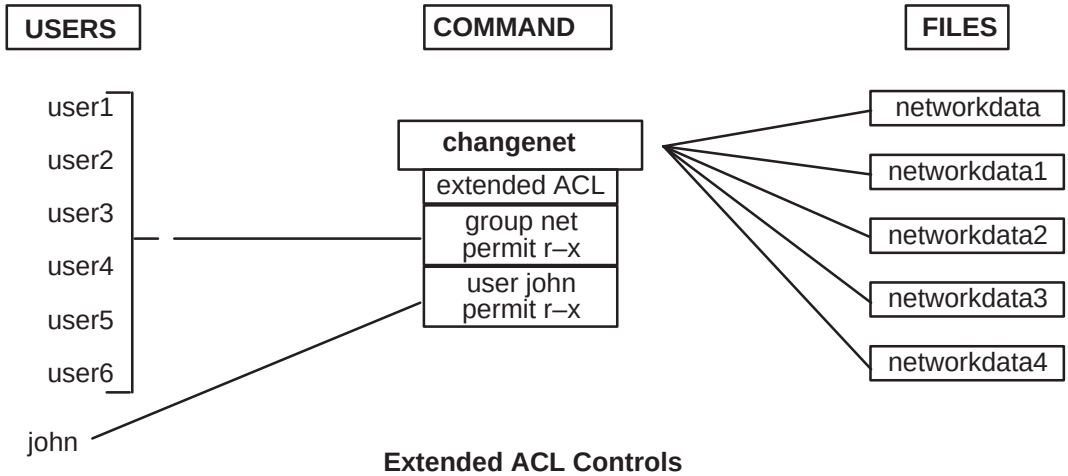
```
owner:  root            rw-
group:  netgroup        rw-
others:                 ---
```

Use the **changenet** command to modify the information in the network file.

```
owner:  root            r-x
group:  netgroup        --s
others: (extended Access Control List entry)
permit  r-x g:net
permit  r-x u:john
```

In this example, group **netgroup** is an administrative privilege group with no members. Only processes running with group **netgroup** (or root user) can read or write the network files. Running the **changenet** command, which is the set group ID (SGID) **netgroup**, gives users the ability to change the network files. Only root users, members of group `net`, and user `john` can execute this command.

The Extended ACL Controls figure shows how the extended ACL is associated with the command instead of with the data files themselves.



Extended ACL Controls

The **changenet** command is the gateway to the network configuration files. The extended Access Control List on the **changenet** command is the guard that allows only certain users through.

This example assumes that there are more data files than programs that operate on them. If the reverse were true, it might make more sense to have ACLs on the data files.

Configuration Files

The following example shows the **admin** group as an administrative privilege group with no members. The exact name of the **admin** group depends on what type of configuration file the group applies to (for example, auditing, authentication, and mail).

```
owner:  root    rw–
group:  admin   rw–
others:         r––
mode:           TCB
```

In general, most configuration files should be readable by all users, with the exception of auditing and authentication data.

Device Special Files

In general, devices should not be readable or writable by normal users. Exceptions to this rule are terminal devices that should be writable so users can send messages to each other and floppy disks that should be both readable and writable so users can transfer files.

Trusted Communication Path

The operating system trusted communication path allows for secure communication between users and the Trusted Computing Base. Users start the trusted communication path by pressing the secure attention key (SAK). This allows only trusted processes to access the user's terminal. A trusted communication path is used whenever the user must enter data that must not be compromised (a password, for example). A trusted

communication path can also be used by the person who administers the system to establish a secure environment for administration.

Note: If the **Install Trusted Computing Base** option was not selected during the initial installation, the trusted communications path will be disabled. The path can be properly enabled only by reinstalling the system.

The trusted communication path is based on:

- a trusted command interpreter (**tsh** command) that only executes commands that are marked as being a member of the Trusted Computing Base
- restricting access to a terminal to trusted programs
- a reserved key sequence, called the secure attention key (SAK), which allows the user to request a trusted communication path

After the SAK has been pressed, the **init** command starts either the **getty** command or the **shell** command that:

- changes the owner and mode of the terminal so that only processes run by a user can open the terminal
- issues a **frevoke** subroutine to invalidate all previous **open** calls of the terminal

This assures that only the current **getty** or **shell** command has access to the terminal.

Trusted Command Interpreter

The **getty** command runs the **shell** command in response to a SAK only if the user was logged in to this terminal. This command establishes the terminal modes and runs the trusted shell (**tsh** command).

The **tsh** command provides a subset of the functions of the normal shell (Korn shell). The trusted shell executes only trusted programs (for example, programs tagged with the TCB bit). The built-in **shell** command allows the user to drop the trusted communication path and execute the user login shell.

Restricting Access to a Terminal

As mentioned previously, the **getty** and **shell** commands change the owner and mode of a terminal to prevent untrusted programs from accessing the terminal. The operating system provides a way to configure exclusive terminal access.

Using the Trusted Communication Path

A trusted communication path is established by pressing the SAK reserved key sequence (Ctrl-X, Ctrl-R).

A trusted communication path should be established under the following conditions:

- when logging in to the system

  After you press the SAK:

  – If a new login screen scrolls up, you have a secure path.

  – If the trusted shell prompt appears, the initial login screen was an unauthorized program that may have been trying to steal your password. You should find out who is currently using this terminal with the **who** command and then log off.

- when you want the command you enter to result in a trusted program running

  Some examples of this include:

  – running as root user

  You should run as root user only after establishing a trusted communication path. This ensures that no untrusted programs will be run with root user authority.

  – running the **su**, **passwd**, and **newgrp** commands

  You should only run these commands after establishing a trusted communication path.

Warning: Use caution when using SAK; it kills all processes that attempt to access the terminal and any links to it (for example, **/dev/console** can be linked to **/dev/tty0**).

Configuring the Secure Attention Key

Each terminal can be independently configured so that pressing SAK at that terminal creates a trusted communication path. This is specified by the **sak_enabled** attribute in **/etc/security/login.cfg** file. If the value of this attribute is **true**, recognition of the SAK is enabled.

If a port is to be used for communications, (for example, by the **uucp** command), the specific port used should have the following line in its stanza of the **/etc/security/login.cfg** file:

```
sak_enabled = false
```

This line or no entry disables the SAK for that terminal.

To enable SAK on a terminal, add the following line to the stanza for that terminal:

```
sak_enabled = true
```

Managing Protected Resources with Access Control

Access control also involves managing protected resources using the **setuid** and **setgid** programs and hard-copy labeling. The operating system supports several types of information resources, or objects. These objects allow user processes to store or communicate information.

The most important types of objects are:

- files and directories (used for information storage)
- named pipes, message queues, shared memory segments, and semaphores (used for information transfer between processes)

Each object has an associated owner, group, and mode. The mode defines access permissions for the owner, group, and other users.

The following are the direct access control attributes for the different types of objects:

Owner The owner of a specific object controls its discretionary access attributes. The owner's attributes are set to the creating process's effective user ID. For file system objects, the direct access control attributes for an owner cannot be changed without root privilege.

For System V Interprocess Communication (SVIPC) objects, either the creator or owner can change the owner. SVIPC objects have an associated creator that has all the rights of the owner (including access authorization). However, the creator cannot be changed, even with root privilege.

Group SVIPC objects are initialized to the effective group ID of the creating process. For file system objects, the direct access control attributes are initialized to either the effective group ID of the creating process or the group ID of the parent directory (this is determined by the group inheritance flag of the parent directory).

The owner of an object can change the group; the new group must be either the effective group ID of the creating process or the group ID of the parent directory. The owner of an object can change the group; the new group must be either the effective group or in the supplementary group ID of the owner's current process. (As above, SVIPC objects have an associated creating group that cannot be changed and share the access authorization of the object group.)

For more information about access control lists, see "Access Control Lists" in the *System User's Guide: Operating System and Devices*.

Using setuid and setgid Programs

The permission bits mechanism allows effective access control for resources in most situations. But for more precise access control, the operating system provides **setuid** and **setgid** programs.

Most programs execute with the user and group access rights of the user who invoked them. Program owners can associate the access rights of the user who invoked them by making the program a **setuid** or **setgid** program; that is, a program with the **setuid** or **setgid** bit set in its permissions field. When that program is executed by a process, the process acquires the access rights of the owner of the program. A **setuid** program executes with the access rights of its owner, while a **setgid** program has the access rights of its group and both bits can be set according to the permission mechanism.

Although the process is assigned the additional access rights, these rights are controlled by the program bearing the rights. Thus, the **setuid** and **setgid** programs allow for user-programmed access controls in which access rights are granted indirectly. The program acts as a trusted subsystem, guarding the user's access rights.

Although these programs can be used with great effectiveness, there is a security risk if they are not designed carefully. In particular, the program must never return control to the user while it still has the access rights of its owner, because this would allow a user to make unrestricted use of the owner's rights.

Note: For security reasons, the operating system does not support **setuid** or **setgid** calls within a shell script.

Administrative Access Rights

The operating system provides privileged access rights for system administration. System privilege is based on user and group IDs. Users with effective user or group IDs of 0 are recognized as privileged.

Processes with effective user IDs of 0 are known as root user processes and can:

- read or write any object
- call any system function
- perform certain subsystem control operations by executing **setuid-root** programs

You can manage the system using two types of privilege: the **su** command privilege and **setuid-root** program privilege. The **su** command allows all programs you invoke to function as root user processes, and **su** is a flexible way to manage the system, but it is not very secure.

Making a program into a **setuid-root** program means the program is a root user-owned program with the **setuid** bit set. A **setuid-root** program provides administrative functions that ordinary users can perform without compromising security; the privilege is encapsulated in the program rather than granted directly to the user.

It can be difficult to encapsulate all necessary administrative functions in **setuid-root** programs, but it provides more security to system managers.

Auditing Overview

The auditing subsystem provides the system administrator with the means to record security-relevant information, which can be analyzed to detect potential and actual violations of the system security policy. The auditing subsystem has three functions: event detection, information collection, and information processing. Each of these functions can be configured by the system administrator.

Event detection is distributed throughout the Trusted Computing Base (TCB), both in the kernel (supervisor state code) and the trusted programs (user state code). An auditable event is any security-relevant occurrence in the system. A security-relevant occurrence is any change to the security state of the system, any attempted or actual violation of the system access control or accountability security policies, or both. The programs and kernel modules that detect auditable events are responsible for reporting these events to the system audit logger, which runs as part of the kernel and can be accessed either with a subroutine (for trusted program auditing) or within a kernel procedure call (for supervisor state auditing). The information reported should include the name of the auditable event, the success or failure of the event, and any additional event-specific information that would be relevant to security auditing.

Event detection configuration consists of turning event detection on or off, either at the global (system) level or at the local (process) level. To control event detection at the global level, use the **audit** command to enable or disable the audit subsystem. To control event detection at the local level, you can audit selected users for groups of audit events (audit classes).

Information collection encompasses logging the selected auditable events. This function is performed by the kernel audit logger, which provides both an SVC (subroutine) and an intra-kernel procedure call interface that records auditable events.

The audit logger is responsible for constructing the complete audit record, consisting of the audit header, which contains information common to all events (such as the name of the event, the user responsible, the time and return status of the event), and the audit trail, which contains event-specific information. The audit logger appends each successive record to the kernel audit trail, which can be written in either (or both) of two modes:

BIN mode The trail is written into alternating files, providing for safety and long-term storage.

STREAM mode The trail is written to a circular buffer that is read synchronously through an audit pseudo-device. STREAM mode offers immediate response.

Information collection can be configured at both the front end (event recording) and at the back end (kernel trail processing). Event recording is selectable on a per-user basis. Each user has a defined set of audit events which are actually logged in the kernel trail when they occur. At the back end, the modes are individually configurable, so that the administrator can employ the back-end processing best suited for a particular environment. In addition, BIN mode auditing can be configured to shut down the system in the event of failure.

The operating system provides several options for processing the kernel audit trail. The BIN mode trail can be compressed, filtered or formatted for output, or any reasonable combination of these prior to archival storage of the audit trail, if any. Compression is done through Huffman encoding. Filtering is done with standard query language (SQL)-like audit record selection (using the **auditselect** command) and provides for both selective viewing and selective retention of the audit trail. Formatting of audit trail records can be used to examine the audit trail, to generate periodic security reports, and to print a paper audit trail. The STREAM mode audit trail can be monitored in real time to provide immediate threat

monitoring capability. Configuration of these options is handled by separate programs that can be invoked as daemon processes to filter either BIN or STREAM mode trails, although some of the filter programs are more naturally suited to one mode or the other.

Event Selection

The set of auditable events on the system defines which occurrences can actually be audited and the granularity of the auditing provided. The auditable events must cover the security-relevant events on the system, as defined previously. The level of detail you use for auditable event definition must tread a fine line between insufficient detail, leading to excessive information collection, and too much detail, making it difficult for the administrator to logically understand the selected information. The definition of events takes advantage of similarities in detected events. For the purpose of this discussion, a detected event is any single instance of an auditable event; for instance, a given event may be detected in various places. The underlying principle is that detected events with similar security properties are selected as the same auditable event. The following list shows an event classification:

Security Policy Events
- Subject Events
  - process creation
  - process deletion
  - setting subject security attributes: user IDs, group IDs
  - process group, control terminal
- Object Events
  - object creation
  - object deletion
  - object open (including processes as objects)
  - object close (including processes as objects)
  - setting object security attributes: owner, group, ACL
- Import/Export Events
  - importing or exporting an object
- Accountability Events
  - adding a user, changing user attributes in the password database
  - adding a group, changing group attributes in the group database
  - user login
  - user logoff
  - changing user authentication information
  - trusted path terminal configuration
  - authentication configuration
  - auditing administration: selecting events and audit trails, switching on/off, defining user auditing classes
- General System Administration Events
  - use of privilege
  - file system configuration
  - device definition and configuration
  - system configuration parameter definition
  - normal system IPL and shutdown
  - RAS configuration
  - other system configuration
- Security Violations (potential)
  - access permission refusals
  - privilege failures
  - diagnostically detected faults and system errors
  - (attempted) alteration of the TCB.

The auditing subsystem has a global state variable that indicates whether the auditing subsystem is on or off. In addition, each process has a local state variable that indicates whether the auditing subsystem should record information about this process. Both of these

variables determine whether events are detected by the Trusted Computing Base (TCB) modules and programs. Turning TCB auditing off for a specific process allows that process to do its own auditing and not to bypass the system accountability policy. Permitting a trusted program to audit itself allows for more efficient and effective collection of information.

Information Collection

Information collection addresses event selection and kernel audit trail modes. It is done by a kernel routine that provides interfaces to log information and configuration interfaces. The information is used by the TCB components that detect auditable events, and the configuration interfaces are used by the auditing subsystem to control the audit logger routine.

Audit Logging

Auditable events are logged with one of two interfaces, the user state and supervisor state. The user state part of the TCB uses the **auditlog** or **auditwrite** subroutine, while the supervisor state portion of the TCB uses a set of kernel procedure calls.

For each record, the audit event logger prefixes an audit header to the event-specific information. This header identifies the user and process for which this event is being audited, as well as the time of the event. The code that detects the event supplies the event type and return code or status and, optionally, additional event-specific information (the event tail). Event-specific information consists of object names (for example, files refused access or tty used in failed login attempts), subroutine parameters, and other modified information.

Events are defined symbolically, rather than numerically. This lessens the chances of name collisions, without using an event registration scheme. Also, since subroutines are auditable, the extendable kernel definition, with no fixed SVC numbers, makes it difficult to record events by number, since the number mapping would have to be revised and logged every time the kernel interface was extended or redefined.

Audit Record Format

The audit records consist of a common header, followed by audit trails peculiar to the audit event of the record. The structures for the headers are defined in the **/usr/include/sys/audit.h** file. The format of the information in the audit trails is peculiar to each base event and is shown in the **/etc/security/audit/events** file.

The information in the audit header is generally collected by the logging routine to ensure its accuracy, while the information in the audit trails is supplied by the code that detects the event. The audit logger has no knowledge of the structure or semantics of the audit trails. For example, when the **login** command detects a failed login, it records the specific event with the terminal on which it occurred and writes the record into the audit trail using the **auditlog** subroutine. The audit logger kernel component records the subject-specific information (user IDs, process IDs, time) in a header and appends this to the other information. The caller supplies only the event name and result fields in the header.

Logger Configuration

The audit logger is responsible for constructing the complete audit record. You must select the audit events that you want to be logged.

Event Selection

There are two different types of audit event selection: per process and per object.

Per-Process Auditing

To select process events with reasonable efficiency and usability, the operating system allows the system administrator to define audit classes. An audit class is a subset of the base auditing events in the system. Auditing classes provide for convenient logical groupings of the base auditing events.

For each user on the system, the system administrator defines a set of audit classes that determines the base events that could be recorded for that user. Each process run by the user is tagged with its audit classes.

Per-Object Auditing

The operating system provides for the auditing of object accesses by name, that is, the auditing of specific objects (normally files). Most objects are not that interesting from a security perspective. By-name object auditing prevents having to cover all object accesses to audit the few pertinent objects. In addition, the auditing mode can be specified, so that only accesses of the specified mode (read/write/execute) and results (success/failure) are recorded.

Kernel Audit Trail Modes

Kernel logging can be set to BIN or STREAM modes to define where the kernel audit trail is to be written. If the BIN mode is used, the kernel audit logger must be given (prior to audit startup) at least one file descriptor to which records are to be appended.

BIN mode consists of writing the audit records into alternating files. At auditing startup, the kernel is passed two file descriptors and an advisory maximum bin size. It suspends the calling process and starts writing audit records into the first file descriptor. When the size of the first bin reaches the maximum bin size, and if the second file descriptor is valid, it switches to the second bin and reactivates the calling process. It keeps writing into the second bin until it is called again with another valid file descriptor. If at that point the second bin is full, it switches back to the first bin, and the calling process returns immediately. Otherwise, the calling process is suspended, and the kernel continues writing records into the second bin until it is full. Processing continues this way until auditing is turned off.

