System Administration Guide: Basic Administration

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

Part No: 819–2379–16
December 2008

081212@21808

Contents

Preface

System Administration Guide: Basic Administration is part of a set that includes a significant part of the Solaris™ system administration information. This guide contains information for both SPARC® based and x86 based systems.

This book assumes you have completed the following tasks:

- Installed the Solaris Express Operating System
- Set up all the networking software that you plan to use

For the Solaris release, new features that might be interesting to system administrators are covered in sections called *What's New in ... ?* in the appropriate chapters.

Note – This Solaris release supports systems that use the SPARC and x86 families of processor architectures: UltraSPARC®, SPARC64, AMD64, Pentium, and Xeon EM64T. The supported systems appear in the *Solaris OS: Hardware Compatibility Lists* at `http://www.sun.com/bigadmin/hcl`. This document cites any implementation differences between the platform types.

In this document these x86 related terms mean the following:

- "x86" refers to the larger family of 64-bit and 32-bit x86 compatible products.
- "x64" points out specific 64-bit information about AMD64 or EM64T systems.
- "32-bit x86" points out specific 32-bit information about x86 based systems.

For supported systems, see the *Solaris OS: Hardware Compatibility Lists*.

Who Should Use This Book

This book is intended for anyone responsible for administering one or more systems running the Solaris release. To use this book, you should have 1-2 years of UNIX® system administration experience. Attending UNIX system administration training courses might be helpful.

How the System Administration Guides Are Organized

Here is a list of the topics that are covered by the System Administration Guides.

Book Title	Topics
System Administration Guide: Basic Administration	User accounts and groups, server and client support, shutting down and booting a system, managing services, and managing software (packages and patches)
System Administration Guide: Advanced Administration	Terminals and modems, system resources (disk quotas, accounting, and crontabs), system processes, and troubleshooting Solaris software problems
System Administration Guide: Devices and File Systems	Removable media, disks and devices, file systems, and backing up and restoring data
System Administration Guide: IP Services	TCP/IP network administration, IPv4 and IPv6 address administration, DHCP, IPsec, IKE, Solaris IP filter, Mobile IP, IP network multipathing (IPMP), and IPQoS
System Administration Guide: Naming and Directory Services (DNS, NIS, and LDAP)	DNS, NIS, and LDAP naming and directory services, including transitioning from NIS to LDAP and transitioning from NIS+ to LDAP
System Administration Guide: Naming and Directory Services (NIS+)	NIS+ naming and directory services
System Administration Guide: Network Interfaces and Network Virtualization	Networking stack, NIC driver property configuration, network interface configuration, administration of VLANs and link aggregations, configuring WiFi wireless networking.
System Administration Guide: Network Services	Web cache servers, time-related services, network file systems (NFS and Autofs), mail, SLP, and PPP
System Administration Guide: Solaris Printing	Solaris printing topics and tasks, using services, tools, protocols, and technologies to set up and administer printing services and printers
System Administration Guide: Security Services	Auditing, device management, file security, BART, Kerberos services, PAM, Solaris Cryptographic Framework, privileges, RBAC, SASL, and Solaris Secure Shell
System Administration Guide: Virtualization Using the Solaris Operating System	Resource management features, which enable you to control how applications use available system resources; zones software partitioning technology, which virtualizes operating system services to create an isolated environment for running applications; and virtualization using Sun^{TM} xVM hypervisor technology, which supports multiple operating system instances simultaneously

Book Title	Topics
Solaris CIFS Administration Guide	Solaris CIFS service, which enables you to configure a Solaris system to make CIFS shares available to CIFS clients; and native identity mapping services, which enables you to map user and group identities between Solaris systems and Windows systems
Solaris Trusted Extensions Administrator's Procedures	System installation, configuration, and administration that is specific to Solaris Trusted Extensions
Solaris ZFS Administration Guide	ZFS storage pool and file system creation and management, snapshots, clones, backups, using access control lists (ACLs) to protect ZFS files, using ZFS on a Solaris system with zones installed, emulated volumes, and troubleshooting and data recovery

Related Third-Party Web Site References

Note – Sun is not responsible for the availability of third-party web sites mentioned in this document. Sun does not endorse and is not responsible or liable for any content, advertising, products, or other materials that are available on or through such sites or resources. Sun will not be responsible or liable for any actual or alleged damage or loss caused by or in connection with the use of or reliance on any such content, goods, or services that are available on or through such sites or resources.

Documentation, Support, and Training

The Sun web site provides information about the following additional resources:

- Documentation (`http://www.sun.com/documentation/`)
- Support (`http://www.sun.com/support/`)
- Training (`http://www.sun.com/training/`)

Typographic Conventions

The following table describes the typographic conventions that are used in this book.

TABLE P–1 Typographic Conventions

Typeface	Meaning	Example
`AaBbCc123`	The names of commands, files, and directories, and onscreen computer output	Edit your `.login` file. Use `ls -a` to list all files. `machine_name% you have mail.`

`AaBbCc123`	What you type, contrasted with onscreen computer output	`machine_name%` **`su`**
		`Password:`

aabbcc123	Placeholder: replace with a real name or value	The command to remove a file is `rm` *filename*.
AaBbCc123	Book titles, new terms, and terms to be emphasized	Read Chapter 6 in the *User's Guide*. A *cache* is a copy that is stored locally. Do *not* save the file. **Note:** Some emphasized items appear bold online.

Shell Prompts in Command Examples

The following table shows the default UNIX system prompt and superuser prompt for the C shell, Bourne shell, and Korn shell.

TABLE P–2 Shell Prompts

Shell	Prompt
C shell	`machine_name%`
C shell for superuser	`machine_name#`
Bourne shell and Korn shell	`$`
Bourne shell and Korn shell for superuser	`#`

General Conventions

Be aware of the following conventions used in this book.

- When following steps or using examples, be sure to type double-quotes ("), left single-quotes (‘), and right single-quotes (’) exactly as shown.
- The key referred to as Return is labeled Enter on some keyboards.

- The root path usually includes the `/sbin`, `/usr/sbin`, `/usr/bin`, and `/etc` directories, so the steps in this book show the commands in these directories without absolute path names. Steps that use commands in other, less common, directories show the absolute paths in the examples.
- The examples in this book are for a basic SunOS software installation without the Binary Compatibility Package installed and without `/usr/ucb` in the path.

Caution – If `/usr/ucb` is included in a search path, it should always be at the end of the search path. Commands like `ps` or `df` are duplicated in `/usr/ucb` with different formats and options from the SunOS commands.

CHAPTER 1

Solaris Management Tools (Road Map)

This chapter provides a roadmap to Solaris management tools.

What's New in Solaris Management Tools?

No new Solaris Management tools have been introduced in this Solaris release.

These tools are new or changed in the Solaris 10 initial 3/05 release:

- `admintool` – Starting with the Solaris 10 release, this tool is no longer available
- Package and Patch Tool Enhancements

The following table provides a brief description of new or changed Solaris management tools .

TABLE 1–1 New or Changed Solaris Management Tools in the Solaris Release

Solaris Management Tool	Description	For More Information
`admintool`	This tool is no longer available. Alternative tools include the following: ■ Solaris Management Console to manage users and groups ■ Solaris Product Registry to manage software ■ Solaris Print Manager to manage printers ■ Solaris Management Console to manage terminals and modems	"Setting Up User Accounts (Task Map)" on page 113 "Managing Software With the Solaris Product Registry GUI (Task Map)" on page 373 Chapter 4, "Setting Up Printers (Tasks)," in *System Administration Guide: Solaris Printing* "Setting Up Terminals and Modems With Serial Ports Tool (Overview)" in *System Administration Guide: Advanced Administration*
Package and Patch Tools	Starting with the Solaris 10 release, the package and patch tools have been enhanced. Use the `pkgchk` command with the `-P` option instead of `grep pattern /var/sadm/install/contents`. The `-P` option enables you to use a partial path.	"Package and Patch Tool Enhancements" on page 361 Chapter 21, "Managing Solaris Patches by Using the `patchadd` Command (Tasks)"
Solaris Print Manager	Expanded printer support in Solaris Print Manager includes the following features. These features were introduced in a Solaris 10 release: ■ `Never Print Banner` option ■ Support for raster image processor (RIP) ■ Support for PostScript Printer Description (PPD) files ■ New `-n` option to the `lpadmin` command, which enables you to specify a PPD file when creating a new print queue or modifying an existing print queue ■ The `lpstat` command output displays the PPD for a print queue that was creating by specifying a PPD file	"What's New in Printing?" in *System Administration Guide: Solaris Printing*

Matrix of Solaris Management Tools and Supported Releases

This section provides information about tools that are primarily used to manage users, groups, clients, disks, printers, and serial ports.

This table lists the various Solaris management GUI tools and whether they are currently supported.

TABLE 1–2 Matrix of Solaris Management Tool Support

	Current Solaris Release Express	Solaris 10	Solaris 9
`admintool`	Not supported	Not supported	Supported
Solstice AdminSuite 2.3	Not supported	Not supported	Not supported
Solstice AdminSuite 3.0	Not supported	Not supported	Not supported
Solaris Management Tools 1.0	Not supported	Not supported	Not supported
Solaris Management Tools 2.0	Not supported	Not supported	Not supported
Solaris Management Tools 2.1	Supported	Supported	Supported

TABLE 1–3 Matrix of Solaris Management Tool Support in the Solaris 8 Release

	Solaris 8
`admintool`	Supported
Solstice AdminSuite 2.3	Not supported
Solstice AdminSuite 3.0	Supported
Solaris Management Tools 1.0	Supported
Solaris Management Tools 2.0	Supported (Solaris 8 01/01, 4/01, 7/01, 10/01, 2/02 releases only)
Solaris Management Tools 2.1	Not supported

If you want to perform administration tasks on a system with a text-based terminal as the console, use Solaris Management Console commands instead. For more information, see Table 1–6.

Feature Descriptions for Solaris Management Tools

This table describes the tools that are available in the Solaris Express release.

TABLE 1–4 Descriptions for Solaris Management Tools

Feature or Tool	Supported in Solaris Management Console 2.1?
Computers and Networks tool	Supported
Diskless Client support	A diskless client command-line interface is available
Disks tool	Supported
Enhanced Disk tool (Solaris Volume Manager)	Supported
Job Scheduler tool	Supported
Log Viewer tool	Supported
Mail Alias support	Supported
Mounts and Shares tool	Supported
Name Service support	For users, groups, and network information only
Performance tool	Supported
Printer support	Not Supported, but Solaris Print Manager is available as a separate tool
Projects tool	Supported
role-based access control (RBAC) support	Supported
RBAC Tool	Supported
Serial Port tool	Supported
Software Package tool	Not supported
System Information tool	Supported
User/Group tool	Supported

Feature Descriptions for Solaris 9 Management Tools

This table describes the tools available in the Solaris 9 releases.

TABLE 1–5 Feature Descriptions for Solaris 9 Management Tools

Feature or Tool	Supported in `admintool`?	Supported in Solaris Management Console 2.1?
Computers and Networks tool	Not supported	Supported
Diskless Client support	Not supported	A diskless client command-line interface is available
Disks tool	Not supported	Supported
Enhanced Disk tool (Solaris Volume Manager)	Not supported	Supported
Job Scheduler tool	Not supported	Supported
Log Viewer tool	Not supported	Supported
Mail Alias support	Not supported	Supported
Mounts and Shares tool	Not supported	Supported
Name Service support	Not supported	For users, groups, and network information only
Performance tool	Not supported	Supported
Printer support	Supported	Not supported, but Solaris Print Manager is available as a separate tool
Projects tool	Not supported	Supported
RBAC support	Not supported	Supported
RBAC tool	Not supported	Supported
Serial Port tool	Supported	Supported
Software Package tool	Supported	Not supported
System Information tool	Not supported	Supported
User/Group tool	Supported	Supported

Availability of Solaris Management Commands

This series of tables lists commands that perform the same tasks as the Solaris management tools. For information on diskless client support, see Chapter 7, "Managing Diskless Clients (Tasks)."

Solaris System Management Commands

This table describes the commands that provide the same functionality as the Solaris management tools. You must be superuser or assume an equivalent role to use these commands. Some of these commands are for the local system only. Others commands operate in a name service environment. See the appropriate man page and refer to the `-D` option.

TABLE 1–6 Descriptions for Solaris Management Commands

Command	Description	Man Page
`smc`	Starts the Solaris Management Console	`smc`(1M)
`smcron`	Manages `crontab` jobs	`smcron`(1M)
`smdiskless`	Manages diskless client support	`smdiskless`(1M)
`smexec`	Manages entries in the `exec_attr` database	`smexec`(1M)
`smgroup`	Manages group entries	`smgroup`(1M)
`smlog`	Manages and views WBEM log files	`smlog`(1M)
`smmultiuser`	Manages bulk operations on multiple user accounts	`smmultiuser`(1M)
`smosservice`	Adds Operating System (OS) services and diskless client support	`smosservice`(1M)
`smprofile`	Manages profiles in the `prof_attr` and `exec_attr` databases	`smprofile`(1M)
`smrole`	Manages roles and users in role accounts	`smrole`(1M)
`smserialport`	Manages serial ports	`smserialport`(1M)
`smuser`	Manages user entries	`smuser`(1M)

This table describes the commands you can use to manage RBAC from the command line. You must be superuser or assume an equivalent role to use these commands. These commands cannot be used to manage RBAC information in a name service environment.

TABLE 1–7 RBAC Command Descriptions

Command	Description	References
`auths`	Displays authorizations granted to a user	`auths`(1)

TABLE 1–7 RBAC Command Descriptions *(Continued)*

Command	Description	References
`profiles`	Displays execution profiles for a user	`profiles`(1)
`roleadd`	Adds a new role to the system	`roleadd`(1M)
`roles`	Displays roles granted to a user	`roles`(1)

This table describes the commands you can use to manage users, groups, and RBAC features from the command line. You must be superuser or assume an equivalent role to use these commands. These commands cannot be used to manage user and group information in a name service environment.

TABLE 1–8 Solaris User/Group Command Descriptions

Command	Description	References
`useradd`, `usermod`, `userdel`	Adds, modifies, or removes a user	`useradd`(1M), `usermod`(1M), `userdel`(1M)
`groupadd`, `groupmod`, `groupdel`	Adds, modifies, or removes a group	`groupadd`(1M), `groupmod`(1M), `groupdel`(1M)

For More Information About Solaris Management Tools

This table identifies where to find more information about Solaris management tools.

TABLE 1–9 For More Information About Solaris Management Tools

Tool	Availability	For More Information
Solaris Management Console 2.1 suite of tools	Solaris 9, 10, and Solaris Express releases	This guide and the console online help
Solaris Management Console 2.0 suite of tools	Solaris 8 1/01, 4/01, 7/01, 10/01, and 2/02 releases	Solaris Management Console online help
`admintool`	Solaris 9 and previous Solaris releases	`admintool`
AdminSuite 3.0	Solaris 8, Solaris 8 6/00, and Solaris 8 10/00 releases	*Solaris Easy Access Server 3.0 Installation Guide*

TABLE 1–9 For More Information About Solaris Management Tools *(Continued)*

Tool	Availability	For More Information
Diskless Client command-line interface	Solaris 8 1/01, 4/01, 7/01, 10/01, 2/02, Solaris 9, Solaris 10, and Solaris Express releases	Chapter 7, “Managing Diskless Clients (Tasks)”

◆ ◆ ◆ CHAPTER 2

Working With the Solaris Management Console (Tasks)

This chapter describes the Solaris management tools that are used to perform system administration tasks. Topics include starting the Solaris Management Console (console), setting up role-based access control (RBAC) to use with the console, and working with the Solaris management tools in a name service environment.

For information on the procedures associated with performing system management tasks by using the Solaris Management Console, see these task maps:

- "Using the Solaris Management Tools With RBAC (Task Map)" on page 40
- "Using the Solaris Management Tools in a Name Service Environment (Task Map)" on page 45

For information on troubleshooting Solaris Management Console problems, see "Troubleshooting the Solaris Management Console" on page 53.

Solaris Management Console (Overview)

The following sections provide information about the Solaris Manager Console.

What Is the Solaris Management Console?

The Solaris Management Console is a container for GUI-based management tools that are stored in collections referred to as *toolboxes*.

The console includes a default toolbox with many basic management tools, including tools for managing the following:

- Users
- Projects
- `cron` jobs for mounting and sharing file systems

- `cron` jobs for managing disks and serial ports

For a brief description of each Solaris management tool, see Table 2–1.

You can add tools to the existing toolbox, or you can create new toolboxes.

The Solaris Management Console has three primary components:

- **The Solaris Management Console client**

  Called the *console*, this component is the visible interface and contains the GUI tools used to perform management tasks.
- **The Solaris Management Console server**

  This component is located either on the same machine as the console or remotely. This component provides all the *back-end* functionality that allows management through the console.
- **The Solaris Management Console toolbox editor**

  This application, which looks similar to the console, is used to add or modify toolboxes, to add tools to a toolbox, or to extend the scope of a toolbox. For example, you could add a toolbox to manage a name service domain.

The default toolbox is visible when you start the console.

Solaris Management Console Tools

This table describes the tools included in the default Solaris Management Console toolbox. Cross-references to background information for each tool are provided.

TABLE 2–1 Solaris Management Console Tool Suite

Category	Tool	Description	For More Information
System Status	System Information	Monitors and manages system information such as date, time, and time zone	Chapter 5, "Displaying and Changing System Information (Tasks)," in *System Administration Guide: Advanced Administration*
	Log Viewer	Monitors and manages the Solaris Management Console tools log and system logs	Chapter 14, "Troubleshooting Software Problems (Overview)," in *System Administration Guide: Advanced Administration*

TABLE 2–1 Solaris Management Console Tool Suite *(Continued)*

Category	Tool	Description	For More Information
	Processes	Monitors and manages system processes	"Processes and System Performance" in *System Administration Guide: Advanced Administration*
	Performance	Monitors system performance	Chapter 11, "Managing System Performance (Overview)," in *System Administration Guide: Advanced Administration*
System Configuration	Users	Manages users, rights, roles, groups, and mailing lists	"What Are User Accounts and Groups?" on page 86 and "Role-Based Access Control (Overview)" in *System Administration Guide: Security Services*
	Projects	Creates and manages entries in the `/etc/project` database	Chapter 2, "Projects and Tasks (Overview)," in *System Administration Guide: Virtualization Using the Solaris Operating System*
	Computers and Networks	Creates and monitors computer and network information	Solaris Management Console online help
Services	Scheduled Jobs	Creates and manages scheduled `cron` jobs	"Ways to Automatically Execute System Tasks" in *System Administration Guide: Advanced Administration*
Storage	Mounts and Shares	Mounts and shares file systems	Chapter 19, "Mounting and Unmounting File Systems (Tasks)," in *System Administration Guide: Devices and File Systems*
	Disks	Creates and manages disk partitions	Chapter 10, "Managing Disks (Overview)," in *System Administration Guide: Devices and File Systems*
	Enhanced Storage	Creates and manages volumes, hot spare pools, state database replicas, and disk sets	*Solaris Volume Manager Administration Guide*
Devices and Hardware	Serial Ports	Sets up terminals and modems	Chapter 1, "Managing Terminals and Modems (Overview)," in *System Administration Guide: Advanced Administration*

Context–sensitive help is available after you start a tool. For broader, more in-depth online information than the context help provides, see the expanded help topics. You can access these help topics from the console Help menu.

Why Use the Solaris Management Console?

The console provides a set of tools with many benefits for administrators.

The console does the following:

- **Supports all experience levels**

  Inexperienced administrators can complete tasks by using the GUI, which includes dialog boxes, wizards, and context help. Experienced administrators find that the console provides a convenient, secure alternative to using `vi` to manage hundreds of configuration parameters spread across dozens or hundreds of systems.
- **Controls user access to the system**

  Although any user can access the console by default, only superuser can make changes in the initial configuration. As described in "Role-Based Access Control (Overview)" in *System Administration Guide: Security Services*, it is possible to create special user accounts called *roles* can be created and assigned to users, typically administrators, who are permitted to make specific system changes.

  The key benefit of RBAC is that roles can be limited so that users have access to only those tasks that are necessary for doing their jobs. RBAC is *not* required for using the Solaris management tools. You can run all tools as superuser without making any changes.
- **Provides a command line interface**

  If preferred, administrators can operate the Solaris management tools through a command-line interface (CLI). Some commands are written specifically to mimic the GUI tool functions, such as the commands for managing users. These new commands are listed in Table 1–6, which includes the names and brief descriptions of each command. There is also a man page for each command.

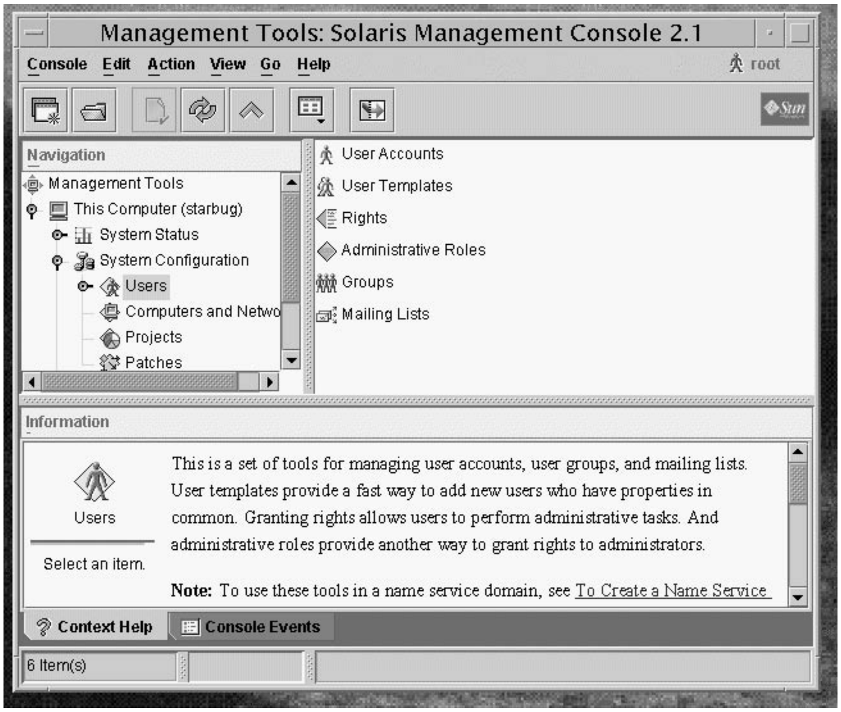
  For Solaris management tools that have no special commands, such as the Mounts and Shares tool, use the standard UNIX commands.

For in-depth information about how RBAC works, its benefits, and how to apply those benefits to your site, see "Role-Based Access Control (Overview)" in *System Administration Guide: Security Services*.

To learn more about using RBAC with the Solaris management tools, see "Using the Solaris Management Tools With RBAC (Task Map)" on page 40.

Organization of the Solaris Management Console

In the following figure, the console is shown with the Users tool open.



This is a set of tools for managing user accounts, user groups, and mailing lists. User templates provide a fast way to add new users who have properties in common. Granting rights allows users to perform administrative tasks. And administrative roles provide another way to grant rights to administrators.

Note: To use these tools in a name service domain, see To Create a Name Service

FIGURE 2–1 Solaris Management Console – Users Tool

The main part of the console consists of three panes:

- **Navigation pane** (at the left) – For accessing tools (or sets of tools), folders, or other toolboxes. Icons in the navigation pane are called *nodes* and are expandable if they are folders or toolboxes.
- **View pane** (at the right) – For viewing information related to the node selected in the navigation pane. The view pane shows either the contents of the selected folder, subordinate tools, or the data associated with the selected tool.
- **Information pane** (at the bottom) – For displaying context-sensitive help or console events.

Changing the Solaris Management Console Window

The layout of the console window is highly configurable. You can use the following features to change the console window layout:

- **View menu** – Use the Show option in the View menu to hide or display the optional bars and panes. The other options in the View menu control the display of nodes in the view pane.
- **Console menu** – Use the Preferences option to set the following: the initial toolbox, the orientation of panes, clicking or double-clicking for selection, text or icons in the tool bar, fonts, default tool loading, authentication prompts, and advanced logins.
- **Context Help or Console Events toggles** – Use the icons at the bottom of the information pane to toggle between the display of context-sensitive help and console events.

Solaris Management Console Documentation

The main source of documentation for using the console and its tools is the online help system. Two forms of online help are available: context-sensitive help and expanded help topics.

- **Context-sensitive help responds to your use of the console tools.**

  Clicking the cursor on tabs, entry fields, radio buttons, and so forth, causes the appropriate help to appear in the Information pane. You can close, or reopen the Information pane by clicking the question mark button on dialog boxes and wizards.
- **Expanded help topics are available from the Help menu or by clicking cross reference links in some context-sensitive help.**

  These topics appear in a separate viewer and contain more in-depth information than is provided by the context help. Topics include overviews of each tool, explanations of how each tool works, files used by a specific tool, and troubleshooting.

For a brief overview of each tool, refer to Table 2–1.

How Much Role-Based Access Control?

As described in "Why Use the Solaris Management Console?" on page 34, a major advantage of using the Solaris management tools is the ability to use Role-Based Access Control (RBAC). RBAC provides administrators with access to just the tools and commands they need to perform their jobs.

Depending on your security needs, you can use varying degrees of RBAC.

RBAC Approach	Description	For More Information
No RBAC	Allows you to perform all tasks as superuser. You can log in as yourself. When you select a Solaris management tool, you specify `root` as the user and the `root` password.	"How to Become Superuser (root) or Assume a Role" on page 38
`root` as a role	Eliminates anonymous `root` logins and prevents users from logging in as `root`. This approach requires users to log in as themselves before they assume the `root` role. Note that you can apply this approach whether or not you are using other roles.	"How to Plan Your RBAC Implementation" in *System Administration Guide: Security Services*
Single role only	Uses the Primary Administrator role, which is roughly equivalent to having `root` access only.	"Creating the Primary Administrator Role" on page 41
Suggested roles	Uses three roles that are easily configured: Primary Administrator, System Administrator, and Operator. These roles are appropriate for organizations with administrators at different levels of responsibility whose job capabilities roughly fit the suggested roles.	"Role-Based Access Control (Overview)" in *System Administration Guide: Security Services*
Custom roles	You can add your own roles, depending on your organization's security needs.	"Managing RBAC" in *System Administration Guide: Security Services* and "How to Plan Your RBAC Implementation" in *System Administration Guide: Security Services*

Becoming Superuser (`root`) or Assuming a Role

Most administration tasks, such as adding users, file systems, or printers, require that you first log in as `root` (UID=0) or assume a role if you are using RBAC. The `root` account, also known as the *superuser* account, is used to make system changes and can override user file protection in emergency situations.

The superuser account and roles should be used only to perform administrative tasks to prevent indiscriminate changes to the system. The security problem associated with the superuser account is that a user has complete access to the system even when performing minor tasks.

In a non-RBAC environment, you can either log in to the system as superuser or use the `su` command to change to the superuser account. If RBAC is implemented, you can assume roles through the console or use `su` and specify a role.

When you use the console to perform administration tasks, you can do one of the following:

- Log in to the console as yourself and then supply the `root` user name and password
- Log in to the console as yourself and then assume a role

A major benefit of RBAC is that roles can be created to give limited access to specific functions only. If you are using RBAC, you can run restricted applications by assuming a role rather than by becoming superuser.

For step-by-step instructions on creating the Primary Administrator role, see "How to Create the First Role (Primary Administrator)" on page 42. For an overview on using RBAC, see Chapter 9, "Using Role-Based Access Control (Tasks)," in *System Administration Guide: Security Services*.

▼ How to Become Superuser (root) or Assume a Role

Become superuser or assume a role by using one of the following methods. Each method requires that you know either the superuser password or the role password.

1 Become superuser. Select one of the following methods to become superuser:

- **Log in as a user, start the Solaris Management Console, select a Solaris management tool, and then log in as `root`.**

  This method enables to you perform any management task from the console.

  For information on starting the Solaris Management Console, see "How to Start the Solaris Management Console in a Name Service Environment" on page 51.

- **Log in as superuser on the system console.**

  ```
  hostname console: root
  Password: root-password
  #
  ```

  The pound sign (#) is the Bourne shell prompt for the superuser account.

  This method provides complete access to all system commands and tools.

- **Log in as a user, and then change to the superuser account by using the `su` command at the command line.**

  ```
  % su
  Password: root-password
  #
  ```

  This method provides complete access to all system commands and tools.

- **Log in remotely as superuser.**

  This method is not enabled by default. You must modify the `/etc/default/login` file to remotely log in as superuser on the system console. For information on modifying this file, see Chapter 3, "Controlling Access to Systems (Tasks)," in *System Administration Guide: Security Services*.

  This method provides complete access to all system commands and tools.

2 **Assume a role. Select one of the following methods to assume a role:**

- **Log in as user, and then change to a role by using the `su` command at the command line.**

  ```
  % su role
  Password: role-password
  $
  ```

  This method provides access to all the commands and tools that the role has access to.

- **Log in as a user, start the Solaris Management Console, select a Solaris management tool, and then assume a role.**

  For information on starting the Solaris Management Console, see "How to Start the Console as Superuser or as a Role" on page 44.

  This method provides access to the Solaris management tools that the role has access to.

Using the Solaris Management Tools With RBAC (Task Map)

This task map describes the tasks to do if you want to use the RBAC security features rather than the superuser account to perform administration tasks.

Note – The information in this chapter describes how to use the console with RBAC. RBAC overview and task information is included to show you how to initially set up RBAC with the console.

For detailed information on RBAC and how to use it with other applications, see "Role-Based Access Control (Overview)" in *System Administration Guide: Security Services*.

Task	Description	For Instructions
1. Start the console.	If your user account is already set up, start the console as yourself. Then, log in to the console as `root`. If you do not have a user account set up, become superuser first, and then start the console.	"How to Start the Console as Superuser or as a Role" on page 44
2. Add a user account for yourself.	Add a user account for yourself, if you do not have an account already.	Solaris Management Console online help "If You Are the First to Log in to the Console" on page 41
3. Create the Primary Administrator role	Create the Primary Administrator role. Then, add yourself to this role.	"How to Create the First Role (Primary Administrator)" on page 42
4. Assume the Primary Administrator role.	Assume the Primary Administrator role after you have created this role.	"How to Assume the Primary Administrator Role" on page 43
5. (Optional) Make `root` a role.	Make `root` a role and add yourself to the root role so that no other user can use the `su` command to become `root`.	"How to Plan Your RBAC Implementation" in *System Administration Guide: Security Services*
6. (Optional) Create other administrative roles.	Create other administrative roles and grant the appropriate rights to each role. Then, add the appropriate users to each role.	Chapter 9, "Using Role-Based Access Control (Tasks)," in *System Administration Guide: Security Services*

The following sections provide overview information and step-by-step instructions for using the Solaris Management Console and the RBAC security features.

If You Are the First to Log in to the Console

If you are the first administrator to log in to the console, start the console as a user (yourself). Then, log in as superuser. This method gives you complete access to all the console tools.

Here are the general steps, depending on whether you are using RBAC:

- **Without RBAC** – If you choose not to use RBAC, continue working as superuser. All other administrators will also need `root` access to perform their jobs.
- **With RBAC** – You will need to do the following:
  - Set up your user account, if you do not already have an account.
  - Create the role called Primary Administrator.
  - Assign the Primary Administrator right to the role that you are creating.
  - Assign your user account to this role.

    For step-by-step instructions on creating the Primary Administrator role, see "How to Create the First Role (Primary Administrator)" on page 42.

    For an overview on using RBAC, see Chapter 9, "Using Role-Based Access Control (Tasks)," in *System Administration Guide: Security Services*.

Creating the Primary Administrator Role

An *administrator role* is a special user account. Users who assume a role are permitted to perform a predefined set of administrative tasks.

The Primary Administrator role is permitted to perform all administrative functions, similar to superuser.

If you are superuser, or a user assuming the Primary Administrator role, you can define which tasks other administrators are permitted to perform. With the help of the Add Administrative Role wizard, you can create a role, grant rights to the role, and then specify which users are permitted to assume that role. A *right* is a named collection of commands, or authorizations, for using specific applications. A right enables you to perform specific functions within an application. The use of rights can be granted or denied by an administrator.

You are prompted for the following information when you create the Primary Administrator role.

TABLE 2–2 Field Descriptions for Adding a Role by Using the Solaris Management Console

Field name	Description
Role name	Selects the name an administrator uses to log in to a specific role.
Full name	Provides a full, descriptive name of this role. (Optional)
Description	Provides further description of this role.
Role ID number	Selects the identification number assigned to this role. This number is the same as the set of identifiers for UIDs.
Role shell	Selects the shell that runs when a user logs in to a terminal or console window and assumes a role in that window.
Create a role mailing list	Creates a mailing list with the same name as the role, if checked. You can use this list to send email to everyone assigned to the role.
Role password and confirm Password	Sets and confirms the role password.
Available rights and granted Rights	Assigns rights to this role by choosing from the list of Available Rights and adding them to the list of Granted Rights.
Select a home directory	Selects the home directory server where this role's private files will be stored.
Assign users to this role	Adds specific users to the role so that they can assume the role to perform specific tasks.

For detailed information about role-based access control, and instructions on how to use roles to create a more secure environment, see "Role-Based Access Control (Overview)" in *System Administration Guide: Security Services*.

▼ How to Create the First Role (Primary Administrator)

This procedure describes how to create the Primary Administrator role and then assign it to your user account. This procedure assumes that your user account is already created.

1 **Start the console as yourself.**

```
% /usr/sadm/bin/smc &
```

For additional information on starting the console, see "How to Start the Console as Superuser or as a Role" on page 44.

The console online help provides more information about creating a user account for yourself.

2 **Click on the This Computer icon in the Navigation pane.**

3 **Click on System Configuration->Users -> Administrative Roles.**

4 **Click Action->Add Administrative Role.**

The Add Administrative Role wizard opens.

5 **Create the Primary Administrator role with the Administrative Role wizard by following these steps.**

a. **Identify the role name, full role name, description, role ID number, role shell, and whether you want to create a role mailing list. Click Next.**

b. **Set and confirm the role password. Click Next.**

c. **Select the Primary Administrator right from the Available Rights column and add it to Granted Rights column. Click Next.**

d. **Select the home directory for the role. Click Next.**

e. **Assign yourself to the list of users who can assume the role. Click Next.**

If necessary, see Table 2–2 for a description of the role fields.

6 **Click Finish.**

▼ How to Assume the Primary Administrator Role

After you have created the Primary Administrator role, log in to the console as yourself, and then assume the Primary Administrator role.

When you assume a role, you take on all the attributes of that role, including the rights. At the same time, you relinquish all of your own user properties.

1 **Start the console.**

```
% /usr/sadm/bin/smc &
```

For information on starting the console, see "How to Start the Console as Superuser or as a Role" on page 44.

2 **Log in with your user name and password.**

A list shows which roles you are permitted to assume.

3 **Log in to the Primary Administrator role and provide the role password.**

Starting the Solaris Management Console

The following procedure describes how to start the console and gain access to the Solaris management tools.

For instructions on what to do if you are the first user to log in to the console, see "If You Are the First to Log in to the Console" on page 41.

▼ How to Start the Console as Superuser or as a Role

If you start the console as a user with your own user account, you have limited access to the Solaris management tools. For greater access, you can log in as yourself and then log in as one of the roles you are allowed to assume. If you are permitted to assume the role of Primary Administrator, you then have access to all the Solaris management tools. This role is equivalent to that of superuser.

1. **Verify that you are in a window environment, such as the CDE environment.**

2. **Start the console in one of the following ways:**

   - **From the command line, type the following command:**

     ```
     % /usr/sadm/bin/smc &
     ```

     It might take a minute or two for the console to come up the first time.

   - **Start the console from the Tools menu of the CDE front panel.**

   - **Double-click the Solaris Management Console icon in CDE's Applications Manager or File Manager.**

   The Solaris Management Console window is displayed.

   Note – Open a console in your window environment to display the Solaris Management Console startup messages. Do not attempt to start the Solaris Management Console server manually before starting the Solaris Management Console. The server starts automatically when you start the Solaris Management Console. For information on troubleshooting console problems, see "Troubleshooting the Solaris Management Console" on page 53.

   ---

3. **Double-click the This Computer icon under the Management Tools icon in the Navigation pane.**

   A list of categories is displayed.

4 **(Optional) Select the appropriate toolbox.**

If you want to use a toolbox other than the default toolbox, select the appropriate toolbox from the Navigation pane. Or, select Open Toolbox from the console menu and load the toolbox you want.

For information about using different toolboxes, see "How to Create a Toolbox for a Specific Environment" on page 48.

5 **Double-click the category icon to access a particular tool.**

Use the online help to identify how to perform a specific task.

6 **Double-click the tool icon.**

A pop-up Log-In window is displayed.

7 **Decide if you want to use the tool as superuser or as a role. If you are logging in a as superuser, enter the `root` password.**

8 **If you are logging in as yourself, backspace over the `root` user name. Then supply your user ID and user password.**

A list of roles you can assume is displayed.

9 **Select the Primary Administrator role, or an equivalent role, and supply the role password.**

For step-by-step instructions on creating the Primary Administrator role, see "How to Create the First Role (Primary Administrator)" on page 42.

The main tool menu is displayed.

Using the Solaris Management Tools in a Name Service Environment (Task Map)

By default, the Solaris management tools are set up to operate in a local environment. For example, the Mounts and Shares tool enables you to mount and share directories on specific systems, but not in an NIS or NIS+ environment. However, you can manage information with the Users and Computers and Networks tools in a name service environment.

To work with a console tool in a name service environment, you need to create a name service toolbox, and then add the tool to that toolbox.

Task	Description	For Instructions
1. Verify prerequisites.	Verify you have completed the prerequisites before attempting to use the console in a name service environment.	"Prerequisites for Using the Solaris Management Console in a Name Service Environment" on page 47
2. Create a toolbox for the name service.	Use the New Toolbox wizard to create a toolbox for your name service tools.	"How to Create a Toolbox for a Specific Environment" on page 48
3. Add a tool to the name service toolbox.	Add the Users tool, or any other name service tool, to your name service toolbox.	"How to Add a Tool to a Toolbox" on page 50
4. Select the toolbox that was just created.	Select the toolbox you just created to manage name service information.	"How to Start the Solaris Management Console in a Name Service Environment" on page 51

RBAC Security Files

The RBAC security files that work with the Solaris Management Console are created when you upgrade to or install at least the Solaris 9 release. If you do not install the Solaris Management Console packages, the RBAC security files are installed without the necessary data for using RBAC. For information on the Solaris Management Console packages, see "Troubleshooting the Solaris Management Console" on page 53.

The RBAC security files if you are running at least the Solaris 9 release are included in your name service so that you can use the Solaris Management Console tools in a name service environment.

The security files on a local server are populated into a name service environment as part of a standard upgrade by the `ypmake`, `nispopulate`, or equivalent LDAP commands.

The following name services are supported:

- NIS
- NIS+
- LDAP
- files

Note – The `projects` database is not supported in the NIS+ environment.

The RBAC security files are created when you upgrade to or install at least the Solaris 9 release.

This table briefly describes the predefined security files that are installed on a system that is running at least the Solaris 9 release.

TABLE 2–3 RBAC Security Files

Local File Name	Table or Map Name	Description
`/etc/user_attr`	`user_attr`	Associates users and roles with authorizations and rights profiles
`/etc/security/auth_attr`	`auth_attr`	Defines authorizations and their attributes and identifies associated help files
`/etc/security/prof_attr`	`prof_attr`	Defines rights profiles, lists the rights profiles assigned to the authorizations, and identifies associated help files
`/etc/security/exec_attr`	`exec_attr`	Defines the privileged operations assigned to a rights profile

For unusual upgrade cases, you might have to use the `smattrpop` command to populate RBAC security files in the following instances:

- When creating or modifying rights profiles
- When you need to include users and roles by customizing the `usr_attr` file

For more information, see "Role-Based Access Control (Overview)" in *System Administration Guide: Security Services*.

Prerequisites for Using the Solaris Management Console in a Name Service Environment

The following table identifies what you need to do before you can use the Solaris Management Console in a name service environment.

Prerequisite	For More Information
Install at least the Solaris 9 release.	*Solaris Express Installation Guide: Basic Installations*
Set up your name service environment.	*System Administration Guide: Naming and Directory Services (DNS, NIS, and LDAP)*
Select your management scope.	"Management Scope" on page 48
Make sure your `/etc/nsswitch.conf` file is configured so that you can access your name service data.	"`/etc/nsswitch.conf` File" on page 48

Management Scope

The Solaris Management Console uses the term *management scope* to refer to the name service environment that you want to use with the selected management tool. The management scope choices for the Users tool and the Computers and Networks tool are LDAP, NIS, NIS+, or files.

The management scope that you select during a console session should correspond to the primary name service identified in the `/etc/nsswitch.conf` file.

`/etc/nsswitch.conf` File

The `/etc/nsswitch.conf` file on each system specifies the policy for name service lookups (where data is read from) on that system.

Note – You must make sure that the name service accessed from the console, which you specify through the console Toolbox Editor, appears in the search path of the `/etc/nsswitch.conf` file. If the specified name service does not appear there, the tools might behave in unexpected ways, resulting in errors or warnings.

When you use the Solaris management tools in a name service environment, you might impact many users with a single operation. For example, if you delete a user in the NIS name service, that user is deleted on all systems that are using NIS.

If different systems in your network have different `/etc/nsswitch.conf` configurations, unexpected results might occur. So, all systems to be managed with the Solaris management tools should have a consistent name service configuration.

▼ How to Create a Toolbox for a Specific Environment

Applications for administering the Solaris Operating System are called tools. Those tools are stored in collections referred to as *toolboxes*. A toolbox can be located on a local server, where the console is located, or on a remote machine.

Use the Toolbox Editor to add a new toolbox, to add tools to an existing toolbox, or to change the scope of a toolbox. For example, use this tool to change the domain from local files to a name service.

Note – You can start the Toolbox Editor as a normal user. However, if you plan to make changes and save them to the default console toolbox, `/var/sadm/smc/toolboxes`, you must start the Toolbox Editor as `root`.

---

1 **Start the Toolbox Editor.**

```
# /usr/sadm/bin/smc edit &
```

2 **Select Open from the Toolbox menu.**

3 **Select the This Computer icon in the Toolboxes: window.**

4 **Click Open.**

The This Computer toolbox opens in the window.

5 **Select the This Computer icon again in the Navigation pane.**

6 **Select Add Folder from the Action menu.**

7 **Use the Folder wizard to add a new toolbox for your name service environment.**

a. **Name and Description – Provide a name in the Full Name window. Click Next.**

For example, provide "NIS tools" for the NIS environment.

b. **Provide a description in the Description window. Click Next.**

For example, "tools for NIS environment" is an appropriate example.

c. **Icons – Use the default value for the Icons. Click Next.**

d. **Management Scope – Select Override.**

e. **Select your name service under the Management Scope pull-down menu.**

f. **Add the name service master name in the Server field, if necessary.**

g. **Add the domain managed by the server in the Domain field.**

h. **Click Finish.**

The new toolbox appears in the left Navigation pane.

8 **Select the new toolbox icon and select Save As from the Toolbox menu.**

9 **Enter the toolbox path name in the Local Toolbox Filename dialog box. Use the `.tbx` suffix.**

/var/sadm/smc/toolboxes/this_computer/*toolbox-name*.tbx

10 **Click Save.**

The new toolbox appears in the Navigation pane in the console window.

See Also After you have created a name service toolbox, you can put a name service tool into it. For more information, see "How to Add a Tool to a Toolbox" on page 50.

▼ How to Add a Tool to a Toolbox

In addition to the default tools that ship with the console, additional tools that can be launched from the console are being developed. As these tools become available, you can add one or more tools to an existing toolbox.

You can also create a new toolbox, for either local management or network management. Then, you can add tools to the new toolbox.

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 **Start the Toolbox Editor, if necessary.**

```
# /usr/sadm/bin/smc edit &
```

3 **Select the toolbox.**

If you want to work in a name service, select the toolbox you just created in the Toolbox Editor. For more information, see "How to Create a Toolbox for a Specific Environment" on page 48.

4 **Select Add Tool from the Action menu.**

5 **Use the Add Tool wizard to add the new tool.**

a. **Server Selection – Add the name service master in the Server window. Click Next.**

b. **Tools Selection – Select the tool you want to add from the Tools window. Click Next.**

If this toolbox is a name service toolbox, choose a tool you want to work in a name service environment. For example, choose the Users tool.

c. **Name and Description – Accept the default values. Click Next.**

d. **Icons – Accept the default values, unless you have created custom icons. Click Next.**

   e. **Management Scope – Accept the default value "Inherit from Parent." Click Next.**

   f. **Tool Loading – Accept the default "Load tool when selected." Click Finish.**

6. **Select Save from the Toolbox menu to save the updated toolbox.**

   The Local Toolbox window is displayed.

▼ How to Start the Solaris Management Console in a Name Service Environment

After you have created a name service toolbox and added tools to it, you can start the Solaris Management Console and open that toolbox to manage a name service environment.

Before You Begin Verify that the following prerequisites are met:

- Ensure that the system you are logged in to is configured to work in a name service environment.
- Verify that the `/etc/nsswitch.conf` file is configured to match your name service environment.

1. **Start the Solaris Management Console.**

   For more information, see "How to Start the Console as Superuser or as a Role" on page 44.

2. **Select the toolbox you created for the name service, which appears in the Navigation pane.**

   For information on creating a toolbox for a name service, see "How to Create a Toolbox for a Specific Environment" on page 48.

Adding Tools to the Solaris Management Console

This section describes how to add legacy tools or unbundled tools to the console. If you want to add authentication to these tools, see "Managing RBAC" in *System Administration Guide: Security Services*.

▼ How to Add a Legacy Tool to a Toolbox

A legacy tool is any application that was not designed specifically as a Solaris management tool. You can add three types of legacy tool applications to a console toolbox: X applications, command-line interface, and HTML. Each tool you add to a toolbox can then be launched from the Solaris Management Console.

1 **Become superuser or assume an equivalent role.**

2 **Start the Solaris Management Console Toolbox Editor, if necessary.**

```
# /usr/sadm/bin/smc edit &
```

3 **Open the toolbox to which you want to add the legacy application.**

The toolbox selected is opened in the Toolbox Editor.

4 **Select the node in the toolbox to which you want to add the legacy application.**

A legacy application can be added to the top node of a toolbox or to another folder.

5 **Click Action->Add Legacy Application.**

The first panel of the Legacy Application Wizard: General is displayed.

6 **Follow the instructions in the wizard.**

7 **Save the toolbox in the Toolbox Editor.**

▼ How to Install an Unbundled Tool

Follow this procedure if you want to add a new tool package that can be launched from the Solaris Management Console.

1 **Become superuser or assume an equivalent role.**

2 **Install the new tool package.**

```
# pkgadd ABCDtool
```

3 **Restart the console so that it recognizes the new tool.**

a. **Stop the console server.**

```
# /etc/init.d/init.wbem stop
```

b. **Start the console server.**

```
# /etc/init.d/init.wbem start
```

4 **Start the console to verify that the new tool is displayed.**

For more information, see "How to Start the Console as Superuser or as a Role" on page 44.

Troubleshooting the Solaris Management Console

Before using this troubleshooting procedure, make sure that the following packages are installed:

- `SUNWmc` – Solaris Management Console 2.1 (Server Components)
- `SUNWmcc` – Solaris Management Console 2.1 (Client Components)
- `SUNWmccom` – Solaris Management Console 2.1 (Common Components)
- `SUNWmcdev` – Solaris Management Console 2.1 (Development Kit)
- `SUNWmcex` – Solaris Management Console 2.1 (Examples)
- `SUNWwbmc` – Solaris Management Console 2.1 (WBEM Components)

These packages provide the basic Solaris Management Console launcher. You must install the `SUNWCprog` cluster to use the Solaris Management Console and all of its tools.

▼ How to Troubleshoot the Solaris Management Console

The client and the server are started automatically when you start the Solaris Management Console.

If the console is visible and you are having trouble running the tools, it might be that the server might not be running. Or, the server might be in a problem state that can be resolved by stopping and restarting it.

1. **Become superuser or assume an equivalent role.**

2. **Determine whether the console server is running.**

   ```
   # /etc/init.d/init.wbem status
   ```

   If the console server is running, you should see a message similar the following:

   ```
   SMC server version 2.1.0 running on port 898.
   ```

3. **If the console server is not running, start it.**

   ```
   # /etc/init.d/init.wbem start
   ```

   After a short time, you should see a message similar to the following:

   ```
   SMC server is ready.
   ```

4. **If the server is running and you are still having problems, stop the console server. Then, restart it.**

   a. **Stop the console server.**

      ```
      # /etc/init.d/init.wbem stop
      ```

You should see a message similar to the following:

```
Shutting down SMC server on port 898.
```

b. **Start the console server.**

```
# /etc/init.d/init.wbem start
```

◆ ◆ ◆ CHAPTER 3

Working With the Sun Java Web Console (Tasks)

This chapter describes the Sun Java™ Web Console, which is used to administer web-based Sun system management applications that are installed and registered on your system.

Topics in this chapter include the following:

For information about the procedures that are associated with using the Java Web Console, see "Getting Started With the Java Web Console (Task Map)" on page 58 and "Troubleshooting the Java Web Console Software (Task Map)" on page 70.

What's New in Administering the Java Web Console?

This section includes features that are new in this Solaris release.

Java Web Console Server Management

Solaris 10 11/06: The Java Web Console server is managed as a service by the Service Management Facility (SMF). For more information about SMF, see Chapter 16, "Managing Services (Overview)."

Applications That Are Available to the Java Web Console

The ZFS™ web-based management tool is available in the Java Web Console. This tool enables you to perform much of the administration tasks that you can perform with the command-line interface (CLI). These capabilities include setting parameters, viewing the various pools and file systems, and making updates to them.

The following are examples of typical procedures that you might perform with the tool:

- Create a new storage pool.
- Add capacity to an existing pool.
- Move (export) a storage pool to another system.
- Import a previously exported storage pool, to make it available on another system.
- View tables of information about storage pools.
- Create a file system.
- Create a `zvol` (virtual volume).
- Take a snapshot of a file system or a `zvol` volume.
- Roll back a file system to a previous snapshot.

For more information about using the Solaris ZFS web-based management tool, see *Solaris ZFS Administration Guide*.

Note – The Sun Java Enterprise System software includes several management applications that run in the Java Web Console.

Java Web Console (Overview)

The Java Web Console provides a common location for users to access web-based system management applications. You access the web console by logging in through a secure `https` port with one of several supported web browsers. The single entry point that the web console provides eliminates the need to learn URLs for multiple applications. In addition, the single entry point provides user authentication and authorization for all applications that are registered with the web console.

All web console-based applications conform to the same user interface guidelines, which enhances ease of use. The web console also provides auditing of user sessions and logging service for all registered applications.

What Is the Java Web Console?

The Java Web Console is a web page where you can find the Sun system management web-based applications that are installed and registered on your system. Registration is automatically a part of an application's installation process. Thus, registration requires no administrator intervention.

The Java Web Console provides the following:

- **A single point of entry for login and the launching of browser-based system management applications**

  The Java Web Console is Sun's current direction for system management applications. The console provides a central location from which you can start browser-based management applications simply by clicking the application names. No compatibility exists between the Java Web Console and the Solaris Management Console. The Java Web Console is a web application that you access through a browser, and Solaris Management Console is a Java application that you start from a command line. Because the consoles are completely independent, you can run both consoles on the same system at the same time.
- **Single sign-on through a secure https port**

  Single sign-on in this context means that you do not have to authenticate yourself to each management application after you authenticate yourself to the web console. You enter your user name and password just once per console session.
- **Dynamically organized and aggregated applications**

  Applications are installed and displayed on the console launch page under the category of management tasks that is most applicable.

  Categories include the following:

  - Systems
  - Storage
  - Services
  - Desktop applications
  - Other
- **A common look and feel**

  All web console applications use the same user interface (UI) components and behavior, thereby reducing the learning curve for administrators.
- **Standard, extensible authentication, authorization, and auditing mechanisms**

  The Java Web Console supports Pluggable Authentication Module (PAM), role-based access control (RBAC) roles, and Basic Security Module (BSM) auditing.

Java Web Console Management Commands

The Java Web Console includes the following management commands:

- `smcwebserver` – This command starts and stops the console's web server.
- `wcadmin` – **Starting with the Solaris 10 11/06 release**, this command is used to configure the console, and to register and deploy console applications. For more information, see the `wcadmin(1M)` man page.
- `smreg` – If you are *not* running at least the Solaris Express 5/06release, this command is used to register all console applications.

  , use this command only to register legacy applications that were created for a version of the console that is not at least Java Web Console 3.0.

The commands are used to perform various tasks that this chapter describes.

For more information about each command, see the `smcwebserver(1M)`, `wcadmin(1M)`, and the `smreg(1M)` man pages.

Supported Web Browsers

The Java Web Console can be used in any of the following browsers while running the Solaris OS:

- Mozilla (at least Version, 1.4)
- Netscape (at least Version, 6.2)
- Firefox (at least Version, 1.0)

Getting Started With the Java Web Console (Task Map)

Task	Description	For Instructions
Start applications from the Java Web Console's launch page.	The Java Web Console's launch page lists all the registered system management applications that you have permission to use. You connect to a specific application by clicking its application name.	"How to Start Applications From the Java Web Console's Launch Page" on page 60

Task	Description	For Instructions
Start, stop, enable, and disable the console server.	You can manage the web server that is used to run the console and the registered applications.	"How to Start the Console Service" on page 62 "How to Enable the Console Service to Run at System Start" on page 62 "How to Stop the Console Service" on page 63 "How to Disable the Console Service" on page 63
Change the Java Web Console's properties.	You should not have to change any of the web console's default properties. Properties that you might choose to change include the following: ■ Console session timeout ■ Logging level ■ Audit implementation	"How to Change Java Web Console Properties" on page 66

Getting Started With the Java Web Console

The Java Web Console's launch page lists the registered system management applications that you have permission to use, and displays a brief description of each application. You connect to a specific application by clicking its application name, which is a link to the actual application. By default, the selected application opens in the web console window. You can choose to open applications in separate browser windows by clicking the Start Each Application in a New Window check box. When you open applications in separate windows, the web console launch page remains available, so you can return to it and launch multiple applications under a single login.

To access the console launch page, type a URL of the following format in the web location field:

`https://`*hostname.domain***`:6789`**

where the following applies:

- `https` specifies a Secure Socket Layer (SSL) connection
- *hostname.domain* specifies the name and domain of the server that is hosting the console
- `6789` is the console's assigned port number

Note – The first time you access the Java Web Console from a particular system, you must accept the server's certificate before the web console's launch page is displayed.

If RBAC is enabled on the system, and your user identity is assigned to a role, you are prompted for a role password after you have successfully logged in. If you assume a role, authorization checks are made for the assumed role. You can opt out of assuming a role by selecting NO ROLE, and then authorization checks are made against your user identity. Following a successful authorization check, the web console launch page is displayed.

▼ How to Start Applications From the Java Web Console's Launch Page

1. **Start a web browser that is compatible with the Java Web Console, such as Firefox 1.0.**

   See "Supported Web Browsers" on page 58 for a list of supported browsers.

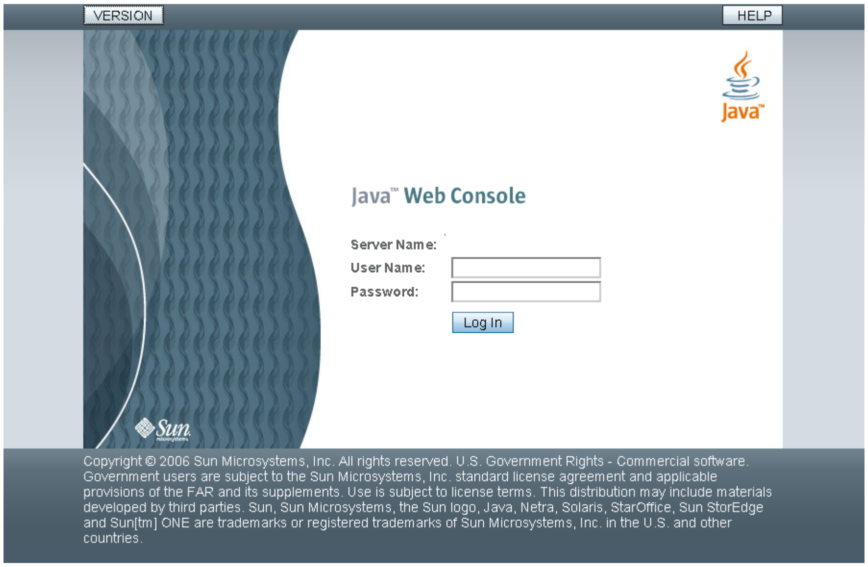
2. **Type the console's URL in the web browser's location field.**

   For example, if the management server host is named `sailfish`, and the domain is `sw`, the URL is `https://sailfish.sw:6789`. This URL takes you to the web console login page.

3. **Accept the server's certificate.**

   You only have to accept the server's certificate once per browser session, not each time you login to the console or start an application.

   The login page is displayed as shown in the following figure.



Copyright © 2006 Sun Microsystems, Inc. All rights reserved. U.S. Government Rights - Commercial software. Government users are subject to the Sun Microsystems, Inc. standard license agreement and applicable provisions of the FAR and its supplements. Use is subject to license terms. This distribution may include materials developed by third parties. Sun, Sun Microsystems, the Sun logo, Java, Netra, Solaris, StarOffice, Sun StorEdge and Sun[tm] ONE are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries.

FIGURE 3–1 Java Web Console Login Page

4 **Enter your user name and password, and optionally your RBAC role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

The console services check your credentials to authenticate them, and ensure that you are authorized to use the console and registered applications.

5 **Click the Start Each Application in a New Window check box if you want to run the application in a new window.**

If you do not select this option, the application will run in the default window, replacing the launch page.

6 **Click the link for the application that you want to run.**

Tip – You can also launch an individual application directly and bypass the launch page by using the following syntax:

`https://`*hostname.domain*`:6789/`*app-context-name*

where *app-context-name* is the name that is used when the application is deployed.

To find the application context name, you can do one of the following:

- Read the application's documentation.
- Run the `wcadmin list -a` or the `smreg list -a` command to see a list of deployed web applications and their context names.
- Run the application from the web console's launch page and note the URL that is displayed in the address location field. You can type the URL directly the next time you use the application. Or, you can bookmark the location and access the application through the bookmark.

Managing the Console Service

Solaris 10 11/06: The Java Web Console service is managed through the Service Management Facility (SMF). You can start, stop, enable, and disable the console service by using SMF commands, or by using the `smcwebserver` script. The FMRI used in SMF for the console is `system/webconsole:console`.

▼ How to Start the Console Service

This procedure starts the server temporarily. If the server was disabled from starting when the system boots, it will continue to be disabled. If the server was enabled, it will continue to be enabled.

Starting with the Solaris 10 11/06 release, the running `enabled` state displays as `true (temporary)`, if the server is running while disabled.

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 **Start the server now, without changing the enabled state.**

```
# smcwebserver start
```

▼ How to Enable the Console Service to Run at System Start

This procedure enables the console service to run each time the system starts. The console is not started in the current session.

Starting with the Solaris 10 11/06 release this procedure sets the `general/enabled` property to true in SMF, so that the server is started at the time the system boots.

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 **Enable the server to be started at system boot.**

```
# smcwebserver enable
```

Solaris 10 11/06: Alternatively, if you want to both start the server now, and enable the server to start when the system boots, use the command:

```
# svcadm enable system/webconsole:console
```

Note – If you are running the Solaris 10 11/06 release, you cannot enable the console by using the `smcwebserver` command. You must use the `svcadm` command.

▼ How to Stop the Console Service

This procedure stops the server temporarily. If the server is disabled from starting when the system boots, it will continue to be disabled. If the server was enabled, it will continue to be enabled.

Starting with the Solaris 10 11/06 release, the running `enabled` state displays as `false (temporary)` if the server is stopped while enabled.

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 **Stop the server now, without changing the enabled state.**

```
# smcwebserver stop
```

▼ How to Disable the Console Service

When the console server is disabled, the server does not start when the system boots.

Starting with the Solaris 10 11/06 release, this procedure sets the console's `general/enabled` property to false in SMF , so that the console server does not start when the system boots.

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 **Disable the server from starting when the system boots.**

```
# smcwebserver disable
```

Solaris 10 11/06: Alternatively, if you want to both stop the server now, and disable the server from starting when the system boots, use the command:

```
# svcadm disable system/webconsole:console
```

Note – If you are running the Solaris 10 11/06 release, you cannot disable the console with the `smcwebserver` command. You must use the `svcadm` command.

Configuring the Java Web Console

The Java Web Console is preconfigured to run without administrator intervention. However, you might choose to change some of the web console's default behavior by overriding the console's configuration properties.

Note – Starting with the Solaris 10 11/06 OS, you must use the `wcadmin` command to change these properties. Previously, the `smreg` command was used. For more information about the `wcadmin` command, see the `wcadmin`(1M) man page.

Properties in the console's configuration files control the behavior of the console. To change the behavior, you define new values for properties to override the default values. The default values of most properties should not be overridden unless there is a specific need that the default values do not provide, such as specifying your own login service.

In general, the property values that you might consider changing are the following:

- **Console session timeout**

  The web console's session timeout period is controlled by the `session.timeout.value` property. This property controls how long a web console page can be displayed without user interaction before the session times out. After the timeout is reached, the user must log in again. The default value is 15 minutes. You can set a new value, in minutes, to conform to your own security policy. However, keep in mind that this property controls the timeout period for all console users and all registered applications.

  See Example 3–1 for an example of how to change the session timeout.

- **Logging level**

  You use logging properties to configure the logging service. The console log files are created in the `/var/log/webconsole/console` directory. The `logging.default.level` property determines which messages are logged. The console logs provide valuable information for troubleshooting problems.

  The logging level applies to any messages that are written through the logging service, which by default uses syslog in the Solaris release The syslog log file is `/var/adm/messages`. The file `/var/log/webconsole/console/console_debug_log` contains log messages written when the debugging service is enabled. This is done by setting the `debug.trace.level` property as described in "Using the Console Debug Trace Log" on page 69. Although the default logging and debug logging services are separate, all Java Web Console logging messages to syslog are also written to the `console_debug_log` to aid in debugging. Generally, the logging service, set with `logging.default.level`, should be always enabled for logging by console applications. Debug logging, set with `debug.trace.level`, should only be enabled to investigate problems.

  The following property values are available for `logging.default.level`:

  - `all`
  - `info`
  - `off`
  - `severe`
  - `warning`

  See Example 3–2 for an example that shows how to change the logging level.

- **Auditing implementation**

  Auditing is the process of generating and logging security-related management events. An event signifies that a specific user has updated the management information on a system. The auditing implementation is used by services and applications that generate audit events.

  The following audit events are defined by the web console:

  - Login
  - Logout
  - Role assumption

When audit events occur, a record of the event is made in an audit log. The location of the audit log varies with the auditing implementation that is in use. The web console's auditing service uses an auditing implementation that is provided by the underlying operating system.

The web console supports three auditing implementations: `Solaris`, `Log`, and `None`. You can select an auditing implementation by specifying one of these keywords for the value of the `audit.default.type` configuration property. Only one auditing implementation is in effect at a time.

The supported auditing implementation types are:

- `Solaris`

  The `Solaris` implementation is the default. This implementation supports the BSM auditing mechanism. The auditing mechanism writes audit records into a system file in the `/var/audit` directory.

  You can display the records with the `praudit` command. For events to be captured, you must enable the BSM auditing mechanism on the system. In addition, the `/etc/security/audit_control` file must contain entries that indicate which events should be generated. You must set the `lo` event as the flag option to see login and logout events for each user. For more information, see the `praudit`(1M) and `bsmconv`(1M) man pages and Part VII, "Solaris Auditing," in *System Administration Guide: Security Services*.
- `Log`

  You can configure this implementation to write to the system's `syslog` service. Audit messages are written to the console log if the logging service has been enabled at the `info` level. See Example 3–2 for more information.
- `None`

  No audit events are generated. Audit messages are written to the debug trace log, if enabled.

See Example 3–5 for an example of specifying the auditing implementation.

▼ How to Change Java Web Console Properties

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 **Depending on which Solaris release you are running, change the selected property value as follows:**

- **If you are running at least the Solaris 10 11/06 release, use this command:**

  ```
  # wcadmin add -p -a console name=value
  ```

`-p`	Specifies that the object type is a property.
`-a console`	Specifies that the property changes are for the application named `console`. The `-a console` option must always be used when you are changing console properties.
name=value	Specifies the property name and the new value for that property.

- **If you are *not* running at least the Solaris Express 5/06 release, use this command:**

  ```
  # smreg add -p -c name
  ```

3 **(Optional) Reset a console property to its default value.**

- **If you are running at least the Solaris 10 11/06 release, use this command:**

  ```
  # wcadmin remove -p -a console name=value
  ```

- **If you are *not* running at least the Solaris Express 5/06 release, use this command:**

  ```
  # smreg remove -p -c name
  ```

`-p`	Specifies that the object type is a property.
`-c`	Specifies that the property changes are for the console application. The `-c` option must always be used when you are changing console properties.
name	Specifies the property name and the new value for that property.

Example 3–1 Changing the Java Web Console's Session Timeout Property

This example shows how to set the session time out value to 5 minutes.

```
# wcadmin add -p -a console session.timeout.value=5
```

Example 3–2 Configuring the Java Web Console Logging Level

This example shows you how to set the logging level to `all`.

```
# wcadmin add -p -a console logging.default.level=all
```

Example 3–3 Resetting the Java Web Console Logging Level to the Default Value

This example shows how to reset the logging level to the default.

```
# wcadmin remove -p -a console logging.default.level
```

Example 3–4 Specifying a Java Version for the Java Web Console

This example shows how to set the Java version for the console.

```
# wcadmin add -p -a console java.home=/usr/java
```

Example 3–5 Choosing an Auditing Implementation for the Java Web Console

This example shows you how to set the auditing implementation to `None`.

```
# wcadmin add -p -a console audit.default.type=None
```

The valid auditing types are:

None	No auditing
Log	Audit messages to syslog
Solaris	Audit messages to BSM

Java Web Console User Identity

By default, the web console runs under the user identity, `noaccess`. However, some system configurations disable the `noaccess` user, or set the login shell for the `noaccess` user to an invalid entry to make this user identity unusable.

When the `noaccess` user is not usable, the web console server cannot be started or configured, so an alternative user identity must be specified. Ideally, the user identity should be changed only once, before the console server is configured at initial startup.

You can configure the web console to run under an alternative non-root user identity by using either of the following commands before the console starts:

```
# smcwebserver start -u username
```

This command starts the web console server under the specified user identity. The web console server runs under this identity each time the server is subsequently started if the command is issued before the first console start.

If you are running at least the Solaris 10 11/06 release, you can also use this command:

```
# wcadmin add -p -a console com.sun.web.console.user=username
```

Note – Starting with the Solaris 10 11/06 release, when the system initially starts, the console also starts and is automatically configured to run under `noaccess`. Consequently, the user identity is set to `noaccess` before you are able to change the user identity. Use the following commands to reset the console to its initial unconfigured state. Then, specify a different user identity when you restart the console.

```
# smcwebserver stop
# /usr/share/webconsole/private/bin/wcremove -i console
# smcwebserver start -u new_user_identity
```

If you are *not* running at least the Solaris Express 5/06 release, use this command:

For the Solaris 10, Solaris 10 1/06, Solaris 10 6/06 releases, use this command:

```
# smreg add -p -c com.sun.web.console.user=username
```

This command causes the web console server to run under the specified user identity the next time the server starts, and each time the server is started.

Using the Console Debug Trace Log

By default, the console does not log debug messages. You can turn on debug logging to help troubleshoot console service problems.

Use the `debug.trace.level` property to turn on debug logging by setting the property to a value other than 0.

Available choices include the following:

- **1** - Use this setting to record potentially severe errors.
- **2** - Use this setting to record important messages, as well as error messages of the 1 level.
- **3** - Use this setting to record all possible messages with full details.

By default, the debug trace log is created in the `/var/log/webconsole` directory . Starting with the Solaris 10 11/06 release, the log is created in the `/var/log/webconsole/console` directory. The log file is named `console_debug_log`. Historical logs, such as `console_debug_log.1` and `console_debug_log.2` might also exist in this directory. There can be up to five (default setting) historical logs stored in this directory before the earliest log is deleted and a new log is created.

EXAMPLE 3–6 Setting the Console Debug Trace Log Level

Use the following command to set the debug trace log level to 3.

For the **Solaris 10 11/06** release, use this command:

```
# wcadmin add -p -a console debug.trace.level=3
```

For the Solaris 10, Solaris 10 1/06, and the Solaris 10 6/06 releases, use this command:

```
# smreg add -p -c debug.trace.level=3
```

EXAMPLE 3–7 Checking the Status of the `debug.trace.level` Property

To check the status of the `debug.trace.level` property, use the `wcadmin list` or `smreg list` command.

Solaris 10 11/06:

```
# wcadmin list -p | grep "debug.trace.level"
```

If you are *not* running at least the Solaris Express 5/06 release, use this command:

```
# smreg list -p | grep "debug.trace.level"
```

Troubleshooting the Java Web Console Software (Task Map)

Task	Description	For Instructions
Check to determine if the console is running and enabled.	Use the `smcwebserver`, `wcadmin`, and `svcs` commands to check if the console is running and enabled. This information is useful for troubleshooting problems.	"How to Check if the Console is Running and Enabled" on page 72
List console resources and properties.	You might need to gather information about the console resources and properties for troubleshooting purposes.	"How to List Console Resources and Properties" on page 72

Task	Description	For Instructions
Determine if an application is a legacy application.	Current applications are registered and deployed with a single command while the console server is running. Legacy applications require the console server to be stopped during registration. If you need to register or unregister an application, you must first determine if the application is a legacy application	"How to Determine if an Application is a Legacy Application" on page 75
List all registered applications.	You can list all applications that are registered with the Java Web Console. Listing all registered applications provides you with information that can be helpful in troubleshooting situations.	"How to List Deployed Applications" on page 75
Register a legacy application with the Java Web Console.	If you need to use a legacy application, you must first register the application with the Java Web Console.	"How to Register a Legacy Application With the Java Web Console" on page 76
Unregister a legacy application from the Java Web Console.	If you do not want a legacy application registered with the Java Web Console, follow the procedure to unregister the legacy application.	"How to Unregister a Legacy Application From the Java Web Console" on page 77
Register a current application with the Java Web Console.	Before using a new application, you need to register the application with the Java Web Console.	"How to Register a Current Application With the Java Web Console" on page 78
Unregister a current application from the Java Web Console.	In some situations, you might need to unregister a current application from the Java Web Console.	"How to Unregister a Current Application from the Java Web Console" on page 78
Enable remote Access to the Java Web Console.	You can enable remote access only to the console, while leaving the other access restrictions in place.	"How to Enable Remote Access to the Java Web Console" on page 83
Change the console's internal passwords	The Java Web Console uses internal passwords. To reduce the possibility of a security breach, you can change these passwords.	"How to Change the Console's Internal Passwords" on page 84

Troubleshooting the Java Web Console Software

The following information is provided to help you troubleshoot any problems that you might encounter when using the Java Web Console software.

Checking Console Status and Properties

You can use the `smcwebserver`, `wcadmin`, and `svcs` commands to get different types of information about the console, which might be useful for troubleshooting problems.

▼ How to Check if the Console is Running and Enabled

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 **Check the server status.**

```
# smcwebserver status
Sun Java(TM) Web Console is running
```

3 **Solaris 10 11/06: Check the console's SMF status and enabled state.**

```
# svcs -l system/webconsole:console
fmri         svc:/system/webconsole:console
name         java web console
enabled      true
state        online
next_state   none
state_time   Wed 17 May 2006 01:22:32 PM EDT
logfile      /var/svc/log/system-webconsole:console.log
restarter    svc:/system/svc/restarter:default
contract_id  129
dependency   require_all/none svc:/milestone/multi-user (online)
```

If you start and stop the server with `smcwebserver` commands without enabling and disabling, the `enabled` property might display as `false (temporary)` or `true (temporary)`.

▼ How to List Console Resources and Properties

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 List the console's resources and properties.

If you are running at least the Solaris 10 11/06 release, use this command:

```
# wcadmin list

Deployed web applications (application name, context name, status):

    console     ROOT            [running]
    console     com_sun_web_ui  [running]
    console     console         [running]
    console     manager         [running]
    legacy      myapp           [running]

Registered jar files (application name, identifier, path):

    console  audit_jar     /usr/lib/audit/Audit.jar
    console  console_jars  /usr/share/webconsole/lib/*.jar
    console  jato_jar      /usr/share/lib/jato/jato.jar
    console  javahelp_jar  /usr/jdk/packages/javax.help-2.0/lib/*.jar
    console  shared_jars   /usr/share/webconsole/private/container/shared/lib/*.jar

Registered login modules (application name, service name, identifier):

    console  ConsoleLogin  userlogin
    console  ConsoleLogin  rolelogin

Shared service properties (name, value):

    ENABLE           yes
    java.home        /usr/jdk/jdk1.5.0_06
```

Note – This ENABLE property is ignored because SMF uses its own enabled property, which is shown in the previous procedure. The ENABLE property is used on older Solaris systems where the console server is not managed by SMF.

For the Solaris 10, Solaris 10 1/06, and Solaris 10 6/06 releases, use this command:

```
# smreg list

 The list of registered plugin applications:

com.sun.web.console_2.2.4       /usr/share/webconsole/console
com.sun.web.ui_2.2.4    /usr/share/webconsole/com_sun_web_ui
com.sun.web.admin.example_2.2.4 /usr/share/webconsole/example

The list of registered jar files:
```

```
com_sun_management_services_api.jar scoped to ALL
com_sun_management_services_impl.jar scoped to ALL
com_sun_management_console_impl.jar scoped to ALL
com_sun_management_cc.jar scoped to ALL
com_sun_management_webcommon.jar scoped to ALL
com_iplanet_jato_jato.jar scoped to ALL
com_sun_management_solaris_impl.jar scoped to ALL
com_sun_management_solaris_implx.jar scoped to ALL

The list of registered login modules for service ConsoleLogin:

com.sun.management.services.authentication.PamLoginModule optional
use_first_pass="true" commandPath="/usr/lib/webconsole";
com.sun.management.services.authentication.RbacRoleLoginModule requisite
force_role_check="true" commandPath="/usr/lib/webconsole";

The list of registered server configuration properties:

session.timeout.value=15
authentication.login.cliservice=ConsoleLogin
logging.default.handler=com.sun.management.services.logging.ConsoleSyslogHandler
logging.default.level=info
logging.default.resource=com.sun.management.services.logging.resources.Resources
logging.default.filter=none
logging.debug.level=off
audit.default.type=None
audit.None.class=com.sun.management.services.audit.LogAuditSession
audit.Log.class=com.sun.management.services.audit.LogAuditSession audit.class.fail=none
authorization.default.type=SolarisRbac
authorization.SolarisRbac.class=
com.sun.management.services.authorization.SolarisRbacAuthorizationService
authorization.PrincipalType.class=
com.sun.management.services.authorization.PrincipalTypeAuthorizationService
debug.trace.level=0
.
.
.
No environment properties have been registered.
```

Problems Accessing the Console

Problems with console access might indicate that the console server is not enabled, or security settings are restrictive. See "Checking Console Status and Properties" on page 72 and "Java Web Console Security Considerations" on page 79 for more information.