The STREAM mode is much simpler than the BIN mode. The kernel writes records into a circular buffer. When the kernel reaches the end of the buffer, it simply wraps to the beginning. Processes read the information through a pseudo-device called **/dev/audit**. When a process opens this device, a new channel is created for that process. Optionally, the events to be read on the channel can be specified as a list of audit classes.

The main purpose of this mode is to allow for timely reading of the audit trail, which is desirable for real-time threat monitoring. Another use is to create a paper trail that is written immediately, preventing any possible tampering with the audit trail, as is possible if the trail is stored on some writable media.

Setting Up Auditing

The following is an overview of the steps you must take to set up an auditing subsystem. Refer to the configuration files noted in these steps for more specific information.

1. Select system activities (events) to audit from the list in the **/etc/security/audit/events** file or edit the file to add a new event.
   - You can only add an event to this file if you have included code to log that event in an application program (using the **auditwrite** or **auditlog** subroutine) or in a kernel extension (using the **audit_svcstart**, **audit_svcbcopy**, and **audit_svcfinis** kernel services).
   - Ensure that formatting instructions for any new audit events are included in the **/etc/security/audit/events** file. These specifications enable the **auditpr** command to write an audit tail when it formats audit records.
2. Group your selected audit events into sets of similar items called audit classes. Define these audit classes in the classes stanza of the **/etc/security/audit/config** file.
3. Assign the audit classes to the individual users and assign audit events to the files (objects) that you want to audit, as follows:
   - To assign audit classes to an individual user, add a line to the users stanza of the **/etc/security/audit/config** file. You can use the **chuser** command to assign audit classes to a user.
   - To assign audit events to an object (data or executable file), add a stanza for that file to the **/etc/security/audit/objects** file.
4. Configure the type of data collection that you want, using BIN collection, STREAM collection, or both methods:
   - *To configure BIN collection*:
     - Edit the **start** stanza in the **/etc/security/audit/config** file to enable BIN collection.
     - Edit the **binmode** stanza in the **/etc/security/audit/config** file to configure the bins and trail, and specify the path of the file containing the binmode back-end processing commands. The default file for back-end commands is the **/etc/security/audit/bincmds** file.
     - Include the shell commands that will process the audit bins in an audit pipe in the **/etc/security/audit/bincmds** file.
   - *To configure STREAM collection*:
     - Edit the **start** stanza in the **/etc/security/audit/config** file to enable STREAM collection.
     - Edit the **streammode** stanza in the **/etc/security/audit/config** file to specify the path to the file containing the **streammode** processing commands. The default file containing this information is the **/etc/security/audit/streamcmds** file.
     - Include the shell commands that will process the stream records in an audit pipe in the **/etc/security/audit/streamcmds** file.
5. When you have finished making any necessary changes to the configuration files, you are ready to enable the audit subsystem using the **audit** command.

Selecting Audit Events

The purpose of an audit is to detect activities that may compromise the security of your system. When performed by an unauthorized user, the following activities violate system security and are candidates for an audit:

- engaging in activities in the Trusted Computing Base
- authenticating users
- accessing the system
- changing the configuration of the system
- circumventing the auditing system
- initializing the system
- installing programs
- modifying accounts
- transferring information into or out of the system

To audit an activity, you must identify the command or process that initiates the audit event and ensure that the event is listed in the **/etc/security/audit/events** file for your system. Then you must add the event either to an appropriate class in the file **/etc/security/audit/config**, or to an object stanza in the **/etc/security/audit/objects** file. See the **/etc/security/audit/events** file on your system for the list of audit events and trail formatting instructions. See the **auditpr** command for a description of how audit event formats are written and used.

Once you have selected the events to audit, you need to combine similar events into audit classes, as described in the section on selecting audit classes. Audit classes are then assigned to users.

Selecting Audit Classes

You can facilitate the assignment of audit events to users by combining similar events into sets called audit classes. These audit classes are defined in the classes stanza of the **/etc/security/audit/config** file.

Some typical audit classes might be:

general	General events alter the state of the system and change user authentication You should audit attempts to circumvent system access controls.
system	Events in the system group modify user and group accounts and install programs.
init	Events in the init group are generated by the **init** program and its immediate descendants, the **login** and **cron** programs.

An example of a stanza in the **/etc/security/audit/config** file follows:

```
classes:
general = USER_SU,PASSWORD_Change,FILE_Unlink,
    FILE_Link,FILE_Rename
system = USER_Change,GROUP_Change,USER_Create,
    GROUP_Create
init = USER_Login,USER_Logout
```

Selecting an Audit Data Collection Method

Your selection of a data collection method depends on how you intend to use the audit data. If you need long-term storage of a large amount of data, you should select bin collection. If you want to process the data as it is collected, select stream collection. If you need both long-term storage and immediate processing, select both methods.

Bin collection Bin collection lets you store a large audit trail for a long time. Audit records are written to a file that serves as a temporary bin. After the file is filled, the data is processed by the **auditbin** daemon, and records are written to an audit trail file for storage.

Stream collection

Stream collection lets you process audit data as it is collected. Audit records are written into a circular buffer within the kernel, and are retrieved by reading **/dev/audit**. The audit records can be displayed, printed to provide a paper audit trail, or converted into bin records by the **auditcat** command.

Chapter 11. Managing the InfoExplorer Program

This chapter contains the following types of information about the InfoExplorer program:

- customizing the InfoExplorer program
- description of the InfoExplorer information bases
- managing InfoExplorer public notes
- accessing InfoExplorer from CD–ROM
- removing InfoExplorer information bases
- changing InfoExplorer languages
- identifying the executable version of the graphical and ASCII versions of InfoExplorer
- creating InfoExplorer public notes
- transferring InfoExplorer bookmarks from one user to another

Customizing the InfoExplorer Program

The InfoExplorer program can be set up to access the library from either a CD-ROM or a fixed disk. Accessing the library from a fixed disk can improve performance, but requires additional storage space due to the size of the information bases.

To customize the InfoExplorer program for your system, you can set up public notes (see page 11-4) to give users information that is specific to your installation or your information needs. You can also set up bookmark lists and history files and transfer them to other users to provide tutorials or similar ordered lists of information.

Understanding the InfoExplorer Information Bases

The InfoExplorer library and code are located in the **/usr/lpp/info** directory. In this directory are different subdirectories that contain executables, information bases, fonts, and public note storage. Following is a listing of these directories:

bin contains executables for ASCII and graphics InfoExplorer tools

The **mergenote** command, used to combine groups of note files into a single file, is also located in this directory.

data contains the **ispaths** file that describes the installed information bases and some system definition files, and provides storage location for public notes

See “Creating InfoExplorer Public Notes” on page 11-12.

data/JP contains definition files for MBCS Japanese language environment

X11fonts/JP contains Japanese fonts used by the InfoExplorer window interface for MBCS Japanese language environment

notes contains system notes if any are installed

lib/*Language* contains installed information bases for the national language specified by the *Language* directory name

The directory name is based on the installed language for the system. For example, on a French Canadian system the directory name would be **fr_CF**. The default is the name **en_US** on U.S. English systems.

lib/Language/*library*
contains additional library subdirectories within a library directory

On a system with multiple languages installed, more than one *Language* directory can exist in the **/usr/lpp/info/lib** directory. For example, on a system that uses German and French, the information base for German can be installed in the **/usr/lpp/info/lib/de_DE** directory, while the information base for French is installed in the **/usr/lpp/info/lib/fr_FR** directory. Users can use either language by changing environment variable settings for **LANG**, **INFOLANG**, or **INFOLOCALE**. For more information, see “Changing InfoExplorer Languages” on page 11-9.

Other files that are not in **/usr/lpp/info** include**:**

/usr/bin/info contains the shell script that determines whether to invoke the ASCII or graphical version of InfoExplorer

/usr/lib/x11/app-defaults/Info_gr
contains the application defaults file that contains system resource definitions

Information Shipped with Licensed Programs

Subsets of the information on the CD-ROM are supplied at no charge with certain licensed programs. See the licensed programs with which information is shipped, and the section on hypertext information base library contents in the *Documentation Overview*.

Note: The ASCII version of the InfoExplorer program does not contain graphical artwork.

Managing InfoExplorer Public Notes

Public notes can be read by any hypertext user. By default, InfoExplorer notes are private, accessible only to the user who created them. These private notes are saved in the user's **$HOME/info** and **$HOME/info/<library>/notes** directories.

Users with write access to the **/usr/lpp/info/data** directory can create public notes, which are stored in this directory, by converting their private notes to public notes with the **mergenote** command.

Accessing InfoExplorer from CD-ROM

The first time you access InfoExplorer databases from your CD-ROM, you must:

- create a CD-ROM file system
- mount the CD-ROM file system
- run the **linkinfocd** script

Note: You can also install databases from the CD-ROM. Some databases on this CD-ROM may have already been installed with the operating system or other licensed products. Run the **lslpp** command or SMIT to get a list of the database packages already installed on your system.

The installation application that you use (SMIT or one of the VSM applications) will create a temporary mount point for the CD-ROM.

Prerequisites

You must have root user authority or be a member of the system group to create and mount the CD-ROM file system and run the **linkinfocd** script.

Create a CD-ROM File System

1. Insert the Hypertext Information Base Library CD-ROM.
2. Use the **smit crcdrfs** fast path to create the CD-ROM file system. The **Add a CDROM File System** menu appears.
3. List the devices available for the field:

   `DEVICE name`

   The DEVICE NAME overlay appears over the previous screen.
4. Specify the available CD-ROM device you plan to use.
5. Highlight the field:

   `MOUNT POINT`
6. Type the following, but do not confirm the value until you get to step 9.

   `/infocd`
7. Highlight the field:

   `Mount AUTOMATICALLY at system restart?`
8. Select one of the following choices:

   a. To mount InfoExplorer every time the system starts, press the Tab key to specify **yes.**

   b. To mount InfoExplorer manually, use the default value **no**.
9. Confirm your choices after you have completed making changes to the entry fields.
10. Exit SMIT.

Mount the CD-ROM File System

Mount your CD-ROM to the file system you created by following the steps below:

1. Enter the SMIT fast path at the system prompt:

   ```
   smit mountfs
   ```

   The **Mount a File System** menu appears.

2. Highlight the field:

   ```
   FILE SYSTEM name
   ```

3. List the file system names. The FILE SYSTEM name overlay appears over the previous menu.

4. Select a line similar to the following:

   ```
   /dev/cdx /infocd cdrfs
   ```

   where *x* is the number of your CD-ROM drive.

5. Press Enter.

6. The system always mounts the CD-ROM as a read-only file system. You can use the tab key to select **yes** or **no** in the field:

   ```
   Mount as READ ONLY file system
   ```

7. Confirm your choice to mount the CD-ROM file system.

8. Exit SMIT when the field

   ```
   Command: status
   ```

   changes to **OK**.

The InfoExplorer databases are now mounted and ready to be accessed from the CD-ROM.

Notes:

1. If the CD-ROM is ejected from the system while it is still mounted, the connection is broken and you cannot access the information. To remove the CD-ROM from the system, unmount that file system using the **unmount** command before ejecting the CD-ROM. To access the CD-ROM again, you must remount the CD-ROM file system, using the **mount** or **smit** command.

2. You can keep copies of the information bases on your fixed disk, in case your CD-ROM becomes inaccessible, or you can delete them. For more information, see “Removing InfoExplorer Information Bases” on page 11-8.

Run the linkinfocd Script

The **linkinfocd** script links the database subdirectories from the **/infocd** CD-ROM file system to the **/usr/lpp/info/lib/en_US/aix41** directory. Each database subdirectory is linked individually, just as each database can be installed separately. This allows you to have databases installed on your fixed disk drive, databases linked from a mounted CD-ROM, or a combination of both. The script also links the **ispaths** file from the **/infocd** CD-ROM file system to the **/usr/lpp/info/data** directory.

The **linkinfocd** script does the following:

- checks to see if InfoExplorer (**/usr/lpp/info**) is installed on the system

  If **/usr/lpp/info** does not exist, the script exits.

- checks to see if the **/usr/lpp/info/lib/en_US/aix41** directory exists and creates it if it is not found
- checks to see if the database subdirectories exist

  If the database subdirectory name is found in **/usr/lpp/info/lib/en_US/aix41** as:

  – a link from the CD-ROM, the script prints a message that the database is already linked from the CD-ROM.

  – a link from elsewhere, the script forces the link from the CD-ROM over the existing link.

  – a directory, the script prints a message that you must deinstall that database if you want to link it from the mounted CD-ROM.

  If the database subdirectory is not found in **/usr/lpp/info/lib/en_US/aix41**, the script links that database subdirectory from **/infocd/usr/lpp/info/lib/en_US/aix41** to **/usr/lpp/info/lib/en_US/aix41**.
- checks to see if the **/usr/lpp/info/data/ispaths** file exists
- If the **ispaths** file name is found as:

  – a link from the CD-ROM, the script prints a message that the **ispaths** file is already linked from the CD-ROM.

  – a link from elsewhere, the script copies the linked **ispaths** file to **ispaths.linked** and links the **ispaths** file from the CD-ROM into **/usr/lpp/info/data**.

  – a file, the script copies the existing **ispaths** to **ispaths.orig** and links the **ispaths** file from the CD-ROM into **/usr/lpp/info/data**.

  If no **/usr/lpp/info/data/ispaths** file is found, the script links the **ispaths** file from the CD-ROM into **/usr/lpp/info/data**.

To run the **linkinfocd** script, enter:

```
/infocd/linkinfocd
```

Removing InfoExplorer Information Bases

The method you use to remove an information base depends on whether it is installed on the fixed disk from an installation media or linked from the mounted hypertext CD–ROM.

Prerequisites

You must have write access to the **/usr/lpp/info/lib/***$LANG* directory. *$LANG* refers to the language you are using for the InfoExplorer program.

Remove Information Bases Installed on Fixed Disk

To determine what database packages are installed on your system, run the **lslpp** command or use SMIT to generate a list. To remove an information base that is installed on the fixed disk, you must remove the corresponding software option. Refer to the “Maintaining Optional Software” chapter in the *Operating System Installation* guide.

Remove Information Bases Linked from CD-ROM

To determine what databases are linked from the hypertext CD-ROM, use the **cd** command to change to the **/infocd/usr/lpp/info/lib/en_US/aix41** directory and run the **ls –l** command. Database directories that are linked from the hypertext CD-ROM will have symbolic links to **/infocd** listed.

Use the **rm** command to delete the symbolic links for any unneeded information bases. To delete the links, use the following form of the **rm** command:

```
rm –f /usr/lpp/info/lib/$LANG/InformationBaseName
```

For example, to remove the symbolic link for the **files** information base (*AIX Version 4.1 Files Reference*) from a system where the language is U.S. English, enter:

```
rm –f /usr/lpp/info/lib/en_US/aix41/files
```

Note: Information bases cannot be deleted from the CD-ROM. For performance enhancements, you may decide to install frequently accessed information bases onto the fixed disk. Refer to the“Installing and Removing Optional Software and Service Updates” chapter in the *Operating System Installation* guide.

Changing InfoExplorer Languages

On a system with multiple languages installed, each language information base has its own directory in the **/usr/lpp/info/lib** directory. For example, the information base for German is installed in the **/usr/lpp/info/lib/de_DE** directory, while the information bases for U.S. English are installed in the **/usr/lpp/info/lib/en_US** directory.

InfoExplorer determines what languages to use for database content separately from the language to use for messages (like menu bar entries or button titles). The language used for messages is determined by the value of the **LANG** or the **LC_MESSAGES** environment variable. If **LC_MESSAGES** is set, that value is used; otherwise, the value of **LANG** is used to determine which messages to use.

InfoExplorer uses several methods for determining the language to use for database content. The precedence is as follows:

1. If the **INFOLANG** environment variable is set, then InfoExplorer attempts to read databases from the **/usr/lpp/info/lib/<$INFOLANG>** directory.
2. If no libraries are found there or the **INFOLANG** environment variable is not set, then InfoExplorer uses the **INFOLOCALE** environment variable. You can specify a list of locales in the **INFOLOCALE** environment variable by separating each locale with a colon. The first locale specified is the first language that InfoExplorer attempts to read. InfoExplorer continues trying to read databases based on the locales specified in the **INFOLOCALE** environment variable until a match is found.
3. If no libraries are found using the **INFOLOCALE** environment variable, the environment variable **LC_MESSAGES** is used.
4. If no libraries are found using the **LC_MESSAGES**, the **LANG** environment variable is used.
5. If no libraries are found, then InfoExplorer defaults to using the libraries installed in **/usr/lpp/info/lib/en_US**.

Identifying the Executable Version (InfoExplorer Graphical Interface)

Procedure

1. Select the **Info** menu option in the menu bar.
2. Hold down the left mouse button to display the Info pull-down menu.
3. Move the pointer down the Info pull-down menu to the **Copyright** option.
4. To display InfoExplorer version and copyright information, release the left button when the **Copyright** option is enclosed in a box.

   Note: To release the Info menu without selecting an item, move the pointer off the menu and release the left button.

Identifying the InfoExplorer Version

Check what version of InfoExplorer by entering:

```
info
```

The version number appears in the introduction window.

Listing the InfoExplorer Version Installed

Use the **lslpp** command to get the Version Release Modification and Fix (VRMF) number of the image installed by entering:

```
lslpp -l x11.info.rte
```

Identifying the Executable Version (InfoExplorer ASCII)

Procedure

1. Press the Ctrl-O key sequence to activate the menu bar.

   The **info** option is the default option highlighted.

2. Press the Enter key to display the **info** pull-down menu options.
3. Use the Down Arrow key to highlight the **Copyright** option.
4. Press the Enter key to display the copyright and edition information for the databases.