Problems with Application Registration

This section contains information about solving possible registration problems with console applications. For information about a particular console application, you should refer to the application's documentation.

Note – Console applications typically are registered as part of their installation process, so you should not normally need to register an application yourself.

Starting with the Solaris 10 11/06 release, the web console has changed the approach to application registration but can still support applications that were developed for earlier versions of the console. Current applications are registered and deployed with a single command while the console server is running. Applications that were developed for the earlier console are known as *legacy* applications, and require the console server to be stopped during registration. If you need to register or unregister an application, you must first determine if the application is a legacy application, as described in the following procedure.

▼ How to Determine if an Application is a Legacy Application

1. **View the application's `app.xml` file.**

   The `app.xml` file is located in the application's `WEB-INF` directory.

2. **Examine the `registrationInfo` tag in the `app.xml` file.**

   For a legacy application, the `registrationInfo` tag is a version 2.*x* . For example, `registrationInfo version="2.2.4"`.

   For a current application, the version in the `registrationInfo` tag is at least 3.0. For example, `registrationInfo version="3.0"`.

▼ How to List Deployed Applications

1. **Become superuser or assume an equivalent role.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2. **List the deployed applications.**

   If you are running at least the Solaris 10 11/06 release, use this command:

   ```
   # wcadmin list -a

   Deployed web applications (application name, context name, status):
   ```

```
    console  ROOT            [running]
    console  com_sun_web_ui  [running]
    console  console         [running]
    console  manager         [running]
    legacy   myapp           [running]
```

The command lists all the registered and deployed applications. Legacy applications are listed with the application name `legacy`. See "How to Determine if an Application is a Legacy Application" on page 75. All other listed applications are current applications, and would be registered as described in "How to Register a Current Application With the Java Web Console" on page 78.

Typically, the status that is shown for the applications contains either `running` or `stopped`. If the status is `running`, the application is currently loaded and available. If the status is `stopped`, then the application is not currently loaded and is unavailable. Sometimes an application registers and deploys successfully, but does not load because of a problem in the application. If so, the application's status is `stopped`. Check the `console_debug_log` to determine if there is an error with a traceback from the console's underlying web container, Tomcat, when attempting to load the application. For more information about the `console_debug_log`, see "Using the Console Debug Trace Log" on page 69.

If all the applications show `stopped` (including the console application), this usually means the console's web container is not running. The list of applications in this case is obtained from the static `context.xml` files registered with the web container.

For the Solaris 10, Solaris 10 1/06, and Solaris 10 6/06 releases, use this command:

```
# smreg list -a

The list of registered plugin applications:

        com.sun.web.console_2.2.4       /usr/share/webconsole/console
        com.sun.web.ui_2.2.4    /usr/share/webconsole/com_sun_web_ui
        com.sun.web.admin.yourapp_2.2.4 /usr/share/webconsole/yourapp
```

▼ How to Register a Legacy Application With the Java Web Console

Note – This procedure applies to all console applications in the Solaris 10, Solaris 10 1/06, and Solaris 10 6/06 releases. Starting with Solaris 10 11/06 release, this procedure also applies *only* to those applications that are identified as legacy applications. See "How to Register a Current Application With the Java Web Console" on page 78 for the registration procedure for current applications. See also "How to Determine if an Application is a Legacy Application" on page 75.

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 Stop the web server.

```
# smcwebserver stop
```

3 Register the application.

```
# smreg add -a /directory/containing/application-files
```

The smreg command manages the information in the Java Web Console's registration table. This script also performs some additional work to deploy the application. For additional options to this command, see the smreg(1M) man page.

4 Restart the web server.

```
# smcwebserver start
```

Example 3–8 Registering a Legacy Application

This example shows how to register a legacy application whose files are located in the /usr/share/webconsole/example directory. Notice that for legacy applications, the console server must be stopped before the application is registered, and started after the application is registered. A warning given by smreg can be ignored because this application is a legacy console application.

```
# smcwebserver stop
# smreg add -a /usr/share/webconsole/example

    Warning: smreg is obsolete and is preserved only for
    compatibility with legacy console applications. Use wcadmin instead.

    Type "man wcadmin" or "wcadmin --help" for more information.

Registering com.sun.web.admin.example_version.

# smcwebserver start
```

▼ How to Unregister a Legacy Application From the Java Web Console

Note – This procedure applies to all console applications in the Solaris 10, Solaris 10 1/06, and Solaris 10 6/06 releases. Starting with Solaris 10 11/06 release, this procedure applies *only* to those applications that are identified as legacy applications. See "How to Unregister a Current Application from the Java Web Console" on page 78 for the procedure that describes how to unregister current applications.

If you do not want a particular legacy application to display in the web console's launch page, but you do not want to uninstall the software, you can use the smreg command to unregister the application. See "How to Determine if an Application is a Legacy Application" on page 75.

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 **Unregister an application.**

```
# smreg remove -a app-name
```

Example 3–9 Unregistering a Legacy Application From the Java Web Console

This example shows how to unregister a legacy application with the *app-name* `com.sun.web.admin.example`.

```
# smreg remove -a com.sun.web.admin.example

 Unregistering com.sun.web.admin.example_version.
```

▼ How to Register a Current Application With the Java Web Console

Solaris 10 11/06: This procedure is for updated console applications that can be registered and deployed without stopping and starting the console server. See "How to Register a Legacy Application With the Java Web Console" on page 76 for the registration procedure for legacy applications and all console applications that are in the Solaris 10, Solaris 10 1/06, Solaris 10 6/06 releases. See also "How to Determine if an Application is a Legacy Application" on page 75.

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 **Register and deploy the application.**

```
wcadmin deploy -a app-name -x app-context-name /full path/to/app-name
```

Example 3–10 Registering Current Applications

This example shows how to register and deploy an application that has been developed or updated for the current web console.

```
# wcadmin deploy -a newexample_1.0 -x newexample /apps/webconsole/newexample
```

▼ How to Unregister a Current Application from the Java Web Console

Solaris 10 11/06: This procedure is for updated console applications, which can be unregistered and undeployed without stopping and starting the console server. See "How to Unregister a Legacy Application From the Java Web Console" on page 77 for the unregistration procedure for legacy applications and all console applications that are in the Solaris 10, Solaris 10 1/06, Solaris 10 6/06 releases. See "How to List Deployed Applications" on page 75 and "How to Determine if an Application is a Legacy Application" on page 75 to determine if an application is a legacy or updated application.

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 **Undeploy and unregister the application.**

```
# wcadmin undeploy -a newexample_1.0 -x newexample
```

Java Web Console Reference Information

This reference section includes the following topics:

- "Java Web Console Security Considerations" on page 79
- "Specifying Authorizations With the `authTypes` Tag" on page 81

Java Web Console Security Considerations

There are several security considerations to keep in mind when you use applications that are in the Java Web Console.

These security considerations include the following:

- **Access to the Java Web Console** – Whether you can connect to the console through a browser.
- **Access to applications** – Whether you can see a particular application in the Java Web Console's launch page.
- **Application permissions** – The levels of permissions that you must have to run parts or all of an application.
- **Application access to remote systems** – How security credentials relate to remote systems
- **Internal passwords used in the console** - Changing the default passwords that are used internally in the console, starting with the Solaris 10 11/06 release.

Access to the Java Web Console

Permissions to the web console launcher application are usually open so that any valid user can log in. However, you can restrict access to the console by specifying the rights in the `authTypes` tag in the web console's `app.xml` file, which is located in the `/usr/share/webconsole/webapps/console/WEB-INF` directory. For more information, see "Specifying Authorizations With the `authTypes` Tag" on page 81.

Some system configurations are set up to be very secure, so that attempts to connect from a remote system to the URLs of the console or registered applications are refused. If your system is configured to prevent remote access, when you try to access the console as `https://hostname.domain:6789`, your browser displays a message such as:

```
Connect to hostname.domain:6789 failed (Connection refused)
```

The SMF profile in effect on the system might be restricting access. See "SMF Profiles" on page 324 for more information about profiles. See "Enabling Remote Access to the Java Web Console" on page 83 for a procedure to allow access to the console from remote systems.

Access to Applications in the Java Web Console

After you successfully log in to the web console, you might not automatically have access to all of the applications that are registered in that console . Typically, applications are installed so that all users can see them in the console launch page. As an administrator, you can grant and restrict access to applications.

To restrict access to an application, specify the rights in the `authTypes` tag, which is in the application's `app.xml` file. You can find the application's `app.xml` file in the *installation-location*`/WEB-INF/` subdirectory. Typically, this directory would be located in `/usr/share/webconsole/webapps/`*app-context-name*`/WEB-INF`.

If the application files are not in the usual location, you can locate the files by using the following command:

```
wcadmin list --detail -a
```

This command lists each deployed application, showing when it was deployed and the path to the application's base directory. The `app.xml` file is located in the subdirectory `WEB-INF` within the base directory.

For more information, see "Specifying Authorizations With the `authTypes` Tag" on page 81.

Application Privileges

If you can see an application's link on the Java Web Console's launch page, you can run that application. However, an application might make additional authorization checks based upon the authenticated user or role identity. These checks are not controlled by the `authTypes` tag, but are explicitly coded into the application itself. For example, an application might grant read access to all authenticated users, but restrict update access to a few users or a few roles.

Application Access to Remote Systems

Having all the appropriate credentials does not guarantee that you can use an application to manage every system within the application's scope of operation. Each system that you administer by using the Java Web Console application has its own security domain. Having read-and-write permissions on the web console system does not guarantee that those credentials are automatically sufficient to administer any other remote system.

In general, access to remote systems depends on how the security is implemented in the web application. Typically, web applications make calls to *agents* that perform actions on behalf of the applications. These applications must be authenticated by the agents based on their web console credentials and the credentials by which they are known on the agent system. Depending upon how this agent authentication is done, an authorization check might also be made on the agent itself, based upon this authenticated identity.

For example, in web applications that use remote WBEM agents, authentication typically uses the user or role identity that initially authenticated to the Java Web Console. If this authentication fails on that agent system, access to that system is denied in the web application. If authentication succeeds on that agent system, access might still be denied if the agent makes an access control check and denies access there. Most applications are written so that the authentication and authorization checks on the agent never fail if you have been successfully authenticated on the web console and assumed the correct role.

Internal Passwords Used in the Console

Starting with the Solaris 10 11/06 release, the Java Web Console uses several password-protected internal user names to perform administrative tasks on the underlying web server, and to encrypt key store and trust store files. The passwords are set to initial values to enable the console to be installed. To reduce the possibility of a security breach, you should change the passwords after installation. See "Changing Internal Passwords for Java Web Console" on page 84

Specifying Authorizations With the `authTypes` Tag

While most system management web applications do not require any administrator intervention to use the `authTypes` tag, in some cases, you might need to change the values of this tag. The `authTypes` tag contains a set of information that describes the level of authorization that is required for a user to view an application in the Java Web Console. The web console determines if a user is authorized to see a particular application, based on the authorization requirements in the application's `app.xml` file. Each application can determine whether a user must have proper authorization to run the application. This determination might be made as part of the application installation process. Or, you might need to supply the information, depending on your own security requirements. The product documentation for the application should contain the information that is necessary to determine whether you need to specify a particular permission.

You can nest several `authType` tags within the `authTypes` tag.

The authTypes tag must contain at least one authType tag that provides the following necessary information:

- Type of authorization check to perform
- Permission subclass name
- Parameters that are required to instantiate the Permission subclass

In the following example, the authType tag has one attribute, name. The required name attribute is the name of the authorization service type. Different authorization types might require different values for the classType and permissionParam tags.

```
<authTypes>
    <authType name="SolarisRbac">
        <classType>
          com.sun.management.solaris.RbacPermission
        </classType>
        <permissionParam name="permission">
          solaris.admin.serialmgr.read
        </permissionParam>
    </authType>
</authTypes>
```

The following table shows the tags that can be nested within an authType tag

TABLE 3–1 Nested authType Tags

Tag	Attribute	Description
classType		The Permission subclass name. This tag is a required tag.
permissionParam	name	The parameters that are required to create an instance of the class specified by classType.

The authTypes tag and nested authType tags are required elements in the app.xml file. If you want to register an application that is available to anyone, specify the authType tag with no content, as shown in the following example.

```
<authTypes>
        <authType name="">
            <classType></classType>
            <permissionParam name=""></permissionParam>
        </authType>
</authTypes>
```

Enabling Remote Access to the Java Web Console

If you can only connect to the console by logging into the system that is running the console, and then using the URL `https://localhost:6789`, the system is using a configuration that prevents remote access. Starting with the Solaris 10 11/06 release, you can enable remote access only to the console, while leaving the other access restrictions in place, by using the following procedure:

▼ How to Enable Remote Access to the Java Web Console

1. **Become superuser or assume an equivalent role on the system where the console is running.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2. **Set a property to allow the console server to respond to network requests and restart the console server.**

```
# svccfg -s svc:/system/webconsole setprop options/tcp_listen = true
# smcwebserver restart
```

Disabling Remote Access to the Java Web Console

You can prevent users from connecting to the console from remote systems. Starting with the Solaris 10 11/06 release, you can disable remote access only to the console, while leaving the other access permissions in place, by using the following procedure:

▼ How to Disable Remote Access to the Java Web Console

1. **Become superuser or assume an equivalent role on the system where the console is running.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2. **Set a property to prevent the console server from responding to network requests, and restart the console server.**

```
# svccfg -s svc:/system/webconsole setprop options/tcp_listen = false
# smcwebserver restart
```

After the restart the console now only responds to a browser on the same system as the console server process. You cannot use a proxy in the browser, only a direct connection. You can also use the `https://localhost:6789/` URL to access the console.

Changing Internal Passwords for Java Web Console

Starting with the Solaris 10 11/06 release, the console uses some internal user names and passwords. The console's internal user names and passwords are used only by the console framework, and are never used directly by a user or system administrator. However, if the passwords were known, a malicious user could potentially interfere with the console applications. To reduce the possibility of such a security breach, you should change the passwords. You do not need to remember the new passwords, because the software uses them invisibly.

▼ How to Change the Console's Internal Passwords

The passwords are known as the administrative password, keystore password, and truststore password. You do not need to know the default initial values in order to change the passwords. This procedure explains how to change all three passwords with separate commands.

1. **Become superuser or assume an equivalent role.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2. **Change the administrative password.**

   ```
   # wcadmin password -a
   ```

   You are prompted to enter the new password twice. The password should be 8 to 32 characters.

3. **Change the key store password.**

   ```
   # wcadmin password -k
   ```

   You are prompted to enter the new password twice. The password should be 8 to 32 characters.

4. **Change the trust store password.**

   ```
   # wcadmin password -t
   ```

   You are prompted to enter the new password twice. The password should be 8 to 32 characters.

◆ ◆ ◆ CHAPTER 4

Managing User Accounts and Groups (Overview)

This chapter provides guidelines and planning information for managing user accounts and groups. This chapter also includes information about customizing the user's work environment.

This is a list of the overview information in this chapter:

- “What's New in Managing Users and Groups?” on page 85
- “What Are User Accounts and Groups?” on page 86
- “Where User Account and Group Information Is Stored” on page 94
- “Tools for Managing User Accounts and Groups” on page 98
- “Customizing a User's Work Environment” on page 102

For step-by-step instructions on managing user accounts and groups, see Chapter 5, “Managing User Accounts and Groups (Tasks).”

What's New in Managing Users and Groups?

This section includes information about new or changed features for managing users and groups in this Solaris release.

`useradd` Default Shell Enhancements

The `useradd` command enables you to specify a default shell by using the `-s` option and a default `skel`directory by using the `-k` option.

For example:

```
# useradd  -D  [-s /usr/bin/ksh]  [-k /export/home] foo
```

You can now also specify a base directory without using the `-D` option.

For example:

```
# useradd [-b /export/home] foo
```

For more information on these changes, see the `useradd(1M)` man page.

Tools for User Account and Group Account Management

The following table describes available tools for user account and group management.

TABLE 4–1 Tools for User Account and Group Management

Tool Name	Description	For More Information
Solaris Management Console	Graphical tool that is used to manage users, groups, roles, rights, mailing lists, disks, terminals, and modems.	"Setting Up User Accounts (Task Map)" on page 113
`smuser`, `smrole`, `smgroup`	Commands that are used to manage users, groups and roles. The SMC services must be running to use these commands.	"Adding a Group and User With the `smgroup` and `smuser` Commands" on page 119
`useradd`, `groupadd`, `roleadd`; `usermod`, `groupmod`, `rolemod`; `userdel`, `groupdel`, `roledel`	Commands that are used to manage users, groups, and roles.	"Adding a Group and User With the `groupadd` and `useradd` Commands" on page 119

Note – The Admintool is not available in this Solaris release.

What Are User Accounts and Groups?

One basic system administration task is to set up a user account for each user at a site. A typical user account includes the information a user needs to log in and use a system, without having the system's root password. The components of user account information are described in "User Account Components" on page 87.

When you set up a user account, you can add the user to predefined groups of users. A typical use of groups is to set up group permissions on a file and directory, which allows access only to users who are part of that group.

For example, you might have a directory containing confidential files that only a few users should be able to access. You could set up a group called `topsecret` that includes the users working on the `topsecret` project. And, you could set up the `topsecret` files with read permission for the `topsecret` group. That way, only the users in the `topsecret` group would be able to read the files.

A special type of user account, called a *role*, is used to give selected users special privileges. For more information, see "Role-Based Access Control (Overview)" in *System Administration Guide: Security Services*.

User Account Components

The following sections describe the specific components of a user account.

User (Login) Names

User names, also called *login names*, let users access their own systems and remote systems that have the appropriate access privileges. You must choose a user name for each user account that you create.

Consider establishing a standard way of assigning user names so that they are easier for you to track. Also, names should be easy for users to remember. A simple scheme when selecting a user name is to use the first name initial and first seven letters of the user's last name. For example, Ziggy Ignatz becomes `zignatz`. If this scheme results in duplicate names, you can use the first initial, middle initial, and the first six characters of the user's last name. For example, Ziggy Top Ignatz becomes `ztignatz`.

If this scheme still results in duplicate names, consider using the following scheme to create a user name:

- The first initial, middle initial, first five characters of the user's last name
- The number 1, or 2, or 3, and so on, until you have a unique name

Note – Each new user name must be distinct from any mail aliases that are known to the system or to an NIS or NIS+ domain. Otherwise, mail might be delivered to the alias rather than to the actual user.

For detailed guidelines on setting up user (login) names, see "Guidelines for Using User Names, User IDs, and Group IDs" on page 93.

User ID Numbers

Associated with each user name is a user identification number (UID). The UID number identifies the user name to any system on which the user attempts to log in. And, the UID number is used by systems to identify the owners of files and directories. If you create user accounts for a single individual on a number of different systems, always use the same user name and ID number. In that way, the user can easily move files between systems without ownership problems.

UID numbers must be a whole number that is less than or equal to 2147483647. UID numbers are required for both regular user accounts and special system accounts. The following table lists the UID numbers that are reserved for user accounts and system accounts.

TABLE 4–2 Reserved UID Numbers

UID Numbers	User or Login Accounts	Description
0 – 99	root, `daemon`, `bin`, `sys`, and so on	Reserved for use by the Solaris OS
100 – 2147483647	Regular users	General purpose accounts
60001 and 65534	`nobody` and `nobody4`	Anonymous users
60002	`noaccess`	Non trusted users

Do not assign UIDs 0 through 99. These UIDs are reserved for allocation by the Solaris Operating System. By definition, `root` always has UID 0, `daemon` has UID 1, and pseudo-user `bin` has UID 2. In addition, you should give `uucp` logins and pseudo user logins, such as `who`, `tty`, and `ttytype`, low UIDs so that they fall at the beginning of the `passwd` file.

For additional guidelines on setting up UIDs, see "Guidelines for Using User Names, User IDs, and Group IDs" on page 93.

As with user (login) names, you should adopt a scheme to assign unique UID numbers. Some companies assign unique employee numbers. Then, administrators add a number to the employee number to create a unique UID number for each employee.

To minimize security risks, you should avoid reusing the UIDs from deleted accounts. If you must reuse a UID, "wipe the slate clean" so that the new user is not affected by attributes set for a former user. For example, a former user might have been denied access to a printer by being included in a printer deny list. However, that attribute might be inappropriate for the new user.

Using Large User IDs and Group IDs

UIDs and group IDs (GIDs) can be assigned up to the maximum value of a signed integer, or 2147483647.

However, UIDs and GIDs over 60000 do not have full functionality and are incompatible with many Solaris features. So, avoid using UIDs or GIDs over 60000.

The following table describes interoperability issues with Solaris products and previous Solaris releases.

TABLE 4–3 Interoperability Issues for UIDs or GIDs Over 60000

Category	Product or Command	Issue
NFS interoperability	SunOS 4.0 NFS software and compatible releases	NFS server and client code truncates large UIDs and GIDs to 16 bits. This situation can create security problems if systems running SunOS 4.0 and compatible releases are used in an environment where large UIDs and GIDs are being used. Systems running SunOS 4.0 and compatible releases require a patch to avoid this problem.
Name service interoperability	NIS name service and file-based name service	Users with UIDs greater than 60000 can log in or use the `su` command on systems running the Solaris 2.5 (and compatible releases). However, their UIDs and GIDs will be set to 60001 (`nobody`).
	NIS+ name service	Users with UIDs greater than 60000 are denied access on systems running Solaris 2.5 (and compatible releases) and the NIS+ name service.

TABLE 4–4 Large UID or GID Limitation Summary

UID or GID	Limitations
60003 or greater	Users who log in to systems running Solaris 2.5 (and compatible releases) and the NIS or files name service get a UID and GID of `nobody`.
65535 or greater	■ Systems running Solaris 2.5 (and compatible releases) with the NFS version 2 software truncate UIDs to 16 bits, creating possible security problems. ■ Users who use the `cpio` command with the default archive format to copy a file see an error message for each file. And, the UIDs and GIDs are set to `nobody` in the archive. ■ x86 based systems: Users that run SVR3-compatible applications will probably see `EOVERFLOW` return codes from system calls. ■ x86 based systems: If users attempt to create a file or directory on a mounted System V file system, the System V file system returns an `EOVERFLOW` error.
100000 or greater	The `ps -l` command displays a maximum five-digit UID. So, the printed column won't be aligned when it includes a UID or GID larger than 99999.
262144 or greater	Users who use the `cpio` command with the `-H odc` format or the `pax -x cpio` command to copy files see an error message returned for each file. And, the UIDs and GIDs are set to `nobody` in the archive.
1000000 or greater	Users who use the `ar` command have their UIDs and GIDs set to `nobody` in the archive.

TABLE 4–4 Large UID or GID Limitation Summary *(Continued)*

UID or GID	Limitations
2097152 or greater	Users who use the `tar` command, the `cpio -H ustar` command, or the `pax -x tar` command have their UIDs and GIDs set to `nobody`.

UNIX Groups

A *group* is a collection of users who can share files and other system resources. For example, users who working on the same project could be formed into a group. A group is traditionally known as a UNIX group.

Each group must have a name, a group identification (GID) number, and a list of user names that belong to the group. A GID number identifies the group internally to the system.

The two types of groups that a user can belong to are as follows:

- **Primary group –** Specifies a group that the operating system assigns to files that are created by the user. Each user must belong to a primary group.
- **Secondary groups –** Specifies one or more groups to which a user also belongs. Users can belong to up to 15 secondary groups.

For detailed guidelines on setting up group names, see "Guidelines for Using User Names, User IDs, and Group IDs" on page 93.

Sometimes, a user's secondary group is not important. For example, ownership of files reflect the primary group, not any secondary groups. Other applications, however, might rely on a user's secondary group memberships. For example, a user has to be a member of the `sysadmin` group (group 14) to use the Admintool software in previous Solaris releases. However, it doesn't matter if group 14 is his or her current primary group.

The `groups` command lists the groups that a user belongs to. A user can have only one primary group at a time. However, a user can temporarily change the user's primary group, with the `newgrp` command, to any other group in which the user is a member.

When adding a user account, you must assign a primary group for a user or accept the default group, `staff` (group 10). The primary group should already exist. If the primary group does not exist, specify the group by a GID number. User names are not added to primary groups. If user names were added to primary groups, the list might become too long. Before you can assign users to a new secondary group, you must create the group and assign it a GID number.

Groups can be local to a system or managed through a name service. To simplify group administration, you should use a name service such as NIS or a directory service such as LDAP. These services enable you to centrally manage group memberships.

User Passwords

You can specify a password for a user when you add the user. Or, you can force the user to specify a password when the user first logs in.

User passwords must comply with the following syntax:

- Password length must at least match the value identified by the `PASSLENGTH` variable in the `/etc/default/passwd` file. By default, `PASSLENGTH` is set to 6.
- The first 6 characters of the password must contain at least two alphabetic characters and have at least one numeric or special character.

Although user names are publicly known, passwords must be kept secret and known only to users. Each user account should be assigned a password. The password can be a combination of six to eight letters, numbers, or special characters.

To make your computer systems more secure, users should change their passwords periodically. For a high level of security, you should require users to change their passwords every six weeks. Once every three months is adequate for lower levels of security. System administration logins (such as root and sys) should be changed monthly, or whenever a person who knows the root password leaves the company or is reassigned.

Many breaches of computer security involve guessing a legitimate user's password. You should make sure that users avoid using proper nouns, names, login names, and other passwords that a person might guess just by knowing something about the user.

Good choices for passwords include the following:

- Phrases (`beammeup`).
- Nonsense words made up of the first letters of every word in a phrase. For example, `swotrb` for SomeWhere Over The RainBow.
- Words with numbers or symbols substituted for letters. For example, `sn00py` for snoopy.

Do not use these choices for passwords:

- Your name (spelled forwards, backwards, or jumbled)
- Names of family members or pets
- Car license numbers
- Telephone numbers
- Social Security numbers
- Employee numbers
- Words related to a hobby or interest
- Seasonal themes, such as Santa in December
- Any word in the dictionary

Home Directories

The home directory is the portion of a file system allocated to a user for storing private files. The amount of space you allocate for a home directory depends on the kinds of files the user creates, their size, and the number of files that are created.

A home directory can be located either on the user's local system or on a remote file server. In either case, by convention the home directory should be created as `/export/home/`*username*. For a large site, you should store home directories on a server. Use a separate file system for each `/export/home`*n* directory to facilitate backing up and restoring home directories. For example, `/export/home1`, `/export/home2`.

Regardless of where their home directory is located, users usually access their home directories through a mount point named `/home/`*username*. When AutoFS is used to mount home directories, you are not permitted to create any directories under the `/home` mount point on any system. The system recognizes the special status of `/home` when AutoFS is active. For more information about automounting home directories, see "Task Overview for Autofs Administration" in *System Administration Guide: Network Services*.

To use the home directory anywhere on the network, you should always refer to the home directory as `$HOME`, not as `/export/home/`*username*. The latter is machine-specific. In addition, any symbolic links created in a user's home directory should use relative paths (for example, `../../../x/y/x`) so that the links are valid no matter where the home directory is mounted.

Name Services

If you are managing user accounts for a large site, you might want to consider using a name or directory service such as LDAP, NIS, or NIS+. A name or directory service enables you to store user account information in a centralized manner instead of storing user account information in every system's `/etc` files. When you use a name or directory service for user accounts, users can move from system to system using the same user account without having site-wide user account information duplicated on every system. Using a name or directory service also promotes centralized and consistent user account information.

User's Work Environment

Besides having a home directory to create and store files, users need an environment that gives them access to the tools and resources they need to do their work. When a user logs in to a system, the user's work environment is determined by initialization files. These files are defined by the user's startup shell, such as the C, Korn, or Bourne shell.

A good strategy for managing the user's work environment is to provide customized user initialization files, such as `.login`, `.cshrc`, `.profile`, in the user's home directory.

Note – Do not use system initialization files, such as `/etc/profile` or `/etc/.login`, to manage a user's work environment. These files reside locally on systems and are not centrally administered. For example, if AutoFS is used to mount the user's home directory from any system on the network, you would have to modify the system initialization files on each system to ensure a consistent environment whenever a user moved from system to system.

For detailed information about customizing user initialization files for users, see "Customizing a User's Work Environment" on page 102.

Another way to customize user accounts is through role-based access control (RBAC). See "Role-Based Access Control (Overview)" in *System Administration Guide: Security Services* for more information.

Guidelines for Using User Names, User IDs, and Group IDs

User names, UIDs, and GIDs should be unique within your organization, which might span multiple domains.

Keep the following guidelines in mind when creating user or role names, UIDs, and GIDs:

- **User names –** They should contain from two to eight letters and numerals. The first character should be a letter. At least one character should be a lowercase letter.

  Note – Even though user names can include a period (.), underscore (_), or hyphen (-), using these characters is not recommended because they can cause problems with some software products.

- **System accounts –** Do not use any of the user names, UIDs, or GIDs that are contained in the default `/etc/passwd` and `/etc/group` files. Do not use the UIDs and GIDs, 0-99. These numbers are reserved for allocation by the Solaris Operating System and should not be used by anyone. Note that this restriction also applies to numbers not currently in use.

  For example, `gdm` is the reserved user name and group name for the GNOME Display Manager daemon and should not be used for another user. For a complete listing of the default `/etc/passwd` and `/etc/group` entries, see Table 4–5 and Table 4–6.

  The `nobody` and `nobody4` accounts should never be used for running processes. These two accounts are reserved for use by NFS. Use of these accounts for running processes could lead to unexpected security risks. Processes that need to run as a non-root user should use the `daemon` or `noaccess` accounts.

- **System account configuration** – The configuration of the default system accounts should never be changed. This includes changing the login shell of a system account that is currently locked. The only exception to this rule is the setting of a password and password aging parameters for the root account.

Where User Account and Group Information Is Stored

Depending on your site policy, user account and group information can be stored in your local system's `/etc` files or in a name or directory service as follows:

- The NIS+ name service information is stored in tables.
- The NIS name service information is stored in maps.
- The LDAP directory service information is stored in indexed database files.

Note – To avoid confusion, the location of the user account and group information is generically referred to as a *file* rather than as a *database*, *table*, or *map*.

Most user account information is stored in the `passwd` file. Password information is stored as follows:

- In the `passwd` file when you are using NIS or NIS+
- In the `/etc/shadow` file when you are using `/etc` files
- In the `people` container when you are using LDAP

Password aging is available when you are using NIS+ or LDAP, but not NIS.

Group information is stored in the `group` file for NIS, NIS+ and files. For LDAP, group information is stored in the `group` container.

Fields in the `passwd` File

The fields in the `passwd` file are separated by colons and contain the following information:

username:*password*:*uid*:*gid*:*comment*:*home-directory*:*login-shell*

For example:

```
kryten:x:101:100:Kryten Series 4000 Mechanoid:/export/home/kryten:/bin/csh
```

For a complete description of the fields in the `passwd` file, see the `passwd(1)` man page.

Default `passwd` File

The default Solaris `passwd` file contains entries for standard daemons. Daemons are processes that are usually started at boot time to perform some system-wide task, such as printing, network administration, or port monitoring.

```
root:x:0:1:Super-User:/:/sbin/sh
daemon:x:1:1::/:
bin:x:2:2::/usr/bin:
sys:x:3:3::/:
adm:x:4:4:Admin:/var/adm:
lp:x:71:8:Line Printer Admin:/usr/spool/lp:
uucp:x:5:5:uucp Admin:/usr/lib/uucp:
nuucp:x:9:9:uucp Admin:/var/spool/uucppublic:/usr/lib/uucp/uucico
smmsp:x:25:25:SendMail Message Submission Program:/:
listen:x:37:4:Network Admin:/usr/net/nls:
gdm:x:50:50:GDM Reserved UID:/:
webservd:x:80:80:WebServer Reserved UID:/:
nobody:x:60001:60001:NFS Anonymous Access User:/:
noaccess:x:60002:60002:No Access User:/:
nobody4:x:65534:65534:SunOS 4.x NFS Anonymous Access User:/:
```

TABLE 4–5 Default `passwd` File Entries

User Name	User ID	Description
`root`	`0`	Superuser account
`daemon`	`1`	Umbrella system daemon associated with routine system tasks
`bin`	`2`	Administrative daemon associated with running system binaries to perform some routine system task
`sys`	`3`	Administrative daemon associated with system logging or updating files in temporary directories
`adm`	`4`	Administrative daemon associated with system logging
`lp`	`71`	Line printer daemon
`uucp`	`5`	Daemon associated with uucp functions
`nuucp`	`6`	Another daemon associated with `uucp` functions
`smmsp`	`25`	Sendmail message submission program daemon
`webservd`	`80`	Account reserved for WebServer access
`gdm`	50	GNOME Display Manager daemon

TABLE 4–5 Default passwd File Entries *(Continued)*

User Name	User ID	Description
listen	37	Network listener daemon
nobody	60001	Account reserved for anonymous NFS access.
noaccess	60002	Assigned to a user or a process that needs access to a system through some application but without actually logging in.
nobody4	65534	SunOS 4.0 or 4.1 version of the nobody user account

Fields in the shadow File

The fields in the shadow file are separated by colons and contain the following information:

username:*password*:*lastchg*:*min*:*max*:*warn*:*inactive*:*expire*

For example:

```
rimmer:86Kg/MNT/dGu.:8882:0::5:20:8978
```

For a complete description of the fields in the shadow file, see the shadow(4) and crypt(1) man pages.

Fields in the group File

The fields in the group file are separated by colons and contain the following information:

group-name:*group-password*:*gid*:*user-list*

For example:

```
bin::2:root,bin,daemon
```

For a complete description of the fields in the group file, see the group(4) man page.

Default group File

The default Solaris group file contains the following system groups that support some system-wide task, such as printing, network administration, or electronic mail. Many of these groups having corresponding entries in the passwd file.

```
root::0:
other::1:
bin::2:root,daemon
```

```
sys::3:root,bin,adm
adm::4:root,daemon
uucp::5:root
mail::6:root
tty::7:root,adm
lp::8:root,adm
nuucp::9:root
staff::10:
daemon::12:root
smmsp::25:
sysadmin::14:
gdm::50:
webservd::80:
nobody::60001:
noaccess::60002:
nogroup::65534:
```

TABLE 4–6 Default group File Entries

Group Name	Group ID	Description
root	0	Superuser group
other	1	Optional group
bin	2	Administrative group associated with running system binaries
sys	3	Administrative group associated with system logging or temporary directories
adm	4	Administrative group associated with system logging
uucp	5	Group associated with uucp functions
mail	6	Electronic mail group
tty	7	Group associated with tty devices
lp	8	Line printer group
nuucp	9	Group associated with uucp functions
staff	10	General administrative group.
daemon	12	Group associated with routine system tasks
sysadmin	14	Administrative group associated with legacy Admintool and Solstice AdminSuite tools
smmsp	25	Daemon for Sendmail message submission program
webservd	80	Group reserved for WebServer access

TABLE 4–6 Default group File Entries (Continued)

Group Name	Group ID	Description
gdm	50	Group reserved for the GNOME Display Manager daemon
nobody	60001	Group assigned for anonymous NFS access
noaccess	60002	Group assigned to a user or a process that needs access to a system through some application but without actually logging in
nogroup	65534	Group assigned to a user who is not a member of a known group

Tools for Managing User Accounts and Groups

The following table lists the recommended tools for managing users and groups. These tools are included in the Solaris Management Console suite of tools. For information about starting and using the Solaris Management Console, see Chapter 2, "Working With the Solaris Management Console (Tasks)."

TABLE 4–7 Tools for Managing Users and Groups

Solaris Management Tool	Purpose
Users	Manage users accounts
User Templates	Create a set of attributes for a specific kind of user like students, engineers, or instructors
Rights	Manage RBAC rights
Administrative Roles	Manage RBAC administrative roles
Groups	Manage group information
Projects	Manage project information
Mailing Lists	Manage mailing lists

Use the Solaris Management Console online help for information on performing these tasks.

For information on the Solaris commands that can be used to manage user accounts and groups, see Table 1–6. These commands provide the same functionality as the Solaris management tools, including authentication and name service support.

Tasks for Solaris User and Group Management Tools

The Solaris user management tools enable you to manage user accounts and groups on a local system or in a name service environment.

This table describes the tasks you can do with the Users tool's User Accounts feature.

TABLE 4–8 Task Descriptions for User Accounts Tool

Task	Description
Add a user	Adds a user to the local system or name service.
Create a user template	Creates a template of predefined user attributes for creating users of the same group, such as students, contractors, or engineers.
Add a user with a user template	Adds a user with a template so that user attributes are predefined.
Clone a user template	Clones a user template if you would like to use a similar set of predefined user attributes. Then, change only some of the attributes as needed.
Set up user properties	Sets up user properties in advance of adding users. Properties include specifying whether a user template is used when adding a user, and whether the home directory or mail box is deleted by default when removing a user.
Add multiple users	Adds multiple users to the local system or name service by specifying a text file, typing each name, or automatically generating a series of user names.
View or change user properties	Displays or changes user properties such as login shell, password, or password options.
Assign rights to users	Assigns RBAC rights to users that will allow them to perform specific administration tasks.
Remove a user	Removes the user from the local system or the name service. Optionally, you can also specify whether the user's home directory or mailbox is removed. The user is also removed from any groups or roles.

For information about adding a user to the local system or name service, see “What Are User Accounts and Groups?” on page 86 and “User Account Components” on page 87.

TABLE 4–9 Task Descriptions for Rights Tool

Task	Description
Grant a right	Grants a user a right to run a specific command or application that was previously only available to an administrator.
View or change existing rights properties	Displays or changes existing rights.
Add an authorization	Adds an authorization, which is a discrete right granted to a role or a user.
View or change an authorization	Displays or changes existing authorizations.

For more information on granting rights to users, see "Contents of Rights Profiles" in *System Administration Guide: Security Services*.

TABLE 4–10 Task Descriptions for Administrative Roles Tool

Task	Description
Add an administrative role	Adds a role that someone would use to perform a specific administrative task.
Assign rights to an administrative role	Assigns specific rights to a role that enable someone to perform a task.
Change an administrative role	Adds or removes rights from a role.

For more information on using administrative roles, see "How to Plan Your RBAC Implementation" in *System Administration Guide: Security Services*.

TABLE 4–11 Task Descriptions for Groups Tool

Task	Description
Add a group	Adds a group to the local system or name service so that the group name is available before you add the user.
Add a user to a group	Adds a user to a group if the user needs access to group-owned files.
Remove a user from a group	Removes a user from a group if the user no longer requires group file access.

For information on adding users to groups, see "UNIX Groups" on page 90.

TABLE 4–12 Task Descriptions for Mailing Lists Tool

Task	Description
Create a mailing list	Creates a mailing list, which is a list of user names for sending email messages.
Change a mailing list name	Changes the mailing list after it is created.
Remove a mailing list	Removes a mailing list if it is no longer used.

For information on creating mailing lists, see the Solaris Management Console's online help.

TABLE 4–13 Task Descriptions for Projects Tool

Task	Description
Create or clone a project	Creates a new project or clones an existing project if the existing project has attributes similar to what you need for the new project.
Modify or view project attributes	Displays or changes existing project attributes.
Delete a project	Removes a project if the project is no longer used.

Managing Users and Resources With Projects

Starting with the Solaris 9 release, users and groups can be members of a *project*, an identifier that indicates a workload component that can be used as the basis of system usage or resource allocation chargeback. Projects are part of the Solaris resource management feature that is used to manage system resources.

Users need to be a member of a project to successfully log in to a system running the Solaris 9 release. By default, users are a member of the `group.staff` project when the Solaris 9 release is installed and no other project information is configured.

User project information is stored in the `/etc/project` file, which can be stored on the local system (files), the NIS name service, or the LDAP directory service. You can use the Solaris Management Console to manage project information.

The `/etc/project` file must exist for users to log in successfully, but requires no administration if you are not using projects.

For more information on using or setting up projects, see Chapter 2, “Projects and Tasks (Overview),” in *System Administration Guide: Virtualization Using the Solaris Operating System*.

Customizing a User's Work Environment

Part of setting up a user's home directory is providing user initialization files for the user's login shell. A *user initialization file* is a shell script that sets up a work environment for a user after the user logs in to a system. Basically, you can perform any task in a user initialization file that you can do in a shell script. However, a user initialization file's primary job is to define the characteristics of a user's work environment, such as a user's search path, environment variables, and windowing environment. Each login shell has its own user initialization file or files, which are listed in the following table.

TABLE 4–14 User Initialization Files for Bourne, C, and Korn Shells

Shell	User Initialization File	Purpose
Bourne	`$HOME/.profile`	Defines the user's environment at login
C	`$HOME/.cshrc`	Defines the user's environment for all C shells and is invoked after login shell
	`$HOME/.login`	Defines the user's environment at login
Korn	`$HOME/.profile`	Defines the user's environment at login
	`$HOME/$ENV`	Defines user's environment at login in the file and is specified by the Korn shell's `ENV` environment variable

The Solaris environment provides default user initialization files for each shell in the `/etc/skel` directory on each system, as shown in the following table.

TABLE 4–15 Default User Initialization Files

Shell	Default File
C	`/etc/skel/local.login`
	`/etc/skel/local.cshrc`
Bourne or Korn	`/etc/skel/local.profile`

You can use these files as a starting point and modify them to create a standard set of files that provide the work environment common to all users. Or, you can modify these files to provide the working environment for different types of users. Although you cannot create customized user initialization files with the Users tool, you can populate a user's home directory with user initialization files located in a specified “skeleton” directory. You can do this by creating a user template with the User Templates tool and specifying a skeleton directory from which to copy user initialization files.

For step-by-step instructions on how to create sets of user initialization files for different types of users, see "How to Customize User Initialization Files" on page 115.

When you use the Users tool to create a new user account and select the create home directory option, the following files are created, depending on which login shell is selected.

TABLE 4–16 Files Created by Users Tool When Adding a User

Shell	Files Created
C	The `/etc/skel/local.cshrc` and the `/etc/skel/local.login` files are copied into the user's home directory and are renamed `.cshrc` and `.login`, respectively.
Bourne and Korn	The `/etc/skel/local.profile` file is copied into the user's home directory and renamed `.profile`.

If you use the `useradd` command to add a new user account and specify the `/etc/skel` directory by using the `-k` and `-m` options, all three `/etc/skel/local*` files and the `/etc/skel/.profile` file are copied into the user's home directory. At this point, you need to rename them to whatever is appropriate for the user's login shell.

Using Site Initialization Files

The user initialization files can be customized by both the administrator and the user. This important feature can be accomplished with centrally located and globally distributed user initialization files, called *site initialization files*. Site initialization files enable you to continually introduce new functionality to the user's work environment, while enabling the user to customize the user's initialization file.

When you reference a site initialization file in a user initialization file, all updates to the site initialization file are automatically reflected when the user logs in to the system or when a user starts a new shell. Site initialization files are designed for you to distribute site-wide changes to users' work environments that you did not anticipate when you added the users.

You can customize a site initialization file the same way that you customize a user initialization file. These files typically reside on a server, or set of servers, and appear as the first statement in a user initialization file. Also, each site initialization file must be the same type of shell script as the user initialization file that references it.

To reference a site initialization file in a C-shell user initialization file, place a line similar to the following at the beginning of the user initialization file:

```
source /net/machine-name/export/site-files/site-init-file
```

To reference a site initialization file in a Bourne-shell or Korn-shell user initialization file, place a line similar to the following at the beginning of the user initialization file:

```
. /net/machine-name/export/site-files/site-init-file
```

Avoiding Local System References

You should not add specific references to the local system in the user initialization file. You want the instructions in a user initialization file to be valid regardless of which system the user logs into.

For example:

- To make a user's home directory available anywhere on the network, always refer to the home directory with the variable `$HOME`. For example, use `$HOME/bin` instead of `/export/home/`*username*`/bin`. The `$HOME` variable works when the user logs in to another system and the home directories are automounted.
- To access files on a local disk, use global path names, such as `/net/`*system-name/directory-name*. Any directory referenced by `/net/`*system-name* can be mounted automatically on any system on which the user logs in, assuming the system is running AutoFS.

Shell Features

The following table lists basic shell features that each shell provides, which can help you determine what you can and can't do when creating user initialization files for each shell.

TABLE 4–17 Basic Features of Bourne, C, and Korn Shells

Feature	Bourne	C	Korn
Known as the standard shell in UNIX	Applicable	N/A	N/A
Compatible syntax with Bourne shell	-	N/A	Applicable
Job control	Applicable	Applicable	Applicable
History list	N/A	Applicable	Applicable
Command-line editing	N/A	Applicable	Applicable
Aliases	N/A	Applicable	Applicable
Single-character abbreviation for login directory	N/A	Applicable	Applicable
Protection from overwriting (`noclobber`)	N/A	Applicable	Applicable
Setting to ignore Control-D (`ignoreeof`)	N/A	Applicable	Applicable

TABLE 4–17 Basic Features of Bourne, C, and Korn Shells *(Continued)*

Feature	Bourne	C	Korn
Enhanced `cd` command	N/A	Applicable	Applicable
Initialization file separate from `.profile`	N/A	Applicable	Applicable
Logout file	N/A	Applicable	N/A

Shell Environment

A shell maintains an environment that includes a set of variables defined by the `login` program, the system initialization file, and the user initialization files. In addition, some variables are defined by default.

A shell can have two types of variables:

- **Environment variables –** Variables that are exported to all processes spawned by the shell. Their settings can be seen with the `env` command. A subset of environment variables, such as `PATH`, affects the behavior of the shell itself.
- **Shell (local) variables –** Variables that affect only the current shell. In the C shell, a set of these shell variables have a special relationship to a corresponding set of environment variables. These shell variables are `user`, `term`, `home`, and `path`. The value of the environment variable counterpart is initially used to set the shell variable.

In the C shell, you use the lowercase names with the `set` command to set shell variables. You use uppercase names with the `setenv` command to set environment variables. If you set a shell variable, the shell sets the corresponding environment variable. Likewise, if you set an environment variable, the corresponding shell variable is also updated. For example, if you update the `path` shell variable with a new path, the shell also updates the `PATH` environment variable with the new path.

In the Bourne and Korn shells, you can use the uppercase variable name equal to some value to set both shell and environment variables. You also have to use the `export` command to activate the variables for any subsequently executed commands.

For all shells, you generally refer to shell and environment variables by their uppercase names.

In a user initialization file, you can customize a user's shell environment by changing the values of the predefined variables or by specifying additional variables. The following table shows how to set environment variables in a user initialization file.

TABLE 4–18 Setting Environment Variables in a User Initialization File

Shell Type	Line to Add to the User Initialization File
C shell	`setenv` *VARIABLE value* Example: `setenv MAIL /var/mail/ripley`
Bourne or Korn shell	*VARIABLE=value*`; export` *VARIABLE* Example: `MAIL=/var/mail/ripley;export MAIL`

The following table describes environment variables and shell variables that you might want to customize in a user initialization file. For more information about variables that are used by the different shells, see the `sh(1)`, `ksh(1)`, or `csh(1)` man pages.

TABLE 4–19 Shell and Environment Variable Descriptions

Variable	Description
`CDPATH`, or `cdpath` in the C shell	Sets a variable used by the `cd` command. If the target directory of the `cd` command is specified as a relative path name, the `cd` command first looks for the target directory in the current directory ("."). If the target is not found, the path names listed in the `CDPATH` variable are searched consecutively until the target directory is found and the directory change is completed. If the target directory is not found, the current working directory is left unmodified. For example, the `CDPATH` variable is set to `/home/jean`, and two directories exist under `/home/jean`, `bin`, and `rje`. If you are in the `/home/jean/bin` directory and type `cd rje`, you change directories to `/home/jean/rje`, even though you do not specify a full path.
`history`	Sets the history for the C shell.
`HOME`, or `home` in the C shell	Sets the path to the user's home directory.
`LANG`	Sets the locale.
`LOGNAME`	Defines the name of the user currently logged in. The default value of `LOGNAME` is set automatically by the login program to the user name specified in the `passwd` file. You should only need to refer to, not reset, this variable.
`LPDEST`	Sets the user's default printer.
`MAIL`	Sets the path to the user's mailbox.
`MANPATH`	Sets the hierarchies of man pages that are available.

TABLE 4–19 Shell and Environment Variable Descriptions *(Continued)*

Variable	Description
PATH, or path in the C shell	Specifies, in order, the directories that the shell searches to find the program to run when the user types a command. If the directory is not in the search path, users must type the complete path name of a command. As part of the login process, the default PATH is automatically defined and set as specified in .profile (Bourne or Korn shell) or .cshrc (C shell). The order of the search path is important. When identical commands exist in different locations, the first command found with that name is used. For example, suppose that PATH is defined in Bourne and Korn shell syntax as PATH=/bin:/usr/bin:/usr/sbin:$HOME/bin and a file named sample resides in both /usr/bin and /home/jean/bin. If the user types the command sample without specifying its full path name, the version found in /usr/bin is used.
prompt	Defines the shell prompt for the C shell.
PS1	Defines the shell prompt for the Bourne or Korn shell.
SHELL, or shell in the C shell	Sets the default shell used by make, vi, and other tools.
TERMINFO	Specifies the path name for an unsupported terminal that has been added to the terminfo file. Use the TERMINFO variable in either the /etc/profile or /etc/.login file. When the TERMINFO environment variable is set, the system first checks the TERMINFO path defined by the user. If the system does not find a definition for a terminal in the TERMINFO directory defined by the user, it searches the default directory, /usr/share/lib/terminfo, for a definition. If the system does not find a definition in either location, the terminal is identified as "dumb."
TERM, or term in the C shell	Defines the terminal. This variable should be reset in either the /etc/profile or /etc/.login file. When the user invokes an editor, the system looks for a file with the same name that is defined in this environment variable. The system searches the directory referenced by TERMINFO to determine the terminal characteristics.
TZ	Sets the time zone. The time zone is used to display dates, for example, in the ls -l command. If TZ is not set in the user's environment, the system setting is used. Otherwise, Greenwich Mean Time is used.

The PATH Variable

When the user executes a command by using the full path, the shell uses that path to find the command. However, when users specify only a command name, the shell searches the directories for the command in the order specified by the PATH variable. If the command is found in one of the directories, the shell executes the command.

A default path is set by the system. However, most users modify it to add other command directories. Many user problems related to setting up the environment and accessing the correct version of a command or a tool can be traced to incorrectly defined paths.

Setting Path Guidelines

Here are some guidelines for setting up efficient `PATH` variables:

- If security is not a concern, put the current working directory (.) first in the path. However, including the current working directory in the path poses a security risk that you might want to avoid, especially for superuser.
- Keep the search path as short as possible. The shell searches each directory in the path. If a command is not found, long searches can slow down system performance.
- The search path is read from left to right, so you should put directories for commonly used commands at the beginning of the path.
- Make sure that directories are not duplicated in the path.
- Avoid searching large directories, if possible. Put large directories at the end of the path.
- Put local directories before NFS mounted directories to lessen the chance of “hanging” when the NFS server does not respond. This strategy also reduces unnecessary network traffic.

Setting a User's Default Path

This is an example of how to set a user's default path.

The following examples show how to set a user's default path to include the home directory and other NFS mounted directories. The current working directory is specified first in the path. In a C-shell user initialization file, you would add the following:

```
set path=(. /usr/bin $HOME/bin /net/glrr/files1/bin)
```

In a Bourne-shell or Korn-shell user initialization file, you would add the following:

```
PATH=.:/usr/bin:/$HOME/bin:/net/glrr/files1/bin
export PATH
```

Locale Variables

The `LANG` and `LC` environment variables specify the locale-specific conversions and conventions for the shell. These conversions and conventions include time zones, collation orders, and formats of dates, time, currency, and numbers. In addition, you can use the `stty` command in a user initialization file to indicate whether the terminal session will support multibyte characters.

The `LANG` variable sets all possible conversions and conventions for the given locale. You can set various aspects of localization separately through these `LC` variables: `LC_COLLATE`, `LC_CTYPE`, `LC_MESSAGES`, `LC_NUMERIC`, `LC_MONETARY`, and `LC_TIME`.

The following table describes some of the values for the `LANG` and `LC` environment variables.

TABLE 4–20 Values for `LANG` and `LC` Variables

Value	Locale
`de_DE.ISO8859-1`	`German`
`en_US.UTF-8`	`American English` (UTF-8)
`es_ES.ISO8859-1`	`Spanish`
`fr_FR.ISO8859-1`	`French`
`it_IT.ISO8859-1`	`Italian`
`ja_JP.eucJP`	`Japanese` (EUC)
`ko_KR.EUC`	`Korean` (EUC)
`sv_SE.ISO8859-1`	`Swedish`
`zh_CN.EUC`	`Simplified Chinese` (EUC)
`zh_TW.EUC`	`Traditional Chinese` (EUC)

For more information on supported locales, see the *International Language Environments Guide*.

EXAMPLE 4–1 Setting the Locale Using the `LANG` Variables

The following examples show how to set the locale by using the `LANG` environment variables. In a C-shell user initialization file, you would add the following:

```
setenv LANG de_DE.ISO8859-1
```

In a Bourne-shell or Korn-shell user initialization file, you would add the following:

```
LANG=de_DE.ISO8859-1; export LANG
```

Default File Permissions (`umask`)

When you create a file or directory, the default file permissions assigned to the file or directory are controlled by the *user mask*. The user mask is set by the `umask` command in a user initialization file. You can display the current value of the user mask by typing `umask` and pressing Return.

The user mask contains the following octal values:

- The first digit sets permissions for the user
- The second digit sets permissions for group
- The third digit sets permissions for other, also referred to as `world`

Note that if the first digit is zero, it is not displayed. For example, if the user mask is set to 022, 22 is displayed.

To determine the `umask` value you want to set, subtract the value of the permissions you want from 666 (for a file) or 777 (for a directory). The remainder is the value to use with the `umask` command. For example, suppose you want to change the default mode for files to 644 (`rw-r--r--`). The difference between 666 and 644 is 022, which is the value you would use as an argument to the `umask` command.

You can also determine the `umask` value you want to set by using the following table. This table shows the file and directory permissions that are created for each of the octal values of `umask`.

TABLE 4–21 Permissions for `umask` Values

`umask` **Octal Value**	**File Permissions**	**Directory Permissions**
`0`	`rw-`	`rwx`
`1`	`rw-`	`rw-`
`2`	`r--`	`r-x`
`3`	`r--`	`r--`
`4`	`-w-`	`-wx`
`5`	`-w-`	`-w-`
`6`	`--x`	`--x`
`7`	`---` (none)	`---` (none)

The following line in a user initialization file sets the default file permissions to `rw-rw-rw-`.

```
umask 000
```

User and Site Initialization Files Examples

The following sections provide examples of user and site initialization files that you can use to start customizing your own initialization files. These examples use system names and paths that you need to change for your particular site.

EXAMPLE 4–2 The `.profile` File

```
(Line 1) PATH=$PATH:$HOME/bin:/usr/local/bin:/usr/ccs/bin:.
(Line 2) MAIL=/var/mail/$LOGNAME
(Line 3) NNTPSERVER=server1
(Line 4) MANPATH=/usr/share/man:/usr/local/man
(Line 5) PRINTER=printer1
(Line 6) umask 022
(Line 7) export PATH MAIL NNTPSERVER MANPATH PRINTER
```

1. Defines the user's shell search path
2. Defines the path to the user's mail file
3. Defines the user's Usenet news server
4. Defines the user's search path for man pages
5. Defines the user's default printer
6. Sets the user's default file creation permissions
7. Sets the listed environment variables

EXAMPLE 4–3 The `.cshrc` File

```
(Line 1) set path=($PATH $HOME/bin /usr/local/bin /usr/ccs/bin)
(Line 2) setenv MAIL /var/mail/$LOGNAME
(Line 3) setenv NNTPSERVER server1
(Line 4) setenv PRINTER printer1
(Line 5) alias h history
(Line 6) umask 022
(Line 7) source /net/server2/site-init-files/site.login
```

1. Defines the user's shell search path.
2. Defines the path to the user's mail file.
3. Defines the user's Usenet news server.
4. Defines the user's default printer.
5. Creates an alias for the `history` command. The user needs to type only `h` to run the `history` command.
6. Sets the user's default file creation permissions.
7. Sources the site initialization file.

EXAMPLE 4–4 Site Initialization File

The following shows an example site initialization file in which a user can choose a particular version of an application.

```
# @(#)site.login
main:
```

EXAMPLE 4–4 Site Initialization File *(Continued)*

```
echo "Application Environment Selection"
echo ""
echo "1. Application, Version 1"
echo "2. Application, Version 2"
echo ""
echo -n "Type 1 or 2 and press Return to set your
application environment: "

set choice = $<

if ( $choice !~ [1-2] ) then
goto main
endif

switch ($choice)

case "1":
setenv APPHOME /opt/app-v.1
breaksw

case "2":
setenv APPHOME /opt/app-v.2
endsw
```

This site initialization file could be referenced in a user's `.cshrc` file (C shell users only) with the following line:

```
source /net/server2/site-init-files/site.login
```

In this line, the site initialization file is named `site.login` and is located on a server named `server2`. This line also assumes that the automounter is running on the user's system.

CHAPTER 5

Managing User Accounts and Groups (Tasks)

This chapter describes how to set up and maintain user accounts and groups.

For information on the procedures associated with setting up and maintaining user accounts and groups, see the following:

- "Setting Up User Accounts (Task Map)" on page 113
- "Maintaining User Accounts (Task Map)" on page 123

For background information about managing user accounts and groups, see Chapter 4, "Managing User Accounts and Groups (Overview)."

Setting Up User Accounts (Task Map)

Task	Description	For Instructions
Gather user information.	Use a standard form to gather user information to help you keep user information organized.	"Gathering User Information" on page 114
Customize user initialization files.	You can set up user initialization files (`.cshrc`, `.profile`, `.login`), so that you can provide new users with consistent environments.	"How to Customize User Initialization Files" on page 115
Add a group.	You can add a group with the following tools: Solaris Management Console's Groups tool Solaris command-line interface tools	"How to Add a Group With the Solaris Management Console's Groups Tool" on page 116 "Adding Groups and Users With Command-Line Tools" on page 119

Task	Description	For Instructions
Add a user.	You can add a user with the following tools: Solaris Management Console's Users tool Solaris command-line interface tools	"How to Add a User With the Solaris Management Console's Users Tool" on page 117 "Adding Groups and Users With Command-Line Tools" on page 119
Set up a user template.	You can create a user template so that you don't have to manually add all similar user properties.	See Solaris Management Console online help
Add rights or a role to a user.	You can add rights or a role to a user so that the user can perform a specific command or task.	See Solaris Management Console online help
Share the user's home directory.	You must share the user's home directory so that the directory can be remotely mounted from the user's system.	"How to Share a User's Home Directory" on page 120
Mount the user's home directory.	You must mount the user's home directory on the user's system.	"How to Mount a User's Home Directory" on page 122

Gathering User Information

You can create a form such as the following to gather information about users before adding their accounts.

Item	Description
User Name:	
Role Name:	
Profiles or Authorizations:	
UID:	
Primary Group:	
Secondary Groups:	
Comment:	
Default Shell:	
Password Status and Aging:	

Item	Description
Home Directory Path Name:	
Mounting Method:	
Permissions on Home Directory:	
Mail Server:	
Department Name:	
Department Administrator:	
Manager:	
Employee Name:	
Employee Title:	
Employee Status:	
Employee Number:	
Start Date:	
Add to These Mail Aliases:	
Desktop System Name:	

▼ How to Customize User Initialization Files

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 **Create a skeleton directory for each type of user.**

```
# mkdir /shared-dir/skel/user-type
```

shared-dir	The name of a directory that is available to other systems on the network.
user-type	The name of a directory to store initialization files for a type of user.

3 **Copy the default user initialization files into the directories that you created for different types of users.**

```
# cp /etc/skel/local.cshrc /shared-dir/skel/user-type/.cshrc
# cp /etc/skel/local.login /shared-dir/skel/user-type/.login
# cp /etc/skel/local.profile /shared-dir/skel/user-type/.profile
```

Note – If the account has profiles assigned to it, then the user has to launch a special version of the shell called a profile shell to use commands (with any security attributes) that are assigned to the profile. There are three *profile shells* corresponding to the types of shells: `pfsh` (Bourne shell), `pfcsh` (C shell), and `pfksh` (Korn shell). For information about profile shells, see "Role-Based Access Control (Overview)" in *System Administration Guide: Security Services*.

---

4 **Edit the user initialization files for each user type and customize them based on your site's needs.**

For a detailed description on the ways to customize the user initialization files, see "Customizing a User's Work Environment" on page 102.

5 **Set the permissions for the user initialization files.**

```
# chmod 744 /shared-dir/skel/user-type/.*
```

6 **Verify that the permissions for the user initialization files are correct.**

```
# ls -la /shared-dir/skel/*
```

Example 5–1 Customizing User Initialization Files

The following example shows how to customize the C-shell user initialization file in the `/export/skel/enduser` directory designated for a particular type of user. For an example of a `.cshrc` file, see Example 4–3.

```
# mkdir /export/skel/enduser
# cp /etc/skel/local.cshrc /export/skel/enduser/.cshrc

(Edit .cshrc file)
# chmod 744 /export/skel/enduser/.*
```

▼ How to Add a Group With the Solaris Management Console's Groups Tool

You can add existing users to the group when you add the group. Or, you can just add the group and then add the user to the group when you add the user.

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 Start the Solaris Management Console.

```
# /usr/sadm/bin/smc &
```

For more information on starting the Solaris Management Console, see "How to Start the Console as Superuser or as a Role" on page 44 or "How to Start the Solaris Management Console in a Name Service Environment" on page 51.

3 Click the This Computer icon under the Management Tools icon in the Navigation pane.

A list of categories is displayed.

4 (Optional) Select the appropriate toolbox for your name service environment.

5 Click the System Configuration icon.

6 Click the User icon and provide the superuser password or the role password.

7 Click the Groups icon. Select Add Group from the Action menu.

Use the Context help to add a group to the system.

8 Identify the group name at the Group Name prompt under Group Identification.

For example, `mechanoids`.

9 Identify the group number at the Group ID number prompt.

For example, GID 101.

10 Click OK.

▼ How to Add a User With the Solaris Management Console's Users Tool

Use the following procedure to add a user with the Solaris Management Console's Users tool.

1 Become superuser or assume an equivalent role.

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 Start the Solaris Management Console.

```
# /usr/sadm/bin/smc &
```

For more information on starting the Solaris Management Console, see "How to Start the Console as Superuser or as a Role" on page 44 or "How to Start the Solaris Management Console in a Name Service Environment" on page 51.

3 **Click the This Computer icon under the Management Tools icon in the Navigation pane.**

A list of categories is displayed.

4 **(Optional) Select the appropriate toolbox for your name service environment.**

5 **Click the System Configuration icon.**

6 **Click the User icon and provide the superuser password or the role password.**

7 **Click the User Accounts icon.**

Use the Context help to add a user to the system.

8 **Select Add User⇒With Wizard from the Action menu.**

Click Next between the steps below.

a. **Identify the user name or login name at the User Name prompt.**

For example, `kryten`

b. **(Optional) Identify the user's full name at the Full Name prompt.**

For example, `kryten series 3000`.

c. **(Optional) Provide a further description of this user at the Description prompt.**

d. **Provide the user ID at the User ID Number prompt.**

For example, `1001`.

e. **Select the User Must Use This Password At First Login option.**

Provide a password for the user at the Password prompt and then confirm the password at the Confirm Password prompt.

f. **Select the user's primary group.**

For example, `mechanoids`.

g. **Create the user's home directory by accepting the defaults at the Server and Path prompts.**

h. **Specify the mail server.**

i. **Review the information you provided and go back to correct the information, if necessary. Otherwise, click Finish.**

Adding Groups and Users With Command-Line Tools

This section provides examples of adding users and groups with command-line tools.

Adding a Group and User With the `groupadd` and `useradd` Commands

The following example shows how to use the `groupadd` and `useradd` commands to add the group `scutters` and the user `scutter1` to files on the local system. These commands cannot be used to manage users in a name service environment.

```
# groupadd -g 102 scutters
# useradd -u 1003 -g 102 -d /export/home/scutter1 -s /bin/csh \
-c "Scutter 1" -m -k /etc/skel scutter1
64 blocks
```

For more information, see the `groupadd`(1M) and `useradd`(1M) man pages.

Adding a Group and User With the `smgroup` and `smuser` Commands

The following example shows how to use the `smgroup` and `smuser` commands to add the group `gelfs` and the user `camille` to the NIS domain `solar.com` on the host `starlite`.

```
# /usr/sadm/bin/smgroup add -D nis:/starlitesolar.com -- -g 103 -n gelfs
# /usr/sadm/bin/smuser add -D nis:/starlite/solar.com -- -u 1004
-n camille -c "Camille G." -d /export/home/camille -s /bin/csh -g gelfs
```

For more information, see the `smgroup`(1M) and `smuser`(1M) man pages.

Setting Up Home Directories With the Solaris Management Console

Keep the following in mind when using the Solaris Management Console tools to manage user home directories:

- If you use the Users tool's Add User Wizard to add a user account and you specify the user's home directory as `/export/home/`*username*, the home directory is automatically set up to automount. Also, the following entry is added to the `passwd` file.

  `/home/`*username*
- There is only way you can use Users tool to set up a user account that does not automount the home directory. First, set up a user account template that disables this feature. Then, add users with this template. You cannot disable this feature with the Add User Wizard.
- You can use the `smuser add` command with the `-x autohome=N` option to add a user without automounting the user's home directory. However, there is no option to the `smuser delete` command to remove the home directory after the user is added. You would have to remove the user and the user's home directory with the Users tool.

▼ How to Share a User's Home Directory

Use the following procedure to share a user's home directory.

1. **Become superuser or assume an equivalent role on the system that contains the home directory.**

2. **Verify that the `mountd` daemon is running.**

   In this release, `mountd` is now started as part of the NFS server service. To see if the `mountd` daemon is running, type the following command:

   ```
   # svcs network/nfs/server
   STATE          STIME    FMRI
   online         Aug_26   svc:/network/nfs/server:default
   ```

3. **If the `mountd` daemon is not running, start it.**

   ```
   # svcadm network/nfs/server
   ```

4. **List the file systems that are shared on the system.**

   ```
   # share
   ```

5 **Select one of the following based on whether the file system that contains the user's home directory is already shared.**

a. **If the user's home directory is already shared, go to the step 8.**

b. **If the user's home directory is not shared, go to Step 6.**

6 **Edit the `/etc/dfs/dfstab` file and add the following line:**

```
share -F nfs /file-system
```

/file-system is the file system that contains the user's home directory that you need to share. By convention, the file system is `/export/home`.

7 **Share the file systems listed in the `/etc/dfs/dfstab` file.**

```
# shareall -F nfs
```

This command executes all the `share` commands in the `/etc/dfs/dfstab` file so that you do not have to wait to reboot the system.

8 **Verify that a user's home directory is shared.**

```
# share
```

Example 5–2 Sharing a User's Home Directory

The following example shows how to share the `/export/home` directory.

```
# svcs network/nfs/server
# svcadm network/nfs/server
# share
# vi /etc/dfs/dfstab

(The line share -F nfs /export/home is added.)
# shareall -F nfs
# share
-               /usr/dist                ro   ""
-               /export/home/user-name     rw   ""
```

See Also If the user's home directory is not located on the user's system, you have to mount the user's home directory from the system where it is located. For detailed instructions, see "How to Mount a User's Home Directory" on page 122.

▼ How to Mount a User's Home Directory

For information on automounting a home directory, see "Task Overview for Autofs Administration" in *System Administration Guide: Network Services*.

1. **Make sure that the user's home directory is shared.**

   For more information, see "How to Share a User's Home Directory" on page 120.

2. **Log in as superuser on the user's system.**

3. **Edit the `/etc/vfstab` file and create an entry for the user's home directory.**

   ```
   system-name:/export/home/user-name - /export/home/username nfs - yes rw
   ```

system-name	The name of the system where the home directory is located.
`/export/home/`*username*	The name of the user's home directory that will be shared. By convention, `/export/home/`*username* contains user home directories. However, you can use a different file system.
-	Required placeholders in the entry.
`/export/home/`*username*	The name of the directory where the user's home directory will be mounted.

   For more information about adding an entry to the `/etc/vfstab` file, see "Mounting File Systems" in *System Administration Guide: Devices and File Systems*.

4. **Create the mount point for the user's home directory.**

   ```
   # mkdir -p /export/home/username
   ```

5. **Mount the user's home directory.**

   ```
   # mountall
   ```

   All entries in the current `vfstab` file (whose `mount at boot` fields are set to `yes`) are mounted.

6. **Verify that the home directory is mounted.**

   ```
   # mount | grep username
   ```

Example 5–3 Mounting a User's Home Directory

The following example shows how to mount user `ripley`'s home directory.

```
# vi /etc/vfstab

(The line venus:/export/home/ripley - /export/home/ripley
nfs - yes rw is added.)
```

```
# mkdir -p /export/home/ripley
# mountall
# mount
/ on /dev/dsk/c0t0d0s0 read/write/setuid/intr/largefiles/xattr/onerror=panic/dev=...
/devices on /devices read/write/setuid/dev=46c0000 on Thu Jan  8 09:38:19 2004
/usr on /dev/dsk/c0t0d0s6 read/write/setuid/intr/largefiles/xattr/onerror=panic/dev=...
/proc on /proc read/write/setuid/dev=4700000 on Thu Jan  8 09:38:27 2004
/etc/mnttab on mnttab read/write/setuid/dev=47c0000 on Thu Jan  8 09:38:27 2004
/dev/fd on fd read/write/setuid/dev=4800000 on Thu Jan  8 09:38:30 2004
/var/run on swap read/write/setuid/xattr/dev=1 on Thu Jan  8 09:38:30 2004
/tmp on swap read/write/setuid/xattr/dev=2 on Thu Jan  8 09:38:30 2004
/export/home on /dev/dsk/c0t0d0s7 read/write/setuid/intr/largefiles/xattr/onerror=...
/export/home/ripley on venus:/export/home/ripley remote/read/write/setuid/xattr/dev=...
```

Maintaining User Accounts (Task Map)

Task	Description	Instructions
Modify a group.	You can modify a group's name or the users in a group by using the Groups tool.	"How to Modify a Group" on page 125
Delete a group.	You can delete a group if it is no longer needed.	"How to Delete a Group" on page 125
Modify a user account.	*Disable a user account* You can temporarily disable a user account if it will be needed in the future. *Change a user's password* You might need to change a user's password if the user forgets it. *Set password aging* You can force users to change their passwords periodically with User Account tool's Password Options menu.	"How to Disable a User Account" on page 127 "How to Change a User's Password" on page 128 "How to Set Password Aging on a User Account" on page 129
Delete a user account.	You can delete a user account if it is no longer needed.	"How to Delete a User Account" on page 129

Modifying User Accounts

Unless you define a user name or UID number that conflicts with an existing one, you should never need to modify a user account's user name or UID number.

Use the following steps if two user accounts have duplicate user names or UID numbers:

- If two user accounts have duplicate UID numbers, use the Users tool to remove one account and add it again with a different UID number. You cannot use the Users tool to modify a UID number of an existing user account.
- If two user accounts have duplicate user names, use the Users tool to modify one of the accounts and change the user name.

If you do use the Users tool to change a user name, the home directory's ownership is changed, if a home directory exists for the user.

One part of a user account that you can change is a user's group memberships. Select the Properties option from Users tool's Action menu to add or delete a user's secondary groups. Alternatively, you can use the Groups tool to directly modify a group's member list.

You can also modify the following parts of a user account:

- Description (comment)
- Login shell
- Passwords and password options
- Home directory and home directory access
- Rights and roles

Disabling User Accounts

Occasionally, you might need to temporarily or permanently disable a user account. Disabling or locking a user account means that an invalid password, `*LK*`, is assigned to the user account, preventing future logins.

The easiest way to disable a user account is to lock the password for an account with Users tool.

You can also enter an expiration date in the account availability section of the User Properties screen. An expiration date enables you to set a limit on how long the account is active.

Other ways to disable a user account: set up password aging or change the user's password.

Deleting User Accounts

When you delete a user account with the Users tool, the software deletes the entries in the `passwd` and `group` files. In addition, the files in the user's home directory and mail directory are deleted also.

▼ How to Modify a Group

Use the following procedure to modify a group.

1. **Become superuser or assume an equivalent role.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2. **Start the Solaris Management Console.**

   ```
   # /usr/sadm/bin/smc &
   ```

   For more information on starting the Solaris Management Console, see "How to Start the Console as Superuser or as a Role" on page 44 or "How to Start the Solaris Management Console in a Name Service Environment" on page 51.

3. **Click the This Computer icon under the Management Tools icon in the Navigation pane.**

   A list of categories is displayed.

4. **(Optional) Select the appropriate toolbox for your name service environment.**

5. **Click the System Configuration icon.**

6. **Click the User icon.**

7. **Provide the superuser password or the role password.**

8. **Click the Groups icon.**

9. **Select the group to modify.**

   For example, select `scutters`.

10. **Modify the selected group in the Group Name: text box. Click OK when you are finished.**

    For example, change `scutters` to `scutter`.

    All the users that were in the `scutters` group are now in the `scutter` group.

▼ How to Delete a Group

Use the following procedure to delete a group.

1. **Become superuser or assume an equivalent role.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 **Start the Solaris Management Console.**

```
# /usr/sadm/bin/smc &
```

For more information on starting the Solaris Management Console, see "How to Start the Console as Superuser or as a Role" on page 44 or "How to Start the Solaris Management Console in a Name Service Environment" on page 51.

3 **Click the This Computer icon under the Management Tools icon in the Navigation pane.**

A list of categories is displayed.

4 **(Optional) Select the appropriate toolbox for your name service environment.**

5 **Click the System Configuration icon.**

6 **Click the User icon.**

7 **Provide the superuser password or the role password.**

8 **Click the Groups icon.**

9 **Select the group to delete.**

For example, select `scutter`.

10 **Click OK in the popup window.**

The group is removed from all the users who were a member of this group.

Administering Passwords

You can use the Users tool for password administration. This tool includes the following capabilities:

- Specifying a normal password for a user account
- Enabling users to create their own passwords during their first login
- Disabling or locking a user account
- Specifying expiration dates and password aging information

Note – Password aging is not supported by the NIS name service.

Using Password Aging

If you are using NIS+ or the `/etc` files to store user account information, you can set up password aging on a user's password. Starting in the Solaris 9 12/02 release, password aging is also supported in the LDAP directory service.

Password aging enables you to force users to change their passwords periodically or to prevent a user from changing a password before a specified interval. If you want to prevent an intruder from gaining undetected access to the system by using an old and inactive account, you can also set a password expiration date when the account becomes disabled. You can set password aging attributes with the `passwd` command or the Solaris Management Console's Users tool.

For information about starting the Solaris Management Console, see "How to Start the Console as Superuser or as a Role" on page 44.

▼ How to Disable a User Account

Use the following procedure if you need to disable a user account.