Identifying the InfoExplorer Version

Check what version of InfoExplorer by entering:

```
info
```

The version number appears at the bottom of the screen while the InfoExplorer session is initializing.

Listing the InfoExplorer Version Installed

Use the **lslpp** command to get the VRMF number of the image installed by entering:

```
lslpp -l bos.info.rte
```

Creating InfoExplorer Public Notes

Public notes can be read by any hypertext user. Public notes are created by merging and relocating private notes files.

Prerequisites

- You must have write access to the **/usr/lpp/info/data** directory.
- You must create and save private notes to a file in the InfoExplorer window or ASCII interface.

Procedure

Use one of the following methods to create public notes:

- For the default InfoExplorer library, use the **mergenote** command to merge private notes files into a single notes file. Designate the **/usr/lpp/info/data** directory as the location of the new public notes list.
- For public libraries other than the default, private notes are stored in the directory **$HOME/info/***LibraryName*. Use the **mergenote** command to merge private notes files into a single notes file. Designate the **/usr/lpp/info/data/***LibraryName* directory as the location of the new public notes list.

Notes:

1. Private notes and note lists are saved in the user's **$HOME/info** and **$HOME/info/***LibraryName***/notes** directories.
2. Public notes for the default InfoExplorer library should be placed in the **/usr/lpp/info/data** directory.

Bookmarks created by one user can be copied for access by other hypertext users. Bookmark files are saved in the user's **$HOME/info** or **$HOME/info/***LibraryName* directory with a **.bmk** extension.

Transferring InfoExplorer Bookmarks from One User to Another

Bookmarks created by one user can be copied for access by other hypertext users. Bookmark files are saved in the user's **$HOME/info** or **$HOME/info/***LibraryName* directory with a **.bmk** extension.

Prerequisites

You must have read and write access to the users' **$HOME/info** directories.

Procedure

1. Use the **cp** command to copy a bookmark file from one user to another.

   For example, to copy the bookmark file `review.bmk` in user `sharon`'s **$HOME** directory to user `donna`'s **$HOME** directory, enter:

   ```
   cd /home/sharon/info
   cp review.bmk /home/donna/info
   ```

2. In the InfoExplorer program, reset the new user's default bookmark file using the Defaults Editor window. For more information about setting the default bookmark file, see the procedures on "How to Set Defaults (InfoExplorer Windows)" or "How to Set Defaults (InfoExplorer ASCII)" in the *System User's Guide: Operating System and Devices*.

Chapter 12. Managing the AIX Common Desktop Environment

AIX includes the new AIX Common Desktop Environment (CDE) 1.0. Help volumes, InfoExplorer information, and hardcopy manuals may refer to the desktop as the AIX Common Desktop Environment, the AIXwindows desktop, the CDE desktop, AIX CDE 1.0, or simply, the desktop.

With the AIX Common Desktop Environment, you can access networked devices and tools without having to be aware of their location. You can exchange data across applications by simply dragging and dropping objects.

System administrators will find many tasks that previously required complex command line syntax can now be done more easily and similarly from platform to platform. They can also maximize their investment in existing hardware and software by configuring centrally and distributing applications to users. They can centrally manage the security, availability, and interoperability of applications for the users they support.

Starting and Stopping the AIX Common Desktop Environment

You can set up the system so that the AIX Common Desktop Environment comes up automatically when you start the system, or you can start AIX Common Desktop Environment manually. You must log in as root to perform each of these tasks.

Enabling and Disabling Desktop Autostart

You may find it more convenient to set up your system to start the AIX Common Desktop Environment automatically when the system is turned on. You can do this from a command line or by using the System Management Interface Tool (SMIT). You can perform each of these tasks in one of three ways:

- from the command line
- with SMIT
- with SMIT fast path

Prerequisite

You must have root user authority to enable or disable desktop autostart.

Enabling at the Command Line

1. At the command line, enter:

   ```
   dtconfig –e
   ```

2. Restart the machine.

Enabling With SMIT

1. Enter the SMIT fast path:

   ```
   /usr/bin/smit dtconfig
   ```

2. Select **CDE Environment**.
3. Confirm your choice and exit SMIT.
4. Restart the machine.

Disabling at the Command Line

1. At the command line, enter:

   ```
   dtconfig –d
   ```

2. Restart the machine.

Disabling With SMIT

1. Enter the SMIT fast path:

   ```
   /usr/bin/smit dtconfig
   ```

2. Select **Command-line Environment.**
3. Confirm your choice and exit SMIT.
4. Restart the machine.

Starting and Stopping AIX Common Desktop Environment Manually

You can start and stop AIX Common Desktop Environment manually.

Start the Desktop Login Manager Manually

1. Log in to your system as root.
2. At the command line, enter:

   ```
   /usr/dt/bin/dtlogin –daemon
   ```

A **Desktop Login** screen will display. When you log in, you will start a desktop session.

Stop the Login Manager Manually

When you manually stop the login manager, all X servers and desktop sessions that the login manager started are stopped.

1. Open a terminal emulator window and log in as root.
2. Obtain the process ID of the Login Manager by entering the following:

   ```
   cat /var/dt/Xpid
   ```

3. Stop the Login Manager by entering:

   `kill –term` *process_id*

Modifying Desktop Profiles

When a user logs in to the desktop, the shell environment file (**.profile** or **.login**) is not automatically read. The desktop runs the X server before the user logs in, so the function provided by the **.profile** file or the **.login** file must be provided by the desktop's login manager.

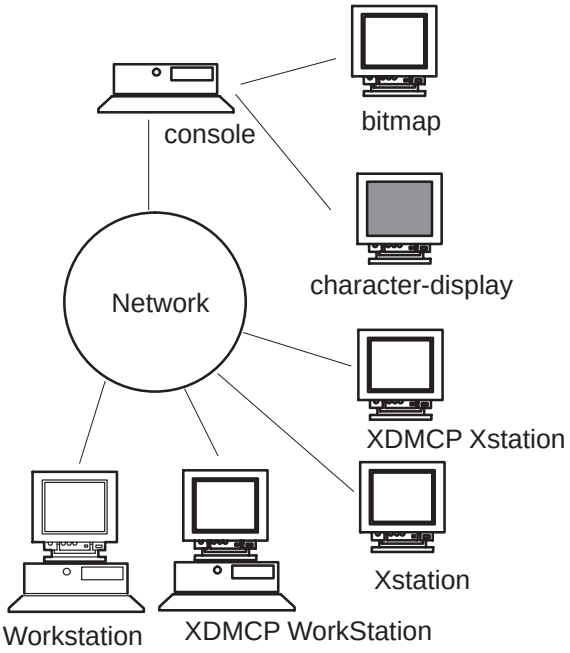
User-specific environment variables are set in */Home Directory/***.dtprofile**. A template for this file is located in **/usr/dt/config/sys.dtprofile**. Place variables and shell commands in **.dtprofile** that apply only to the desktop. Add lines to the end of the **.dtprofile** to incorporate the shell environment file.

System-wide environment variables can be set in Login Manager configuration files. For details on configuring environment variables, see *Common Desktop Environment: Advanced User's and System Administrator's Guide*.

Adding and Removing Displays and Terminals for the AIX Common Desktop Environment

The login manager may be started from a system with a single local bitmap or graphics console. Many other situations are also possible, however. You may want to start AIX Common Desktop Environment from:

- local consoles
- remote consoles
- bitmap and character-displays
- X Display Manager Control Protocol (XDMCP) Xstations
- non-XDMCP Xstations
- X terminal systems running on a host system on the network



An X terminal system consists of a display device, keyboard, and a mouse that run only the X server. Clients, including AIX Common Desktop Environment, are run on one or more host systems on the networks. Output from the clients is directed to the X terminal display.

Wherever possible, you should use terminals that support XDMCP (X Display Manager Control Protocol).

The following Login Manager configuration tasks support many possible configurations:

- adding an Xstation terminal that supports XDMCP
- adding a Non-XDMCP Xstation terminal
- removing a local display
- adding an ASCII or character-display terminal

Adding an Xstation Terminal that supports XDMCP

1. Make sure Login Manager is running on the host system.
2. Enable XDMCP on the X terminal and direct it to contact Login Manager on the host system.

XDMCP provides a mechanism by which X terminals can request login services from a network host. It ensures that the X terminal is communicating with a valid login manager, and provides the protocol for exchanging authentication information between the X terminal and the host login manager. Documentation for your X terminal covers the procedure for enabling XDMCP.

Limiting Access by X Terminals to a Host

1. If the **/etc/dt/config/Xaccess** file does not exist, copy the **/usr/dt/config/Xaccess** file to the **/etc/dt/config** directory.
2. If you have to copy Xaccess to **/etc/dt/config**, you must change the **Dtlogin.servers:** line in **/etc/dt/config/Xconfig** to:

   ```
   Dtlogin.accessFile: /etc/dt/config/Xaccess
   ```

3. Edit **/etc/dt/config/Xaccess** on the host. List only those X terminals permitted to access Login Manager.

If Xaccess is empty, any host can connect.

Using a Workstation as an X Terminal

From a command line, enter:

```
/usr/bin/X11/X –query hostname
```

The X server of the workstation acting as an X terminal must:

- support XDMCP and the **–query** command-line option
- provide xhost permission (in **/etc/X*.hosts**) to the terminal host

Adding a Non-XDMCP Xstation Terminal

1. If the **/etc/dt/config/Xservers** file does not exist, copy the **/usr/dt/config/Xservers** file to the **/etc/dt/config** directory.
2. If you have to copy X servers to **/etc/dt/config**, you must change the **Dtlogin.servers:** line in **/etc/dt/config/Xconfig** to:

   ```
   Dtlogin*servers: /etc/dt/config/Xservers
   ```

3. Edit **/etc/dt/config/Xservers** to include an entry for each terminal. The display type of each terminal must be foreign.
4. Reread the Login Manager configuration files.

When Login Manager receives a SIGHUP, it rereads X config and the X servers file (or the file specified by the **Dtlogin.servers** resource). If it finds a new entry, **dtlogin** starts managing that display. If an entry has been removed, the process associated with that entry is immediately terminated.

The following lines in X servers directs **dtlogin** to manage sessions on two non-XDMCP terminals.

```
ext1:0 NPD200X foreign
ext2:0 QCP–19 foreign
```

Removing a Local Display

To remove a local display, remove its entry in the X servers file in the **/usr/dt/config** directory.

Adding an ASCII or Character-Display Terminal

A character-display console is a configuration in which the console is not a bitmap device.

If No Bitmap Display Is Present

1. If the **/etc/dt/config/Xservers** file does not exist, copy the **/usr/dt/config/Xervers** file to the **/etc/dt/config** directory.
2. If you have to copy X servers to **/etc/dt/config**, you must change the **Dtlogin.servers:** line in **/etc/dt/config/Xconfig** to:

   ```
   Dtlogin*servers: /etc/dt/config/Xservers
   ```

3. Comment out the line in **/etc/dt/config/Xservers** that starts the X server. This will disable the **Login Option** Menu.

   ```
   # * Local local@console /path/X :0
   ```

4. Reread the Login Manager configuration files.

If a Bitmap Display Is Present

1. If the **/etc/dt/config/Xservers** file does not exist, copy the **/usr/dt/config/Xservers** file to the **/etc/dt/config** directory.
2. If you have to copy X servers to **/etc/dt/config**, you must change the **Dtlogin.servers:** line in **/etc/dt/config/Xconfig** to:

   ```
   Dtlogin*servers: /etc/dt/config/Xservers
   ```

3. Edit the line in **/etc/dt/config/Xservers** that starts the X server to read:

   ```
   * Local local@none /path/X :0
   ```

4. Reread the Login Manager configuration files.

Customizing Display Devices for AIX Common Desktop Environment

You can configure AIX Common Desktop Environment Login Manager to run on systems with two or more display devices.

When a system includes multiple displays, the following configuration requirements must be met:

- A server must be started on each display.
- No Windows mode must be configured for each display.

It may be necessary or desirable to use different dtlogin resources for each display.

It may also be necessary or desirable to use different system-wide environment variables for each display device.

Starting the Server on Each Display Device

Procedure

1. If the **/etc/dt/config/Xservers** file does not exist, copy the **/usr/dt/config/Xservers** file to the **/etc/dt/config** directory.
2. If you have to copy X servers to **/etc/dt/config**, you must change the **Dtlogin.servers:** line in **/etc/dt/config/Xconfig** to:

   ```
   Dtlogin*servers: /etc/dt/config/Xservers
   ```

3. Edit **/etc/dt/config/Xservers** to start an X server on each display device.

Syntax

The general syntax for starting the server is:

```
DisplayName DisplayClass DisplayType [ @ite ] Command
```

Only displays with an associated Internal Terminal Emulator (ITE) can operate in **No Windows** mode. **No Windows** mode temporarily disables the desktop for the display and runs a getty process if one is not already started. This allows you to log in and perform tasks not possible under AIX Common Desktop Environment. When you log out, the desktop is restarted for the display device. If a getty is not already running on a display device, Login Manager starts one when **No Windows** mode is initiated.

Default configuration

When ite is omitted, display:0 is associated with the ITE (/dev/console).

Specifying a Different Display as ITE

- On the ITE display, set ITE to the character device.
- On all other displays, set ITE to none.

The following entries in X servers start a server on three local displays on `sysaaa:0`. `Display :0` will be the console (ITE).

```
sysaaa:0 Local local /usr/bin/X11/X :0
sysaaa:1 Local local /usr/bin/X11/X :1
sysaaa:2 Local local /usr/bin/X11/X :2
```

On host **sysbbb**, the bitmap display `:0` is not the ITE; the ITE is associated with device **/dev/tty1**. The following entries in X servers start servers on the two bitmap displays with No Windows Mode enabled on `:1`.

```
sysaaa:0 Local local@none /usr/bin/X11/X :0
sysaaa:1 Local local@ttyi1 /usr/bin/X11/X :1
```

Specifying the Display Name in 'Xconfig'

You cannot use regular hostname:0 syntax for the display name in **/etc/opt/dt/Xconfig**.

- Use underscore in place of the colon.
- In a fully qualified host name, use underscores in place of the periods.

```
Dtlogin.claaa_0.resource: value
Dtlogin.sysaaa_prsm_ld_edu_0.resource: value
```

Using Different Login Manager Resources for Each Display

1. If the **/etc/dt/config/Xconfig** file does not exist, copy the **/usr/dt/config/Xconfig** file to the **/etc/dt/config** directory.
2. Use the resources resource in **/etc/dt/config/Xconfig** to specify a different resource file for each display (this file will be equivalent to **/etc/opt/dt/Xresources**):

   ```
   Dtlogin.DisplayName.resources: path/file
   ```

3. Create each of the resource files specified in Xconfig.
4. In each file, place the dtlogin resources for that display.

The following lines in Xconfig specify different resource files for three displays:

```
Dtlogin.sysaaa_0.resources: /etc/opt/dt/Xresources0
Dtlogin.sysaaa_1.resources: /etc/opt/dt/Xresources1
Dtlogin.sysaaa_2.resources: /etc/opt/dt/Xresources2
```

Running Different Scripts for Each Display

1. If the **/etc/dt/config/Xconfig** file does not exist, copy the **/usr/dt/config/Xconfig** file to the **/etc/dt/config** directory.
2. Use the startup, reset, and setup resources in **/etc/dt/config/Xconfig** to specify different scripts for each display. (These files are run instead of **Xstartup**, **Xreset**, and **Xsetup.**)

   ```
   Dtlogin*DisplayName*sarttup: /path/file
   Dtlogin*DisplayName*startup: /path/file
   Dtlogin*DisplayName*startup: /path/file
   ```

The startup script is run as root after the user has logged in, before the AIX Common Desktop Environment session is started.

The script **/etc/dt/config/Xreset** can be used to reverse the setting made in **Xstartup.** Xreset runs when the user logs out.

The following lines in **Xconfig** specify different scripts for two displays.

```
Dtlogin.sysaaa_0*startup:   /etc/opt/dt/Xstartup0
Dtlogin.sysaaa_1*startup:    /etc/opt/dt/Xstartup1
Dtlogin.sysaaa_0*setup:      /etc/opt/dt/Xsetup0
Dtlogin.sysaaa_1*setup:      /etc/opt/dt/Xsetup1
Dtlogin.sysaaa_0*reset:      /etc/opt/dt/Xreset0
Dtlogin.sysaaa_1*reset:      /etc/opt/dt/Xreset1
```

Setting Different Systemwide Environment Variables for Each Display

1. If the **/etc/dt/config/Xconfig** file does not exist, copy the **/usr/dt/config/Xconfig** file to the **/etc/dt/config** directory.
2. Set the environment resource in **/etc/dt/config/Xconfig** separately for each display:

```
Dtlogin*DisplayName*environment: value
```

The following points apply to environment variables for each display:

- Separate variable assignments with a space or tab.
- Do not use the environment resource to set TZ and LANG.
- There is no shell processing within Xconfig.

The following lines in Xconfig set variables for two displays.

```
Dtlogin*syshere_0*environment:EDITOR=vi SB_DISPLAY_ADDR=0xB00000
Dtlogin*syshere_1*environment: EDITOR=emacs \
                        SB_DISPLAY_ADDR=0xB00000
```

Chapter 13. National Language Support

Many system variables are used to establish the language environment of the system. These variables and their supporting commands, files, and other tools, are referred to as National Language Support (NLS). This chapter describes the following features and procedures for national language support:

- overview for national language support and locale
- locale, locale categories, and locale environment variables
- the locale definition source file and character set description source file
- changing your locale and creating a new collation order
- overviews for converters, the message facility, and national language support for devices
- changing the language environment and the default keyboard map
- list of national language support commands and files

Note: On an FX Series system, messages specific to the FX Series system are not translated.

National Language Support Overview

National Language Support (NLS) provides commands and Standard C Library subroutines for a single worldwide system base. An internationalized system has no built-in assumptions or dependencies on language-specific or cultural-specific conventions such as:

- code sets
- character classifications
- character comparison rules
- character collation order
- numeric and monetary formatting
- date and time formatting
- message-text language

All information pertaining to cultural conventions and language is obtained at process run time.

The following capabilities are provided by NLS to maintain a system running in an international environment:

- localization of information
- separation of messages from programs
- conversion between code sets

Localization of Information

An internationalized system processes information correctly for different locations. For example, in the United States, the date format, 9/6/1990, is interpreted to mean the sixth day of the ninth month of the year 1990. The United Kingdom interprets the same date format to mean the ninth day of the sixth month of the year 1990. The formatting of numerical and monetary data is also country-specific, for example, the U.S. $ (dollar) and the U.K. £ (pound). A *locale* is defined by these language-specific and cultural-specific conventions for processing information.

All locale information must be accessible to programs at run time so that data is processed and displayed correctly for your cultural conventions and language. This process is called localization; it consists of developing a database containing locale-specific rules for formatting data and an interface to obtain the rules. For more information about localization, see “Locale Overview” on page 13-4.

Separation of Messages from Programs

To facilitate translations of messages into various languages and to make the translated messages available to the program based on a user’s locale, it is necessary to keep messages separate from the programs and provide them in the form of message catalogs that a program can access at run time. To aid in this task, commands and subroutines are provided by the message facility. For more information, see “Message Facility Overview” on page 13-18.

Conversion between Code Sets

A *character* is any symbol used for the organization, control, or representation of data. A group of such symbols used to describe a particular language make up a *character set*. A code set contains the encoding values for a character set. It is the encoding values in a code set that provide the interface between the system and its input and output devices.

Historically, the effort was directed at encoding the English alphabet. It was sufficient to use a 7-bit encoding method for this purpose because the number of English characters is not large. To support larger alphabets, such as the Asian languages (for example, Chinese, Japanese, and Korean), additional code sets were developed that contained multibyte encodings.

The following code sets are supported:

- Industry-standard code sets are provided by means of the ISO8859 family of code sets, which provide a range of single-byte code set support that includes Latin-1, Latin-2, Arabic, Cyrillic, Hebrew, Greek, and Turkish. The IBM-eucJP code set is the industry-standard code set used to support the Japanese locale.
- Personal Computer (PC) based code sets IBM-850 and IBM-932 are supported. IBM-850 is a single-byte code set used to support Latin-1 countries (U.S., Canada, and Western Europe). IBM-932 is a multibyte code set used to support the Japanese locale.

As more code sets are supported, it becomes important not to clutter programs with the knowledge of any particular code set. This is known as *code set independence.* To aid in code set independence, NLS supplies converters that translate character encoding values found in different code sets. Using these converters, a system can accurately process data generated in different code set environments. For more information, see “Converters Overview” on page 13-16.

Locale Overview

An internationalized system has no built-in assumptions or dependencies on code set, character classification, character comparison rules, character collation order, monetary formatting, numeric punctuation, date and time formatting, or the text of messages. A *locale* is defined by these language and cultural conventions. All of the information pertaining to language-specific and cultural-specific conventions is obtained at process run time.

All locale information must be accessible to programs at run time so that data is processed and displayed correctly for your language-specific and cultural-specific conventions. The National Language Support (NLS) system provides these localization capabilities. NLS provides a database containing locale-specific rules for formatting data and an interface to obtain these rules.

Understanding Locale

A locale is made up of the language, territory, and code set combination used to identify a set of language conventions. These conventions include information on collation, case conversion and character classification, the language of message catalogs, date-and-time representation, the monetary symbol, and numeric representation.

Locale information contained in locale definition source files must first be converted into a locale database by the **localedef** command. The **setlocale** subroutine can then access this information and set locale information for applications. To deal with locale data in a logical manner, locale definition source files are divided into six categories. Each category contains a specific aspect of the locale data. The **LC_*** environment variables and the **LANG** environment variable can be used in specifying the desired locale.

Locale Naming Conventions

Each locale is named by its locale definition source file name. These files are named for the language, territory, and code set information they describe. The following format is used for naming a locale definition file:

language[_*territory*][.*codeset*][@*modifier*]

For example, the locale for the Danish language spoken in Denmark using the ISO8859-1 code set is `da_DK.ISO8859-1`. The `da` stands for the Danish language and the `DK` stands for Denmark. The short form of `da_DK` is sufficient to indicate this locale. The same language and territory using the IBM-850 code set is indicated by either `Da_DK.IBM-850` or the short form `Da_DK`.

System-defined locale definition files are provided to show the format of locale categories and their keywords. The **/usr/lib/nls/loc** directory contains the locale definition files for system-defined locales. The C, or POSIX, locale defines the ANSI C-defined standard locale inherited by all processes at startup time. The other system-defined locale definition source files are:

Locale	Language	Country	Code Set
Ar_AA	Arabic	Arabic Countries	IBM-1046
ar_AA	Arabic	Arabic Countries	IS08859-6
bg_BG	Bulgarian	Bulgaria	IS08859-5
cs_CZ	Czech	Czech Republic	IS08859-2
Da_DK	Danish	Denmark	IBM-850
da_DK	Danish	Denmark	ISO8859-1
De_CH	German	Switzerland	IBM-850
de_CH	German	Switzerland	ISO8859-1
De_DE	German	Germany	IBM-850
de_DE	German	Germany	ISO8859-1
el_GR	Greek	Greece	ISO8859-7
En_GB	English	Great Britain	IBM-850
en_GB	English	Great Britain	ISO8859-1
En_US	English	United States	IBM-850
en_US	English	United States	ISO8859-1

Es_ES	Spanish	Spain	IBM-850
es_ES	Spanish	Spain	ISO8859-1
Fi_FI	Finnish	Finland	IBM-850
fi_FI	Finnish	Finland	ISO8859-1
Fr_BE	French	Belgium	IBM-850
fr_BE	French	Belgium	ISO8859-1
Fr_CA	French	Canada	IBM-850
fr_CA	French	Canada	ISO8859-1
Fr_FR	French	France	IBM-850
fr_FR	French	France	ISO8859-1
Fr_CH	French	Switzerland	IBM-850
fr_CH	French	Switzerland	ISO8859-1
hr_HR	Croatian	Croatia	ISO8859-2
hu_HU	Hungarian	Hungary	ISO8859-2
Is_IS	Icelandic	Iceland	IBM-850
is_IS	Icelandic	Iceland	ISO8859-1
It_IT	Italian	Italy	IBM-850
it_IT	Italian	Italy	ISO8859-1
Iw_IL	Hebrew	Israel	IBM-856
iw_IL	Hebrew	Israel	ISO8859-8
Ja_JP	Japanese	Japan	IBM-932
ja_JP	Japanese	Japan	IBM-eucJP
ko_KR	Korean	Korea	IBM-eucKR
mk_MK	Macedonian	Former Yugoslav Republic of Macedonia	ISO–8859-5
Nl_BE	Dutch	Belgium	IBM-850
nl_BE	Dutch	Belgium	ISO8859-1
Nl_NL	Dutch	Netherlands	IBM-850
nl_NL	Dutch	Netherlands	ISO8859-1
No_NO	Norwegian	Norway	IBM-850
no_NO	Norwegian	Norway	ISO8859-1
pl_PL	Polish	Poland	ISO8859-2
pt_BR	Brazillian	Brazil	ISO8859-1
Pt_PT	Portuguese	Portugal	IBM-850
pt_PT	Portuguese	Portugal	ISO8859-1
ro_RO	Romanian	Romania	ISO8859-2
ru_RU	Russian	Russia	ISO8859-5
sh_SP	Serbian Latin	Yugoslavia	ISO8859-2

sl_SI	Slovene	Slovenia	ISO8859-2
sk_SK	Slovak	Slovakia	ISO8859-2
sr_SP	Serbian Cyrillic	Yugoslavia	ISO8859-5
Sv_SE	Swedish	Sweden	IBM-850
sv_SE	Swedish	Sweden	ISO8859-1
tr_TR	Turkish	Turkey	ISO8859-9
zh_CN	Simplified Chinese	People's Republic of China	IBM-eucCN
ZH_CN	Chinese	People's Republic of China	UTF-8
zh_TW	Chinese (trad)	Republic of China	IBM-eucTW

Installation Default Locale

The installation default locale refers to the locale selected at installation. For example, when prompted, a user can specify the French language as spoken in Canada during the installation process. The code set automatically defaults to the ISO8859-1 code set. With this information, the system sets the value of the default locale, specified by the **LANG** environment variable, to `fr_CA` (`fr` for ISO8859-1 French and `CA` for Canada). Every process uses this locale unless the **LC_*** or **LANG** environment variables are modified. The default locale can be changed by using the System Management Interface Tool (SMIT) Manage Language Environment menu.

Understanding Locale Categories

A locale *category* is a particular grouping of language-specific and cultural-convention-specific data. For instance, data referring to date-and-time formatting, the names of the days of the week, names of the months, and other time-specific information is grouped into the **LC_TIME** category. Each category uses a set of keywords that describe the particulars of that locale subset.

The following standard categories can be defined in a locale definition source file:

LC_COLLATE	defines character-collation or string-collation information
LC_CTYPE	defines character classification, case conversion, and other character attributes
LC_MESSAGES	defines the format for affirmative and negative responses
LC_MONETARY	defines rules and symbols for formatting monetary numeric information
LC_NUMERIC	defines rules and symbols for formatting non–monetary numeric information
LC_TIME	defines a list of rules and symbols for formatting time and date information

Note: Locale categories can only be modified by editing the locale definition source file; they should not be confused with the environment variables of the same name, which can be set from the command line.

Understanding Locale Environment Variables

National Language Support (NLS) uses several environment variables to influence the selection of locales. You can set the values of these variables to change search paths for locale information:

LANG specifies the installation default locale

Note: The **LANG** value is established at installation. (This is the locale every process will use unless the **LC_*** environment variables are set). The **LANG** environment variable can be changed by using the System Management Interface Tool (SMIT). The C and POSIX locales offer the best performance.

LC_ALL overrides the value of the **LANG** environment variable and the values of any other **LC_*** environment variables

LC_COLLATE specifies the locale to use for **LC_COLLATE** category information

The **LC_COLLATE** category determines character-collation or string-collation rules governing the behavior of ranges, equivalence classes, and multicharacter collating elements.

LC_CTYPE specifies the locale to use for **LC_CTYPE** category information

The **LC_CTYPE** category determines character handling rules governing the interpretation of sequences of bytes of text data characters (that is, single-byte versus multibyte characters), the classification of characters (for example, alpha, digit, and so on), and the behavior of character classes.

LC__FASTMSG specifies that default messages should be used for the C and POSIX locales and that **NLSPATH** will be ignored when **LC__FASTMSG** is set to `true`

Otherwise, POSIX compliant message handling will be performed. The default value will be `LC__FASTMSG=true` in **/etc/environment**.

LC_MESSAGES specifies the locale to use for **LC_MESSAGES** category information

The **LC_MESSAGES** category determines rules governing affirmative and negative responses and the locale (language) for messages and menus.

LC_MONETARY specifies the locale to use for **LC_MONETARY** category information

The **LC_MONETARY** category determines the rules governing monetary-related formatting.

LC_NUMERIC specifies the locale to use for **LC_NUMERIC** category information

The **LC_NUMERIC** category determines the rules governing non–monetary numeric formatting.

LC_TIME specifies the locale to use for **LC_TIME** category information

The **LC_TIME** category determines the rules governing date and time formatting.

LOCPATH specifies the search path for localized information, including binary locale files, input methods, and code-set converters

Note: All **setuid** and **setgid** programs ignore the **LOCPATH** environment variable.

NLSPATH specifies the search path for locating message catalog files

This environment variable is used by the Message Facility component of the NLS subsystem.

The environment variables that affect locale selection can be grouped into three priority classes:

Locale Environment Variable Hierarchy	
Priority Class	**Environment Variables**
High	**LC_ALL**
Medium	**LC_COLLATE**
	LC_CTYPE
	LC_MESSAGES
	LC_MONETARY
	LC_NUMERIC
	LC_TIME
Low	**LANG**

The behavior of an internationalized program is affected by the locale environment variables in the following manner:

- If the **LC_ALL** environment variable is set, the value of the **LC_ALL** variable is used for all categories. For example, if the **LC_ALL** environment variable is equal to `en_US` and the **LANG** environment variable is equal to `fr_FR`, the locale is set to `en_US`.
- If the **LC_ALL** environment variable is not set, the values specified for medium-priority environment variables are used. For example, if the **LANG** environment variable is set to `en_US` and the **LC_TIME** environment variable is set to `fr_FR`, then the **LC_TIME** category will be loaded from the `fr_FR` locale database. The **LC_TIME** environment variable does not affect the behavior of any other category.
- If individual **LC_*** environment variables are not set, the value of the **LANG** environment variable specifies the locale for all remaining categories.
- If the **LANG** variable is not set, the locale for all remaining categories defaults to the C locale.

Understanding the Locale Definition Source File

Unlike environment variables, which can be set from the command line, locales can only be modified by editing and compiling a locale definition source file.

If a desired locale is not part of the library, a binary version of the locale can be compiled by the **localedef** command. Locale behavior of programs is not affected by a locale definition source file unless the file is first converted by the **localedef** command, and the locale object is made available to the program. The **localedef** command converts source files containing definitions of locales into a run-time format and copies the run-time version to the file specified on the command line, which usually is a locale name. Internationalized commands and subroutines can then access the locale information. For information on preparing source files to be converted by the **localedef** command, see “Locale Definition Source File Format” in the *Files Reference*.

Understanding the Character Set Description (charmap) Source File

Using the character set description (charmap) source file, you can assign symbolic names to character encodings.

Developers of character set description (charmap) source files are free to choose their own symbolic names, provided that these names do not conflict with the standardized symbolic names that describe the portable character set.

The charmap file resolves problems with the portability of sources, especially locale definition sources. The standardized portable character set is constant across all locales. The charmap file provides the capability to define a common locale definition for multiple code sets. That is, the same locale definition source can be used for code sets with different encodings of the same extended characters.

A charmap file defines a set of symbols that are used by the locale definition source file to refer to character encodings. The characters in the portable character set can optionally be included in the charmap file, but the encodings for these characters should not differ from their default encodings.

The charmap files are located in the **/usr/lib/nls/charmap** directory.

Changing Your Locale

Changing the NLS Environment with the Manage Language Environment SMIT Interface

A menu provided by this interface allows the user to:

- change the default language environment
- change the keyboard map for the next system restart
- manage fonts
- convert the code set of message catalogs
- convert the code set of flat text files

In addition, you can use the **setmaps** command to set the code set map of a terminal.

Changing the Default Language Environment

The **"LANG = <name>"** string (in the **/etc/environment** file) can be changed through the Manage Language Environment SMIT interface. The setting of the **LANG** environment variable designates the default locale, which is a language-territory-code-set combination. The default locale provides formats for default collation, character classification, case conversion, numeric and monetary formatting, date-and-time formatting, and affirmative or negative responses. The default locale includes reference to the code set.

Changing the Default Keyboard Mapping for the Next System Restart

If more than one code set is supported for a given language-territory combination, multiple LFT keyboard mappings exist. The selected keyboard mapping has to match the code set of the selected language environment.

Managing Fonts

Users can perform such tasks as selecting the active font and selecting the font to load for the next system restart. The selected font has to support the same code set as the selected language environment and LFT keyboard mapping.

Converting the Code Set of Message Catalogs

Message catalogs are shipped in one code set for each translated language-territory combination. The code set of the message catalog has to match the code set of the locale.

Converting the Code Set of Flat Text Files

User-defined flat files of one code set can be converted to another code set through the Manage Language Environment SMIT interface when appropriate (IBM-850 to ISO8859-1, for example).

Typical User Scenarios

There are several NLS-related scenarios some users may encounter on the system. This section lists some of these with suggested actions to be taken.

- User keeps the default code set.

  The user may be satisfied with the default code set for language-territory combinations even where more than one code set is supported. The user may keep the default code set if the current user environment uses that code set, or if the user is new and has no code set preference.

The language-territory selected at system installation time will be defaulted to the appropriate locale based on the default code set. The default keyboard mappings, default font, and message catalogs are all established around the default code set. This scenario requires no special action from the user.

- User changes the code set from the default code set.

  Users of a Latin-1 or Japanese locale may want to migrate their data and NLS environment to the nondefault code set. This can be done in the following fashion:

  – if the user has existing data that requires conversion

    Flat text files that require conversion to the preferred code set may be converted through use of the **iconv** utility or through the Manage the Language Environment SMIT menu. User-defined structured files require conversion via user-written conversion tools, which use the **iconv library functions** to convert the desired text fields in the structured files.

  – when the user wants to change to the other code set

    Where more than one code set is supported for a language-territory combination, the user may change to the nondefault locale by using the **chlang**, **chkbd**, and **chfont** commands, or by using the Manage Language Environment SMIT menu to accomplish the change of language environment, the keyboard mapping, and the font to reflect the nondefault code set.

Changing the NLS Environment with the localedef Command

If a special locale is desired (that is, a locale different from any of those provided), take the following steps with a user ID that allows read or write permissions (for example, root):

1. If you are using a locale source file named `gwm`, copy the provided locale source file that is closest to the desired locale to a file named `gwm.src`. This name cannot be the same as any previously defined locale. The system-defined locales are listed in “Understanding Locale” on page 13-5.

   ```
   cd /usr/lib/nls/loc
   cp en_GB.ISO8859-1.src gwm.src
   ```

2. Edit the newly created locale source file to change the locale variables to the desired values:

   ```
   vi gwm.src
   change d_fmt ″%d%m%y″ to d_fmt ″%m–%d–%y″
   ```

3. Compile the locale definition source file:

   ```
   localedef –f ISO8859-1 –i gwm.src gwm
   ```

4. Set the **LOCPATH** environment variable to the directory containing the new locale file. The default for **LOCPATH** is **/usr/lib/nls/loc**:

   ```
   LOCPATH=/usr/lib/nls/loc; export LOCPATH
   ```

   Note: All **setuid** and **setgid** programs ignore the **LOCPATH** environment variable.