1. **Become superuser or assume an equivalent role.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2. **Start the Solaris Management Console.**

   ```
   # /usr/sadm/bin/smc &
   ```

   For more information on starting the Solaris Management Console, see "How to Start the Console as Superuser or as a Role" on page 44 or "How to Start the Solaris Management Console in a Name Service Environment" on page 51.

3. **Click the This Computer icon under the Management Tools icon in the Navigation pane.**

   A list of categories is displayed.

4. **(Optional) Select the appropriate toolbox for your name service environment.**

5. **Click the System Configuration icon.**

6. **Click the User icon and provide the superuser password or the role password.**

7. **Click the User Accounts icon.**

8. **Double–click the user.**

   For example, select `scutter2`.

9 **Select the Account is Locked option in the Account Availability section of the General tab features.**

10 **Click OK.**

▼ How to Change a User's Password

Use the following procedure when a user forgets her password.

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 **Start the Solaris Management Console.**

```
# /usr/sadm/bin/smc &
```

For more information on starting the Solaris Management Console, see "How to Start the Console as Superuser or as a Role" on page 44 or "How to Start the Solaris Management Console in a Name Service Environment" on page 51.

3 **Click the This Computer icon under the Management Tools icon in the Navigation pane.**

A list of categories is displayed.

4 **(Optional) Select the appropriate toolbox for your name service environment.**

5 **Click the System Configuration icon.**

6 **Click the User icon.**

7 **Provide the superuser password or the role password.**

8 **Click the User Accounts icon, then double–click the user who needs a new password.**

For example, select `scutter1`.

9 **Select the Password tab, then select the User Must Use This Password at Next Login option. .**

10 **Enter the user's new password and click OK.**

▼ How to Set Password Aging on a User Account

Use the following procedure to set password aging on a user account.

1. **Become superuser or assume an equivalent role.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2. **Start the Solaris Management Console.**

   ```
   # /usr/sadm/bin/smc &
   ```

   For more information on starting the Solaris Management Console, see "How to Start the Console as Superuser or as a Role" on page 44 or "How to Start the Solaris Management Console in a Name Service Environment" on page 51.

3. **Click the This Computer icon under the Management Tools icon in the Navigation pane.**

   A list of categories is displayed.

4. **(Optional) Select the appropriate toolbox for your name service environment.**

5. **Click the System Configuration icon.**

6. **Click the User Accounts icon and provide the superuser password or the role password.**

7. **Click the User Accounts icon.**

8. **Double–click the user, then select the Password Options tab.**

   For example, select `scutter2`.

9. **Select the Password Options tab.**

10. **Select the appropriate Password Options in Days option and click OK.**

    For example, select Users Must Change Within to set a date when the user must change his or her password.

▼ How to Delete a User Account

Use the following procedure to remove a user account.

1. **Become superuser or assume an equivalent role.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2. **Start the Solaris Management Console.**

   ```
   # /usr/sadm/bin/smc &
   ```

   For more information on starting the Solaris Management Console, see "How to Start the Console as Superuser or as a Role" on page 44 or "How to Start the Solaris Management Console in a Name Service Environment" on page 51.

3. **Click the This Computer icon under the Management Tools icon in the Navigation pane.**

   A list of categories is displayed.

4. **(Optional) Select the appropriate toolbox for your name service environment.**

5. **Click the System Configuration icon.**

6. **Click the User icon.**

7. **Provide the superuser password or the role password.**

8. **Click the User Accounts icon.**

9. **Double–click the user account to be removed.**

   For example, select `scutter4`.

10. **Click Delete in the popup window if you are sure you want to remove the user account.**

    You are prompted to remove the user's home directory and mailbox contents.

◆ ◆ ◆ CHAPTER 6

Managing Client-Server Support (Overview)

This chapter describes the management of server and client support on a network. Overview information is provided about each system configuration (referred to as a *system type*) that is supported in the Solaris Operating System. This chapter also includes guidelines for selecting the appropriate system type to meet your needs.

Note – Information in this chapter that pertains only to a specific Solaris release, or was introduced in a specific Solaris release, is labeled accordingly.

This is a list of the overview information in this chapter.

- "What's New in Managing Client-Server Support?" on page 131
- "Where to Find Client-Server Tasks" on page 134
- "What Are Servers, Clients, and Appliances?" on page 134
- "What Does Client Support Mean?" on page 135
- "Overview of System Types" on page 135
- "Diskless Client Management Overview" on page 139

For step-by-step instructions about how to manage diskless client support, see Chapter 7, "Managing Diskless Clients (Tasks)."

What's New in Managing Client-Server Support?

This section describes new or changed diskless client features in this Solaris release.

Support for Specifying Platform by Using bootadm -p Command

A new `-p` *platform* argument has been added to the `bootadm` command. This option enables you to specify the platform or machine hardware class of a client system in situations where the client platform differs from the server platform, for example when administering diskless clients.

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

`nfs4_domain` Keyword Impacts Diskless Client Boot

The `set_nfs4_domain` script that was delivered in the Solaris 10 OS is no longer used to set the NFSv4 domain. To set the NVSv4 domain, add the new `nfs4_domain` keyword to the diskless client's `sysidcfg` file. Note that if the `nfs4_domain` keyword exists in the `sysidcfg` file, the first boot of a diskless client sets the domain accordingly.

x86: Diskless Client Changes in the GRUB Boot Environment

An extension has been made to GRUB to enable `kernel$`, `module$`, and `$ISADIR` usage in the `menu.lst` file.

The `bootadm` command installs a default boot entry in the `menu.lst` file that is similar to the following:

```
kernel$ /platform/i86pc/kernel/$ISADIR/unix
module$ /platform/i86pc/kernel/$ISADIR/unix /platform/i86pc/kernel/$ISADIR/unix -B $ZFS-ROOTFS
module$ /platform/i86pc/$ISADIR/boot_archive
```

The `kernel$` and `module$` keywords are identical to the `kernel` and `module` commands that are used in the GRUB multiboot implementation. The `$ISADIR` keyword has the added capability to expand to `amd64` on 64-bit capable hardware. If the x86 based system is not 64-bit capable, the `$ISADIR` keyword is a null value (`""`). In this instance, the system boots the 32-bit kernel.

Note – These changes do not prevent you from booting of a newer Solaris kernel with an older implementation of GRUB. Nor do the changes prevent you from booting of an older Solaris kernel with a newer implementation of GRUB.

Note – GRUB based booting is not available on SPARC based systems.

The following feature enhancements are part of the new diskless boot scheme:

- The OS server is now capable of serving multiple Solaris releases simultaneously.

  With the new diskless boot scheme, you can perform a `pxegrub` based network boot , where multiple releases are presented to a client from the GRUB menu.
- Vendor-specific options are now specified in the boot archive.

  In previous releases, client-specific boot properties, typically defined in the `bootenv.rc` file, were provided by using vendor-specific options for the DHCP setup. The total length of the information that was required frequently exceeded the limit in the DHCP specification.

  With the boot new scheme, this information is part of the boot archive. The PXE/DHCP server is only required to provide the server IP address, the boot file, `pxegrub`, and possibly a client-specific menu file, through Site Option 150.

x86: Changes to the `smdiskless` Command

The `smdiskless` command is used to set up diskless clients. Previously, the `smdiskless` command set up the root (`/`) and `/usr` file systems, then exported these file systems to the client through NFS. To boot the client, you would additionally need to configure the `/tftpboot` area manually. This manual step is no longer a requirement for setting up a diskless client. The `smdiskless` command now automatically invokes a script in the `/usr/sadm/lib/wbem/config_tftp` file, which prepares the `/tftpboot` area for a PXE boot.

After running the `smdiskless` command, the `/tftpboot/01`*ethernet-address* file is displayed as a link to `pxegrub` and the `/tftpboot/menu.lst.01`*ethernet-address* file, which contains the GRUB menu entry. The *ethernet-address* in this instance is 01, followed by the Ethernet address of the client network interface. When supplying the Ethernet address of the client network interface, use uppercase letters and do not include colons.

The boot archive of the diskless client is automatically updated during shutdown. If the client's boot archive is out of date when it is shut down, you might need to run the following command from the OS server to update the boot archive:

```
# bootadm update-archive -f -R /export/root/host-name
```

where *host-name* is the host name of the client system.

For more information, see "x86: How to Boot the Failsafe Archive to Forcibly Update a Corrupt Boot Archive" on page 266 and the `bootadm(1M)` man page.

Note – This information applies to both SPARC based and x86 based OS servers that are serving x86 based clients.

For more information on setting up and configuring DHCP, see Chapter 14, "Configuring the DHCP Service (Tasks)," in *System Administration Guide: IP Services*.

For more information on how to manage diskless clients in the GRUB boot environment, see "Booting an x86 Based System by Using GRUB (Task Map)" on page 252.

Where to Find Client-Server Tasks

Use this table to find step-by-step instructions for setting up server and client support.

Client-Server Services	For More Information
Install or JumpStart clients	*Solaris Express Installation Guide: Network-Based Installations*
Diskless client systems in the Solaris OS	"Diskless Client Management Overview" on page 139 and Chapter 7, "Managing Diskless Clients (Tasks)"
Diskless client systems in the Solaris 7 OS and earlier Solaris releases	*Solstice AdminSuite 2.3 Administration Guide*

What Are Servers, Clients, and Appliances?

Systems on the network can usually be described as one of the system types in this table.

System Type	Description
Server	A system that provides services to other systems in its network. There are file servers, boot servers, web servers, database servers, license servers, print servers, installation servers, appliance servers, and even servers for particular applications. This chapter uses the term *server* to mean a system that provides boot services and file systems for other systems on the network.

System Type	Description
Client	A system that uses remote services from a server. Some clients have limited disk storage capacity, or perhaps none at all. Such clients must rely on remote file systems from a server to function. Diskless systems and appliance systems are examples of this type of client. Other clients might use remote services (such as installation software) from a server. However, they don't rely on a server to function. A stand-alone system is a good example of this type of client. A stand-alone system has its own hard disk that contains the root (`/`), `/usr`, and `/export/home` file systems and swap space.
Appliance	A network appliance such as the Sun Ray™ appliance provides access to applications and the Solaris OS. An appliance gives you centralized server administration, and no client administration or upgrades. Sun Ray appliances also provide *hot desking*. Hot desking enables you to instantly access your computing session from any appliance in the server group, exactly where you left off. For more information, see http://www.sun.com/products/sunray.

What Does Client Support Mean?

Support can include the following:

- Making a system known to the network (host name and Ethernet address information)
- Providing installation services to remotely boot and install a system
- Providing Solaris OS services and application services to a system with limited disk space or no disk space

Overview of System Types

System types are sometimes defined by how they access the root (`/`) and `/usr` file systems, including the swap area. For example, stand-alone systems and server systems mount these file systems from a local disk. Other clients mount the file systems remotely, relying on servers to provide these services. This table lists some of the characteristics of each system type.

TABLE 6–1 Characteristics of System Types

System Type	Local File Systems	Local Swap Space?	Remote File Systems	Network Use	Relative Performance
Server	root (`/`)	Available	Not available	High	High
	`/usr`				
	`/home`				
	`/opt`				
	`/export/home`				
Stand-alone system	root (`/`)	Available	Not available	Low	High
	`/usr`				
	`/export/home`				
OS Server	`/export/root`				
Diskless client	Not available	Not available	root (`/`)	High	Low
			swap	High	Low
			`/usr`		
			`/home`		
Appliance	Not available	Not available	Not available	High	High

Description of a Server

A server system contains the following file systems:

- The `root` (`/`) and `/usr` file systems, plus swap space
- The `/export` and `/export/home` file systems, which support client systems and provide home directories for users
- The `/opt` directory or file system for storing application software

Servers can also contain the following software to support other systems:

- Solaris OS services for diskless systems that are running a different release

---

Note – OS client&hyphen;server configurations, where only one system is running a Solaris release that implements the new boot architecture can result in major incompatibilities. It is therefore recommended that you install or upgrade diskless systems to the same release as the server OS before adding diskless client support. New boot (GRUB) was introduced in the Solaris 10 1/06 release on the x86 platform and in the Solaris 10 10/08 release on the SPARC platform.

- Clients that use a different platform than the server
- Solaris CD image software and boot software for networked systems to perform remote installations
- JumpStart™ directory for networked systems to perform custom JumpStart installations

Stand-Alone Systems

A *networked stand-alone system* can share information with other systems in the network. However, it can continue to function if detached from the network.

A stand-alone system can function autonomously because it has its own hard disk that contains the root (`/`), `/usr`, and `/export/home` file systems and swap space. Thus, the stand-alone system has local access to OS software, executables, virtual memory space, and user-created files.

Note – A stand-alone system requires sufficient disk space to hold its necessary file systems.

A *non-networked stand-alone system* is a stand-alone system with all the characteristics just listed, except it is not connected to a network.

Diskless Clients

A *diskless client* has no disk and depends on a server for all its software and storage needs. A diskless client remotely mounts its `root` (`/`), `/usr`, and `/home` file systems from a server.

A diskless client generates significant network traffic due to its continual need to procure OS software and virtual memory space from across the network. A diskless client cannot operate if it is detached from the network or if its server malfunctions.

For more overview information about diskless clients, see "Diskless Client Management Overview" on page 139.

Description of an Appliance

An appliance, such as the Sun Ray appliance, is an X display device that requires no administration. There is no CPU, fan, disk, and very little memory. An appliance is connected to a Sun display monitor. However, the appliance user's desktop session is run on a server and displayed back to the user.

The X environment is set up automatically for the user and has the following characteristics:

- Relies on a server to access other file systems and software applications
- Provides centralized software administration and resource sharing
- Contains no permanent data, making it a field-replaceable unit (FRU)

Guidelines for Choosing System Types

You can determine which system types are appropriate for your environment by comparing each system type based on the following characteristics:

Centralized administration

- Can the system be treated as a field-replaceable unit (FRU)?

  This means that a broken system can be quickly replaced with a new system without any lengthy backup and restore operations and no loss of system data.
- Does the system need to be backed up?

  Large costs in terms of time and resources can be associated with backing up a large number of desktop systems.
- Can the system's data be modified from a central server?
- Can the system be installed quickly and easily from a centralized server without handling the client system's hardware?

Performance

- Does this configuration perform well in desktop usage?
- Does the addition of systems on a network affect the performance of other systems already on the network?

Disk space usage

How much disk space is required to effectively deploy this configuration?

This table describes how each system type scores in terms of each characteristic. A ranking of 1 is most efficient. A ranking of 4 is least efficient.

TABLE 6–2 Comparison of System Types

System Type	Centralized Administration	Performance	Disk Space Usage
Stand-alone system	4	1	4
Diskless client	1	4	1
Appliance	1	1	1

Diskless Client Management Overview

The following sections and Chapter 7, "Managing Diskless Clients (Tasks)," describe how to manage diskless client support in the Solaris Operating System (Solaris OS).

A *diskless client* is a system that depends on an *OS server* for its operating system, software, and storage. A diskless client mounts its root (`/`), `/usr`, and other file systems from its OS server. A diskless client has its own CPU and physical memory and can process data locally. However, a diskless client cannot operate if it is detached from its network or if its OS server malfunctions. A diskless client generates significant network traffic because of its continual need to function across the network.

Starting with the Solaris 9 release, the diskless client commands, `smosservice` and `smdiskless`, enable you to manage OS services and diskless client support. In the Solaris 8 release, diskless clients were managed with the Solstice™ GUI management tools.

OS Server and Diskless Client Support Information

Caution – Attempts to add diskless client support using an OS client-server configuration where one system implements the new boot architecture, but the other system does not, can result in major incompatibilities. New boot (GRUB) was implemented on the x86 platform, starting with the Solaris 10 1/06 release and on the SPARC platform, starting with the Solaris 10 10/8 release. Note that adding diskless support on systems that are running a Solaris release that is more recent than that which is running on the OS server is also an unsupported configuration. To avoid potential problems, it is recommended that you install the latest Solaris release before adding diskless client support.

The Solaris releases and architecture types that are supported by the `smosservice` and `smdiskless` commands include the following:

- **SPARC based servers:** Supported in the Solaris 8, Solaris 9, Solaris 10, and Solaris Express releases
- **SPARC based clients:** Supported in the Solaris 8, Solaris 9, Solaris 10, and Solaris Express releases
- **x86 based servers:** Supported in the Solaris 9, Solaris 10, and Solaris Express releases
- **x86 based clients:** Supported in the Solaris 9, Solaris 10, and Solaris Express releases

The following table illustrates the x86 OS client-server configurations that are supported by the `smosservice` and `smdiskless` commands. This information applies to the Solaris 9 and the Solaris 10 FCS (3/05) release.

If you are running at least the Solaris 10 1/06 release, or a Solaris Express Community release, it is recommended that you install or upgrade to the same release before adding diskless client support.

TABLE 6–3 x86 OS Client-Server Support

	Diskless Client OS	
Server OS	**Solaris 10 3/05**	**Solaris 9**
Solaris 10 3/05	Supported	Supported
Solaris 9	Not supported	Supported

The following table describes the SPARC OS client-server configurations that are supported by the `smosservice` and `smdiskless` commands. This information applies to the Solaris 8 and Solaris 9 releases, and the Solaris 10 release, up through the Solaris 10 5/08 OS.

If you are running at least the Solaris 10 10/08 release, or a Solaris Express Community release, it is recommended that you install or upgrade to the same release before adding diskless client support.

TABLE 6–4 SPARC OS Client-Server Support

	Diskless Client OS		
Server OS	**Solaris 10 3/05 through Solaris 10 5/08**	**Solaris 9**	**Solaris 8**
Solaris 10 3/05 through Solaris 10 5/08	Supported	Supported	Supported
Solaris 9	Not supported	Supported	Supported

TABLE 6–4 SPARC OS Client-Server Support *(Continued)*

Solaris 8	Not supported	Not supported	Supported

Diskless Client Management Features

You can use the `smosservice` and `smdiskless` commands to add and maintain diskless client support on a network. By using a name service, you can manage system information in a centralized manner so that important system information, such as host names, do not have to be duplicated for every system on the network.

You can perform the following tasks with the `smosservice` and `smdiskless` commands:

- Add and modify diskless client support
- Add and remove OS services
- Manage diskless client information in the LDAP, NIS, NIS+, or files name service environment

If you are performing a GRUB based boot on an x86 system, you need to manually set up the DHCP configuration. See "x86: How to Prepare for Adding Diskless Clients in a GRUB Based Boot Environment" on page 148 for more information.

Note – You can only use the diskless client commands to set up diskless client booting. You cannot use these commands to set up other services, such as remote installation or profile services. Set up remote installation services by including diskless client specifications in the `sysidcfg` file. For more information, see *Solaris Express Installation Guide: Custom JumpStart and Advanced Installations*.

Working With Diskless Client Commands

By writing your own shell scripts and using the commands shown in the following table, you can easily set up and manage your diskless client environment.

TABLE 6–5 Diskless Client Commands

Command	Subcommand	Task
`/usr/sadm/bin/smosservice`	`add`	Add OS services
	`delete`	Delete OS services
	`list`	List OS services
	`patch`	Manage OS service patches

TABLE 6–5 Diskless Client Commands *(Continued)*

Command	Subcommand	Task
`/usr/sadm/bin/smdiskless`	`add`	Add a diskless client to an OS server
	`delete`	Delete a diskless client from an OS server
	`list`	List the diskless clients on an OS server
	`modify`	Modify the attributes of a diskless client

You can obtain help on these commands in two ways:

- Use the `-h` option when you type the command, subcommand, and required options, as shown in the following example.

  ```
  % /usr/sadm/bin/smdiskless add -p my-password -u my-user-name -- -h
  ```

- View the `smdiskless`(1M) and `smosservice`(1M) man pages.

Required RBAC Rights for Diskless Client Management

You can use the `smosservice` and `smdiskless` commands as superuser. If you are using role-based access control (RBAC), you can use either a subset of or all of the diskless client commands, according to the RBAC rights to which they are assigned. The following table lists the RBAC rights that are required to use the diskless client commands.

TABLE 6–6 Required RBAC Rights for Diskless Client Management

RBAC Right	Command	Task
Basic Solaris User, Network Management	`smosservice list`	List OS services
	`smosservice patch`	List OS service patches
	`smdiskless list`	List diskless clients on an OS server
Network Management	`smdiskless add`	Add diskless clients
System Administrator	All commands	All tasks

Adding OS Services

A Solaris OS server is a server that provides operating system (OS) services to support diskless client systems. You can add support for an OS server or convert a stand-alone system to an OS server by using the `smosservice` command.

For each platform group and Solaris release that you want to support, you must add the particular OS service to the OS server. For example, if you want to support SPARC sun-4u systems running the Solaris Express release, you must add the Sun-4u/Solaris Express OS services to the OS server. For each diskless client that you support, you must add the OS service for that client to the OS server. For example, you would need to add OS services to support SPARC sun-4m systems or x86 based systems that run the Solaris 10 or Solaris 9 release, because they are different platform groups.

You must have access to the appropriate Solaris software CD or disk image to add OS services.

Adding OS Services When the OS Server Has Been Patched

When adding OS services to an OS server, you might see an error message stating that you have inconsistent versions of the OS running on the server and the OS that you are trying to add. This error message occurs when the installed version of the OS has packages that were previously patched, and the OS services being added do not have those packages patched, because the patches have been integrated into the packages.

For example, you might have a server that is running the current Solaris release or the Solaris 10 release. You might also have additional OS services loaded on this server, including the Solaris 9 SPARC sun-4m OS services that have been patched. If you try to add the Solaris 8 SPARC sun-4u OS services from a CD-ROM to this server, you could get the following error message:

```
Error: inconsistent revision, installed package appears to have been
patched resulting in it being different than the package on your media.
You will need to backout all patches that patch this package before
retrying the add OS service option.
```

Disk Space Requirements for OS Servers

Before you set up your diskless client environment, ensure that you have the required disk space available for each diskless client directory.

In previous Solaris releases, you were prompted about diskless client support during the installation process. Starting with the Solaris 9 release, you must manually allocate an `/export` file system either during installation or create it after installation. See the following table for specific disk space requirements.

TABLE 6–7 Disk Space Recommendations for Solaris OS Servers and Diskless Clients

Server OS/Architecture Type	Directory	Required Disk Space
Solaris Express and Solaris 10 SPARC based OS server	`/export`	5 to 6.8 Gbytes

TABLE 6–7 Disk Space Recommendations for Solaris OS Servers and Diskless Clients *(Continued)*

Server OS/Architecture Type	Directory	Required Disk Space
Solaris Express and Solaris 10 x86 based OS server	`/export`	5 to 6.8 Gbytes
Solaris Express and Solaris 10 SPARC based diskless client	`/export`	Reserve 200 to 300 Mbytes per diskless client.
Solaris Express and Solaris 10 x86 based diskless client	`/export`	Reserve 200 to 300 Mbytes per diskless client.

Note – Disk space recommendations can vary, depending on the Solaris release that is installed. For specific information about the disk space recommendations in the current Solaris release, see "Disk Space Recommendations for Software Groups" in *Solaris Express Installation Guide: Planning for Installation and Upgrade*.

CHAPTER 7

Managing Diskless Clients (Tasks)

This chapter describes how to manage diskless clients in the Solaris Operating System (Solaris OS).

For information on the procedures that are associated with managing diskless clients, see "Managing Diskless Clients (Task Map)" on page 145. For information about installation problems that are associated with managing diskless clients, see "Troubleshooting Diskless Client Installation Problems" on page 166. For overview information on managing diskless clients, see Chapter 6, "Managing Client-Server Support (Overview)."

Managing Diskless Clients (Task Map)

The following table identifies the procedures that are required to manage diskless clients.

Task	Description	For Instructions
1. (Optional) Enable Solaris Management Console logging to view diskless client error messages.	Choose Log Viewer from the console main window to view diskless client error messages.	"Starting the Solaris Management Console" on page 44
2. Prepare for adding a diskless client.	Verify supported releases and identify the platform, media path, and cluster (or software group) of each diskless client.	"x86: How to Prepare for Adding Diskless Clients in a GRUB Based Boot Environment" on page 148 "How to Prepare for Adding Diskless Clients in the Solaris 10 OS" on page 151

Task	Description	For Instructions
3. Add required OS services to an OS server.	Add the OS services for the diskless clients you want to support by using the `smosservice` command. You must identify the platform, media path, and each diskless client platform that you want to support.	"How to Add OS Services for Diskless Client Support" on page 152
4. Locate and install any `ARCH=all` packages that were missed when you added OS services to the server. **Note –** To avoid having to add these packages to each diskless client individually, perform this task *prior* to adding diskless client support .	The `smosservice add` command does not install any root (`/`) or `/usr` packages that are designated `ARCH=all`. These packages must be installed manually after adding the OS services to the OS server. This behavior has existed since the Solaris 2.1 OS. The behavior applies to both SPARC based and x86 based platforms. Missing `ARCH=all` packages vary, depending on which Solaris OS you are running.	"How to Locate and Install Missing `ARCH=all` Packages" on page 166
5. Add a diskless client.	Add diskless client support by specifying all required information by using the `smdiskless` command.	"x86: How to Add a Diskless Client in the GRUB Based Boot Environment" on page 155 "How to Add a Diskless Client in the Solaris 10 OS" on page 158
6. Boot the diskless client.	Verify that a diskless client was successfully added by booting the diskless client.	"x86: How to Boot a Diskless Client With GRUB" on page 160 "SPARC: How to Boot a Diskless Client in the Solaris 10 OS" on page 161
7. (Optional) Delete diskless client support.	Delete support for a diskless client if it is no longer required.	"How to Remove Diskless Client Support" on page 161
8. (Optional) Delete OS services for a diskless client.	Delete OS services for a diskless client if they are no longer needed.	"How to Remove OS Services for Diskless Clients" on page 162
9. (Optional) Patch OS services.	Add, delete, list, or synchronize patches for diskless client OS services.	"How to Add an OS Patch for a Diskless Client" on page 164

Preparing for Managing Diskless Clients

These sections describe the preparations that are necessary for managing diskless clients.

Keep the following key points in mind when managing diskless clients:

- The Solaris installation program doesn't prompt you to set up diskless client support. You must manually create an `/export` partition to support diskless clients. You create the `/export` partition during or after the installation process.
- The `/export` partition must contain a minimum of 5 Gbytes, depending upon the number of clients supported. For specific information, see .
- The name service identified in the `smosservice` or `smdiskless` commands must match the primary name service identified in the `/etc/nsswitch.conf` file. If you don't specify a name service in the `smdiskless` or `smosservice` commands, the default name service is `files`.

  Use the `-D` option to the `smosservice` and `smdiskless` commands to specify a name server. For more information, see the `smosservice`(1M) and `smdiskless`(1M) man pages.

  Starting with the Solaris 10 8/07 release, the `set_nfs4_domain` script that was delivered in the Solaris 10 OS is no longer used to set the NFSv4 domain. To set the NFSv4 domain, add the `nfs4_domain` keyword to the diskless client's `sysidcfg` file, for example, `server:/export/root/client/etc/sysidcfg`.

  If the `nfs4_domain` keyword exists in the client system's `sysidcfg` file, the first boot of a diskless client sets the domain accordingly. Also, the OS server should be up and running, and the diskless client's NFSv4 domain setting must match the setting in the OS server's `/var/run/nfs4_domain` file.

  For more information, see "Preconfiguring With the sysidcfg File" in *Solaris Express Installation Guide: Network-Based Installations*.
- The OS server and the diskless client must be on the same subnet.
- You cannot provide client services on a multiterabyte UFS file system, because OS and diskless client services *cannot* be added to a UFS file system that resides on an EFI-labeled disk.

Note – Attempts to add OS and diskless client services to a UFS file system that resides on an EFI-labeled disk result in an erroneous insufficient disk space message similar to the following:

```
The partition /export does not have enough free space.
1897816 KB (1853.34 MB) additional free space required.
Insufficient space available on
/dev/dsk/c0t5d0s0 /export
```

After you determine the platform, media path, and cluster for each diskless client, you are ready to add OS services.

The following directories are created and populated for each OS service that you add:

- `/export/Solaris_`*version*`/Solaris_`*version*`-`*instruction-set*`.all` (symbolic link to `/export/exec/Solaris_`*version*`/Solaris_`*version*`-`*instruction-set*`.all`)
- `/export/Solaris_`*version*
- `/export/Solaris_`*version*`/var`
- `/export/Solaris_`*version*`/opt`
- `/export/share`
- `/export/root/templates/Solaris_`*version*
- `/export/root/clone`
- `/export/root/clone/Solaris_`*version*
- `/export/root/clone/Solaris_`*version*`/`*machine-class*

The following default directories are created and populated on the OS server for each diskless client that you add:

- `/export/root/`*diskless-client*
- `/export/swap/`*diskless-client*
- `/tftpboot/`*diskless-client-ipaddress-in-hex*`/export/dump/`*diskless-client* (if you specify the `-x` *dump* option)

Note – You can modify the default locations of the root (`/`), `/swap`, and `/dump` directories by using the `-x` option to the `smosservice` and `smdiskless` commands. However, do not create these directories under the `/export` file system.

▼ x86: How to Prepare for Adding Diskless Clients in a GRUB Based Boot Environment

Use this procedure to prepare for adding a diskless client. This procedure includes general information for x86 based systems.

When you use the `smosservice add` command to add OS services, you must specify the platform, media path, and cluster (or software group) of each diskless client platform that you want to support.

Before You Begin Ensure that the system that is intended to be the OS service is running a supported release. Also, verify that the OS server release and diskless client release combination is supported. For more information, see "OS Server and Diskless Client Support Information" on page 139.

1 **Identify the diskless client platform by using this format:**

instruction-set.machine-class.Solaris-version

For example:

```
i386.i86pc.Solaris_11
```

The following are the possible platform options:

InstructionSet	MachineClass	SolarisVersion
sparc	sun4v	Starting with the Solaris 10 1/06 OS
	sun4u, sun4m, sun4d, and sun4c	Solaris Express, Solaris 10, Solaris 9, and Solaris 8
i386	i86pc	Solaris Express, Solaris 10, Solaris 9, and Solaris 8

Note – The sun-4c architecture is not supported in the Solaris 8, Solaris 9, Solaris 10, or Solaris Express releases. The sun-4d architecture is not supported in the Solaris 9, Solaris 10, or Solaris Express releases. The sun-4m architecture is not supported in the Solaris Express Developer Editionor the Solaris 10 release.

---

2 **Identify the media path.**

The media path is the full path to the disk image that contains the OS that you want to install for the diskless client.

The Solaris OS is delivered on multiple CDs. However, you cannot use the `smosservice` command to load OS services from a multiple CD distribution. You must run the scripts that are found on the Solaris software CDs (and optional Language CD) to do the following:

3 **Create an install image on a server. For information on setting up an install server, refer to** *Solaris Express Installation Guide: Network-Based Installations*.

4 **Load the required OS services from the CD image.**

Use one of the following scripts:

- CD 1 – `/media/Solaris_11/Tools/setup_install_server`
- Additional Solaris Software CDs – `/media/Solaris_11/Tools/add_to_install_server`
- Language CD – `/media/Solaris_11/Tools/add_to_install_server`

For example, if you are using the `setup_install_server` script from the Solaris Express Software 1 CD on a locally connected CD-ROM device, the syntax looks similar to the following:

```
# mkdir /export/install/sol_11_x86
# cd /cd_mount_point/Solaris_11/Tools
# ./setup_install_server /export/install/sol_11_x86
```

5 **Add the `BootFile` and `BootSrvA` DHCP options to your DHCP server configuration to enable a PXE boot.**

For example:

```
Boot server IP (BootSrvA) : svr-addr
(BootFile) : 01client-macro
```

where *svr-addr* is the IP address of the OS server and *client-macro* is named by the client's Ethernet type (01) and the mac address of the client. This number is also the name of the file that is used in the `/tftpboot` directory on the installation server.

Note – The notation for the *client-macro* consists of uppercase letters. This notation should not contain any colons.

You can add these options from the command-line, or by using DHCP Manager. See Example 7–4 for more information.

For more information, see "x86: How to Perform a GRUB Based Boot From the Network" on page 278, "Preconfiguring System Configuration Information With the DHCP Service (Tasks)" in *Solaris Express Installation Guide: Network-Based Installations*, and Part II, "DHCP," in *System Administration Guide: IP Services*.

6 **After the Solaris CD image is installed on the disk, note the disk media path. For example:**

```
/net/export/install/sol_11_x86_
```

This is the disk media path that needs to be specified when you use the `smosservice` command.

7 **Identify the `SUNWCXall` cluster when you add OS services.**

You must use the same cluster for diskless clients that run the same OS on the same system.

Note – Always specify `SUNWCXall` as the cluster.

▼ How to Prepare for Adding Diskless Clients in the Solaris 10 OS

When you use the `smosservice add` command to add OS services, you must specify the platform, media path, and cluster (or software group) of each diskless client platform that you want to support.

Before You Begin Ensure that the system that is intended to be the OS service is running a supported release. Also verify that the combination of OS server release and diskless client release is supported. For more information, see "OS Server and Diskless Client Support Information" on page 139.

1 **Identify the diskless client platform by using this format:**

instruction-set.*machine-class*.Solaris-*version*

For example:

```
sparc.sun4u.Solaris_10
```

The following are the possible platform options:

instruction-set	*machine-class*	**Solaris**_*version*
sparc	sun4v	Starting with the Solaris 10 1/06 OS
	sun4c, sun4d, sun4m, sun4u,	Solaris_10, Solaris_9, and Solaris_8
i386	i86pc	Solaris_10, Solaris_9, and Solaris_8

Note – The sun-4c architecture is not supported in the Solaris 8, Solaris 9, or Solaris 10 releases. The sun-4d architecture is not supported in the Solaris 9 or 10 releases. The sun-4m architecture is not supported in the Solaris 10 release.

2 **Identify the media path.**

The media path is the full path to the disk image that contains the OS that you want to install for the diskless client.

The Solaris OS is delivered on multiple CDs. However, you cannot use the `smosservice` command to load OS services from a multiple CD distribution. You must run the scripts that are found on the Solaris software CDs (and optional Language CD) to do the following:

3 **Create an install image on a server. For information on setting up an install server, refer to *Solaris Express Installation Guide: Network-Based Installations*.**

4 **Load the required OS services from the CD image.**

Use one of the following scripts:

- CD 1 – `/media/Solaris_11/Tools/setup_install_server`
- Additional Solaris Software CDs – `/media/Solaris_11/Tools/add_to_install_server`
- Language CD – `/media/Solaris_11/Tools/add_to_install_server`

For example, if you are using the `setup_install_server` script from the Solaris Express Community Edition Software 1 CD on a locally connected CD-ROM device, the syntax looks similar to the following:

```
# mkdir /export/install/sparc_11
# cd /cd_mount_point/Solaris_11/Tools
# ./setup_install_server /export/install/sparc_11
```

5. **After the Solaris CD image is installed on the disk, specify the disk media path. For example:**

   `/export/install/sparc_11`

6. **Identify the `SUNWCXall` cluster when you add OS services.**

   You must use the same cluster for diskless clients that run the same OS on the same system.

   For example, consider the following Solaris 9 diskless clients:

   - `sparc.sun4m.Solaris_9`
   - `sparc.sun4u.Solaris_9`

   To set up these diskless clients, you would need to specify the `SUNWCXall` cluster for each diskless client because the `sun4u` and `sun4m` systems require the `SUNWCXall` cluster. In addition, diskless clients that run the same operating release (in this example, Solaris_9) on the same system must use the same cluster.

Note – If you are using a `sun4u` system, or if you are using a system with an accelerated 8-bit color memory frame buffer (`cgsix`), you *must* specify `SUNWCXall` as the cluster.

▼ How to Add OS Services for Diskless Client Support

Use this procedure to add OS services for a diskless client on the server.

Note – When adding OS services with the `smosservice add` command, root (`/`) and `/usr` packages with the `ARCH=all` type are not installed. These packages are skipped. No warning or error messages are displayed. After you add the OS services to the OS server, you must install the missing packages manually. For instructions, see How to Locate and Install Missing `ARCH=all` Packages.

---

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 **Verify that the Solaris Management Console server is running and that the diskless client tools are available on the system.**

```
# /usr/sadm/bin/smosservice list -H host-name:898 --
```

3 **Add the OS services.**

```
# /usr/sadm/bin/smosservice add -H host-name:898 -- -o host-name
-x mediapath=path -x platform=instruction-set.machine-class.Solaris_version
-x cluster=cluster-name -x locale=locale-name
```

`add`
Adds the specified OS service.

`-H` *host-name*:898
Specifies the host name and port to which you want to connect. If you do not specify a port, the system connects to the default port, 898.

Note – The `-H` option is not a required option when using the `smossservice` command to add OS services.

`--`
Identifies that the subcommand arguments start after this point.

`-x` *mediapath=path*
Specifies the full path to the Solaris image.

`-x` *platform=instruction-set.machine-class.*.Solaris_*version*
Specifies the instruction architecture, machine class, and the Solaris version to be added.

`-x` *cluster=cluster-name*
Specifies the Solaris cluster to install.

`-x` *locale=locale-name*
Specifies the locale to install.

Note – The installation process can take about 45 minutes, depending on the server speed and the OS service configuration you choose.