5. Set the corresponding environment variable or variables:

   ```
   export LC_TIME=gwm
   ```

Creating a New Collation Order

1. If you are using a locale source file named gwm, copy the provided locale source file that is closest to the desired character collation order to a file named gwm.src. This name cannot be the same as any previously defined locale. The system-defined locales are listed in “Understanding Locale” on page 13-5.

   ```
   cd /usr/lib/nls/loc
   cp en_GB.ISO8859-1.src gwm.src
   ```

2. Edit the newly created gwm.src file to change the lines that are associated with the **LC_COLLATE** category that is associated with the characters you want to change:

   ```
   vi gwm.src
           change
                   <a>     <a>;<non-accent>;<lower-case>;IGNORE
                   <b>     <b>;<non-accent>;<lower-case>;IGNORE
                   <c>     <c>;<non-accent>;<lower-case>;IGNORE
                   <d>     <d>;<non-accent>;<lower-case>;IGNORE
           to
                   <a>     <d>;<non-accent>;<lower-case>;IGNORE
                   <b>     <c>;<non-accent>;<lower-case>;IGNORE
                   <c>     <b>;<non-accent>;<lower-case>;IGNORE
                   <d>     <a>;<non-accent>;<lower-case>;IGNORE
   ```

3. Generate the new gwm locale:

   ```
   localedef -f ISO08859-1 -i gwm.src gwm
   ```

4. Set the **LOCPATH** environment variable to the directory containing the new locale. If the new locale is in /u/foo, then enter:

   ```
   LOCPATH=/u/foo:/usr/lib/nls/loc; export LOCPATH
   ```

   The default for **LOCPATH** is **/usr/lib/nls/loc**.

   Note: All **setuid** and **setgid** programs ignore the **LOCPATH** environment variable.

5. Change the **LC_COLLATE** environment variable to the name of the newly defined gwm locale binary:

   ```
   LC_COLLATE=gwm; export LC_COLLATE
   ```

   Any command will now use the collation order specified in the gwm locale. In this example, the characters a-d are sorted in reverse order by commands such as **li**, **ls**, and **sort**.

Converters Overview

National Language Support (NLS) provides a base for internationalization to allow data to be changed from one code set to another. You may need to convert text files or message catalogs. There are several standard converters for this purpose.

When a program sends data to another program residing on a remote host, the data can require conversion from the code set of the source machine to that of the receiver. For example, when communicating with an IBM VM system, the system converts its ISO8859-1 data to EBCDIC. Code sets define character and control function assignments to code points. These coded characters must be converted when a program receives data in one code set but displays it in another code set.

There are two interfaces for doing conversions:

iconv command allows you to request a specific conversion by naming the from and to code sets

libiconv functions allow applications to request converters by name

The system provides a library of converters that is ready to use. You supply the name of the converter you want to use. The converter libraries are found in the following directories: **/usr/lib/nls/loc/iconv/*** and **/usr/lib/nls/loc/iconvTable/***.

In addition to code set converters, the converter library also provides a set of network interchange converters. In a network environment, the code sets of the communications systems and the protocols of communication determine how the data should be converted.

Interchange converters are used to convert data sent from one system to another. Conversions done from one internal code set to another require code set converters. Whether data must be converted from a sender's code set to a receiver's code set, or 8-bit data must be converted into 7-bit data form, a uniform interface is required. The **iconv** subroutines provide this interface.

Standard Converters

There are standard converters for use with the **iconv** command and subroutines. The following list describes the different types of converters. For a list of converters, see *General Programming Concepts: Writing and Debugging Programs*.

Converter Type

Table converter
converts single-byte stateless code sets and performs a table translation from one byte to another byte

Multibyte converter
provides conversions between multibyte code sets, for example, between Japanese PC Code (IBM-932) or Japanese AIX Code (IBM-eucJP) and IBM Japanese Host Code Sets (IBM-930 and IBM-939)

Interchange Converter Types

7-bit converter converts between internal code sets and standard interchange formats (7-bit)

8-bit converter converts between internal code sets and standard interchange formats (8-bit)

Compound text converter
converts between compound text and internal code sets

uucode converter
: provides the same mapping as the **uuencode** and **uudecode** commands

Miscellaneous converters
: used by some of the converters listed above

Understanding iconv Libraries

The iconv facility consists of a set of functions that contain the data and logic to convert from one code set to another. The utility also includes the **iconv** command, which converts data. A single system can have several converters. The **LOCPATH** environment variable determines the converter that the **iconv** subroutines use.

Note: All **setuid** and **setgid** programs ignore the **LOCPATH** environment variable.

Using the iconv Command

Any converter installed in the system can be used through the **iconv** command, which uses the **iconv** library. The **iconv** command acts as a filter for converting from one code set to another. For example, the following command filters data from PC Code (IBM-850) to ISO8859-1:

```
cat File | iconv -f IBM-850 -t ISO8859-1 | tftp -p - host /tmp/fo
```

The **iconv** command converts the encoding of characters read from either standard input or the specified file and then writes the results to standard output.

Universal UCS Converter

UCS-2 is a universal 16-bit encoding (see the code set overview in *General Programming Concepts: Writing and Debugging Programs*) that can be used as an interchange medium to provide conversion capability between virtually any code sets. The conversion can be accomplished using the Universal UCS Converter, which converts between any two code sets XXX and YYY as follows:

```
XXX <–> UCS-2 <–> YYY
```

The XXX and YYY conversions must be included in the supported List of UCS-2 Interchange Converters, and must be installed on the system.

The universal converter is installed as the file **/usr/lib/nls/loc/iconv/Universal_UCS_Conv**. A new conversion can be supported by creating a new link with the appropriate name in the **/usr/lib/nls/loc/iconv** directory. For example, to support new converters between IBM-850 and IBM-437, you can execute the following commands:

```
ln –s /usr/lib/nls/loc/iconv/Universal_UCS_Conv
/usr/lib/nls/loc/iconv/IBM-850_IBM-437
```

```
ln –s /usr/lib/nls/loc/iconv/Universal_UCS_Conv
/usr/lib/nls/loc/iconv/IBM-437_IBM-850
```

Warning: If a converter link is created for incompatible code sets (for example, ISO8859-1 and IBM-eucJP), and if the source data contains characters that don't exist in the target code set, significant data loss can result.

Message Facility Overview

To facilitate translation of messages into various languages and to make them available to a program based on a user's locale, it is necessary to keep messages separate from the program and provide them in the form of message catalogs that a program can access at run time. To aid in this task, commands and subroutines are provided by the Message Facility. Message source files containing application messages are created by the programmer and converted to message catalogs. These catalogs are used by the application to retrieve and display messages, as needed. Message source files can be translated into other languages and converted to message catalogs without changing and recompiling a program.

The Message Facility includes the following two commands for displaying messages with a shell script or from the command line:

dspcat displays all or part of a message catalog

dspmsg displays a selected message from a message catalog

These commands use the **NLSPATH** environment variable to locate the specified message catalog. The **NLSPATH** environment variable lists the directories containing message catalogs. These directories are searched in the order in which they are listed. For example:

```
NLSPATH=/usr/lib/nls/msg/%L/%N:/usr/lib/nls/msg/prime/%N
```

The **%L** and **%N** special variables are defined as follows:

%L specifies the locale-specific directory containing message catalogs

The value of the **LC_MESSAGES** category or the **LANG** environment variable is used for the directory name. The **LANG**, **LC_ALL**, or **LC_MESSAGES** environment variable can be set by the user to the locale for message catalogs.

%N specifies the name of the catalog to be opened

If the **dspcat** command cannot find the message, the default message is displayed. You must enclose the default message in single-quotation marks if the default message contains **%***n***$** format strings. If the **dspcat** command cannot find the message and you do not specify a default message, a system-generated error message is displayed.

The following example uses the **dspcat** command to display all messages in the existing `msgerrs.cat` message catalog:

```
/usr/lib/nls/msg/$LANG/msgerrs.cat:
dspcat msgerrs.cat
```

The following output is displayed:

```
1:1 Cannot open message catalog %s
Maximum number of catalogs already open
1:2 File %s not executable
2:1 Message %d, Set %d not found
```

By displaying the contents of the message catalog in this manner, you can find the message ID numbers assigned to the `msgerrs` message source file by the **mkcatdefs** command to replace the symbolic identifiers. Symbolic identifiers are not readily usable as references for the **dspmsg** command, but using the **dspcat** command as shown can give you the necessary ID numbers.

The following is a simple shell script called `runtest` that shows how to use the **dspmsg** command:

```
if [ – x ./test ]
    ./test;
else
    dspmsg  msgerrs.cat –s 1 2 ’%s NOT EXECUTABLE \n’ ”test”;
    exit;
```

Note: If you do not use a full path name, as in the preceding examples, be careful to set the **NLSPATH** environment variable so that the **dspcat** command searches the proper directory for the catalog. The **LC_MESSAGES** category or the value of the **LANG** environment variable also affects the directory search path.

National Language Support Overview for Devices

National Language Support (NLS) uses the locale setting to define its environment. The locale setting is dependent on the user's requirements for data processing and language that determines input and output device requirements. The system administrator is responsible for configuring devices that are in agreement with user locales.

Terminals (tty Devices)

The **setmaps** command is used to set the terminal and code-set map for a given tty or pty. The **setmaps** file format defines the text of the code-set map file and the terminal map file.

The code-set map file defines the length (number of bytes) and the width (number of columns) for a given character. The code-set maps are used for canonical editing in the **termio** terminal data processing. A code-set map is configured according to the code set of the locale.

The text of a code set map file is a description of the code set, including the type (single byte or multibyte), the memory and screen widths (for multibyte code sets), and the optional NO TAGconverter modules to push on the stream. The code set map file is located in the **/usr/lib/nls/csmap** directory and has the same name as the code set.

The terminal-map-file rules associate a pattern string with a replacement string. The operating system uses an input map file to map input from the keyboard to an application and uses an output map file to map output from an application to the display.

Printers

Virtual printers inherit the default code set of incoming jobs from the **LANG** entry in the **/etc/environment** file. A printer subsystem can support several virtual printers. If more than one virtual printer is supported, each can have a different code set. There are three suggested printer subsystem scenarios:

- The first scenario involves several queues, several virtual printers, and one physical printer. Each virtual printer has its own code set. The print commands specify which queue to use. The queue in turn specifies the virtual printer with the appropriate code set. In this scenario, the user needs to know which queue is attached to which virtual printer and the code set that is associated with each.
- The second scenario is similar to the first, but each virtual printer is attached to a different printer.
- The third scenario involves using the **qprt** print command to specify the code set. In this option, there are several queues available and one virtual printer. The virtual printer uses the inherited default code set.

The user specifies the queue and code set by using the **qprt** command with the **–P–x** flags. If the **–P** flag is not specified, the default queue is used. If the **–x** flag is not used, the default code set for the virtual printer is used.

Low-Function Terminals

Key Maps

Low-function terminals (LFTs) support single-byte code-set languages using key maps. An LFT key map translates a key stroke into a character string in the code set. A list of all available key maps is in the **/usr/lib/nls/loc** directory. LFT does not support languages that require multibyte code sets.

The default LFT keyboard setting and associated font setting are based on the language selected during installation. The possible default code sets are:

- ISO8859-1
- ISO8859-2
- ISO8859-5
- ISO8859-6
- ISO8859-7
- ISO8859-8
- ISO8859-9

There are several ways to change the default settings:

- To change the default font for next reboot, use the **chfont** command with the **–n** flag.
- To change the default keyboard for next reboot, use the **chkbd** command with the **–n** flag.

The **lsfont** and **lskbd** commands list all the fonts and keyboard maps that are currently available to the LFT.

Fonts

The LFT font libraries for all the supported code sets are in the **/usr/lpp/fonts** directory.

Changing the Language Environment

A number of system operations are affected by the language environment. Some of these operations include collation, time of day and date representation, numeric representation, monetary representation, and message translation. The language environment is determined by the value of the **LANG** environment variable, and you can change that value with the **chlang** command. The **chlang** command can be run from the command line or from SMIT.

Using the Shell Command

If you know the correct value for the language environment you want, you can change the language environment by entering:

```
chlang Language
```

Using SMIT

If you do not know the correct value for the language environment, or if you want to see a list of the possible values, use the SMIT fastpath.

1. Use the **smit chlang** fast path to access the **Change Language Environment** menu.
2. Select the language environment you want to change to.
3. Confirm your choice to change your language environment.

Changing the Default Keyboard Map

NLS also enables you to specify the correct keyboard for the language you want to use. The operating system provides a number of keyboard maps for this purpose. You can change the default keyboard map for LFT terminals with the **chkbd** command, which you can run using SMIT. The change does not go into effect until you restart the system.

1. Use the **smit chkbd** fast path to access the **Select the Keyboard Map for the Next System Restart** menu.
2. Select the keyboard map you want to change to.
3. Confirm your choice to change your keyboard map for the next system startup.

List of National Language Support Commands and Files

National Language Support (NLS) provides several commands and files for system internationalization.

Converter Command

NLS provides a base for internationalization in which data may be changed from one code set to another. The following command can be used for this conversion:

iconv converts the encoding of characters from one code set encoding scheme to another

Input Method Command

The Input Method is a set of subroutines that translate key strokes into character strings in the code set specified by a locale. The Input Method subroutines include logic for locale-specific input processing and keyboard controls (Ctrl, Alt, Shift, Lock, Alt Graphic). The following command allows for the customizing of input method mapping for the use of input method subroutines:

keycomp compiles a keyboard mapping file into an input method keymap file

For more information about these methods, see the Input Methods overview in *General Programming Concepts: Writing and Debugging Programs*.

Locale Commands and Files

NLS provides a database containing locale-specific rules for formatting data and an interface to obtain these rules.

Locale Commands

The following commands are provided for the creation and display of locale information:

locale writes information about the current locale or all public locales

localedef converts locale definition source files and character set description (charmap) source files to produce a locale database

Locale Source Files

The following files are provided for the specification of rules for formatting locale-specific data:

character set description (charmap)
defines character symbols as character encodings

locale definition
contains one or more categories that describe a locale

The following categories are supported:

LC_COLLATE defines character or string collation information

LC_CTYPE defines character classification, case conversion, and other character attributes

LC_MESSAGES
defines the format for affirmative and negative responses

LC_MONETARY
defines rules and symbols for formatting monetary numeric information

LC_NUMERIC defines a list of rules and symbols for formatting non–monetary numeric information

LC_TIME defines a list of rules and symbols for formatting time and date information

Message Facility Commands

The Message Facility consists of standard defined (X/Open) subroutines, commands, and value-added extensions to support externalized message catalogs. These catalogs are used by an application to retrieve and display messages, as needed. The following Message Facility commands create message catalogs and display their contents:

dspcat displays all or part of a message catalog

dspmsg displays a selected message from a message catalog

gencat creates and modifies a message catalog

mkcatdefs preprocesses a message source file for input to the **gencat** command

runcat pipes output from the **mkcatdefs** command to the **gencat** command

Chapter 14. Managing Power Management on a Standard AIX System

Power Management is a technique that enables hardware and software to minimize system power consumption. It is especially important for products that operate with batteries and desktop products.

This chapter includes the following sections that discuss functions of Power Management:

- “Enabling and Disabling Power Management”
- “Configuring and Unconfiguring Power Management”
- “Starting System State Transition from the Enable State”
- “Changing/Showing Characteristics of Power Management”
- “Changing Power Management Time”
- “Managing Display Power Management”
- “Specifying Power Management Characteristics of Each Device”
- “Managing the Battery”

You can perform each of these tasks from a command line or by using the System Management Interface Tool (SMIT). These tasks can also be done using GUI. To start the Power Management application, enter `/usr/lpp/x11/bin/xpowerm`.

Note: On an FX Series system, there are no equivalent commands (such as **pmctrl**) and there is no SMIT interface to directly manage power management. Instead, the system itself actively and continuously monitors power consumption and reports faults, when necessary. Refer to *Administering Your Fault Tolerant System* for more information.

Enabling and Disabling Power Management

Power Management can be enabled or disabled. When Power Management is enabled, each device can enter a power-savings mode and the system can enter Power Management states. With Power Management disabled, no system state transition occurs and devices operate in the full-on mode.

You can perform each of these tasks from the command line or with SMIT.

Prerequisite

You must have root user authority to enable or disable Power Management.

Enabling Events

Using the Command Line

At the command line, enter:

```
pmctrl –e –a enable
```

Using SMIT

1. Type `smit` from the command line.
2. Select **Performance and Resource Scheduling**.
3. Select **Power Management**.
4. Select **Enable/Disable Power Management**.
5. Enter `enable` in the field:

   ```
   Enable/Disable Power Management
   ```
6. Confirm your choice to enable Power Management.

Disabling Events

Using the Command Line

At the command line, enter:

```
pmctrl –e –a full_on
```

Using SMIT

1. Type `smit` from the command line.
2. Select **Performance and Resource Scheduling**.
3. Select **Power Management**.
4. Select **Enable/Disable Power Management**.
5. Enter `disable` in the field:

   ```
   Enable/Disable Power Management
   ```
6. Confirm your choice to disable Power Management.

Configuring and Unconfiguring Power Management

You can configure and unconfigure Power Management from the command line or with SMIT.

Note: GUI cannot be used to configure Power Management.

Prerequisite

You must have root user authority to configure or unconfigure Power Management.

Configuring Power Management

Using the Command Line

Configuring Power Management will cause the power management kernel extensions to be loaded and the pmc0 or power management controller device to go to the available state.

```
mkdev –l pmc0
```

Using SMIT

1. Type `smit` from the command line.
2. Select **Performance and Resource Scheduling**.
3. Select **Power Management**.
4. Select **Configure/Unconfigure Power Management**.
5. Select **Configure Power Management** to start configuring the Power Management subsystem.

Unconfiguring Power Management

Using the Command Line

Unconfiguring Power Management causes the power management kernel extensions to be unloaded and the pmc0 or power management controller device to go to the defined state. The system operates as if Power Management were not installed. This is different from Disabling Power Management because when it is only disabled, the power management kernel extensions are still loaded, but not active.

Use the **rmdev** command to unconfigure the Power Management subsystem:

```
rmdev -l pmc0
```

Using SMIT

1. Type `smit` from the command line.
2. Select **Performance and Resource Scheduling**.
3. Select **Power Management**.
4. Select **Configure/Unconfigure Power Management**.
5. Select **Unconfigure Power Management** to start unconfiguring the Power Management subsystem.

Starting System State Transition from the Enable State

System state transition can be initiated to one of the following states: standby, suspend, hibernation, or shutdown.

Note: All of the states may not be supported on a given platform.

You can perform this task from the command line or with SMIT.

Prerequisite

None, but permission for general user is set by root user.

Using the Command Line

Use the **pmctrl** command to start system suspend state by entering:

```
pmctrl –e –a suspend
```

Using SMIT

1. Type `smit` from the command line.
2. Select **Performance and Resource Scheduling**.
3. Select **Power Management**.
4. Select **System State Transition from the Enable State**.
5. Enter the state (standby, suspend, hibernation, or shutdown).
6. Confirm your choice to start the system state transition.

Changing/Showing Characteristics of Power Management

This procedure describes how to change parameters such as system idle time, action when main power is switched off, limitation of state transition for general users, enable or disable resume password, TTY session handling for suspend or hibernation, and so on. You can perform this task from the command line or with SMIT.

Prerequisite

You must have root user authority to change parameters.

Using the Command Line

Use the **pmctrl** command to change parameters:

- Change system idle time to 20 minutes by entering:

  ```
  pmctrl –t 20
  ```

- Enable password checking on resume by entering:

  ```
  pmctrl –w on
  ```

- Terminate TTY session when system enters suspend or hibernation by entering:

  ```
  pmctrl –y on
  ```

Using SMIT

1. Type `smit` from the command line.
2. Select **Performance and Resource Scheduling**.
3. Select **Power Management**.
4. Select **Change/Show Characteristics of Power Management**.
5. Select the appropriate values.
6. Confirm your choice to change the parameters.

Changing Power Management Timer

Power Management has two timers for executing system state transitions at a specified time. The timers are resume timer and suspend or hibernation timer. These timers can be disabled by setting the time to zero. You can change the timer setting from the command line or with SMIT.

Prerequisite

You must have root user authority to change timer setting.

Using the Command Line

Use the **pmctrl** command to change the timer setting.

- Set the resume time to 5:00 a.m. every day by entering:

  ```
  pmctrl –R 0500
  ```

- Set the hibernation time to 5:00 p.m. on March 13, 1998 by entering:

  ```
  pmctrl –S 9503131700 hibernation
  ```

Using SMIT

1. Type `smit` from the command line.
2. Select **Performance and Resource Scheduling**.
3. Select **Power Management**.
4. Select **Power Management Timer**.
5. Select the appropriate action and values.
6. Confirm your choice to start the new setting.

Managing Display Power Management

Display Power Management System (DPMS) has three values that you can change: dim time, suspend time, and turn-off time. If Power Management is unconfigured, DPMS is still available. You can change the values from the command line or with SMIT.

Prerequisite

You must have root user authority to change the DPMS time setting.

Using the Command Line

Use the **pmctrl** command to set the dim, suspend, and turn-off time to 2, 3, and 4 by entering:

```
pmctrl -d lft0 -t 2 3 4
```

Note: If you want to change 1 or 2 of the values, you must still specify the values for the different timers. Set the values that you do not want to change to -1 and they will not be changed.

Use the following command if Power Management is unconfigured:

```
chdev -1 lft0 -P -a pwr_mgr_t1='2' -a pwr_mgr_t2='3'
```

Using SMIT

1. Type smit from the command line.
2. Select **Performance and Resource Scheduling**.
3. Select **Power Management**.
4. Select **Display Power Management**.
5. Specify the time settings.
6. Confirm your choice to change the time settings.

Changing Idle Time for Each Device

Each Power Management-aware device driver has a user configurable idle time. When a device has been idle for its configured idle time, it enters device idle mode and attempts to put itself in a power savings mode. You can alter the idle time of each device from the command line or with SMIT.

Prerequisite

You must have root user authority to change the idle time of each device.

Using the Command Line

Use the **pmctrl** command to change the idle and standby time of the CD-ROM device to 5 minutes and 2 minutes by entering:

```
pmctrl -d cd0 -t 5 2
```

Note: If you want to change `1` of the values, you must still specify the values for the different timers. Set the value that you do not want to change to `-1` and it will not be changed.

Using SMIT

1. Type `smit` from the command line.
2. Select **Performance and Resource Scheduling**.
3. Select **Power Management**.
4. Select **Power Management Characteristics of Each Device**.
5. Select the device from the list.
6. Enter the **device idle time** in minutes.
7. Confirm your choice to change the idle time.

Managing the Battery

You can display information about the battery or discharge the battery as described in this section.

Showing Battery Information

You can display information about the battery such as battery capacity, remaining capacity, and so on from the command line or with SMIT.

Prerequisite

None.

Using the Command Line

Use the **battery** command to show battery information by entering:

```
battery
```

Using SMIT

1. Type `smit` from the command line.
2. Select **Performance and Resource Scheduling**.
3. Select **Power Management**.
4. Select **battery**.
5. Select the **Show Battery Information**.
6. Confirm your choice.

Discharging the Battery

The battery can be discharged to prevent memory effect. You can perform this task from the command line or with SMIT.

Prerequisite

None.

Using the Command Line

Use the **battery** command to discharge the battery by entering:

```
battery –d
```

Using SMIT

1. Type `smit` from the command line.
2. Select **Performance and Resource Scheduling**.
3. Select **Power Management**.
4. Select **battery**.
5. Select the **Start Battery Discharge**.
6. Confirm your choice to start battery discharging.

Power Management Limitation Warnings

Users of Power Management need to be aware of the following limitations:

Changing configuration during suspend/hibernation

Altering the system configuration, such as memory size, devices, and so on, while the system is in the suspend or hibernation state can cause unpredictable results. This could cause loss of data, file system corruption, system crashes, or a failure to resume from the suspend or hibernation state.

Non-PM-aware device drivers

If a device driver is installed that is not Power Management-aware, unpredictable results could occur when resuming from suspend or hibernation. If a non-PM-aware device driver is to be installed, the suspend and hibernation states must never be used. The following command can be run with root authority to disable the suspend and hibernation states effective on the next system boot.

The following command frees the hibernation logical volume and disallows future selections of the suspend or hibernation states:

```
/usr/lib/boot/disable_hibernation
```

If you want to re–enable these functions, the following command will enable the suspend and hibernation states effective on the next system boot, provided the hardware platform supports such states:

```
/usr/lib/boot/enable_hibernation
```

Booting from CD-ROM or other media after hibernation

Accessing the **rootvg** from maintenance mode such as CD-ROM boot when a valid hibernation image exists can result in loss of data and file system corruption.

Maintenance modes should only be used after normal system shutdown or power-off, not after a hibernation power-off.

Network connections during suspend/hibernation

Network connections are disconnected during the suspend and hibernation states. These connections may have to be re-established by the user after resuming. Since locally cached data won't be available to other nodes on the network during this time and network activity cannot be monitored by the local node during this time, it is recommended that the suspend and hibernation states not be used when using network interfaces such as TCP/IP, NFS, AFS, DCE, SNA, OSI, NetWare, NetBIOS, and so on.

The following command frees the hibernation logical volume and disallows future selections of the suspend or hibernation states:

```
/usr/lib/boot/disable_hibernation
```

If you want to re–enable these functions, the following command will enable the suspend and hibernation states effective on the next system boot, provided the hardware platform supports such states:

```
/usr/lib/boot/enable_hibernation
```

Chapter 15. AIX for BSD System Administrators

This chapter explains the differences and the similarities between AIX and 4.3 BSD UNIX or System V operating systems. First it describes the major differences between 4.3 BSD Systems and AIX. Then it provides more detailed information on the differences for many UNIX features. These features, which are presented in alphabetical order, include:

- accounting
- backup
- boot and startup
- commands for AIX system administration
- cron
- devices
- file comparison table for 4.3 BSD, SVR4, and AIX
- file systems
- finding and examining files
- networking
- NFS and NIS
- online documentation and the man command
- paging space
- passwords
- performance measurement and tuning
- printers
- terminals
- UUCP

Introduction to AIX for BSD System Managers

The following hints will help you get started managing the system:

- You should start by logging in as root at the graphics console.
- You should perform system management from the system console until you become experienced with the system. It is easier to work from the system console than a remote terminal. Once you are experienced with the system, you can work remotely from an xterm or an ASCII terminal.
- Several AIX facilities are involved in system management tasks. They include:
  - The System Management Interface Tool (SMIT)

    SMIT provides an interface between system managers and configuration and management commands. SMIT can help system managers perform most system administration tasks. For more information, see the “System Management Interface Tool (SMIT)” on page 3-2.
  - The Object Data Manager (ODM)

    The ODM provides routines that access objects from the ODM databases. The ODM databases contain device configuration information.
  - The System Resource Controller (SRC)

    The SRC provides access and control of daemons and other system resources through a single interface. For more information, see the “System Resource Controller Overview” on page 8-2.

Major Differences between 4.3 BSD and AIX

This section summarizes the major differences between AIX and 4.3 BSD systems.

Configuration Data Storage

4.3 BSD usually stores configuration data in ASCII files. Related pieces of information are kept on the same line and record processing (sorting and searching) can be done on the ASCII file itself. Records can vary in length and are terminated by a line feed. 4.3 BSD provides tools to convert some potentially large ASCII files to a database (dbm) format. Relevant library functions search the pair of dbm files if they exist, but search the original ASCII file if the dbm files are not found.

Some AIX configuration data is stored in ASCII files, but often in a *stanza* format. A stanza is a set of related pieces of information stored in a group of several lines. Each piece of information has a label to make the contents of the file more understandable.

AIX also supports dbm versions of password and user information. Furthermore, the **/etc/passwd**, **/etc/group**, and **/etc/inittab** files are examples of AIX files where the information is stored in traditional form rather than in stanza form.

Other AIX configuration data are stored in files maintained by the Object Data Manager (ODM). The System Management Interface Tool (SMIT) can manipulate and display information in ODM files. Alternately, you can use the ODM commands directly to view these files. To query the ODM files, use the following commands:

- **odmget**
- **odmshow**

The following ODM commands alter ODM files:

- **odmadd**
- **odmcreate**
- **odmdrop**
- **odmchange**
- **odmdelete**

Note: Altering ODM files incorrectly may cause the system to fail, and may prevent you from successfully restarting the system. You should only use ODM commands directly on ODM files when task-specific commands, such as those generated by SMIT, are unsuccessful.

Configuration Management

When an AIX system starts up, a set of configuration-specific commands is invoked by the Configuration Manager. These configuration-specific commands are called *methods*. Methods identify the devices on the system and update the appropriate ODM files in the **/etc/objrepos** directory.

Device special files in the **/dev** directly are not pre-installed. Some special files, such as those for hard disks, are created automatically during the start-up configuration process. Other special files, such as those for ASCII terminals, must be created by the system administrator by using the SMIT Devices menu. This information is retained in the ODM for later use by the system.

Disk Management

In AIX, disk drives are referred to as *physical volumes*. Partitions are referred to as *logical volumes*. As in 4.3 BSD, a single physical volume can have multiple logical volumes. However, unlike 4.3 BSD, a single volume in AIX can span multiple physical volumes. To do this, you must make several physical volumes into a *volume group* and create logical volumes on the volume group.

AIX commands used for file system and volume management include:

- **crfs**
- **varyonvg**
- **varyoffvg**
- **lsvg**
- **importvg**
- **exportvg**

The following 4.3 BSD commands are also available:

- **mkfs**
- **fsck**
- **fsdb**
- **mount**
- **umount**

Differences between the 4.3 BSD version and the AIX version of these commands are discussed in “File Systems” on page 15-18.

4.3 BSD maintains a list of file systems in the **/etc/fstab** file. AIX maintains a stanza for each file system in the **/etc/filesystems** file.

The 4.3 BSD file system usually reads in large blocks of 8KB, but can store several small files in one block using *fragments* usually consisting of 1KB. The AIX file system does not support fragments and each file consumes at least one block. The block size is 4KB.

New Commands

To handle new configuration and disk management systems, AIX has about 150 new commands that are new to 4.3 BSD administrators. For more information, see “Commands for AIX System Administration” on page 15-10.

Boot and Startup

AIX supports automatic identification and configuration of devices. Consequently, the boot and startup process is very different from 4.3 BSD systems. In addition to the kernel, an image of a boot file system and the previous base device configuration information is loaded to a RAM disk. In the first phase of startup, sufficient configuration information is loaded and checked to permit accessing logical volumes. The paging space device is identified to the kernel and the hard disk root file system is checked. At this time, AIX changes the root file system from the RAM disk to the hard disk and completes the startup procedure, including configuring other devices.

User Authorization

4.3 BSD, and versions of AT&T UNIX operating systems prior to SVR4, store all user authentication information, including encrypted passwords, in the **/etc/passwd** file. Traditionally, the **/etc/passwd** file could be read by all.

On SVR4 systems, encrypted passwords are removed from the **/etc/passwd** file and stored in the **/etc/shadow** file. Only users with root authority and trusted programs (such as the **/bin/login** program) can read the **/etc/shadow** file.

AIX stores encrypted passwords in the **/etc/security/passwd** file. Other files in the **/etc/security** directory are the **user** and **limits** files. These three files define the way a user is allowed to access the system (such as using the **rlogin** or **telnet** commands) and the user's resource limits (such as file size and address space).

Printing

Most 4.3 BSD printing commands are supported with minor differences. One difference is that the **/etc/qconfig** file is the configuration file in AIX.

The AIX line printing system can inter-operate with the 4.3 BSD line printing system, both for submitting print jobs to 4.3 BSD systems and for printing jobs submitted from a 4.3 BSD system.

Shells

AIX supports the Bourne shell, C shell and Korn shell. The full path name for the Bourne shell program is **/bin/bsh**. The **/bin/sh** file is a hard link to the **/bin/ksh** file. This file may be changed by the administrator.

Notes:

1. AIX has no shell scripts that rely on the **/bin/sh**. However, many shell scripts from other systems rely on **/bin/sh** being the Bourne shell.
2. Although the Bourne shell and Korn shell are similar, the Korn shell is not a perfect superset of the Bourne shell.

Accounting

Both the AIX accounting files in the **/usr/lib/acct** directory and the system activity reporting tools in the **/usr/lib/sa** directory are identical to those available with AT&T System V Release 4 (SVR4) with the addition of 4.3 BSD accounting utilities.

Many of the accounting commands are in the **/usr/lib/acct** directory. To begin system accounting, use the **/usr/lib/acct/startup** command. If accounting is not started, commands such as **lastcomm**(1) cannot return information.

AIX provides these 4.3 BSD accounting facilities:

l**ast**(1)	indicates last logins of users and terminals
lastcomm(1)	shows in reverse order the last commands executed
acct(3)	enables and disables process accounting
ac(8)	login accounting
accton(8)	turns system accounting on or off
sa(8)	generally maintains system accounting files

AIX also provides these System V Interface Definition (SVID) Issue II accounting commands and library functions:

acctcms(1)	produces command usage summaries from accounting records
acctcom(1)	displays selected process-accounting record summaries
acctcon1(1)	converts login/logoff records to session records
acctcon2(1)	converts login/logoff records to total accounting records
acctdisk(1)	generates total accounting records from **diskusg**(1) command output
acctmerg(1)	merges total accounting files into an intermediary file
accton(1)	turns on accounting
acctprc1(1)	processes accounting information from **acct**(3) command
acctprc2(1)	processes output of **acctprc1**(1) command into total accounting records
acctwtmp(1)	manipulates connect-time accounting records
chargefee(1)	charges to login name
ckpacct(1)	checks size of **/usr/adm/pacct** file
diskusg(1)	generates disk accounting information
dodisk(1)	performs disk accounting
fwtmp(1)	converts binary records (**wtmp** file) to formatted ASCII
	Note: The **wtmp** file is in the **/var/adm** directory.
lastlogin(1)	updates last date on which each person logged in
monacct(1)	creates monthly summary files
prctmp(1)	prints session record file produced by **acctcon1**(1) command
prdaily(1)	formats a report of yesterday's accounting information
prtacct(1)	formats and prints any total accounting file

runacct(1)	runs daily accounting
shutacct(1)	called by system shutdown to stop accounting and log the reason
startup(1)	called by system initialization to start accounting
turnacct(1)	turns process accounting on or off
wtmpfix(1)	corrects time/date stamps in a file using **wtmp** format

Backup

The **tar** and **cpio** commands can move data between systems. The AIX **tar** command is not fully compatible with the 4.3 BSD **tar** command. The AIX **tar** command requires the **–B** option (blocking input) if it is reading from a pipe. The AT&T **cpio** command is compatible with the AIX version.

AIX can read and write in **dump** and **restore** command format. For example, the AIX **backup** command with the syntax:

```
backup –0uf Device FilesystemName
```

is the same as the 4.3 BSD **dump** command with the syntax:

```
dump 0uf Device FilesystemName
```

Similarly, the AIX **restore** command with the syntax:

```
restore –mivf Device
```

is the same as the 4.3 BSD **restore** command with the syntax:

```
restore ivf Device
```

AIX also has the 4.3 BSD **rdump** and **rrestore** commands. The only difference between the two versions is that for AIX each argument must be preceded by a **–** (dash). For example, the following command:

```
rdump –0 –f orca:/dev/rmt0 /dev/hd2
```

is equivalent to the 4.3 BSD command:

```
rdump 0f orca:/dev/rmt0 /dev/hd2
```

The AIX **backup** command with the following syntax:

```
backup –0f /dev/rmt0 /dev/hd2
```

is equivalent to the 4.3 BSD **dump** command with this syntax:

```
dump 0f /dev/rmt0 /dev/hd2
```

Non-IBM SCSI Tape Support

AIX does not directly support non-IBM SCSI tape drives. However, you can add your own header and interface that use the IBM SCSI driver. For more information, see the information on adding an unsupported device to the system in *Kernel Extensions and Device Support Programming Concepts*.