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

4 **(Optional) Continue to add the other OS services.**

5 **When you are finished adding OS services, verify that the OS services were installed.**

```
# /usr/sadm/bin/smosservice list -H host-name:898 --
```

Example 7–1 SPARC: Adding an OS Service for Diskless Client Support

This example shows how to add Solaris 10 SPARC based OS services on the server `jupiter`. The server `jupiter` is running the Solaris 10 release. The CD image of the Solaris 10 SPARC based OS is located on the installation server, myway, in `/export/s10/combined.s10s_u2wos/61`.

```
# /usr/sadm/bin/smosservice add -H jupiter:898 -- -o jupiter
-x mediapath=/net/myway/export/s10/combined.s10s_u2wos/61
-x platform=sparc.sun4u.Solaris_10
-x cluster=SUNWCXall -x locale=en_US

# /usr/sadm/bin/smosservice list - H jupiter:898
Authenticating as user: root

Type /? for help, pressing enter accepts the default denoted by [ ]
Please enter a string value for: password :: xxxxxx
Loading Tool: com.sun.admin.osservermgr.cli.OsServerMgrCli
from jupiter:898
Login to jupiter as user root was successful.
Download of com.sun.admin.osservermgr.cli.OsServerMgrCli from jupiter:898
was successful.
```

Example 7–2 x86: Adding an OS Service for Diskless Client Support

This example shows how to add Solaris 10 x86 based OS services on the server `orbit`. The server `orbit` is running the Solaris 10 release. The CD image of the Solaris 10 x86 based OS is located on the installation server, `seriously`, in `/export/s10/combined.s10x_u2wos/03`.

```
# /usr/sadm/bin/smosservice add -H orbit:898 -- -o orbit -x
mediapath=/net/seriously/export/s10u2/combined.s10x_u2wos/03 -x
platform=i386.i86pc.Solaris_10 -x cluster=SUNWCXall -x locale=en_US

# /usr/sadm/bin/smosservice list - H orbit:898
Type /? for help, pressing <enter> accepts the default denoted by [ ]
Please enter a string value for: password ::
Starting Solaris Management Console server version 2.1.0.
endpoint created: :898
Solaris Management Console server is ready.
Loading Tool: com.sun.admin.osservermgr.cli.OsServerMgrCli from orbit:898
Login to orbit as user root was successful.
Download of com.sun.admin.osservermgr.cli.OsServerMgrCli from orbit:898 was successful.
Client           Root Area
                Swap Area
                Dump Area
--------------------------------------------------------------------------------
.
.
.
#
```

Next Steps Locate and install any `ARCH=all` packages that were missed when you ran the `smosservice add` command to add the OS services to the OS server. For more information, see How to Locate and Install Missing `ARCH=all` Packages.

x86: How to Add a Diskless Client in the GRUB Based Boot Environment

Starting with the Solaris 10 1/06 release, use this procedure to add a diskless client after you have added OS services.

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 **Add the diskless client.**

```
# /usr/sadm/bin/smdiskless add -- -i ip-address -e ethernet-address
-n client-name -x os=instruction-set.machine-class.Solaris_version
-x root=/export/root/client-name -x swap=/export/swap/client-name
-x swapsize=size -x tz=time-zone -x locale=locale-name
```

`add`
Adds the specified diskless client.

`--`
Identifies that the subcommand arguments start after this point.

`-i` *ip-address*
Identifies the IP address of the diskless client.

`-e` *ethernet-address*
Identifies the Ethernet address of the diskless client.

`-n` *client-name*
Specifies the name of the diskless client.

`-x os=`*instruction-set.machine-class*.Solaris_*version*
Specifies the instruction architecture, machine class, OS, and the Solaris version for the diskless client.

`-x root=root=/export/root/`*client-name*
: Identifies the root (`/`) directory for the diskless client.

`-x swap=root=/export/root/`*client-name*
: Identifies the swap file for the diskless client.

`-x swapsize=`*size*
: Specifies the size of the swap file in Mbytes. The default is 24 Mbytes.

`-x tz=`*time-zone*
: Specifies the time-zone for the diskless client.

`-x locale=`*locale-name*
: Specifies the locale to install for the diskless client.

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

3 **If not already created, add the `BootSrva` and `BootFile` DHCP options to your DHCP server configuration to enable a PXE boot.**

For example:

```
Boot server IP (BootSrvA) : svr-addr
Boot file (BootFile) : 01client-macro
```

where *svr-addr* is the IP address of the server and *client-macro* is named by the client's Ethernet type (01) and the mac address of the client. This number is also the name of the file that is used in the `/tftpboot` directory on the installation server.

Note – The *client-macro* notation consists of uppercase letters. The notation should not contain any colons.

The following files and directories are created in the `/tftpboot` directory:

```
drwxr-xr-x   6 root sys     512 Dec 28 14:53 client-host-name
lrwxrwxrwx   1 root root     31 Dec 28 14:53 menu.lst.01ethernet-address
                        -> /tftpboot/client-host-name/grub/menu.lst
-rw-r--r--   1 root root 118672 Dec 28 14:53 01ethernet-address
```

4 **If the console is on a serial port, edit the `/tftpboot/menu.lst.01`*ethernet-address* file. Uncomment the line that specifies the `tty` setting.**

To change the default `menu.lst` file that is created on the client, edit the `echo` lines in the `/usr/sadm/lib/wbem/config_tftp` file.

For more information, see "Booting an x86 Based System from the Network" on page 276.

5 **Verify that the diskless clients were installed.**

```
# /usr/sadm/bin/smdiskless list -H host-name:898 --
```

6 **(Optional) Continue to use the `smdiskless add` command to add each diskless client.**

Example 7–3 x86: Adding Diskless Client Support to an x86 Based System in the GRUB Boot Environment

This example shows how to add a Solaris 10 x86 based diskless client, `mikey1`.

```
rainy-01# /usr/sadm/bin/smdiskless add -H sdts-01-qfe0 -- -o sdts-01-qfe0
-n mikey1 -i 192.168.20.22 -e 00:E0:88:55:33:BC -x os=i386.i86pc.Solaris_10
-x root=/export/root/mikey1 -x swap=/export/swap/mikey1
Loading Tool: com.sun.admin.osservermgr.cli.OsServerMgrCli
from  sdts-01-qfe0
Login to rainy-01-qfe0 as user root was successful.
Download of com.sun.admin.osservermgr.cli.OsServerMgrCli from
rainy-01-qfe0 was successful.

# /usr/sadm/bin/smdiskless list -H mikey1:898 --
Loading Tool: com.sun.admin.osservermgr.cli.OsServerMgrCli from mikey1:898
Login to mikey1 as user root was successful.
Download of com.sun.admin.osservermgr.cli.OsServerMgrCli from mikey1:898 was
successful.
Platform
--------------------------------------------------------------------------------
i386.i86pc.Solaris_10
sparc.sun4us.Solaris_10
sparc.sun4u.Solaris_10
i386.i86pc.Solaris_9
```

Example 7–4 x86: Adding the `BootSrvA` and `BootFile` DHCP Options to the DHCP Server Configuration

This example shows how to add the `BootSrva` and `BootFile` DHCP options that are necessary for enabling a PXE boot.

```
rainy-01# pntadm -A mikey1 -m  0100E0885533BC -f 'MANUAL+PERMANENT' \
-i 0100E0885533BC 192.168.0.101

rainy-01# dhtadm  -A -m 0100E0885533BC -d \
":BootSrvA=192.168.0.1:BootFile=0100E0885533BC:"
```

In the preceding examples, the server address is the IP address of the server, and the client macro is named by the client's Ethernet type (01) and its mac address. This number is also the name of the file that is used in the `/tftpboot` directory on the installation server. Note that the notation for the client macro consists of uppercase letters and should not contain any colons.

▼ How to Add a Diskless Client in the Solaris 10 OS

Use this procedure to add a diskless client after you have added OS services. Unless otherwise noted, this procedure includes general information for both SPARC based and x86 based systems.

1. **Become superuser or assume an equivalent role.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2. **Add the diskless client.**

```
# /usr/sadm/bin/smdiskless add -- -i ip-address -e ethernet-address
-n client-name -x os=instruction-set.machine-class.Solaris_version
-x root=/export/root/client-name -x swap=/export/swap/client-name
-x swapsize=size -x tz=time-zone -x locale=locale-name
```

`add`
   Adds the specified diskless client.

`--`
   Identifies that the subcommand arguments start after this point.

`-i` *ip-address*
   Identifies the IP address of the diskless client.

`-e` *ethernet-address*
   Identifies the Ethernet address of the diskless client.

`-n` *client-name*
   Specifies the name of the diskless client.

`-x os=`*instruction-set.machine-class.*.Solaris_*version*
   Specifies the instruction architecture, machine class, OS, and the Solaris version for the diskless client.

`-x root=root=/export/root/`*client-name*
   Identifies the root (`/`) directory for the diskless client.

`-x swap=root=/export/root/`*client-name*
   Identifies the swap file for the diskless client.

`-x swapsize=`*size*
   Specifies the size of the swap file in Mbytes. The default is 24 Mbytes.

`-x tz=`*time-zone*
   Specifies the time-zone for the diskless client.

`-x locale=`*locale-name*
   Specifies the locale to install for the diskless client.

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

3 **(Optional) Continue to use the `smdiskless add` command to add each diskless client.**

4 **Verify that the diskless clients were installed.**

```
# /usr/sadm/bin/smdiskless list -H host-name:898 --
```

Example 7–5 SPARC: Adding Diskless Client Support to a SPARC Based System

This example shows how to add Solaris 10 sun4u diskless client, `starlite`, from the server `bearclaus`.

```
# /usr/sadm/bin/smdiskless add -- -i 172.20.27.28 -e 8:0:20:a6:d4:5b
-n starlite -x os=sparc.sun4u.Solaris_10 -x root=/export/root/starlite
-x swap=/export/swap/starlite -x swapsize=128 -x tz=US/Mountain
-x locale=en_US

# /usr/sadm/bin/smdiskless list -H starlite:898 --
Loading Tool: com.sun.admin.osservermgr.cli.OsServerMgrCli from line2-v480:898
Login to line2-v480 as user root was successful.
Download of com.sun.admin.osservermgr.cli.OsServerMgrCli from line2-v480:898 was
successful.
Platform
--------------------------------------------------------------------------------
i386.i86pc.Solaris_10
sparc.sun4us.Solaris_10
sparc.sun4u.Solaris_10
i386.i86pc.Solaris_9
sparc.sun4m.Solaris_9
sparc.sun4u.Solaris_9
sparc.sun4us.Solaris_9
```

Note that the `smdiskless list -H` command output lists both SPARC based and x86 based systems.

Example 7–6 x86: Adding Diskless Client Support to an x86 Based System in the Solaris 10 OS

This example shows how to add a Solaris 10 x86 based diskless client, `mars`, from the server `bearclaus`.

```
# /usr/sadm/bin/smdiskless add -- -i 172.20.27.176 -e 00:07:E9:23:56:48
-n mars -x os=i386.i86pc.Solaris_10 -x root=/export/root/mars
-x swap=/export/swap/mars -x swapsize=128 -x tz=US/Mountain
-x locale=en_US
```

▼ x86: How to Boot a Diskless Client With GRUB

If you have installed or upgraded your system to at least the Solaris 10 1/06 OS, the procedure for booting a diskless client has changed. Follow these steps to boot a diskless client with GRUB.

Note – Starting with the Solaris Express 3/06 release, the GRUB failsafe interaction has changed. When booting the failsafe archive, you are no longer prompted by the system to automatically update the boot archives. The system prompts you to update the boot archives only if inconsistent boot archives are detected. For more information, see "How to Boot the Failsafe Archive on an x86 Based System by Using GRUB" on page 264.

Before You Begin To ensure that the system boots from the network, verify the following prerequisites on the OS server:

- Confirm that the name service used to add the diskless client and the OS services matches the primary name in the server's `/etc/nsswitch.conf` file.
- Verify that the DHCP and `tftp` boot services are running.
- Configure the system BIOS to boot the system from the network by enabling the PXE ROM option.

  Some PXE-capable network adapters have a feature that enables PXE boot if you type a particular keystroke in response to a brief boot-time prompt. See your hardware documentation for information about how to set the boot priority in the BIOS.

1. **Boot the diskless client by typing the correct keystroke combination.**

   The GRUB menu is displayed.

   Depending on the configuration of your network installation server, the GRUB menu that is displayed on your system might vary from the GRUB menu that is shown here.

2. **Use the arrow keys to select a boot entry, then press Enter.**

   If you do not make a selection, the default OS instance is automatically booted after several seconds.

   - **If you need to modify the GRUB kernel behavior by editing the GRUB menu at boot time, use the arrow keys to select a boot entry, then type `e` to edit the entry.**

     Note – The previous example shows the GRUB multiboot implementation. The GRUB menus vary, depending on the Solaris release you are running.

     The boot command that you want to edit is displayed in the GRUB edit screen.

     For more information about modifying kernel behavior at boot time, see Chapter 11, "Modifying Solaris Boot Behavior (Tasks)."

- **To save the edits and return to the GRUB menu, press Enter.**

  The GRUB menu is displayed, showing the edits you made to the boot command.

- **Type `b` to boot the system from the network.**

▼ SPARC: How to Boot a Diskless Client in the Solaris 10 OS

Before You Begin Verify the following prerequisites on the OS server:

- Confirm that the name service used to add the diskless client and the OS services matches the primary name in the server's `/etc/nsswitch.conf` file.

  Otherwise, the diskless client will not boot.

- Confirm that the `rpc.bootparamd` daemon is running. If it is not running, start it.

● **Boot the diskless client.**

```
ok boot net
```

▼ How to Remove Diskless Client Support

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 **Remove the diskless client support.**

```
# /usr/sadm/bin/smdiskless delete -- -o host-name:898 -n client-name
```

3 **Verify that the diskless client support has been removed.**

```
# /usr/sadm/bin/smosservice list -H host-name:898 --
```

Example 7–7 Removing Diskless Client Support

This example shows how to remove the diskless client `holoship` from the OS server `starlite`.

```
# /usr/sadm/bin/smdiskless delete -- -o starlite:898 -n holoship
Authenticating as user: root

Type /? for help, pressing enter accepts the default denoted by [ ]
Please enter a string value for: password ::
Starting SMC server version 2.0.0.
```

```
endpoint created: :898
SMC server is ready.

# /usr/sadm/bin/smosservice list -H starlite:898 --
Loading Tool: com.sun.admin.osservermgr.cli.OsServerMgrCli from starlite
Login to starlite as user root was successful.
Download of com.sun.admin.osservermgr.cli.OsServerMgrCli from starlite
was successful.
```

▼ How to Remove OS Services for Diskless Clients

1. **Become superuser or assume an equivalent role.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2. **Remove the OS services for the diskless clients.**

   ```
   # /usr/sadm/bin/smosservice delete -H $HOST:$PORT -u root -p $PASSWD --
   -x instruction-set.all.Solaris_version
   ```

   Note – Only the machine-class, `all`, is supported.

3. **Verify that the OS services have been removed.**

   ```
   # /usr/sadm/bin/smosservice list -H host-name:898 --
   ```

Example 7–8 Removing OS Services for Diskless Clients

The following example shows how to removing the diskless client OS services (`sparc.all.Solaris_10`) from the server `starlite`.

```
# /usr/sadm/bin/smosservice delete -H starlite:898 -u root
-p xxxxxx -- -x sparc.all.solaris_10
Authenticating as user: root
Type /? for help, pressing enter accepts the default denoted by [ ]
Please enter a string value for: password ::

# /usr/sadm/bin/smosservice list -H starlite:898 --
Loading Tool: com.sun.admin.osservermgr.cli.OsServerMgrCli from starlite:898
Login to starlite as user root was successful.
Download of com.sun.admin.osservermgr.cli.OsServerMgrCli from starlite:898
was successful
```

Patching Diskless Client OS Services

You use the `smosservice patch` command to do the following:

- Establish the `/export/diskless/Patches` patch spool directory on an OS server.
- Add patches to the patch spool directory. If the patch you are adding obsoletes an existing patch in the spool, the obsolete patch is moved to `/export/diskless/Patches/Archive`.
- Delete patches from the patch spool directory.
- List the patches in the patch spool directory.
- Synchronize spooled patches out to clients. You must reboot each synchronized client for the client to recognize the patch update.

Note – Keep your OS servers up to date by installing recommended OS patches on a timely basis.

For information on downloading patches, see "How to Download and Apply a Solaris Patch" on page 426.

Displaying OS Patches for Diskless Clients

Diskless client patches are logged in different directories, depending on the type of patch:

- Kernel patches are logged in the diskless client's `/var/sadm/patch` directory. To display kernel patches, type the following command on the diskless client:

  ```
  % patchadd –p
  ```

  Note – You must be logged in to the diskless client when you run this command. Running the `patchadd -p` command on the OS server displays kernel patches for the OS server only.

- `/usr` patches are logged in the OS server's `/export/Solaris_`*version*`/var/patch` directory. A directory is created for each patch ID. To display `/usr` patches, type the following command on the OS server:

  ```
  % patchadd -S Solaris_version -p
  Patch: 111879-01 Obsoletes: Requires: Incompatibles: Packages: SUNWwsr
  ```

To list all spooled patches by OS and architecture, use the `smosservice` command with the `-P` option.

▼ How to Add an OS Patch for a Diskless Client

1. **Become superuser or assume an equivalent role.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2. **Log in to the diskless client system and shut it down.**

   ```
   # init 0
   ```

3. **Add the patch to a spool directory.**

   ```
   # /usr/sadm/bin/smosservice patch -- -a /var/patches/patch-ID-revision
   ```

   If the patch to add depends on another patch, adding the patch fails with the following message:

   ```
   The patch patch-ID-revision could not be added
   because it is dependent on other patches which have not yet been spooled.
   You must add all required patches to the spool first.
   ```

4. **Verify that the patch has been spooled.**

   ```
   # /usr/sadm/bin/smosservice patch -- -P
   ```

5. **Push the spooled patch to the diskless client.**

   ```
   # /usr/sadm/bin/smosservice patch -- -m -U
   ```

   Note – Pushing and synchronizing the patch to the diskless client can take up to 90 minutes per patch.

   ---

6. **Verify the patch is applied to the diskless client.**

   ```
   # /usr/sadm/bin/smosservice patch -- -P
   ```

Example 7–9 Adding an OS Patch for a Diskless Client

This example shows how to add a Solaris 8 patch (111879-01) to the diskless client's OS services on the server.

```
# /usr/sadm/bin/smosservice patch -- -a /var/patches/111879-01
Authenticating as user: root

Type /? for help, pressing <enter> accepts the default denoted by [ ]
Please enter a string value for: password ::
Loading Tool: com.sun.admin.osservermgr.cli.OsServerMgrCli from starlite
Login to starlite as user root was successful.
Download of com.sun.admin.osservermgr.cli.OsServerMgrCli from starlite
```

```
was successful..
.
# /usr/sadm/bin/smosservice patch -- -P
Patches In Spool Area
Os Rel Arch   Patch Id  Synopsis
-------------------------------------------------------------------------
8      sparc  111879-01 SunOS 5.8: Solaris Product Registry patch SUNWwsr

Patches Applied To OS Services
Os Service                              Patch
-------------------------------------------------------------------------
Solaris_8

Patches Applied To Clone Areas
Clone Area                              Patch
-------------------------------------------------------------------------
Solaris_8/sun4u          Patches In Spool Area
Os Rel Arch   Patch Id  Synopsis
----------------------------------------------------------------------------
8      sparc  111879-01 SunOS 5.8: Solaris Product Registry patch SUNWwsr
.
.
.
# /usr/sadm/bin/smosservice patch -- -m -U
Authenticating as user: root

Type /? for help, pressing <enter> accepts the default denoted by [ ]
Please enter a string value for: password ::
Loading Tool: com.sun.admin.osservermgr.cli.OsServerMgrCli from starlite
Login to starlite as user root was successful.
Download of com.sun.admin.osservermgr.cli.OsServerMgrCli from starlite
was successful.

# /usr/sadm/bin/smosservice patch -- -P
Authenticating as user: root
.
.
.
Patches In Spool Area
Os Rel Arch   Patch Id  Synopsis
----------------------------------------------------------------------------
8      sparc  111879-01 SunOS 5.8: Solaris Product Registry patch SUNWwsr

Patches Applied To OS Services
Os Service                              Patch
----------------------------------------------------------------------------
Solaris_8
```

```
Patches Applied To Clone Areas
Clone Area                          Patch
-----------------------------------------------------------------------------
Solaris_8/sun4u
```

Troubleshooting Diskless Client Problems

This section describes problems that are encountered when managing diskless clients and possible solutions.

Troubleshooting Diskless Client Installation Problems

The `smosservice add` command does not install any packages that are designated `ARCH=all` in the root (`/`) or `/usr` file systems. As a result, these packages are skipped. No warning or error messages are displayed. You must add these packages to the newly-created Solaris OS service manually. This behavior has existed since the Solaris 2.1 OS. The behavior applies to both SPARC based and x86 based clients. Note that the list of missing packages varies, depending on which Solaris OS you are running.

▼ How to Locate and Install Missing `ARCH=all` Packages

This procedure shows you how to locate and install missing `ARCH=all` packages after you have created the Solaris OS service on the server. Examples that are provided in this procedure apply to the Solaris 10 6/06 OS.

1. **Locate all the packages with the `ARCH=all` parameter.**

   a. **Change directories to the `Product` directory of the media for the Solaris 10 image. For example:**

   ```
   % cd /net/server/export/Solaris/s10u2/combined.s10s_u2wos/latest/Solaris_10/Product
   ```

   b. **List all the packages in the `pkginfo` file that have the `ARCH=all` parameter.**

   ```
   % grep -w ARCH=all */pkginfo
   ```

   If an error message indicating the arguments list is too long is displayed, you can alternately run the following command to generate the list:

   ```
   % find . -name pkginfo -exec grep -w ARCH=all {} /dev/null \;
   ```

   Note that running this command takes longer to produce results.

   The output is similar to the following:

   ```
   ./SUNWjdmk-base/pkginfo:ARCH=all
   ./SUNWjhdev/pkginfo:ARCH=all
   ```

```
./SUNWjhrt/pkginfo:ARCH=all
./SUNWjhdem/pkginfo:ARCH=all
./SUNWjhdoc/pkginfo:ARCH=all
./SUNWmlibk/pkginfo:ARCH=all
```

The information that is provided in this list enables you to determine which packages are installed in the `/usr` file system and which packages are installed in the root (`/`) file system.

c. **Check the value of the `SUNW_PKGTYPE` parameter in the package list you generated.**

Packages that belong in the `/usr` file system are designated as `SUNW_PKGTYPE=usr` in the `pkginfo` file. Packages that belong in the root (`/`) file system are designated as `SUNW_PKGTYPE=root` in the `pkginfo` file. In the preceding output, all the packages belong in the `/usr` file system.

2 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

3 **Create the temporary installation administration files.**

You must create a separate installation administration file for packages that are installed in the root (`/`) file system and a separate installation administration file for packages that are installed in the `/usr` file system.

- For `ARCH=all` packages that are installed in the `/usr` file system, create the following temporary installation administration file:

```
# cat >/tmp/admin_usr <<EOF
mail=
instance=unique
partial=nocheck
runlevel=nocheck
idepend=nocheck
rdepend=nocheck
space=nocheck
setuid=nocheck
conflict=nocheck
action=nocheck
basedir=/usr_sparc.all
EOF
#
```

- For `ARCH=all` packages that are installed in the root (`/`) file system, if any exist, create the following temporary installation administration file:

```
# cat >/tmp/admin_root <<EOF
mail=
instance=unique
```

```
partial=nocheck
runlevel=nocheck
idepend=nocheck
rdepend=nocheck
space=nocheck
setuid=nocheck
conflict=nocheck
action=nocheck
EOF
#
```

4 **Install the missing `ARCH=all` packages.**

a. **If the current directory is not the Product directory of the media for the Solaris 10 image, change directories to that directory. For example:**

```
# cd /net/server/export/Solaris/s10u2/combined.s10s_u2wos/latest/Solaris_10/Product
```

You can run the `pwd` command to determine the current directory.

b. **Install the missing `ARCH=all` packages in the `/usr` file system.**

```
# pkgadd -R /export/Solaris_10 -a /tmp/admin_usr -d ‘pwd‘ [package-A package-B ...]
```

Multiple packages can be listed when you running the `pkgadd` command.

c. **Check that the `ARCH=all` packages were installed.**

```
# pkginfo  -R /export/Solaris_10  [package-A package-B ...]
```

d. **Install the missing `ARCH=all` packages that in the root (`/`) file system.**

Note that it is possible that none of these packages exist.

```
# pkgadd  -R /export/root/clone/Solaris_10/sun4u  -a /tmp/admin_root -d ‘pwd‘  [package-X package-Y ...]
```

e. **Check that the `ARCH=all` packages were installed.**

```
# pkginfo  -R /export/root/clone/Solaris_10/sun4u [package-X package- ...]
```

5 **After you have finished adding the missing `ARCH=all` packages, remove the temporary installation administration file.**

```
# rm /tmp/administration-file
```

Example 7–10 Locating and Installing Missing `ARCH=all`Packages

This example shows how to install the missing `ARCH=all` package, `SUNWjdmk-base`, in the `/usr` file system.

```
% uname -a
SunOS t1fac46 5.10 Generic_118833-02 sun4u sparc SUNW,UltraSPARC-IIi-cEngine
```

```
% cat /etc/release
                       Solaris 10 6/06 s10s_u2wos_03 SPARC
           Copyright 2006 Sun Microsystems, Inc.  All Rights Reserved.
                        Use is subject to license terms.
                           Assembled 06 February 2006

% cd /net/ventor/export/Solaris/s10u2/combined.s10s_u2wos/latest/Solaris_10/Product

% grep -w ARCH=all */pkginfo
Arguments too long

% find . -name pkginfo -exec grep -w ARCH=all {} /dev/null \;
./SUNWjdmk-base/pkginfo:ARCH=all
./SUNWjhdev/pkginfo:ARCH=all
./SUNWjhrt/pkginfo:ARCH=all
./SUNWjhdem/pkginfo:ARCH=all
./SUNWjhdoc/pkginfo:ARCH=all
./SUNWmlibk/pkginfo:ARCH=all

% grep -w SUNW_PKGTYPE=usr ./SUNWjdmk-base/pkginfo ./SUNWjhdev/pkginfo ...
./SUNWjdmk-base/pkginfo:SUNW_PKGTYPE=usr
./SUNWjhdev/pkginfo:SUNW_PKGTYPE=usr
./SUNWjhrt/pkginfo:SUNW_PKGTYPE=usr
./SUNWjhdem/pkginfo:SUNW_PKGTYPE=usr
./SUNWjhdoc/pkginfo:SUNW_PKGTYPE=usr

% grep -w SUNW_PKGTYPE=root ./SUNWjdmk-base/pkginfo ./SUNWjhdev/pkginfo ...

% su
Password: xxxxxx

# cat >/tmp/admin_usr <<EOF
mail=
instance=unique
partial=nocheck
runlevel=nocheck
idepend=nochec> k
rdepend=nocheck
space=nocheck
setuid=nocheck
conflict=nocheck
action=nocheck
basedir=/usr_sparc.all
EOF

# pwd
/net/ventor/export/Solaris/s10u2/combined.s10s_u2wos/latest/Solaris_10/Product
```

```
# pkginfo  -R /export/Solaris_10  SUNWjdmk-base
ERROR: information for "SUNWjdmk-base" was not found

# pkgadd  -R /export/Solaris_10  -a /tmp/admin_usr  -d ‘pwd‘  SUNWjdmk-base

Processing package instance <SUNWjdmk-base> </net/ventor/export/Solaris/s10u2/combined.s10s_u2wos...


Java DMK 5.1 minimal subset(all) 5.1,REV=34.20060120
Copyright 2005 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.
Using </export/Solaris_10/usr_sparc.all>
## Processing package information.
## Processing system information.

Installing Java DMK 5.1 minimal subset as <SUNWjdmk-base>

## Installing part 1 of 1.
2438 blocks

Installation of <SUNWjdmk-base> was successful.

# pkginfo -R /export/Solaris_10 SUNWjdmk-base
application SUNWjdmk-base Java DMK 5.1 minimal subset

# rm /tmp/admin_usr
```

Troubleshooting General Diskless Client Problems

This section lists some common problems with diskless clients that you might encounter and possible solutions.

Problem: Diskless client reports `Owner of the module /usr/lib/security/pam_unix_session.so.1 is not root`, when attempting to log in, the `/usr` file system is owned by `nobody`.

Solution: To correct the problem, follow this workaround:

1. Using a text editor, modify the diskless client's `server:/export/root/client/etc/default/nfs` file.
2. Change the `#NFSMAPID_DOMAIN=domain` line to the following:

   ```
   NFSMAPID_DOMAIN=the_same_value_as_in_server’s_/var/run/nfs4_domain
   ```
3. Ensure that the OS server and the diskless client have the same `nfsmapid` domain. To verify this information, check the `/var/run/nfs4_domain` file.

Caution – If the diskless client's `nfs4_domain` file contains a different value than the OS server's `/var/run/nfs4_domain` file, you will not be able to log in to the system after the diskless client boots.

---

4. Reboot the diskless client.

For more information, see Chapter 3, "NFS Tunable Parameters," in *Solaris Tunable Parameters Reference Manual* and `nfsmapid`(1M).

Problem: The OS server fails to do the following:

- Respond to client Reverse Address Resolution Protocol (RARP) requests
- Respond to client `bootparam` requests
- Mount a diskless client root (`/`) file system

Solution: The following solutions apply in a files environment.

- Verify that `files` is listed as the first source for `hosts`, `ethers`, and `bootparams` in the `/etc/nsswitch.conf` file on the OS server.
- Verify that the client's IP address appears in the `/etc/inet/hosts` file.

Note – If you are *not* running at least the Solaris 10 8/07 release, you must also verify that the client's IP address appears in the `/etc/inet/ipnodes` file.

In this Solaris release, there is no longer two separate `hosts` files. The `/etc/inet/hosts` file is a single file that contains both IPv4 and IPv6 entries. You do not need to maintain IPv4 entries in two `hosts` files that always require synchronization. For backward compatibility, the `/etc/inet/ipnodes` file is replaced with a symbolic link of the same name to the `/etc/inet/hosts` file. For more information, see the `hosts`(4) man page.

- Verify that the client's Ethernet address appears in the `/etc/ethers` file.
- Verify that the `/etc/bootparams` file contains the following paths to the client's root (`/`) directory and swap areas.

```
client root=os-server:/export/root/client swap=os-server:
/export/swap/client
```

The swap size varies depending on whether you specify the `-x` *swapsize* option when you add the diskless client. If you specify the `-x` *dump* option when you add the diskless client, the following line is present.

```
dump=os-server:/export/dump/client dumpsize=512
```

The dump size varies depending on whether you specify the `-x` *dumpsize* option when you add the diskless client.

- Verify that the OS server's IP address appears in the `/export/root/`*client*`/etc/inet/hosts` file.

Problem: The OS server fails to do the following:

- Respond to client RARP requests
- Respond to client `bootparam` requests
- Mount a diskless client root (`/`) file system

Solution: The following solutions apply in a name service environment.

- Verify that both the OS server's and the client's Ethernet address and IP address are correctly mapped.
- Verify that the `/etc/bootparams` file contains the paths to the client's root (`/`) directory and swap areas.

```
client root=os-server:/export/
root/client swap=os-server:/export/
swap/client swapsize=24
```

The swap size varies depending on whether you specify the `-x` *swapsize* option when you add the diskless client. If you specify the `-x` *dump* option when you add the diskless client, the following line is present:

```
dump=os-server:/export/dump/client dumpsize=24
```

The dump size varies depending on whether you specify the `-x` *dumpsize* option when you add the diskless client.

Problem: Diskless client panics

Solution: Verify the following:

- The OS server's Ethernet address is correctly mapped to its IP address. If you physically moved a system from one network to another, you might have forgotten to remap the system's new IP address.
- The client's host name, IP address, and Ethernet address do not exist in the database of another server *on the same subnet* that responds to the client's RARP, Trivial File Transfer Protocol (TFTP), or `bootparam` requests. Often, test systems are set up to install their OS from an install server. In these cases, the install server answers the client's RARP or `bootparam` request, returning an incorrect IP address. This incorrect address might result in the download of a boot program for the wrong architecture, or a failure to mount the client's root (`/`) file system.

- The diskless client's TFTP requests are not answered by an install server (or previous OS server) that transfers an incorrect boot program. If the boot program is of a different architecture, the client immediately panics. If the boot program loads from a non-OS server, the client might obtain its root partition from the non-OS server and its `/usr` partition from the OS server. In this situation, the client panics if the root and `/usr` partitions are of conflicting architectures or versions.
- If you are using both an install server and an OS server, verify that the following entry exists in the `/etc/dfs/dfstab` file.

  ```
  share -F nfs -o -ro /export/exec/Solaris_version-instruction-set.all/usr
  ```

  where *version*= `8`, `9`,`10`, 11, and *instruction-set*=`sparc` or `i386`.
- Verify that the diskless client's root (`/`), `/swap`, and `/dump` (if specified) partitions have share entries:

  ```
  share -F nfs -o rw=client,root=client /export/root/client
  share -F nfs -o rw=client,root=client /export/swap/client
  share -F nfs -o rw=client,root=client /export/dump/client
  ```
- On the OS server, type the following command to check which files are shared:

  ```
  % share
  ```

  The OS server must share `/export/root/`*client* and `/export/swap/`*client-name* (defaults), or the root, `/swap`, and `/dump` partitions that you specified when you added the diskless client.

  Verify that the following entries exist in the `/etc/dfs/dfstab` file:

  ```
  share -F nfs -o ro /export/exec/Solaris_version-instruction-set.all/usr
  share -F nfs -o rw=client,root=client /export/root/client
  share -F nfs -o rw=client,root=client /export/swap/client
  ```

Problem: OS server is not responding to diskless client's RARP request

Solution: From the client's intended OS server, run the `snoop` command as superuser (`root`) by using the client's Ethernet address:

```
# snoop xx:xx:xx:xx:xx:xx
```

Problem: Boot program downloads but panics early in the process

Solution: Use the `snoop` command to verify that the intended OS server is answering the client's TFTP and NFS requests.

Problem: Diskless client hangs.

Solution: Restart the following daemons on the OS server:

```
# /usr/sbin/rpc.bootparamd
# /usr/sbin/in.rarpd -a
```

Problem: Incorrect server responds to diskless client's RARP request

Solution: Restart the following daemons on the OS server:

```
# /usr/sbin/rpc.bootparamd
# svcadm enable network/rarp
```

CHAPTER 8

Introduction to Shutting Down and Booting a System

The Solaris Operating System (Solaris OS) is designed to run continuously so that electronic mail and network resources are available to users. This chapter provides guidelines for shutting down and booting a system.

This is a list of the information in this chapter:

For an overview of all of the boot features and methods that are available in the Solaris release, see Chapter 9, "Shutting Down and Booting a System (Overview)"

For instructions on booting a Solaris system, see Chapter 12, "Booting a Solaris System (Tasks)."

What's New in Shutting Down and Booting a System

This section describes new boot features in the Solaris release.

x86: iSCSI Boot

Solaris Express Community Edition, build 104: The iSCSI Boot feature enables you to initialize an operating system over the network from a remote location, such as a storage disk array. This version of iSCSI Boot supports booting from x86 based systems. iSCSI Boot is typically loaded onto an initiator or a diskless client, while the hard disk resides on a SCSI target that is attached to the network. Because the feature uses a standard Ethernet-based

infrastructure, data, storage, and networking traffic can be consolidated on a standard network. More information about iSCSI Boot can be found at
`http://wikis.sun.com/display/OpenSolarisInfo/iSCSI+Boot`.

Fast Reboot on the x86 Platform

Solaris Express Community Edition, build 100, and the OpenSolaris 2008.11 release: The Fast Reboot feature enables you to reboot an x86 based system, bypassing the firmware and boot loader processes. Fast Reboot implements an in-kernel boot loader that loads the kernel into memory and then switches to that kernel, so that the reboot process occurs within seconds. This feature is implemented on both 32-bit and 64-bit kernels.

For more information about this feature, see "x86: Introducing Fast Reboot" on page 194.

ZFS Boot Support

Solaris Express Community Edition, builds 88 and 90: This release includes ZFS ™ installation and boot support. Systems can now be installed and booted from a ZFS root file system. This implementation applies to both SPARC and x86 based systems. Booting, system operations, and installation procedures have been modified to support this change.

For more information, see "Booting From a ZFS Root File System" on page 195.

x86: New `findroot` Command

Solaris Express Community Edition, build 88: All Solaris installation methods, including Solaris Live Upgrade, now use the `findroot` command for specifying which disk slice on an x86 based system to boot. This implementation supports booting systems with ZFS roots, as well as UFS roots. Previously, the `root` command, `root (hd0.0.a)`, was used to explicitly specify which disk slice to boot. This information is located in the `menu.lst` file that is used by GRUB.

The most common form of the GRUB `menu.lst` entry is now:

```
findroot (rootfs0,0,a)
kernel$ /platform/i86pc/kernel/$ISADIR/unix -B $ZFS-BOOTFS
module$ /platform/i86pc/$ISADIR/boot_archive
```

For more information, see "x86: Implementation of the `findroot` Command" on page 228.

Support for Specifying Platform by Using `bootadm` Command

Solaris Express Community Edition, build 87: A new `-p` option has been added to the `bootadm` command.

This option enables you to specify the platform or machine hardware class of a client system in situations where the client platform differs from the server platform, for example when administering diskless clients.

Note – The `-p` option must be used with the `-R` option.

```
# bootadm -p platform -R [altroot]
```

The specified platform must be one of the following:

- `i86pc`
- `sun4u`
- `sun4v`

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

Solaris SPARC Bootstrap Process Redesigned

Solaris Express Community Edition, build 80: The Solaris SPARC bootstrap process has been redesigned to increase commonality with the Solaris x86 boot architecture.

Other enhancements include an improved boot architecture that supports booting a system from additional file system types, for example a ZFS file system or a single miniroot for installation, as well as booting from DVD, NFS, or HTTP. These enhancements increase flexibility and reduce maintenance requirements on SPARC based systems.

As part of this redesign, the Solaris boot archives and the `bootadm` command, previously only available on the Solaris x86 based platform, are now an integral part of the Solaris SPARC boot architecture.