Boot and Startup

On 4.3 BSD systems, the **init** program is the last step in the boot procedure. The main role of the **init** program is to create processes for each available terminal port. The available terminal ports are found by reading the **/etc/ttys** file.

On System V, the **init** program is started at system initialization. The **init** process starts processes according to entries in the **/etc/inittab** file.

AIX follows the System V initialization procedure. You can edit the AIX **/etc/inittab** file by directly editing the file, using the **telinit** command, or by using the following AIX commands:

chitab(1)	changes records in the **/etc/inittab** file
lsitab(1)	lists records in the **/etc/inittab** file
mkitab(1)	makes records in the **/etc/inittab** file
rmitab(1)	removes records in the **/etc/inittab** file

Changes made to the **/etc/inittab** file take effect the next time the system is rebooted, or when the **telinit q** command is run.

Commands for AIX System Administration

This list contains commands that are specifically for administering the AIX environment.

bosboot(1)	initializes a boot device
bootlist(1)	alters the list of boot devices (or the ordering of these devices in the list) available to the system
cfgmgr(1)	configures devices by running the programs in the **/etc/methods** directory
chcons(1)	redirects the system console to device or file, effective next startup
chdev(1)	changes a device's characteristics
chdisp(1)	changes the display used by the low-function terminal (LFT) subsystem
checkcw(1)	prepares constant-width text for the **troff** command
checkeq(1)	checks documents formatted with memorandum macros
checkmm(1)	checks documents formatted with memorandum macros
checknr(1)	checks **nroff** and **troff** files
chfont(1)	changes the default font selected at boot time
chfs(1)	changes attributes of a file system
chgroup(1)	changes attributes for groups
chgrpmem(1)	changes the administrators or members of a group
chhwkbd(1)	changes the low function terminal (LFT) keyboard attributes stored in the Object Data Manager (ODM) database
chitab(1)	changes records in the **/etc/inittab** file
chkbd(1)	changes the default keyboard map used by the low-function terminal (LFT) at system startup
chkey(1)	changes your encryption key
chlang	sets **LANG** environment variable in **/etc/environment** file for next login
chlicense(1)	There are two types of user licensing, fixed and floating. Fixed licensing is always enabled, and the number of licenses can be changed through the **–u** option. Floating licensing can be enabled or disabled (on or off) through the **–f** option.
chlv(1)	changes the characteristics of a logical volume
chnamsv(1)	changes TCP/IP-based name service configuration on a host
chprtsv(1)	changes a print service configuration on a client or server machine
chps(1)	changes attributes of a paging space
chpv(1)	changes the characteristics of a physical volume in a volume group
chque(1)	changes the queue name
chquedev(1)	changes the printer or plotter queue device names
chssys(1)	changes a subsystem definition in the subsystem object class
chtcb(1)	changes or queries the trusted computing base attribute of a file

chtz	changes the system time zone information
chuser(1)	changes attributes for the specified user
chvfs(1)	changes entries in the **/etc/vfs** file
chvg(1)	sets the characteristics of a volume group
chvirprt(1)	changes the attribute values of a virtual printer
crfs(1)	adds a file system
crvfs(1)	creates entries in the **/etc/vfs** file
exportvg(1)	exports the definition of a volume group from a set of physical volumes
extendvg(1)	adds physical volumes to a volume group
grpck(1)	verifies the correctness of a group definition
importvg(1)	imports a new volume group definition from a set of physical volumes
lsallq(1)	lists the names of all configured queues
lsallqdev(1)	lists all configured printer and plotter queue device names within a specified queue
lsdisp(1)	lists the displays currently available on the system
lsfont(1)	lists the fonts available for use by the display
lsfs(1)	displays the characteristics of file systems
lsgroup(1)	displays the attributes of groups
lsitab(1)	lists the records in the **/etc/inittab** file
lskbd(1)	lists the keyboard maps currently available to the low-function terminal (LFT) subsystem
lslicense(1)	displays the number of fixed licenses and the status of floating licensing
lslpp(1)	lists optional program products
lsnamsv(1)	shows name service information stored in the database
lsprtsv(1)	shows print service information stored in the database
lsps	lists paging space and attributes
lsque(1)	displays the queue stanza name
lsquedev(1)	displays the device stanza name
lssrc(1)	gets the status of a subsystem, a group of subsystems, or a subserver
lsuser(1)	displays attributes of user accounts
lsvfs(1)	lists entries in the **/etc/vfs** file
mkcatdefs(1)	preprocesses a message source file
runcat(1)	pipes the output data from the **mkcatdefs** command to the **gencat** command
mkdev(1)	adds a device to the system
mkfont(1)	adds the font code associated with a display to the system
mkfontdir(1)	creates a **fonts.dir** file from a directory of font files

mkgroup(1)	creates a new group
mkitab(1)	makes records in the **/etc/inittab** file
mklv(1)	creates a logical volume
mklvcopy(1)	adds copies to a logical volume
mknamsv(1)	configures TCP/IP-based name service on a host for a client
mknotify(1)	adds a notify method definition to the notify object class
mkprtsv(1)	configures TCP/IP-based print service on a host
mkps(1)	add an additional paging space to the system
mkque(1)	adds a printer queue to the system
mkquedev(1)	adds a printer queue device to the system
mkserver(1)	adds a subserver definition to the subserver object class
mkssys(1)	adds a subsystem definition to the subsystem object class
mksysb	backs up mounted file systems in the **rootvg** volume group for subsequent reinstallation
mkszfile	records size of mounted file systems in the **rootvg** volume group for reinstallation
mktcpip(1)	sets the required values for starting TCP/IP on a host
mkuser(1)	creates a new user account
mkuser.sys(1)	customizes a new user account
mkvg(1)	creates a volume group
mkvirprt(1)	makes a virtual printer
odmadd(1)	adds objects to created object classes
odmchange(1)	changes the contents of a selected object in the specified object class
odmcreate(1)	produces the **.c** (source) and **.h** (include) files necessary for ODM application development and creates empty object classes
odmdelete(1)	deletes selected objects from a specified object class
odmdrop(1)	removes an object class
odmget(1)	retrieves objects from the specified object classes and places them into an **odmadd** input file
odmshow(1)	displays an object class definition on the screen
pwdck(1)	verifies the correctness of local authentication information
redefinevg	redefines the set of physical volumes of the given volume group in the device configuration database
reducevg(1)	removes physical volumes from a volume group When all physical volumes are removed from the volume group, the volume group is deleted.
reorgvg(1)	reorganizes the physical partition allocation for a volume group
restbase(1)	restores customized information from the boot image

rmdel(1)	removes a delta from a Source Code Control System (SCCS) file
rmdev(1)	removes a device from the system
rmf(1)	removes folders and the messages they contain
rmfs(1)	removes a file system
rmgroup(1)	removes a group
rmitab(1)	removes records in the **/etc/inittab** file
rmlv(1)	removes logical volumes from a volume group
rmlvcopy(1)	removes copies from a logical volume
rmm(1)	removes messages
rmnamsv(1)	unconfigures TCP/IP-based name service on a host
rmnotify(1)	removes a notify method definition from the notify object class
rmprtsv(1)	unconfigures a print service on a client or server machine
rmps(1)	removes a paging space from the system
rmque(1)	removes a printer queue from the system
rmquedev(1)	removes a printer or plotter queue device from the system
rmserver(1)	removes a subserver definition from the subserver object class
rmssys(1)	removes a subsystem definition from the subsystem object class
rmuser(1)	removes a user account
rmvfs(1)	removes entries in the **/etc/vfs** file
rmvirprt(1)	removes a virtual printer
savebase(1)	saves base customized device data in the ODM onto the boot device
syncvg(1)	synchronizes logical volume copies that are not current
usrck(1)	verifies the correctness of a user definition
varyoffvg(1)	deactivates a volume group
varyonvg(1)	activates a volume group

Cron

The AIX **cron** daemon is similar to the System V Release 2 **cron** daemon. An entry in the **/etc/inittab** file starts the **cron** daemon.

Devices

A device on a 4.3 BSD system is accessible to an application only when:

- The device is physically installed and functioning.
- The driver for the device is in the kernel.
- The device special files for the device exist in the **/dev** directory.

A device on an AIX is accessible to an application only when:

- The device is physically installed and functioning.
- The driver for the device is in the kernel or in a loaded kernel extension.
- The device special files for the device exist in the **/dev** directory.
- The object database in the **/etc/objrepos** directory contains entries for the device that match the physical configuration.

The device specific programs called *methods*, found in the **/etc/methods** directory, maintain the object database. The methods are invoked by the Configuration Manager (accessed through the **cfgmgr** command) and other commands.

If a device can no longer be accessed by an application program, it may mean the hardware is faulty or it may mean that the configuration database in the **/etc/objrepos** directory is damaged.

The **cfgmgr** command processes the configuration database in the **/etc/objrepos** directory and is processed at startup time by the **cfgmgr** command (the Configuration Manager).

The pseudocode below shows the Configuration Manager logic:

```
/* Main */
While there are rules in the Config_Rules database
        {
        Get the next rule and execute it
        Capture stdout from the last execution
        Parse_Output(stdout)
        }
/* Parse Output Routine */
/* stdout will contain a list of devices found */
Parse_OutPut(stdout)
        {
        While there are devices left in the list
                {
                Lookup the device in the database
                if (!defined)
                        Get define method from database and
execute
                if (! configured)
                        {
                        Get config method from database and
execute
                        Parse_Output(stdout)
                         }
                }
        }
```

File Comparison Table for 4.3 BSD, SVR4, and AIX

The following table compares file names and functions between the 4.3 BSD, SVR4, and AIX operating systems.

File Comparison Table				
4.3 BSD File	**SVR4 File**	**AIX File**	**Database**	**Type (odm/dbm)**
L–Devices	Devices	Devices	no	
L–dialcodes	Dialcodes	Dialcodes	no	
L.cmds	Permissions	Permissions	no	
L.sys	Systems	System	no	
USERFILE	Permissions	Permissions	no	
aliases	mail/namefiles	aliases	aliasesDB/DB	dbm
fstab	vfstab	filesystems	no	
ftpusers	ftpusers	ftpusers	no	
gettytab		N/A		
group	group	group	no	
hosts	hosts	hosts	no	
hosts.equiv	hosts.equiv	hosts.equiv	no	
inetd.conf	inetd.conf	inetd.conf	no	
map3270	N/A	map3270	no	
motd	motd	motd	no	
mtab	mnttab	N/A	no	
named.boot	named.boot	named.boot	no	
named.ca		named.ca	no	
named.hosts		named.data (See note)	no	
named.local		named.local	no	
named.pid	named.pid	named.pid	no	
named.rev		named.rev	no	
networks	networks	networks	no	
passwd	passwd	passwd	no	
printcap	qconfig	qconfig		
protocols		protocols	no	
remote	remote	remote	no	
resolv.conf	resolv.conf	resolv.conf	no	
sendmail.cf	sendmail.cf	sendmail.cf	sendmail.cfDB	neither
services		services	no	
shells	shells	N/A		
stab		N/A		

4.3 BSD File	SVR4 File	AIX File	Database	Type (odm/dbm)
syslog.conf		syslog.conf	no	
syslog.pid		syslog.pid	no	
termcap	terminfo	terminfo		
ttys	ttys	N/A	yes	odm
types		N/A		
utmp	utmp	utmp		
vfont		N/A		
vgrindefs		vgrindefs		
wtmp	wtmp	wtmp		

Note: The file names **named.ca**, **named.hosts**, **named.local**, and **named.rev** are user definable in the **named.boot** file. However, these are the names used for these files in the AIX documentation.

File Systems

This information offers a brief comparison of AIX file systems to other systems' file systems and provides an outline of the supported file system types on AIX systems.

The AIX system uses the **/etc/filesystem** file to list file system device information, and has similar commands for mounting and unmounting file systems.

/etc/filesystems File and /etc/fstab File

4.3 BSD systems store lists of block devices and mount points in the **/etc/fstab** file.

SVR4 systems store block devices and mount point information in the **/etc/vfstab** file.

AIX stores block device and mount point information in the **/etc/filesystems** file. The **crfs**, **chfs**, and **rmfs** commands update the **/etc/filesystems** file.

4.3 BSD system administrators may be interested in the **check** variable in the **/etc/filesystems** file. The **check** variable can be set to the value True, False or to a number. For example, you can specify `check=2` in the **/etc/filesystems** file. The number specifies the pass of the **fsck** command that will check this file system. The **check** parameter corresponds to the fifth field in an **/etc/fstab** file record.

There is no dump frequency parameter in the **/etc/filesystems** file.

File System Support on AIX

AIX supports disk quotas. AIX does not allow mounting of diskettes as file systems.

The syntax of the AIX **mount** and **umount** commands differs from 4.3 BSD and from SVR4 versions of these commands. The commands to mount and unmount all file systems at once are shown for all three systems in the following table:

mount and unmount Commands			
Function	**AIX Syntax**	**4.3 BSD Syntax**	**SVR4 Syntax**
mount all file systems	**mount all**	**mount –a**	**mountall**
unmount all file systems	**umount all**	**umount –a**	**umountall**

See the "File Systems" chapter in *Managing System Storage* for more information.

Finding and Examining Files

AIX supports the following 4.3 BSD file commands:

- **which**
- **whereis**
- **what**
- **file**

AIX does not support the 4.3 BSD **fast find** syntax of the **find** command. At this time, there is no replacement function. The following **ffind** shell script may be used to simulate the functionality:

```
#!/bin/bsh
PATH=/bin
for dir in /bin /etc /lib /usr
do
find $dir –print | egrep $1
done
```

The syntax for the **ffind** script is:

```
ffind FileName
```

Paging Space

The following AIX commands assist in managing paging space (also known as swap space):

chps(1)	changes attributes of a paging space
lsps(1)	lists attributes of a paging space
mkps(1)	adds an additional paging space to the system
rmps(1)	removes a paging space from the system
swapon(1)	specifies additional devices for paging and swapping

If a large paging space is required, you should place one paging logical volume for each hard disk. This allows scheduling of paging across multiple disk drives.

Networking

This section describes how to use 4.3 BSD ASCII network configurations on an AIX system, additional AIX commands and command options, name and address resolution on AIX systems, and differences between 4.3 BSD network management and AIX network management.

How to Change Default Startup to Permit 4.3 BSD ASCII Configurations

You can administer AIX network interfaces through the SMIT and ODM files, or through 4.3 BSD ASCII configuration files.

To administer network interfaces through 4.3 BSD ASCII configuration files, uncomment the commands in the **/etc/rc.net** file below the heading:

```
# Part II – Traditional Configuration
```

Then if you want flat file configuration and SRC support, edit the **/etc/rc.net** file and uncomment the **hostname**, **ifconfig**, and **route** commands with the appropriate parameters.

If you want flat file configuration without SRC support, use the **smit setbootup_option** fast path to change the system to BSD-style **rc** configuration. This option configures the system to use the **/etc/rc.bsdnet** file at startup. You will also have to edit the **/etc/rc.bsdnet** file and uncomment the **hostname**, **ifconfig**, and **route** commands with the appropriate parameters.

Additional Options for ifconfig and netstat Commands

The AIX **ifconfig** command has the following additional options:

mtu — The **mtu** variable specifies the maximum transmission unit (MTU) used on the local network (and local subnets) and the MTU used for remote networks. To maximize compatibility with Ethernet and other networks, set both the Token-Ring and Ethernet default **mtu** value to 1500.

allcast — The **allcast** flag sets the Token-Ring broadcast strategy. Setting the **allcast** flag optimizes connectivity through Token-Ring bridges. Clearing the **allcast** flag (by specifying **–allcast**) minimizes excess traffic on the ring.

The AIX **netstat** command has the **–v** flag. The **netstat –v** command prints driver statistics such as transmit byte count, transmit error count, receive byte count, and receive error count.

Additional Network Management Commands

The following additional commands are supported on AIX:

securetcpip The **securetcpip** shell script enables controlled access mode, which provides enhanced network security. It disallows execution of several unsecured TCP/IP programs, such as the **tftp**, **rcp**, **rlogin**, and **rsh** programs. It also restricts the use of the **.netrc** file.

gated The **gated** command provides MIB support for SNMP.

no The **no** command sets network options that include:

- **dogticks** sets timer granularity for **ifwatchdog** routines
- **subnetsarelocal** determines if packet address is on the local network
- **ipsendredirects** specifies whether the kernel should send redirect signals
- **ipforwarding** specifies whether the kernel should forward packets
- **tcp_ttl** specifies the time-to-live for Transmission Control Protocol (TCP) packets
- **udp_ttl** specifies the time-to-live for User Datagram Protocol (UDP) packets
- **maxttl** specifies the time-to-live for Routing Information Protocol (RIP) packets
- **ipfragttl** specifies the time-to-live for Internet Protocol (IP) fragments
- **lowclust** specifies a low water mark for cluster **mbuf** pool
- **lowmbuf** specifies a low water mark for the **mbuf** pool
- **thewall** specifies the maximum amount of memory that will be allocated to the **mbuf** and cluster **mbuf** pool
- **arpt_killc** specifies the time in minutes before an inactive complete Address Resolution Protocol (ARP) entry will be deleted

iptrace The **iptrace** command provides interface-level packet tracing for Internet protocols.

ipreport The **ipreport** command formats the trace into human-readable form. An example of using this command is the following:

```
iptrace –i en0 /tmp/iptrace.log
# kill iptrace daemon
kill ‘ps ax | grep iptrace | awk ’{ print $1 }’‘
ipreport /tmp/iptrace.log | more
```

Name and Address Resolution

The **gethostbyname** and **gethostbyaddr** subroutines in the **libc** library provide support for Domain Name Service, Network Information Services (NIS, formerly called Yellow Pages), and the **/etc/hosts** database. If the **/etc/resolv.conf** file exists, the name server is always checked first. If the name is not resolved and NIS is running, NIS is checked. If NIS is not running, the **/etc/hosts** file is checked.