The primary difference between the SPARC and x86 boot architectures is how the boot device and file are selected at boot time. The SPARC based platform continues to use the OpenBoot™ PROM (OBP) as the primary administrative interface, with boot options selected by using OBP commands. On x86 based systems, these options are selected through the BIOS and the GRand Unified Bootloader (GRUB) menu.

Note – Although the implementation of the Solaris SPARC boot has changed, no administrative procedures for booting a SPARC based system have been impacted. Boot tasks that are performed by the system administrator remain the same as they were prior to the boot architecture redesign.

For more information, see the `boot(1M)` and `bootadm(1M)` man pages.

For more information in this document, see "Understanding the New Solaris SPARC Boot Architecture" on page 189.

x86: Support for Booting the Solaris OS as a Virtualized Control Domain

Solaris Express Community Edition, build 75: In this release, the GRUB boot loader is capable of booting a Solaris release that runs with hypervisor technology. The hypervisor can securely execute multiple virtual machines simultaneously, each running its own operating system, on a single physical system. You can decide at boot time whether to run the Solaris OS as a virtualized domain0, also called a dom0, or as a stand-alone operating system.

You can run the Solaris OS as a virtualized dom0. Whether to run the Solaris OS as a virtualized dom0 or as a stand-alone operating system is a decision you can make at boot time. To run the Solaris OS as a virtualized dom0, there must first be an entry in the `menu.lst` file that specifies the hypervisor. The entry can be the default boot entry, or it can be an option that you select manually at boot time. Note that when you upgrade your system to a Solaris release that includes this capability, the `bootadm` command automatically adds a menu entry for the hypervisor to the `menu.lst` file.

The `bootadm` command installs a default boot entry in the `menu.lst` file that is similar to the following:

```
kernel$ /boot/$ISADIR/xen.gz
module$ /platform/i86xpv/kernel/$ISADIR/unix /platform/i86xpv/kernel/$ISADIR/unix -B $ZFS-BOOTFS
module$ /platform/i86pc/$ISADIR/boot_archive
```

Note – The format for entries that are used in the `menu.lst` file for booting the Solaris OS as a control domain differs slightly from the format that is used for other `menu.lst` entries. The `kernel$` line must identify the hypervisor binary to use and includes any options that are accepted by the hypervisor. The first `module$` line in the file identifies the Solaris kernel to use, for example `unix`, and include any options the `unix` kernel accepts. Due to an implementation detail that exists in this version of GRUB, the specified `unix` kernel must be named twice on the first `module$` line. The second `module$` line identifies the boot archive to use.

For more information, see "Description of a `menu.lst` File That Supports Hypervisor Technology" on page 317.

For more information about administering Solaris systems that support the hypervisor, see `http://www.opensolaris.org/os/community/xen/docs/` and *System Administration Guide: Virtualization Using the Solaris Operating System*

x86: GRUB Support for Directly Loading and Booting the `unix` Kernel

Solaris Express Community Edition, build 57: This release includes changes to the GRUB based boot environment to enable the boot loader to directly load and boot the `unix` kernel.

Note – The GRUB `multiboot` module is no longer used.

This implementation integrates the previous multiboot functionality directly into the platform-specific `unix` kernel module. These changes reduce the time, as well as memory requirements, that are needed to boot the Solaris OS on x86 based systems.

Two new keywords, `kernel$` and `module$`, have been added to GRUB to assist in creating `menu.lst` entries that work with either 32-bit or 64-bit systems. In addition, the `bootadm` command that manages the `menu.lst` file has been modified to create file entries for the platform-specific `unix` module that is loaded by GRUB. During an upgrade, the `bootadm` command converts any existing multiboot `menu.lst` entries to `unix` entries.

The `kernel$` and `module$` keywords are identical to the `kernel` and `module` commands that are used in the GRUB multiboot implementation, with the addition of the `$ISADIR` keyword. This keyword provides the capability to expand to amd64 on 64-bit capable hardware. If the x86 based system is not 64-bit capable, the `$ISADIR` keyword is a null value (`""`). In this case, the system boots the 32-bit kernel.

Note – These changes do not prevent you from booting a newer Solaris kernel with an older implementation of GRUB. Nor do the changes prevent you from booting an older Solaris kernel with a newer implementation of GRUB.

For information about booting an x86 based system with GRUB, see "x86: Administering the GRUB Bootloader" on page 192.

x86: Support for Using Power Button to Initiate System Shutdown

Pressing and releasing the power button on x86 based systems initiates a clean system shutdown and turns the system off. This functionality is equivalent to using the `init 5` command to shut down a system. On some x86 based systems, the BIOS configuration might prevent the power button from initiating shutdown. To enable use of the power button to perform a clean system shutdown, reconfigure the BIOS.

Note – On certain x86 based systems that were manufactured before 1999 and are running an older Solaris release, pressing the power button immediately turns off system power without safely shutting down the system. This same behavior occurs when pressing the power button on systems that are running with ACPI support that is disabled through the use of `acpi-user-options`.

For more information about `acpi-user-options`, see the `eeprom(1M)` man page.

Where to Find Shut Down and Boot Tasks

Use these references to find step-by-step instructions for shutting down and booting a system.

Shut Down and Boot Task	For More Information
Shut down a SPARC based system or an x86 based system	Chapter 10, "Shutting Down a System (Tasks)"
Modify boot behavior	Chapter 11, "Modifying Solaris Boot Behavior (Tasks)"
Boot a SPARC based system or an x86 based system	Chapter 12, "Booting a Solaris System (Tasks)"

Shut Down and Boot Task	For More Information
Manage the Solaris boot archives	Chapter 14, "Managing the Solaris Boot Archives (Tasks)"
Troubleshoot boot behavior on a SPARC or an x86 based system	"Troubleshooting Booting on the SPARC Platform (Task Map)" on page 281

Shut Down and Boot Terminology

This section describes the terminology that is used in shutting down and booting a system.

Run levels and init states	A *run level* is a letter or digit that represents a system state in which a particular set of system services are available. The system is always running in one of a set of well-defined run levels. Run levels are also referred to as *init states* because the `init` process maintains the run level. System administrators use the `init` command or the `svcadm` command to initiate a run-level transition. This book refers to init states as run levels.
Boot options	A *boot option* describes how a system is booted. Different boot options include the following: ▪ **Interactive boot** – You are prompted to provide information about how the system is booted, such as the kernel and device path name. ▪ **Reconfiguration boot** – The system is shutdown and rebooted to add new devices, if the devices are not hot-pluggable.The system is reconfigured to support newly added hardware or new pseudo devices. ▪ **Recovery boot** – The system is hung or an invalid entry is prohibiting the system from booting successfully or from allowing users to log in.

For terminology that is specific to GRUB based booting, see "x86: GRUB Terminology" on page 310.

Guidelines for Shutting Down a System

Keep the following in mind when you shut down a system:

- Use the `init` and `shutdown` commands to shut down a system. Both commands perform a clean system shutdown, which means that all system processes and services are terminated normally.

  ---

  x86 only – For x86 based systems that are running at least the Solaris 10 6/06 release, you can initiate a clean system shutdown by pressing and releasing the power button. Shutting down an x86 based system in this manner is equivalent to using the `init 5` command to shut down a system. On some x86 based systems, the BIOS configuration might prevent the power button from initiating a system shutdown. To use the power button, reconfigure the BIOS.

- Use the `shutdown` command to shut down a server. Logged-in users and systems that mount resources from the server are notified before the server is shut down. Additional notification of system shutdowns by electronic mail is also recommended so that users can prepare for system downtime.
- You need superuser privileges to use the `shutdown` or `init` command to shut down a system.
- Both `shutdown` and `init` commands take a run level as an argument.

  The three most common run levels are as follows:

  - **Run level 3** – All system resources are available and users can log in. By default, booting a system brings it to run level 3, which is used for normal day-to-day operations. This run level is also known as multiuser level with NFS resources shared.
  - **Run level 6** – Stops the operating system and reboots to the state that is defined by the `initdefault` entry in the `/etc/inittab` file.
  - **Run level 0** – The operating system is shut down, and it is safe to turn off power. You need to bring a system to run level 0 whenever you move a system, or add or remove hardware.

  Run levels are fully described in Chapter 16, "Managing Services (Overview)."

Guidelines for Booting a System

Keep the following in mind when you boot a system:

- After a SPARC based system is shut down, it is booted by using the `boot` command at the PROM level.
- After an x86 based system is shut down, it is booted by selecting an OS instance in the GRUB menu.
- In the Solaris 9 release and some Solaris 10 releases, after an x86 based system is shut down, it is booted by using the `boot` command at the Primary Boot Subsystem menu.
- A system can be rebooted by turning the power off and then back on.

Caution – This method is not considered a clean shutdown, unless you have an x86 based system that is running a Solaris release that supports this shutdown method. See "x86: Support for Using Power Button to Initiate System Shutdown" on page 180. Use this shutdown method only as an alternative in emergency situations. Because system services and processes are terminated abruptly, file system damage is likely to occur. The work required to repair this type of damage could be substantial and might require the restoration of various user and system files from backup copies.

- SPARC and x86 based systems use different hardware components for booting. These differences are described in Chapter 15, "x86: GRUB Based Booting (Reference)."

When to Shut Down a System

The following table lists system administration tasks and the type of shutdown that is needed to initiate the task.

TABLE 8–1 Shutting Down a System

Reason for System Shutdown	Appropriate Run Level	For More Information
To turn off system power due to anticipated power outage	Run level 0, where it is safe to turn off power	Chapter 10, "Shutting Down a System (Tasks)"
To change kernel parameters in the `/etc/system` file	Run level 6 (reboot the system)	Chapter 10, "Shutting Down a System (Tasks)"
To perform file system maintenance, such as backing up or restoring system data	Run level S (single-user level)	Chapter 10, "Shutting Down a System (Tasks)"

TABLE 8–1 Shutting Down a System *(Continued)*

Reason for System Shutdown	Appropriate Run Level	For More Information
To repair a system configuration file such as `/etc/system`	See "When to Boot a System" on page 184	N/A
To add or remove hardware from the system	Reconfiguration boot (also to turn off power when adding or removing hardware)	"Adding a Peripheral Device to a System" in *System Administration Guide: Devices and File Systems*
	Reconfiguration boot (shut down and turn off power when adding or removing devices, if the devices are not hot-pluggable)	
To repair an important system file that is causing system boot failure	See "When to Boot a System" on page 184	N/A
To boot the kernel debugger (`kmdb`) to track down a system problem	Run level 0, if possible	Chapter 10, "Shutting Down a System (Tasks)"
To recover from a hung system and force a crash dump	See "When to Boot a System" on page 184	N/A
Reboot the system by using the kernel debugger (`kmdb`), if the debugger can't be loaded at runtime.	Run level 6 (reboot the system)	**For SPARC based systems:** "SPARC: How to Boot the System With the Kernel Debugger (`kmdb`)" on page 286 **For x86 based systems:** ,"x86: How to Boot a System With the Kernel Debugger in the GRUB Boot Environment (`kmdb`)" on page 290

For examples of shutting down a server or a stand-alone system, see Chapter 10, "Shutting Down a System (Tasks)."

When to Boot a System

The following table lists system administration tasks and the corresponding boot option that is used to complete the task.

TABLE 8–2 Booting a System

Reason for System Reboot	Appropriate Boot Option	Information for SPARC Based Systems	Information for x86 Based Systems
Turn off system power due to anticipated power outage.	Turn system power back on	Chapter 10, "Shutting Down a System (Tasks)"	Chapter 10, "Shutting Down a System (Tasks)"

TABLE 8–2 Booting a System *(Continued)*

Reason for System Reboot	Appropriate Boot Option	Information for SPARC Based Systems	Information for x86 Based Systems
Change kernel parameters in the `/etc/system` file.	Reboot the system to run level 3 (multiuser level with NFS resources shared)	"SPARC: How to Boot a System to Run Level 3 (Multiuser Level)" on page 235	"x86: How to Boot a System to Run Level 3 (Multiuser)" on page 253
Perform file system maintenance, such as backing up or restoring system data.	Press Control-D from run level S to bring the system back to run level 3	"SPARC: How to Boot a System to Run Level S (Single-User Level)" on page 236	"x86: How to Boot a System to Run Level S (Single-User Level)" on page 254
Repair a system configuration file such as `/etc/system`.	Interactive boot	"SPARC: How to Boot a System Interactively" on page 237	"x86: How to Boot a System Interactively" on page 257
Add or remove hardware from the system.	Reconfiguration boot (turn on system power after adding or removing devices, if devices are not hot-pluggable)	"Adding a System Disk or a Secondary Disk (Task Map)" in *System Administration Guide: Devices and File Systems*	"Adding a System Disk or a Secondary Disk (Task Map)" in *System Administration Guide: Devices and File Systems*
	Reconfiguration boot (also to turn on system power after adding or removing hardware)		
Boot the system by using the kernel debugger (`kmdb`) to track down a system problem.	Booting `kmdb`	"SPARC: How to Boot the System With the Kernel Debugger (`kmdb`)" on page 286	"x86: How to Boot a System With the Kernel Debugger in the GRUB Boot Environment (`kmdb`)" on page 290
Boot the system in failsafe mode to repair an important system file that is causing system boot failure.	Booting the failsafe archive	"How to Boot the Failsafe Archive on a SPARC Based System" on page 246	"How to Boot the Failsafe Archive on an x86 Based System by Using GRUB" on page 264
To recover from a hung system and force a crash dump.	Recovery boot	"SPARC: How to Force a Crash Dump and Reboot of the System" on page 283	"x86: How to Force a Crash Dump and Reboot of the System" on page 289

◆ ◆ ◆ CHAPTER 9

Shutting Down and Booting a System (Overview)

This chapter provides an overview of booting a system. The Solaris boot design, boot processes, and various methods of booting a system in the Solaris OS are described.

This is a list of the information in this chapter.

- "Fundamentals of the Solaris Boot Design" on page 188
- "Understanding the New Solaris SPARC Boot Architecture" on page 189
- "Implementation of the Boot Archives on Solaris SPARC" on page 191
- "x86: Administering the GRUB Bootloader" on page 192
- "x86: Introducing Fast Reboot" on page 194
- "Booting From a ZFS Root File System" on page 195

For instructions on booting a Solaris system, see Chapter 12, "Booting a Solaris System (Tasks)"

For what's new in shutting down and booting a system, see "What's New in Shutting Down and Booting a System" on page 175.

For overview information and instructions on administering boot loaders and modifying Solaris boot behavior, see Chapter 11, "Modifying Solaris Boot Behavior (Tasks)."

For information about managing boot services through the Service Management Facility (SMF), see "SMF and Booting" on page 327.

Fundamentals of the Solaris Boot Design

The Solaris boot design, for both the SPARC and x86 platforms, includes the following characteristics:

- **Use of a boot archive**

  The boot archive is a ramdisk image that contains all of the files that are required for booting a system. When you install the Solaris OS, two boot archives are created, one primary archive and one failsafe archive. For more information, see "Implementation of the Boot Archives on Solaris SPARC" on page 191.

  The `bootadm` command has also been modified for use on the SPARC platform. This command functions the same way that it does on the Solaris x86 platform. The `bootadm` command handles the details of archive update and verification automatically. During an installation or system upgrade, the `bootadm` command creates the initial boot archive. During the process of a normal system shutdown, the shutdown process checks the boot archive contents against the root file system. If there are any inconsistencies, the system rebuilds the boot archive to ensure that on reboot, the boot archive and root (`/`) file system are synchronized. You can also use the `bootadm` command to manually update the boot archives. See "Using the `bootadm` Command to Manage the Boot Archives" on page 299.

  Note – Some options of the `bootadm` command cannot be used on SPARC based systems.

  For more information, see the `bootadm(1M)` and `boot(1M)` man pages.

- **Use of a ramdisk image as the root file system during installation and failsafe operations**

  This process is now the same on the Solaris SPARC and Solaris x86 platforms. The ramdisk image is derived from the boot archive and is then transferred to the system from the boot device.

  Note – On the SPARC platform, the OpenBoot™ PROM continues to be used to access the boot device and to transfer the boot archive to the system's memory. Conversely, on the x86 platform, the system is initially controlled by the BIOS. The BIOS is used to initiate a transfer of the boot archive from a network device or to run a boot loader. In the Solaris OS, the x86 boot loader that is used to transfer the boot archive from disk is GRUB. See "x86: Boot Processes" on page 309.

  In the case of a software installation, the ramdisk image is the root file system that is used for the entire installation process. Using the ramdisk image for this purpose eliminates the need to boot the system from removable media. The ramdisk file system type can be either a High Sierra File System (HSFS) or UFS.

Understanding the New Solaris SPARC Boot Architecture

The boot processes on the Solaris SPARC platform have been redesigned and improved to increase commonality with the Solaris x86 boot experience. The new Solaris SPARC boot design enables the addition of new features, for example new file system types, without necessitating any changes to multiple portions of the boot chain. Changes also include the implementation of boot phase independence.

Highlights of these improvements include:

- Commonality in boot processes on the Solaris SPARC and x86 platforms
- Commonality in the network boot experience
- Boot architecture flexibility that enables booting a system from different file system types more easily

The following four boot phases are now independent of each other:

1. **Open Boot PROM (OBP) phase**

   The OBP phase of the boot process on the Solaris SPARC platform is unchanged.

   For disk devices, the firmware driver usually uses the OBP label package's *load* method, which parses the VTOC label at the beginning of the disk to locate the specified partition. Sectors 1-15 of the partition are then read into the system's memory. This area is commonly called the boot block and usually contains a file system reader.
2. **Booter phase**

   During this phase the boot archive is read and executed. Note that this is the only phase of the boot process that requires knowledge of the boot file system format. In some instances, the boot archive might also be the installation miniroot. Protocols that are used for the transfer of the boot loader and the boot archive include local disk access, NFS, and HTTP.
3. **Ramdisk phase**

   The ramdisk is a boot archive that is comprised of kernel modules or an installation miniroot.

   The Solaris SPARC boot archive is identical to a Solaris x86 boot archive. The boot archive file system format is private. Therefore, knowledge of the file system type that is used during a system boot, for example an HSFS or a UFS file system, is not required by the booter or the kernel. The ramdisk extracts the kernel image from the boot archive and then executes it. To minimize the size of the ramdisk, in particular, the installation miniroot that resides in the system's memory, the contents of the miniroot are compressed. This compression is performed on a per-file level and is implemented within the individual file system. The `/usr/sbin/fiocompress` utility is then used to compress the file and mark the file as compressed.

Note – This utility has a private interface to the file compression file system, `dcfs`.

4. **Kernel phase**

   The kernel phase is the final stage of the boot process. During this phase, the Solaris OS is initialized and a minimal root file system is mounted on the ramdisk that was constructed from the boot archive. If the boot archive is the installation miniroot, the OS continues executing the installation process. Otherwise, the ramdisk contains a set of kernel files and drivers that is sufficient to mount the root file system on the specified root device.

   The kernel then extracts the remainder of the primary modules from the boot archive, initializes itself, mounts the real root file system, then discards the boot archive.

Packing and Unpacking the Miniroot

The ramdisk-based miniroot is packed and unpacked by the `root_archive` command. Note that only SPARC based systems that support the new boot architecture have the ability to pack and unpack a compressed version of the miniroot.

Caution – The Solaris Express version of the `root_archive` tool is not compatible with the Solaris 10 version of the tool. Therefore, ramdisk manipulation should only be performed on a system that is running the same Solaris release as the archives.

For more information about packing and unpacking the miniroot, see the `root_archive`(1M) man page.

Software Installation and Upgrades

To install or upgrade the Solaris OS, you need to boot the miniroot from either CD/DVD or from the network. In both instances, the miniroot's root file system is the ramdisk. This process enables you to eject the Solaris boot CD without having to reboot the system. Note that the boot archive contains the entire miniroot. The construction of the installation CD has been modified to use an HSFS boot block. The miniroot is then packed into a single UFS file that is loaded as the ramdisk. Note that the miniroot is used for all OS installation types.

Installation Memory Requirements

If you are running a Solaris Express Community release or an OpenSolaris release, the minimum memory requirements to install a system is 512 Mbytes of memory.

For the Solaris 10 release, the minimum memory requirements to install a system have been increased from 256 Mbytes of memory to minimum of 384 Mbytes of memory. This amount of memory enables a text-based installation *only*. To run the installation GUI program requires a minimum of 768 Mbytes of memory.

Changes to the Network Boot Server Setup Process

The network boot server setup process has been modified. The boot server now serves a bootstrap program, as well as the ramdisk, which is downloaded and booted as a single miniroot for all installations, whether booting from CD/DVD or performing a network installation by using NFS or HTTP. The administration of a network boot server for a network boot over both NFS or the `wanboot` program (HTTP) remains the same. However, the internal implementation of the network boot process has been modified as follows:

1. The boot server transfers a bootstrap in the form of a boot archive to the target system.
2. The target system unpacks the boot archive in a ramdisk.
3. The boot archive is then mounted as the initial read-only root device.

For more information about booting a SPARC based system, see “Booting a SPARC Based System (Task Map)” on page 233.

Support for Booting Multiple Solaris Kernels

On SPARC based systems, when you type `boot` at the `ok` prompt, the default boot device is automatically selected. An alternate boot device can be specified by changing the `boot-device` NVRAM variable. You can also specify an alternate boot device or alternate kernel (boot file) from the command line at boot time. See “SPARC: How to Boot a Solaris Kernel Other Than the Default Kernel” on page 239.

Implementation of the Boot Archives on Solaris SPARC

The Solaris boot archives, previously only available on the x86 platform, are now an integral part of the Solaris SPARC boot architecture.

The `bootadm` command has been modified for use on the Solaris SPARC platform. This command functions the same as it does on the Solaris x86 platform. The `bootadm` command handles the details of archive update and verification. On the x86 platform the `bootadm` command updates the GRUB menu during an installation or system upgrade. You can also use the `bootadm` command to manually manage the boot archives.

The boot archive service is managed by the Service Management Facility (SMF). The service instance for the boot archive is `svc:/system/boot-archive:default`. To enable, disable, or refresh this service use the `svcadm` command. For information about managing services by using SMF, see Chapter 16, “Managing Services (Overview).”

On supported Solaris releases, for both SPARC and x86 based systems, there are two kinds of boot archives:

- Primary boot archive
- Failsafe boot archive

Note – On x86 based systems, when you install the Solaris OS, two primary boot archives are created, one 32-bit boot archive and one 64-bit boot archive.

The files that are included in the Solaris SPARC boot archives are located in the `/platform` directory.

The contents of the `/platform` directory is divided into two groups of files:

- Files that are required for a `sun4u` boot archive
- Files that are required for `sun4v` boot archive

For information about managing the boot archives, see “Managing the Solaris Boot Archives (Task Map)” on page 293.

x86: Administering the GRUB Bootloader

The open source GRand Unified Bootloader (GRUB) is the default boot loader on x86 based systems. GRUB is responsible for loading a boot archive into the system's memory. A boot archive is a collection of critical files that is needed during system startup before the root file system is mounted. The boot archive is the interface that is used to boot the Solaris OS. You can find more information about GRUB at `http://www.gnu.org/software/grub/grub.html`. See also the `grub`(5) man page.

How GRUB Based Booting Works

After an x86 based system is powered on, the Basic Input/Output System (BIOS) initializes the CPU, the memory, and the platform hardware. When the initialization phase has completed, the BIOS loads the boot loader from the configured boot device and then transfers control of the system to the boot loader. The *boot loader* is the first software program that runs after you turn on a system. This program starts the boot process.

GRUB implements a menu interface that includes boot options that are predefined in a configuration file called the `menu.lst` file. GRUB also has a command-line interface that is accessible from the GUI menu interface that can be used to perform various boot functions, including modifying default boot behavior. In the Solaris OS, the GRUB implementation is compliant with the Multiboot Specification, which is described in detail at `http://www.gnu.org/software/grub/grub.html`.

Because the Solaris kernel is fully compliant with the Multiboot Specification, you can boot x86 based systems by using GRUB. With GRUB, you can boot various operating systems that are installed on a single x86 based system. For example, you can individually boot the Solaris OS, Linux, or Windows by selecting the boot entry in the GRUB menu at boot time or by configuring the `menu.lst` file to boot a specific OS by default.

Because GRUB is intuitive about file systems and kernel executable formats, you can load an operating system without recording the physical position of the kernel on the disk. With GRUB-based booting, the kernel is loaded by specifying its file name, and the drive and the partition where the kernel resides. For more information see "Naming Conventions That Are Used for Configuring GRUB" on page 312.

For step-by-step instructions on booting a system with GRUB, see "Booting an x86 Based System by Using GRUB (Task Map)" on page 252.

See also the following man pages:

- `boot`(1M)
- `bootadm`(1M)
- `grub`(5)
- `installgrub`(1M)

GRUB Support for New `findroot` Command

GRUB support for a new `findroot` command has been implemented in this Solaris release. The `findroot` command, which functions similarly to the `root` command that was previously used by GRUB, has enhanced capabilities for discovering a targeted disk, regardless of the boot device. The `findroot` command also supports booting from a ZFS root file system.

The most common format for the `menu.lst` entry for this command is:

```
findroot (rootfs0,0,a)
kernel$ /platform/i86pc/kernel/$ISADIR/unix
module$ /platform/i86pc/$ISADIR/boot_archive
```

For more information, see "x86: Implementation of the `findroot` Command" on page 228.

For GRUB reference information, see Chapter 15, "x86: GRUB Based Booting (Reference)."

x86: Introducing Fast Reboot

This feature is available, starting with build 100 of the Solaris Express Community Edition release, and in the OpenSolaris 2008.11 OS.

Fast Reboot implements an in-kernel boot loader that loads the kernel into memory, then switches to that kernel, thus enabling the reboot process to occur within seconds. With Fast Reboot, you can reboot to a new kernel without experiencing the long delays that can be imposed by the BIOS and boot loader. The ability to fast reboot a system drastically reduces down time and improves efficiency.

Note – Fast Reboot is currently not available on the SPARC platform.

The following sections provide a general overview of the Fast Reboot feature. For task related information, see "Using Fast Reboot on the x86 Platform (Task Map)" on page 268.

Modifications to the `reboot` Command to Support Fast Reboot

To support Fast Reboot, the `reboot` command has been modified to include two new options:

`-f`	Initiates the fast reboot process, when used with the `reboot` command.
`-e`	Initiates a fast reboot to an alternate boot environment (BE), when used in conjunction with the `-f` option,

---

Note – The `-e` option cannot be used to fast reboot to an alternate BE in the OpenSolaris 2008.11 release. For instructions on initiating a fast reboot to an alternate BE in this release, see "x86: Initiating a Fast Reboot to an Alternate Boot Environment in the OpenSolaris 2008.11 OS" on page 273.

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

Implementation of the `quiesce` Function

The system's capability to bypass the firmware when booting a new OS image has dependencies on device drivers' implementation of a new device operation entry point, `quiesce`. On supported drivers, this implementation `quiesces` a device, so that at completion of the function, the driver no longer generates interrupts or access memory. This implementation also resets the device to a hardware state, from which the device can be correctly configured by the driver's attach routine, without a power cycle of the system or being configured by the firmware. For more information about this functionality, see the quiesce(9E) and dev_ops(9S) man pages.

Note – Not all device drivers implement the quiesce function. For troubleshooting instructions, see "x86: Troubleshooting Conditions That Might Prevent Fast Reboot From Working" on page 275.

New uadmin Function

Other changes that support Fast Reboot on the x86 platform include the new uadmin function, AD_FASTREBOOT. This function resets the system, enabling the reboot command to bypass the BIOS and the boot loader phases.

Note – The uadmin 2 8 command has limited functionality. If the command is used to facilitate a fast reboot of the system, neither the boot archive, nor the menu.lst file are updated. For this reason, the reboot -f command is the preferred method for initiating a fast reboot.

For more information, see the uadmin(2)man page.

Booting From a ZFS Root File System

Support for booting a system from a ZFS root file system has been added to the Solaris OS. The Solaris installation software also includes support for system upgrades and patching of systems with ZFS roots. Booting, system operations, and installation procedures have been modified to support this change. Changes to booting include the implementation of boot loaders the SPARC boot architecture to increase commonality with the Solaris x86 boot architecture. This implementation is described in the following sections.

For more information about ZFS, including a complete list of terms, see "ZFS Terminology" in *Solaris ZFS Administration Guide*.

Solaris Installation Requirements for ZFS

Before performing a new installation of the Solaris software or using Solaris Live Upgrade to migrate a UFS root file system to a ZFS root file system, make sure the following requirements are met:

- **Solaris release information:**

  The ability to install and boot from a ZFS root file system is available in the Solaris 10 11/06 release. To perform a Solaris Live Upgrade operation to migrate to a ZFS root file system, you must have installed or upgraded to the Solaris 10 11/06 release.
- **ZFS storage pool space requirements:**

  The minimum amount of available pool space that is required for a bootable ZFS root file system is larger than for a bootable UFS root file system because swap and dump devices are not shared in a ZFS root environment.

  If you select a ZFS root file system during an initial software installation, or if you use Solaris Live Upgrade to migrate from a UFS root file system to a ZFS root file system, a swap area is created on a ZFS volume in the ZFS root pool. The default swap area is sized at 1/2 the size of physical memory, but no more than 2 Gbytes. A ZFS volume is also created for the dump device. The dump device is sized at 1/2 the size of physical memory, but no more than 2 Gbytes. Currently, the swap area and the dump device must reside on separate ZFS volumes.

  For more information, see "ZFS Support for Swap and Dump Devices" in *Solaris ZFS Administration Guide*.

How Booting From a ZFS Root File System Works

Booting from a ZFS root file system works differently than booting from a UFS file system. Because ZFS applies several new concepts for installation and booting, some basic administrative practices for booting a system have changed. The most significant difference between booting from a ZFS root file system and booting from a UFS root file system is that with ZFS a device identifier does *not* uniquely identify a root file system, and thus a BE. With ZFS, a device identifier uniquely identifies a *storage pool*. A storage pool can contain multiple bootable datasets (root file systems). Therefore, in addition to specifying a boot device, a root file system within the pool that was identified by the boot device must also be specified.

On an x86 based system, if the boot device that is identified by GRUB contains a ZFS storage pool, the `menu.lst` file that is used to create the GRUB menu is located in the dataset at the root of that pool's dataset hierachy. This dataset has the same name as the pool. There is one such dataset in each pool.

A *default bootable dataset* is the bootable dataset for the pool that is mounted at boot time and is defined by the root pool's `bootfs` property. When a device in a root pool is booted, the dataset that is specified by this property is then mounted as the root file system.

The new `bootfs` pool property is a mechanism that is used by the system to specify the default bootable dataset for a given pool. When a device in a root pool is booted, the dataset that is mounted by default as the root file system is the one that is identified by the `bootfs` pool property.

On a SPARC based system, the default `bootfs` pool property is overridden by using the new `-Z` *dataset* option of the `boot` command.

On an x86 based system, the default `bootfs` pool property is overridden by selecting an alternate boot environment in the GRUB menu at boot time.

SPARC: Boot Options That Support Booting From a ZFS Root File System

On the SPARC platform, the following two boot options are new:

- The `-L` option, which is used to print a list of all the available BEs on a system.

  ```
  ok boot -L
  ```

  Note – The `-L` option is run from the `ok` prompt. This option only presents the list of available BEs on the system. To boot the system, use the`- Z` boot option.

- The `-Z` option of the `boot` command enables you to specify a bootable dataset other than the default dataset that is specified by the `bootfs` pool property.

  ```
  ok boot -Z dataset
  ```

The list of BEs that are displayed when you use the `-L` option on a device that has a ZFS boot loader reflect the `menu.lst` entries that are available on that particular system. Along with the list of available BEs, instructions for selecting a BE and using the `-Z` option to boot the system are also provided. The dataset specified by the `bootfs` value for the menu item is used for all subsequent files that are read by the booter, for example, the boot archive and various configuration files that are located in the `/etc` directory. This dataset is then mounted as the root file system.

For step-by-step instructions, see "Booting From a ZFS Root File System on a SPARC Based System" on page 241.

x86: Boot Options That Support Booting From a ZFS Root File System

On the x86 platform, a new GRUB keyword, `$ZFS-BOOTFS` has been introduced. When booting an x86 based system, if the root file system that corresponds with the GRUB menu entry is a ZFS dataset, the GRUB menu entry contains the `-B` option with the `$ZFS-BOOTFS` token by default. If you install or upgrade your system with a Solaris release that supports a ZFS boot loader, the GRUB `menu.lst` file is updated with this information automatically. The default bootable dataset is identified by the `bootfs` property.

On x86 based systems that are running a Solaris release that supports a ZFS boot loader, this information is included in the GRUB menu.

The following is an example of a default `menu.lst` file for a GRUB implementation that supports a ZFS boot loader:

```
title Solaris 11 s10x_90 X86
findroot (pool_rpool,0,a)
kernel$ /platform/i86pc/kernel/$ISADIR/unix -B $ZFS-BOOTFS
module$ /platform/i86pc/$ISADIR/boot_archive

title Solaris 11 failsafe
findroot (pool_rpool,0,a)
kernel /boot/platform/i86pc/kernel/unix -s -B console=ttyb
module /boot/x86.miniroot-safe
```

This example shows a `menu.lst` file that has been manually edited to include an alternate bootable dataset entry:

```
title Solaris-alternate-dataset
bootfs myrootpool/bootenv-alt
kernel$ /platform/i86pc/kernel/$ISADIR/unix -B $ZFS-BOOTFS
module$ /platform/i86pc/$ISADIR/boot_archive
```

For step-by-step instructions on booting a system from ZFS, see “Booting From a ZFS Root File System on an x86 Based System” on page 259.

◆ ◆ ◆ CHAPTER 10

Shutting Down a System (Tasks)

This chapter describes the procedures for shutting down systems. This is a list of the step-by-step instructions in this chapter.

This is a list of the overview information in this chapter.

- "System Shutdown Commands" on page 200
- "User Notification of System Down Time" on page 201
- "Turning Off Power to All Devices" on page 208

For overview information about system run levels, see Chapter 16, "Managing Services (Overview)."

For information on the procedures associated with run levels and boot files, see "Shutting Down the System (Task Map)" on page 199.

Shutting Down the System (Task Map)

Task	Description	For Instructions
Determine who is logged in to a system.	Use the `who` command to determine who is logged in to a system.	"How to Determine Who Is Logged in to a System" on page 202
Shut down a server.	Use the `shutdown` command with the appropriate options to shut down a server.	"How to Shut Down a Server" on page 202
Shut down a stand-alone system.	Use the `init` command and indicate the appropriate run-level to shut down a stand-alone system.	"How to Shut Down a Stand-Alone System" on page 206

Task	Description	For Instructions
Turn off power to all devices.	Powering down a system includes the following devices: ■ CPU ■ Monitor ■ External devices, such as disks, tapes, and printers	"How to Turn Off Power to All Devices" on page 208

Shutting Down the System

Solaris software is designed to run continuously so that the electronic mail and network software can work correctly. However, some system administration tasks and emergency situations require that the system is shut down to a level where it is safe to remove power. In some cases, the system needs to be brought to an intermediate level, where not all system services are available.

Such cases include the following:

- Adding or removing hardware
- Preparing for an expected power outage
- Performing file system maintenance, such as a backup

For a complete list of system administration tasks that require a system shutdown, see Chapter 9, "Shutting Down and Booting a System (Overview)."

For information on using your system's power management features, see the `pmconfig`(1M) man page.

System Shutdown Commands

The use of the `init` and `shutdown` commands are the primary ways to shut down a system. Both commands perform a *clean shutdown* of the system. As such, all file system changes are written to the disk, and all system services, processes, and the operating system are terminated normally.

The use of a system's Stop key sequence or turning a system off and then on are not clean shutdowns because system services are terminated abruptly. However, sometimes these actions are needed in emergency situations. For instructions on system recovery techniques, see Chapter 12, "Booting a Solaris System (Tasks)," andChapter 14, "Managing the Solaris Boot Archives (Tasks)."

Note – On x86 systems that are running at least the Solaris 10 6/06 release, pressing and releasing the power button initiates a clean system shutdown. This method is equivalent to using the `init 5` command.

The following table describes the various shutdown commands and provides recommendations for using them.

TABLE 10–1 Shutdown Commands

Command	Description	When To Use
`shutdown`	An executable shell script that calls the `init` program to shut down the system. The system is brought to run level S by default.	Recommended for servers operating at run level 3 because users are notified of the impending shutdown. Also notified are the systems that are mounting resources from the server that is being shut down.
`init`	An executable that kills all active processes and synchronizes the disks before changing run levels.	Recommended for stand-alone systems when other users will not be affected. Provides a faster system shutdown because users are not notified of the impending shutdown.
`reboot`	An executable that synchronizes the disks and passes boot instructions to the `uadmin` system call. In turn, this system call stops the processor.	The `init` command is the preferred method.
`halt`, `poweroff`	An executable that synchronizes the disks and stops the processor.	Not recommended because it doesn't shutdown all processes, and unmount any remaining file systems. Stopping the services, without doing a clean shutdown, should only be done in an emergency or if most of the services are already stopped.

User Notification of System Down Time

When the `shutdown` command is initiated, a warning followed by a final shutdown message is broadcast to all users who are currently logged in to the system and all systems that are mounting resources from the affected system.

For this reason, the `shutdown` command is preferred instead of the `init` command when you need to shut down a server. When you use either command, you might want to give users more notice by sending them a mail message about any scheduled system shutdown.

Use the `who` command to determine which users on the system need to be notified. This command is also useful for determining a system's current run level. For more information, see “Determining a System's Run Level” on page 330 and the `who(1)` man page.