Differences between AIX and 4.3 BSD

On AIX systems, the network daemons are started from the **/etc/rc.tcpip** file, not the **/etc/rc.local** file. The **/etc/rc.tcpip** shell script is invoked from the **/etc/inittab** file, not the **/etc/rc** file.

If the System Resource Controller (SRC) is running, the TCP/IP daemons run under SRC control. If you do not want the TCP/IP daemons running under SRC control, use the **smit setbootup_option** fast path to change the system to BSD-style **rc** configuration.

These network management functions available on 4.3 BSD are supported by AIX:

- kernel-level SYSLOG logging facilities
- Xerox Network Systems (XNS) support
- access rights for UNIX domain sockets

The tn3270 Command

The **tn3270** command is a link to the **telnet** command, but it uses the **/etc/map3270** file and the current **TERM** environment variable value to provide 3270 keyboard mappings. Thus, the **tn3270** command operates exactly like the BSD version.

If you want to change the escape sequences from the defaults used by the **tn3270**, **telnet**, or **tn** commands, set the **TNESC** environment variable before starting these commands.

Online Documentation and man Command

AIX supports the **man –k**, **apropos**, and **whatis** commands, but the database used by these commands must first be created with the **catman –w** command.

The AIX **man** command first searches for flat text pages in the **/usr/man/cat?** files. Next, it searches **nroff**-formatted pages in **/usr/man/man?** files. New man pages can be added in flat text or **nroff** form.

Notes:

1. The **man** command text pages are not provided with the system. The **catman** command creates the database from these text pages. These pages can be either flat text pages stored in the **/usr/man/cat**? files or **nroff** formatted pages stored the in **/usr/man/man**? files.
2. The Text Formatting licensed program must be installed for the **nroff** command to be available for the **man** command to read **nroff**-formatted man pages.

AIX supports the InfoExplorer tool for viewing the full system documentation in hypertext format. The **info** command starts InfoExplorer. The **info** command works in either a window-based interface, or through a curses-based interface. These options can be overridden by options on the command line.

Optional software products are often stored in a subdirectory of the **/usr/lpp** directory. **README** files and other informational documents are often provided with each optional software product in subdirectories of the **/usr/lpp** directory.

You can use the fast path **smit lslpp** (or the **lslpp** command) to list the optional program products installed on the system.

NFS and NIS (formerly Yellow Pages)

Network File System (NFS) and Network Information Services (NIS) daemons are started from the **/etc/rc.nfs** file. However, before the NFS and NIS daemons can be started, the **portmap** daemon must be started in the **/etc/rc.tcpip** file. By default, the **/etc/rc.nfs** file is not invoked by the **/etc/inittab** file. If you add a line in the **/etc/inittab** file to invoke the **/etc/rc.nfs** script, it should be invoked after the **/etc/rc.tcpip** script.

If NIS is active, you should include a root entry prior to the **+**:: (plus sign, colon, colon) entry in the **/etc/passwd** file and a system entry prior to the **+**:: entry in the **/etc/group** file. This allows a system administrator to log in as root and make changes if the system is unable to communicate with the NIS server.

NFS can be configured by using the fast path **smit nfs**. The SMIT menus refer to NIS (formerly Yellow Pages) as NIS. Many of the NFS and NIS commands are found in the **/etc** and **/usr/etc** directories.

Some NFS environments use an **arch** command to identify machine families and types of machines. For example if you are using the IBM RISC System/6000, we suggest you specify the **power** identifier for family (CPU), and the **ibm6000** identifier for type (machine).

Passwords

The following information details the differences between managing passwords in AIX systems and 4.3 BSD systems.

Setting a User Password

When you use the AIX **/bin/passwd** command as the root user, you are prompted for the current root user password. An example of using the AIX **/bin/passwd** command follows:

```
# passwd cslater
Changing password for ”cslater”
Enter root’s Password or
cslater’s Old password:
cslater’s New password:
Re–enter cslater’s
new password:
#
```

The 4.3 BSD version does not prompt for the current root user password. An example of the 4.3 BSD version follows:

```
# passwd cslater
New password:
Retype new password:
#
```

Importing a 4.3 BSD Password File

You can import a 4.3 BSD password file by first copying it to the **/etc/passwd** file and entering:

```
pwdck –y ALL
```

Then the **/etc/security/limits** file must be updated with a null stanza for any new users. The **usrck** command does this, but using the **usrck** command can cause problems unless the **/etc/group** file is imported with the **/etc/passwd** file.

Note: If the **/etc/security/limits** file is modified, the stack must not exceed 65,536 bytes. If it does, running the **usrck** command may cause problems. Change the stack size to 65,536 and run **usrck** command again.

You should also run the **grpck** and **usrck** command to verify group and user attributes.

Editing the Password File

In AIX the **lsuser**, **mkuser**, **chuser**, and **rmuser** commands are provided for managing passwords. All of these commands can be used by running SMIT. However, all of these commands deal with only one user at a time.

Note: Using an editor to change several user name entries at one time requires editing of several files at once, because passwords are stored in the **/etc/security/passwd** file, authorization information is stored in the **/etc/security/user** file, and the remaining user data is stored in the **/etc/passwd** file.

AIX does not support the **vipw** command but does support the **mkpasswd** command. However, you can still administer passwords on an AIX system in a 4.3 BSD manner. Use the following procedure:

1. Put a 4.3 BSD password file in the **/etc/shadow** file.
2. Change the permissions to the file by entering:

   ```
   chmod 000 /etc/shadow
   ```

3. Place the following **vipw** shell script in the **/etc** directory:

```
------------------------------------------------------
#!/bin/bsh
#
# vipw for AIX V3. Uses pwdck for now. May use usrck someday
#
PATH=/bin:/usr/bin:/etc:/usr/ucb # Add to this if your editor is
                                 # some place else
if [ –f /etc/ptmp ] ; then
                echo ”/etc/ptmp exists. Is someone else using
vipw?”
        exit 1
fi
if [ ! –f /‘which ”$EDITOR” | awk ’{ print $1 }’‘ ] ; then
        EDITOR=vi
fi
cp /etc/shadow /etc/ptmp
if (cmp /etc/shadow /etc/ptmp) ; then
        $EDITOR /etc/ptmp
else
        echo cannot copy shadow to ptmp
        exit 1
fi
if (egrep ”^root:” /etc/ptmp >/dev/null) ; then
        cp /etc/ptmp /etc/shadow ; cp /etc/ptmp /etc/passwd
        chmod 000 /etc/passwd /etc/shadow
        pwdck –y ALL 2>1 >/dev/null # return code 114 may change
                rc=$?
        if [ $rc –eq 114 ]; then
                chmod 644 /etc/passwd
                        rm –f /etc/passwd.dir /etc/passwd.pag
                mkpasswd /etc/passwd
                        # update /etc/security/limits, or ftp
                        # will fail
                else
                         pwdck –y ALL
                fi
else
        echo bad entry for root in ptmp
fi
rm /etc/ptmp
------------------------------------------------------------
```

4. SMIT and the **mkuser**, **chuser**, **rmuser** do not use the **mkpasswd** command. If you use these and use the **vipw** shell script or the **mkpasswd** command, you must run:

   ```
   mkpasswd /etc/passwd
   ```

   to update the **/etc/passwd.dir** and **/etc/passwd.pag** files.

Warning: Initialization of the **IFS** variable and the **trap** statements guard against some of the common methods used to exploit security holes inherent in the **setuid** feature. However, the **vipw** and **passwd** shell scripts are intended for relatively open environments where

compatibility is an important consideration. If you want a more secure environment, use only the standard AIX commands.

5. Put the following **passwd** shell script in the **/usr/ucb** directory:

```
------------------------------------------------------
#!/bin/ksh
#
# matches changes to /etc/security/passwd file with changes to
#/etc/shadow
#
IFS=" "
PATH=/bin
trap "exit 2" 1 2 3 4 5 6 7 8 10 12 13 14 15 16 17 18 21 22 \
        23 24 25 27 28 29 30 31 32 33 34 35 36 60 61 62
if [ -n "$1" ]; then
        USERNAME=$1
else
        USERNAME=$LOGNAME
fi
if [ -f /etc/ptmp ]; then
        echo password file busy
        exit 1
fi
        trap "rm /etc/ptmp; exit 3" 1 2 3 4 5 6 7 8 10 12 13 \
                14 15 16 17 18 21 22 23 24 25 27 28 29 30 31 \
                32 33 34 35 36 60 61 62
if (cp /etc/security/passwd /etc/ptmp) ; then
        chmod 000 /etc/ptmp else
        rm -f /etc/ptmp exit 1
fi
if ( /bin/passwd $USERNAME ) ; then
        PW=' awk ' BEGIN { RS = "" }
                $1 == user { print $4 } ' user="$USERNAME:" \
/etc/security/passwd '
else
        rm -f /etc/ptmp
        exit 1
fi
rm -f /etc/ptmp
awk -F: '$1 == user { print $1":"pw":"$3 ":"$4":"$5":"$6":"$7 }
        $1 != user { print $0 }' user="$USERNAME" pw="$PW" \
                /etc/shadow > /etc/ptmp
chmod 000 /etc/ptmp
mv -f /etc/ptmp /etc/shadow
------------------------------------------------------
```

6. Change the permissions to the **passwd** script by entering:

```
chmod 4711 /usr/ucb/passwd
```

7. Ensure that each user's **PATH** environmental variable specifies that the **/usr/ucb** directory be searched prior to the **/bin** directory.

Performance Measurement and Tuning

All devices on AIX have attributes associated with them. To view device attributes, enter:

```
lsattr –E –l DeviceName
```

Any attributes with the value True can be modified with the command:

```
chdev –l DeviceName –a attr=value
```

Warning: Changing device parameters incorrectly can damage your system.

By default, the maximum number of processes per user is 40. The default value may be too low for users who have many windows open simultaneously. There is no SMIT interface to change the maximum number of processes per user, so you must do it from the command line. The following command can be used to change the value systemwide:

```
hdev –l sys0 –a maxuproc=100
```

This example changes the maximum number to 100. The new value is set once the system has rebooted.

To view the current setting of this and other system attributes type:

```
lsattr –E –l sys0
```

The **maxmbuf** attribute is not currently supported by the **mbuf** services.

AIX supports the **vmstat** and **iostat** commands, but not the **systat** command or load averages.

Printers

AIX printing is managed by programs and configurations in the **/usr/lpd** directory. The design, configuration, queueing mechanism, and daemon processes of the 4.3 BSD and AIX printer subsystems are different. However, they both use the **lpd** protocol for remote printing. Both systems use **/etc/hosts.lpd** if it exists and **/etc/host.equiv** otherwise. The AIX printer subsystem offers a gateway to 4.3 BSD printer subsystems, so AIX systems can submit print jobs to 4.3 BSD systems and accept print jobs submitted by 4.3 BSD systems.

The **/etc/printcap** file of 4.3 BSD does not exist in AIX. This file is a combination of a spooler configuration and a printer-capability data base. Users need to understand the format and keywords of the **printcap** file to set up a printer correctly.

The **/etc/qconfig** file of AIX contains only the spooler configuration information. The printer capability is defined in the ODM predefined/customized data base. You can use the **mkvirprt** (make virtual printer) command to define the capabilities of a particular printer to the system.

To make the printer **lp0** available to print on the remote host `viking`, put the following in a 4.3 BSD system **/etc/printcap** file:

```
lp0|Print on remote printer attached to viking:Z
:lp=:rm=viking:rp=lp:st=/usr/spool/lp0d
```

To do the same on an AIX system, put the following in the **/etc/qconfig** file:

```
lp0:
        device = dlp0
        host = viking
        rq = lp
dlp0:
        backend = /usr/lib/lpd/rembak.)b.lp
```

For more information about the printer subsystem, see *AIX Version 4.1 Guide to Printers and Printing*.

AIX supports the following printer commands and library functions:

cancel(1)	cancels requests to a line printer
chquedev(1)	changes the printer or plotter queue device names
chvirprt(1)	changes the attribute values of a virtual printer
disable(1)	disables a printer queue
enable(1)	enables a printer queue
hplj(1)	postprocesses **troff** output for HP LaserJetII with the K cartridge
ibm3812(1)	postprocesses **troff** output for IBM 3812 Mod 2 Pageprinter
ibm3816(1)	postprocesses **troff** output for IBM 3816 Pageprinter
ibm5587G(1)	postprocesses **troff** output for IBM 5587G with 32x32/24x24 cartridge
lp(1)	sends requests to a line printer
lpr(1)	enqueues print jobs
lprm(1)	removes jobs from the line printer spooling queue
lpstat(1)	displays line printer status information

lptest(1)	generates the line printer ripple pattern
lsallqdev(1)	lists all configured printer queue device names within a queue
lsvirprt(1)	displays the attribute values of a virtual printer
mkque(1)	adds a printer queue to the system
mkquedev(1)	adds a printer queue device to the system
mkvirprt(1)	makes a virtual printer
pac(1)	prepares printer/plotter accounting records
piobe(1)	print Job Manager for the printer backend
pioburst(1)	generates burst pages (header and trailer pages) for printer output
piocmdout(3)	subroutine that outputs an attribute string for a printer formatter
piodigest(1)	digests attribute values for a virtual printer definition and stores them
pioexit(3)	subroutine that exits from a printer formatter
pioformat(1)	drives a printer formatter
piofquote(1)	converts certain control characters destined for PostScript printers
piogetstr(3)	subroutine that retrieves an attribute string for a printer formatter
piogetvals(3)	subroutine that initializes Printer Attribute database variables for a printer formatter
piomsgout(3)	subroutine that sends a message from a printer formatter
pioout(1)	printer backend's device driver interface program
piopredef(1)	creates a predefined printer data stream definition
proff(1)	formats text for printers with personal printer data streams
prtty(1)	prints to the printer port of the terminal
qadm(1)	performs system administration for the printer spooling system
qconfig(4)	configures a printer queueing system
qstatus(1)	provides printer status for the print spooling system
restore(3)	restores the printer to its default state
rmque(1)	removes a printer queue from the system
rmquedev(1)	removes a printer or plotter queue device from the system
rmvirprt(1)	removes a virtual printer
splp(1)	changes or displays printer driver settings
xpr(1)	formats a window dump file for output to a printer

Terminals

Terminal Ports

Traditionally, 4.3 BSD system managers enable or disable terminal ports by modifying the **/etc/ttys** file and sending a **HUP** signal to the **init** program.

AIX stores terminal port information in the ODM and starts terminals when the **init** program reads the **/etc/inittab** file. In AIX, you should use SMIT to configure terminal ports.

There is no fixed mapping between the port and the device special file name in the **/dev** directory. Consequently, it is confusing to system managers who are new to AIX which port should be configured. When using SMIT, the first planar serial port (physically labeled s1) is referred to as location `00-00-S1`, adapter `sa0`, and port `s1` in the SMIT menus. The second planar serial port (physically labeled s2) is referred to as location `00-00-S2`, adapter `sa1`, and port `s2`.

Use the **penable** and **pdisable** commands to enable and disable a port.

termcap and terminfo

Like System V, AIX uses **terminfo** entries in **/usr/lib/terminfo/?/*** files. Users familiar with 4.3 BSD systems may find the following commands helpful:

captoinfo(1)	converts a **termcap** file to a **terminfo** file
tic(1)	translates the **terminfo** files from source to compiled format

AIX includes source for many **terminfo** entries. Some of these may need to be compiled with the **tic** command. The **termcap** file is provided in the **/lib/libtermcap/termcap.src** file.

Dave Regan has donated his program **untic** to the public domain. This program uncompiles **terminfo** entries so that the source form may be modified and recompiled with **tic**. It is available from sites that archive **comp.sources.unix**.

UUCP

AIX provides System V Basic Networking Utilities (BNU) which are often referred to as the HDB UUCP.

Dialers(4)	lists modems used for BNU remote communications links
Maxuuxqts(4)	limits the number of instances of the BNU **uuxqt** daemons that can run
Permissions(4)	specifies BNU command permissions for remote systems
Poll(4)	specifies when the BNU program should poll remote systems
Systems(4)	lists remote computers with which the local system can communicate
rmail(1)	handles remote mail received through BNU
uucheck(1)	checks for files and directories required by BNU
uuclean(1)	removes files from the BNU spool directory
uucleanup(1)	deletes selected files from the BNU spooling directory
uucpadm(1)	enters basic BNU configuration information
uudemon.admin(1)	provides periodic information on the status of BNU file transfers
uudemon.cleanu(1)	cleans up BNU spooling directories and log files
uudemon.hour(1)	initiates file transport calls to remote systems using the BNU program
uudemon.poll(1)	polls the systems listed in the BNU Poll file
uulog(1)	provides information about BNU file-transfer activities on a system
uupoll(1)	forces a poll of a remote BNU system
uuq(1)	displays the BNU job queue and deletes specified jobs from the queue
uusnap(1)	displays the status of BNU contacts with remote systems
uustat(1)	reports the status of and provides limited control over BNU operations

AIX also provides the 4.3 BSD **uuencode** and **uudecode** commands. The HDB **uugetty** command is not supported.

For more information, see the lists of BNU files, file formats, and directories in the *System User's Guide: Communications and Networks*.

Index

Symbols

A

K

L

M

N

P

R

S

T

U

V

W