▼ How to Determine Who Is Logged in to a System

1. **Log into the system to be shut down.**

2. **Display all users who are logged in to the system.**

   ```
   $ who
   ```

Example 10–1 Determining Who Is Logged in to a System

The following example shows how to display who is logged in to the system.

```
$ who
holly       console      May  7 07:30
kryten      pts/0        May  7 07:35   (starlite)
lister      pts/1        May  7 07:40   (bluemidget)
```

- Data in the first column identifies the user name of the logged-in user
- Data in the second column identifies the terminal line of the logged-in user
- Data in the third column identifies the date and time that the user logged in
- Data in the forth column, if present, identifies the host name if a user is logged in from a remote system

▼ How to Shut Down a Server

1. **Become superuser or assume an equivalent role.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2. **Find out if users are logged in to the system.**

   ```
   # who
   ```

   A list of all logged-in users is displayed. You might want to send mail or broadcast a message to let users know that the system is being shut down.

3. **Shut down the system.**

   ```
   # shutdown -iinit-level -ggrace-period -y
   ```

   -i*init-level* Brings the system to an init level that is different from the default of S. The choices are 0, 1, 2, 5, and 6.

   Run levels 0 and 5 are reserved states for shutting the system down. Run level 6 reboots the system. Run level 2 is available as a multi-user operating state.

`-g`*grace-period*	Indicates a time (in seconds) before the system is shut down. The default is 60 seconds.
`-y`	Continues to shut down the system without intervention. Otherwise, you are prompted to continue the shutdown process after 60 seconds.

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

4 **If you are asked for confirmation, type `y`.**

```
Do you want to continue? (y or n): y
```

If you used the `shutdown -y` command, you will not be prompted to continue.

5 **Type the superuser password, if prompted.**

```
Type Ctrl-d to proceed with normal startup,
(or give root password for system maintenance): xxxxxx
```

6 **After you have finished the system administration tasks, press Control-D to return to the default system run level.**

7 **Use the following table to verify that the system is at the run level that you specified in the `shutdown` command.**

Specified Run Level	SPARC Based System Prompt	x86 Based System Prompt
S (single-user level)	`#`	`#`
0 (power-down level)	`ok` or `>`	`Press any key to reboot`
Run level 3 (multiuser level with remote resources shared)	*hostname* `console login:`	*hostname* `console login:`

Example 10–2 SPARC: Bringing a Server to Run Level S

In the following example, the `shutdown` command is used to bring a SPARC based system to run level S (single-user level) in three minutes.

```
# who
root   console      Jun 14 15:49    (:0)

# shutdown -g180 -y

Shutdown started.    Mon Jun 14 15:46:16 MDT 2004

Broadcast Message from root (pts/4) on venus Mon Jun 14 15:46:16...
The system venus will be shut down in 3 minutes .
.
```

```
.
Broadcast Message from root (pts/4) on venus Mon Jun 14 15:46:16...
The system venus will be shut down in 30 seconds .
.
.
INIT: New run level: S
The system is coming down for administration.  Please wait.
Unmounting remote filesystems: /vol nfs done.
Shutting down Solaris Management Console server on port 898.
Print services stopped.
Jun 14 15:49:00 venus syslogd: going down on signal 15
Killing user processes: done.

Requesting System Maintenance Mode
SINGLE USER MODE

Root password for system maintenance (control-d to bypass): xxxxxx
single-user privilege assigned to /dev/console.
Entering System Maintenance Mode
#
```

Example 10–3 SPARC: Bringing a Server to Run Level 0

In the following example, the shutdown command is used to bring a SPARC based system to run level 0 in 5 minutes without requiring additional confirmation.

```
# who
root       console       Jun 17 12:39
userabc         pts/4        Jun 17 12:39   (:0.0)
# shutdown -i0 -g300 -y
Shutdown started.    Thu Jun 17 12:40:25 MST 2004

Broadcast Message from root (console) on pretend Thu Jun 17 12:40:25...
The system pretend will be shut down in 5 minutes
.
.
.
Changing to init state 0 - please wait
#
INIT: New run level: 0
The system is coming down.  Please wait.
System services are now being stopped.
.
.
.
The system is down.
syncing file systems... done
Program terminated
```

```
Type  help  for more information
ok
```

If you are bringing the system to run level 0 to turn off power to all devices, see "How to Turn Off Power to All Devices" on page 208.

Example 10–4 SPARC: Rebooting a Server to Run Level 3

In the following example, the `shutdown` command is used to reboot a SPARC based system to run level 3 in two minutes. No additional confirmation is required.

```
# who
root           console      Jun 14 15:49    (:0)
userabc    pts/4        Jun 14 15:46    (:0.0)
# shutdown -i6 -g120 -y
Shutdown started.    Mon Jun 14 15:46:16 MDT 2004

Broadcast Message from root (pts/4) on venus Mon Jun 14 15:46:16...
The system venus will be shut down in 2 minutes


Changing to init state 6 - please wait
#
INIT: New run level: 6
The system is coming down.  Please wait.
.
.
.
The system is down.
syncing file systems... done
rebooting...
.
.
.
venus console login:
```

See Also Regardless of why you shut down a system, you'll probably want to return to run level 3 where all file resources are available and users can log in. For instructions on bringing a system back to a multiuser level, see Chapter 12, "Booting a Solaris System (Tasks)."

▼ How to Shut Down a Stand-Alone System

Use this procedure when you need to shut down a stand-alone system.

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 **Shut down the system.**

```
# init 5
```

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

- **Alternately, you can use the `uadmin` command to shut down the system.**

  ```
  # uadmin 2 0
  ```

- **If you have an x86 based system that is running at least the Solaris 10 6/06 release, you can press and release the power button to initiate a clean system shutdown and turn off the system. This functionality is equivalent to using the `init 5` command to shut down a system. For more information, see "What's New in Shutting Down and Booting a System" on page 175.**

3 **Use the following table to verify that the system is at the run level that you specified in the `init` command.**

Specified Run Level	SPARC Based System Prompt	x86 Based System Prompt
S (single-user level)	`#`	`#`
2 (multiuser level)	`#`	`#`
0 (power-down level)	`ok` or `>`	`Press any key to reboot`
3 (multiuser level with NFS resources shared)	*hostname* `console login:`	*hostname* `console login:`

Example 10–5 Using the `uadmin` command to Shut Down a System

```
# uadmin 2 0
syncing file systems... done
Program terminated
```

Example 10–6 Bringing a Stand-Alone System to Run Level 0

In this example, the `init` command is used to bring an x86 based stand-alone system to the level where it is safe to turn off power.

```
# init 0
#
INIT: New run level: 0
The system is coming down.  Please wait.
.

.

.
The system is down.
syncing file systems... [11] [10] [3] done
Press any key to reboot
```

If you are bringing the system to run level 0 to turn off power to all devices, see "How to Turn Off Power to All Devices" on page 208.

Example 10–7 SPARC: Bringing a Stand-Alone System to Run Level S

In this example, the `init` command is used to bring a SPARC based stand-alone system to run level S (single-user level).

```
# init s
#
INIT: New run level: S
The system is coming down for administration.  Please wait.
Unmounting remote filesystems: /vol nfs done.
Print services stopped.
syslogd: going down on signal 15
Killing user processes: done.

SINGLE USER MODE

Root password for system maintenance (control-d to bypass): xxxxxx
single-user privilege assigned to /dev/console.
Entering System Maintenance Mode
#
```

See Also Regardless of why you shut down the system, you'll probably want to return to run level 3 where all file resources are available and users can log in. For instructions on bringing a system back to a multiuser level, see Chapter 12, "Booting a Solaris System (Tasks)."

Turning Off Power to All Devices

You need to turn off power to all system devices when you do the following:

- Replace or add hardware.
- Move the system from one location to another.
- Prepare for an expected power outage or natural disaster such as an approaching electrical storm.

Turn the power off for system devices, including the CPU, the monitor, and external devices such as disks, tapes, and printers.

Before you turn off power to all system devices, you should shut down the system cleanly, as described in the preceding sections.

▼ How to Turn Off Power to All Devices

1. **Select one of the following methods to shut down the system:**
   - **If you are shutting down a server, see "How to Shut Down a Server" on page 202.**
   - **If you are shutting down a stand-alone system, see "How to Shut Down a Stand-Alone System" on page 206.**
2. **Turn off the power to all devices after the system is shutdown. If necessary, also unplug the power cables.**
3. **After power can be restored, use the following steps to turn on the system and devices.**
   a. **Plug in the power cables.**
   b. **Turn on the monitor.**
   c. **Turn on disk drives, tape drives, and printers.**
   d. **Turn on the CPU.**
   The system is brought to run level 3.

◆ ◆ ◆ CHAPTER 11

Modifying Solaris Boot Behavior (Tasks)

This chapter provides information about modifying boot behavior on Solaris systems.

The following is list of the information in this chapter:

- "Modifying Boot Behavior on SPARC Based Systems (Task Map)" on page 209
- "Modifying Solaris Boot Behavior on x86 Based Systems (Task Map)" on page 217

For what's new in booting and general overview information about the boot process, see Chapter 8, "Introduction to Shutting Down and Booting a System."

For step-by-step instructions on booting a Solaris system, see Chapter 12, "Booting a Solaris System (Tasks)."

Modifying Boot Behavior on SPARC Based Systems (Task Map)

Task	Description	For Instructions
Identify the PROM revision number.	Use the banner command at the ok prompt to display the PROM revision number for a system.	"SPARC: How to Find the PROM Revision Number for a System" on page 210
Identify devices on the system that can be booted.	Before modifying boot behavior by using the boot PROM, identify the devices on the system.	"SPARC: How to Identify Devices on a System" on page 211
Display the current boot device.	Use this procedure to determine the current default boot device from which the system will boot.	"SPARC: How to Determine the Default Boot Device" on page 213

Task	Description	For Instructions
Change the default boot device.	To change the default boot device, use one of the following methods: ■ Change the `boot-device` parameter at the boot PROM. ■ Change the `boot-device` parameter by using the `eeprom` command.	"SPARC: How to Change the Default Boot Device by Using the Boot PROM" on page 213 "SPARC: How to Change the Default Boot Device by Using the `eeprom` Command" on page 215
Reset the system.	When you reset the system, the system runs diagnostic tests on the hardware, then reboots.	"SPARC: Resetting the System" on page 215
Change the default boot file.	To change the default kernel that the system boots, use one of the following methods: ■ Change the `boot-file` parameter by using the boot PROM. ■ Change the`boot-file` parameter by using the `eeprom` command.	"SPARC: How to Change the Default Kernel by Using the Boot PROM" on page 216 "SPARC: How to Change the Default Kernel by Using the `eeprom` Command" on page 216

SPARC: Using the Boot PROM

The boot PROM is used to boot a system. You might need to change the way the system boots. For example, you might want to reset the device to boot from or run hardware diagnostics before you bring the system to a multiuser level.

System administrators typically use the PROM level to boot a system. You can also change the default boot file and boot device at the PROM level.

If you need to perform any of the following tasks, you need to change the default boot device:

- Add a new drive to the system either permanently or temporarily
- Change the network boot strategy
- Temporarily boot a stand-alone system from the network

For a complete list of PROM commands, see `monitor(1M)` or `eeprom(1M)`.

▼ SPARC: How to Find the PROM Revision Number for a System

● **Display a system's PROM revision number by using the `banner` command.**

```
ok banner
Sun Ultra 5/10 UPA/PCI (UltraSPARC-IIi 333MHz), No Keyboard
OpenBoot 3.15, 128 MB memory installed, Serial #number.
Ethernet address number, Host ID: number.
```

Hardware configuration information, including the revision number of the PROM, is displayed. In this example, the PROM revision number is 3.15.

▼ SPARC: How to Identify Devices on a System

You might need to identify the devices on the system to determine what are the appropriate devices to boot from.

Before You Begin Before you can safely use the `probe` commands to determine what devices are attached to the system, you need to do the following:

- Change the PROM `auto-boot?` parameter to false.

  ```
  ok setenv auto-boot? false
  ```
- Issue the `reset-all` command to clear system registers.

  ```
  ok reset-all
  ```

You can view the `probe` commands that are available on your system by using the `sifting probe` command:

```
ok sifting probe
```

If you run the `probe` commands without clearing the system registers, the following message is displayed:

```
ok probe-scsi
This command may hang the system if a Stop-A or halt command
has been executed.  Please type reset-all to reset the system
before executing this command.
Do you wish to continue? (y/n) n
```

1 **Identify the devices on the system.**

```
ok probe-device
```

2 **(Optional) If you want the system to reboot after a power failure or after using the `reset` command, then reset the `auto-boot?` parameter to true.**

```
ok setenv auto-boot? true
auto-boot? =          true
```

3 **Boot the system to multiuser mode.**

```
ok reset-all
```

Example 11–1 SPARC: Identifying the Devices on a System

The following example shows how to identify the devices connected to an Ultra™ 10 system.

```
ok setenv auto-boot? false
auto-boot? =          false
ok reset-all
Resetting ...

Sun Ultra 5/10 UPA/PCI (UltraSPARC-IIi 333MHz), No Keyboard
OpenBoot 3.15, 128 MB memory installed, Serial #10933339.
Ethernet address 8:0:20:a6:d4:5b, Host ID: 80a6d45b.

ok probe-ide
  Device 0  ( Primary Master )
          ATA Model: ST34321A

  Device 1  ( Primary Slave )
         Not Present

  Device 2  ( Secondary Master )
         Removable ATAPI Model: CRD-8322B

  Device 3  ( Secondary Slave )
         Not Present

ok setenv auto-boot? true
auto-boot? =          true
```

Alternatively, you can use the `devalias` command to identify the device aliases and the associated paths of devices that *might* be connected to the system. For example:

```
ok devalias
screen                   /pci@1f,0/pci@1,1/SUNW,m64B@2
net                      /pci@1f,0/pci@1,1/network@1,1
cdrom                    /pci@1f,0/pci@1,1/ide@3/cdrom@2,0:f
disk                     /pci@1f,0/pci@1,1/ide@3/disk@0,0
disk3                    /pci@1f,0/pci@1,1/ide@3/disk@3,0
disk2                    /pci@1f,0/pci@1,1/ide@3/disk@2,0
disk1                    /pci@1f,0/pci@1,1/ide@3/disk@1,0
disk0                    /pci@1f,0/pci@1,1/ide@3/disk@0,0
ide                      /pci@1f,0/pci@1,1/ide@3
floppy                   /pci@1f,0/pci@1,1/ebus@1/fdthree
ttyb                     /pci@1f,0/pci@1,1/ebus@1/se:b
ttya                     /pci@1f,0/pci@1,1/ebus@1/se:a
keyboard!                /pci@1f,0/pci@1,1/ebus@1/su@14,3083f8:forcemode
keyboard                 /pci@1f,0/pci@1,1/ebus@1/su@14,3083f8
mouse                    /pci@1f,0/pci@1,1/ebus@1/su@14,3062f8
name                     aliases
```

▼ SPARC: How to Determine the Default Boot Device

1 **Bring the system to the `ok` PROM prompt.**

For more information, see "How to Shut Down a Stand-Alone System" on page 206.

2 **Use the `printenv` command to determine the default boot device.**

```
ok printenv boot-device
```

`boot-device`	Identifies the parameter for setting the device from which to boot.
device[*n*]	Identifies the `boot-device` value such as a disk or the network. The *n* can be specified as the *disk number*.

The default boot-device is displayed in a format that is similar to the following:

```
boot-device = /pci@1f,4000/scsi@3/disk@1,0:a
```

If the default boot-device is a network boot device, the output is similar to the following:

```
boot-device = /sbus@1f,0/SUNW,fas@e,8800000/sd@a,0:a \
/sbus@1f,0/SUNW,fas@e,8800000/sd@0,0:a disk net
```

▼ SPARC: How to Change the Default Boot Device by Using the Boot PROM

You might need to identify the devices on the system before you can change the default boot device to some other device. For information on identifying devices on the system, see "SPARC: How to Identify Devices on a System" on page 211.

1 **Change to run level 0.**

```
# init 0
```

The `ok` PROM prompt is displayed. For more information, see the`init(1M)` man page.

2 **Change the value of the `boot-device` parameter.**

```
ok setenv boot-device device[n]
```

Use one of the `probe` commands if you need help identifying the disk number.

3 **Verify that the default boot device has been changed.**

```
ok printenv boot-device
```

4 **Save the new `boot-device` value.**

```
ok reset-all
```

The new `boot-device` value is written to the PROM.

Example 11–2 SPARC: Changing the Default Boot Device

In this example, the default boot device is set to disk.

```
# init 0
#
INIT: New run level: 0
.
.
.
The system is down.
syncing file systems... done
Program terminated
ok setenv boot-device /pci@1f,4000/scsi@3/disk@1,0
boot-device =         /pci@1f,4000/scsi@3/disk@1,0
ok printenv boot-device
boot-device           /pci@1f,4000/scsi@3/disk@1,0
ok boot
Resetting ...

screen not found.
Can't open input device.
Keyboard not present.  Using ttya for input and output.

Sun Enterprise 220R (2 X UltraSPARC-II 450MHz), No Keyboard
OpenBoot 3.23, 1024 MB memory installed, Serial #13116682.
Ethernet address 8:0:20:c8:25:a, Host ID: 80c8250a.

Rebooting with command: boot disk1
Boot device: /pci@1f,4000/scsi@3/disk@1,0  File and args:
```

In this example, the default boot device is set to the network.

```
# init 0
#
INIT: New run level: 0
.
.
.
The system is down.
syncing file systems... done
Program terminated
```

```
ok setenv boot-device net
boot-device =         net
ok printenv boot-device
boot-device           net                    disk
ok reset
Sun Ultra 5/10 UPA/PCI (UltraSPARC-IIi 333MHz), No Keyboard
OpenBoot 3.15, 128 MB memory installed, Serial #number.
Ethernet address number, Host ID: number.



Boot device: net  File and args:
.
.
.
pluto console login:
```

▼ SPARC: How to Change the Default Boot Device by Using the `eeprom` Command

1. **Become superuser or assume an equivalent role.**

2. **Specify the alternate kernel to boot.**

   ```
   # eeprom boot-device new-boot-device
   ```

3. **Verify that the new parameter has been set.**

   ```
   # eeprom boot-device
   ```

   The output should display the new `eeprom` value for the `boot-device` parameter.

SPARC: Resetting the System

Run the following command from the `ok` prompt:

```
ok reset-all
```

The self-test program, which runs diagnostic tests on the hardware, is executed. Then, if the `auto-boot?` parameter is set to true, the system is rebooted.

▼ SPARC: How to Change the Default Kernel by Using the Boot PROM

1. **Change to run level 0.**

   ```
   # init 0
   ```

   The ok PROM prompt is displayed. For more information, see theinit(1M) man page.

2. **Set the `boot-file` property to an alternate kernel.**

   ```
   ok setenv boot-file boot-file
   ```

3. **Verify that the default boot device has been changed.**

   ```
   ok printenv boot-file
   ```

4. **Save the new `boot-file` value.**

   ```
   ok reset-all
   ```

   The new `boot-file` value is written to the PROM.

▼ SPARC: How to Change the Default Kernel by Using the `eeprom` Command

1. **Become superuser or assume an equivalent role.**

2. **Specify the alternate kernel to boot.**

   ```
   # eeprom boot-file new boot-file
   ```

   For example:

   ```
   # eeprom boot-file=kernel.name/sparcv9/unix
   ```

3. **Verify that the new parameter has been set.**

   ```
   # eeprom boot-file
   ```

   The output should display the new `eeprom` value for the specified parameter.

Modifying Solaris Boot Behavior on x86 Based Systems (Task Map)

TABLE 11–1 x86: Modifying Boot Behavior: Task Map

Task	Description	For Information
Set boot file parameters by using the `eeprom` command.	Modify boot behavior on an x86 based system by using the `eeprom` command. Boot options that are set by using the `eeprom` command persist over a system reboot, unless these options are overridden by modifying kernel behavior in the GRUB menu at boot time.	"x86: How to Modify Boot Behavior by Using the `eeprom` Command" on page 218
Modify boot behavior by editing the GRUB menu at boot time.	Modify boot behavior by editing GRUB menu at boot time. Boot options that are specified by modifying the boot behavior in the GRUB menu persist only until the next system reboot.	"x86: How to Modify Boot Behavior by Editing the GRUB Menu at Boot Time" on page 221
Modify boot behavior by manually editing the `menu.lst` file.	Modify boot behavior by editing the `menu.lst` configuration file to add new OS entries or redirect the console. Changes you make to the file persist over system reboots.	"x86: How to Modify Boot Behavior by Editing the `menu.lst` File" on page 224
Modify the `menu.lst` file to include entries that support the `findroot` command.	Additional menu entries that use the `findroot` command can be added to the `menu.lst` file menu after an installation or upgrade.	"x86: How to Add GRUB Menu Entries That Use the `findroot` Command" on page 230

Modifying Boot Behavior on x86 Based Systems

The primary methods for modifying boot behavior on an x86 based system are as follows:

- By using the `eeprom` command.

  The `eeprom` command is used to assign a different value to a standard set of properties. These values, which are equivalent to the SPARC OpenBoot PROM NVRAM variables, are stored either in the `/boot/solaris/bootenv.rc` file or in the `menu.lst` file. Changes that are made to boot behavior by using the `eeprom` command persist over each system reboot and are preserved during a software upgrade. See the `eeprom`(1M) man page for more information.

Caution – If you directly edit the `menu.lst` file, certain boot properties (boot-file, boot-args, and console) may not be changed at a later time by using the `eeprom` command.

- By editing the GRUB menu at boot time.

  Changes that are made by modifying the GRUB kernel behavior at boot time override options that you set by using the `eeprom` command. However, these changes only remain in effect until the next time you boot the system. See the `kernel`(1M) man page for more information.
- By manually editing the GRUB `menu.lst` file.

Caution – Any system generated changes made to `menu.lst` entries are changed or lost during a system upgrade. However, any new boot entries that you manually added remain after an upgrade. You can override `eeprom` settings by editing the GRUB menu at boot time or by editing the `menu.lst` file. Changes made by editing the GRUB menu at boot time do not persist. Changes that are made to `menu.lst` file persist over system reboots.

▼ x86: How to Modify Boot Behavior by Using the `eeprom` Command

You can display or set boot parameters by using the `eeprom` command. These parameters are found in the `/boot/solaris/bootenv.rc` file. Changes that are made by using the `eeprom` command persist over a system reboot. You can override `eeprom` settings by editing the GRUB menu at boot time to specify alternative boot behavior.

See "x86: How to Modify Boot Behavior by Editing the GRUB Menu at Boot Time" on page 221. For more information about changes to the `eeprom` command in this release, see the `eeprom`(1M) man page.

1. **Become superuser or assume an equivalent role.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2. **To change the specified parameter, type the `eeprom` command with the appropriate arguments .**

   ```
   # eeprom parameter=new-value
   ```

3. **Verify that the new parameter has been set.**

   ```
   # eeprom parameter
   ```

   The output should display the new `eeprom` value for the specified parameter.

Example 11–3 x86: Setting `boot-file` Parameters by Using the `eeprom` Command

This example shows how to manually specify that the system boot a 64-bit kernel. Note that the system must support 64-bit computing.

```
# eeprom boot-file=kernel/amd64/unix
```

This example shows how to manually boot a 32-bit kernel on a 64-bit capable system.

```
# eeprom boot-file=kernel/unix
```

This example shows how to restore the default auto detected boot behavior on a system.

```
# eeprom boot-file=""
```

x86: Modifying Boot Behavior by Editing the GRUB Menu at Boot Time

The following is an example of a GRUB main menu in a Solaris release that supports booting a system from a ZFS root file system. This menu is based on the contents of the `menu.lst` configuration file and includes menu entries for all of the bootable OS instances on the system. The first entry in the menu is the default, unless otherwise specified. To specify another boot entry as the default, add the `default=`*n* command to the `menu.lst` file, where *n* is a number, starting from 0 (the first boot entry).

```
GNU GRUB  version 0.95  (637K lower / 3144640K upper memory)
 +-------------------------------------------------------------------------+
be1)
be1 failsafe
be3
be3 failsafe
be2
be2 failsafe
+-------------------------------------------------------------------------+
      Use the ^ and v keys to select which entry is highlighted.
      Press enter to boot the selected OS, 'e' to edit the
      commands before booting, or 'c' for a command-line.
```

Note – The information that is contained in the `menu.lst` file varies, depending on the Solaris release and the installation method that was used.

To edit a boot entry in the GRUB menu, use the arrow keys to select the entry, then type `e`.

```
GNU GRUB  version 0.95  (637K lower / 3144640K upper memory)
 +-------------------------------------------------------------------------+
findroot (BE_be1,0,a)
bootfs rpool/ROOT/szboot_0508
kernel$ /platform/i86pc/kernel/$ISADIR/unix -B $ZFS-BOOTFS
module$ /platform/i86pc/$ISADIR/boot_archive
 +-------------------------------------------------------------------------+
      Use the ^ and v keys to select which entry is highlighted.
      Press enter to boot the selected OS, 'e' to edit the
      commands before booting, or 'c' for a command-line.
```

For instructions on editing the GRUB menu at boot time, see "x86: How to Modify Boot Behavior by Editing the GRUB Menu at Boot Time" on page 221.

Description of the GRUB Edit Menu

The following examples show the edit menu in the various GRUB implementations:

GRUB ZFS Support:

```
grub edit> kernel$ /platform/i86pc/kernel/$ISADIR/unix -B $ZFS-BOOTFS,prop=value [,prop=value...]][-asrvxk]
[-m smf-options] [-i altinit]
```

Note – When adding boot arguments on a system with ZFS support, any additional `-B` options should be added after the default `-B $ZFS-BOOTFS` argument.

GRUB hypervisor Support:

```
grub edit> module$ /platform/i86xpv/$ISADIR/unix /platform/i86xpv/$ISADIR/unix -B $ZFS-BOOTFS,\
prop=value [,prop=value...]][-asrvxk] [-m smf-options] [-i altinit]
```

GRUB UFS Support:

```
grub edit> kernel$ /platform/i86pc/kernel/$ISADIR/unix [-asrvxk]
            [-m smf-options] [-i altinit][-B prop=value [,prop=value...]]
```

Boot Arguments You Can Specify When Editing the GRUB Menu at Boot Time

The following list describes the boot arguments and options that can be specified by editing the GRUB menu at boot time:

unix	Specifies the kernel to boot.

`-a`	Prompts the user for configuration information.
`-s`	Boots the system in single-user mode.
`-r`	Specifies a reconfiguration boot. The system probes all attached hardware devices and then assigns nodes in the file system to represent only those devices that are actually found.
`-v`	Boots the system with verbose messages enabled.
`-x`	Does not boot in clustered mode.
`-k`	Boots the system with the kernel debugger enabled.
`-m` *smf-options*	Controls the boot behavior of the Service Management Facility (SMF). Included are two categories of options, recovery options and messages options.
`-i altinit`	Specifies an alternative executable as the primordial process. `altinit` is a valid path to an executable.
`-B` *prop=value [,prop=value]...*	Specifies kernel boot properties.

The following are various ways you can modify boot behavior in the GRUB menu by using the `-B` *prop=val* option:

`-B console=ttya`	Redirects the console to `ttya`.
`-B acpi-enum=off`	Disables Advanced Configuration and Power Interface (ACPI) enumeration of devices.
`-B console=ttya,acpi-enum=off`	Redirects the console to `ttya` and disables the ACPI enumeration of devices.
`-B acpi-user-options=0x2`	Disables ACPI entirely.

Note – When properties are specified by using the `eeprom` command *and* on the GRUB command line, the GRUB command takes precedence.

▼ x86: How to Modify Boot Behavior by Editing the GRUB Menu at Boot Time

When you modify the GRUB kernel behavior by editing the GRUB menu at boot time, the changes do not persist over a system reboot. Default boot behavior is restored the next time you boot the system.

1. **Reboot the system.**

   When the boot sequence begins, the GRUB main menu is displayed.

2. **Use the arrow keys to select the boot entry to edit, then type `e` to access the GRUB edit menu.**

3. **Use the arrow keys to select the `kernel` or `kernel$` line in this menu.**

4. **Type `e` to add boot arguments to the line.**

5. **Type any additional boot arguments that you want to specify.**

6. **Press Return to save your changes and return to the previous menu.**

   Note – Pressing the Escape key returns you to the GRUB main menu without saving your changes.

   ---

7. **To boot the system, type `b`.**

   Changes you make take affect when the system is booted.

Example 11–4 x86: Booting a 32-Bit Kernel on a 64-Bit Enabled System

To boot a 32-bit kernel on a 64-bit capable system, add the `kernel/unix` argument.

```
grub edit> kernel /platform/i86pc/multiboot kernel/unix
```

Example 11–5 x86: Redirecting the Serial Console

To redirect the serial console to `ttyb`, add the `-B console=ttyb` argument.

```
grub edit> kernel /platform/i86pc/multiboot -B console=ttyb
```

Alternatively, you can use `input-device/output-device` property, as shown in the following example:

```
grub edit> kernel /platform/i86pc/multiboot -B input-device=ttyb,output-device=ttyb
```

This example shows how you would override the serial line speed:

```
grub edit> kernel /platform/i86pc/multiboot -B ttyb-mode="115200,8,n,1,-"
```

Caution: In the preceding example, the property value contains commas, which is also a property separator. To avoid confusing the property parser, use double quotation marks around the entire property value.

x86: Modifying Boot Behavior by Editing the `menu.lst` File

The GRUB menu, which is based on the `menu.lst` configuration file, can be customized. When you install or upgrade the Solaris release, the `bootadm` command automatically updates the `menu.lst` file to reflect menu entries that are supported for that particular GRUB implementation. Any newly installed OS that is listed in this file is displayed as a boot entry in the GRUB menu when the system is rebooted. Note that when installing an operating system other than the Solaris OS, you will need to manually add the menu entry to the `menu.lst` file afterwards.

The following is an example of a typical GRUB main menu, based on the contents of the `menu.lst` file. The GRUB main menu consists of boot entries that are available, plus a failsafe archive.

```
GNU GRUB  version 0.95  (631K lower / 2095488K upper memory)
 +-------------------------------------------------------------------------+
 | Solaris 10.1 ... X86                                                    |
 | Solaris failsafe                                                        |
 |                                                                         |
 +-------------------------------------------------------------------------+
```

A configurable timeout is available to boot the default OS entry. The default OS boot entry that is booted is configurable through the `default` command. The Solaris installation software typically sets this command to boot one of the valid Solaris boot entries. To boot a different instance of the Solaris OS (if applicable), or to boot a different OS, use the arrow keys to highlight a different boot entry. Then press Enter to boot that entry. Note that if the `default` command is not set, the first boot entry in the GRUB menu is booted.

Only the *active* `menu.lst` file is used to boot the system. To modify the GRUB menu that is displayed when you boot the system, edit the active GRUB `menu.lst` file. Changing any other `menu.lst` file has no effect on the menu that is displayed when you boot the system To determine the location of the active `menu.lst` file, use the `list-menu` subcommand of the `bootadm` command. For more information about using the `bootadm` command, see "Using the `bootadm` Command to Manage the Boot Archives" on page 299.

For a complete description of the `menu.lst` file in each of the GRUB implementations in the Solaris OS, see "x86: Supported GRUB Implementations" on page 315.

▼ x86: How to Modify Boot Behavior by Editing the `menu.lst` File

You might need to modify the `menu.lst` file for one of the following reasons:

- To add new OS entries
- To add GRUB console redirection information

Before You Begin Because only the *active* GRUB `menu.lst` file is used to boot the system, make sure you edit the correct file. Changing any other GRUB `menu.lst` file has no effect on the menu that is displayed when you boot the system.

The location of the active `menu.lst` file varies, depending on whether you have a system with a UFS root or a ZFS root.

- For a UFS root, the active `menu.lst` file is `/boot/grub/menu.lst`.
- For a ZFS root, the active `menu.lst` file is /*pool-name*/`boot/grub/menu.lst`

You can determine the location of the active GRUB `menu.lst` file by using the `bootadm` command with the `list-menu` subcommand.

```
# bootadm list-menu
```

For more information about the `bootadm` command, see the `bootadm`(1M) man page.

1 **Become superuser or assume an equivalent role.**

2 **To add a new OS entry to the active `menu.lst` file, use a text editor to modify the file.**

The comments within the `menu.lst` file provide you with the necessary information for adding a new OS entry.

The following is an example of a `menu.lst` file on a system that is running a Solaris release with ZFS boot support. Boot entries in the `menu.lst` file vary, depending on the Solaris release you are running.

```
#---------- ADDED BY BOOTADM - DO NOT EDIT ----------
title Solaris 11 s10x_90 X86
findroot (pool_rpool,0,a)
kernel$ /platform/i86pc/kernel/$ISADIR/unix -B $ZFS-BOOTFS
module$ /platform/i86pc/$ISADIR/boot_archive
#---------------------END BOOTADM--------------------
```

Caution – Do not directly edit the original contents of the `menu.lst` file. To make changes to any of the OS entries in the file, edit the file manually, duplicating the existing content. Then, make the modifications to the duplicated content.

Also note when manually adding new user entries to the file, do not include guard comments that are reserved for use by the system, such as “Added by bootadm” or “Added by Live Upgrade”. Not using these comments for manually-added entries ensures that these entries remain intact during a software upgrade.

- **To specify a 32-bit Solaris kernel as the default kernel to boot, remove all instances of the `$ISADIR` keyword from the `menu.lst` file.**

  ```
  title Solaris Nevada snv_53 X86
  kernel$ /platform/i86pc/kernel/unix [*]
  module$ /platform/i86pc/boot_archive
  ```

  To specify a 32-bit kernel as the boot method, you must also change the line that begins with the `kernel` keyword in the Solaris failsafe entry:

  ```
  title Solaris failsafe
  kernel /boot/platform/i86pc/kernel/unix [-B *] -s [*]
  module /boot/x86.miniroot-safe
  ```

If you have added any additional entries, beyond the default entries, make equivalent changes manually.

The [`-B` *] and [*] flags must be preserved, if these flags exist in the original `menu.lst` file. Also, the failsafe entry should always have an `-s` flag.

3 **After adding the required information, save the file.**

Note that any changes you make to the file take effect at the next system reboot.

Tip – If you are running the Linux OS, and install the Solaris OS, the Linux OS entry is not displayed in the GRUB menu when the system is rebooted. Before installing the Solaris software, save a copy of the `menu.lst` file that contains the Linux information. After the installation, add the Linux information back to the newly-created `menu.lst` file in the Solaris partition.

Because changes you make to the `menu.lst` file are not directly related to the Solaris OS, you cannot make them by using the `eeprom` command. You must edit the file directly. Note that the Solaris software upgrade process preserves any changes that you make to the `menu.lst` file.

Caution – Solaris GRUB is capable of booting both the Linux OS and the Solaris OS. However, Linux GRUB is not capable of booting the Solaris OS.

Always ensure that one of the following conditions are met:

- The Solaris `fdisk` partition is active, that it has GRUB installed , and that the `menu.lst` file is the *active* GRUB menu
- That Solaris GRUB is installed to the Master Boot Record (MBR) and that it refers to a `menu.lst` in the Solaris `fdisk` partition.

For a detailed description of the GRUB `menu.lst` that pertains to each Solaris release, see "x86: Supported GRUB Implementations" on page 315.

Example 11–6 `menu.lst` File on a System With a ZFS Boot Loader

The following examples show what a `menu.lst` file looks like on a system that has a ZFS boot loader. By default, this system will boot from a ZFS root file system. Note that the contents of the file varies, depending on the installation type.

New installation or standard upgrade:

```
title Solaris 11 s10x_90 X86
findroot (pool_rpool,0,a)
kernel$ /platform/i86pc/kernel/$ISADIR/unix -B $ZFS-BOOTFS
module$ /platform/i86pc/$ISADIR/boot_archive

title Solaris 11 failsafe
findroot (pool_rpool,0,a)
kernel /boot/platform/i86pc/kernel/unix -B console=ttyb
module /boot/x86.miniroot-safe
```

Solaris Live Upgrade:

```
title be1
findroot (BE_be1,0,a)
bootfs rpool/ROOT/szboot_0508
kernel$ /platform/i86pc/kernel/$ISADIR/unix -B $ZFS-BOOTFS
module$ /platform/i86pc/$ISADIR/boot_archive

title be1 failsafe
findroot (BE_be1,0,a)
kernel /boot/platform/i86pc/kernel/unix -s -B console=ttyb
module /boot/x86.miniroot-safe
```

Example 11–7 `menu.lst` File on a System With a UFS Boot Loader

The following examples show what a `menu.lst` file looks like on a system that has a UFS root file system installed. By default, this system will boot from a UFS root file system.

New installation or standard upgrade:

```
title Solaris 11 s10x_90 X86
findroot (rootfs0,0,a)
kernel$ /platform/i86pc/kernel/$ISADIR/unix
module$ /platform/i86pc/$ISADIR/boot_archive

title Solaris 11 failsafe
findroot (rootfs0,0,a)
kernel /boot/platform/i86pc/kernel/unix -s -B console=ttyb
module /boot/x86.miniroot-safe
```

Solaris Live Upgrade:

```
title be1
findroot (BE_be1,0,a)
kernel$ /platform/i86pc/kernel/$ISADIR/unix
module$ /platform/i86pc/$ISADIR/boot_archive

title be1 failsafe
findroot (BE_be1,0,a)
kernel /boot/platform/i86pc/kernel/unix -s -B console=ttyb
module /boot/x86.miniroot-safe
```

Example 11–8 x86: `menu.lst` File With Hypervisor Support

The following examples are for a GRUB implementation that supports booting the Solaris OS as a dom0, with hypervisor support.

In this example, the `menu.lst` file has been modified to enable a 64-bit capable system to boot the Solaris OS as a dom0, in 32-bit mode. This modification includes removing all instances of `$ISADIR` from the file.

```
title 32-bit Solaris on xVM
kernel$ /boot/xen.gz
module$ /platform/i86xpv/kernel/unix /platform/i86xpv/kernel/unix -B $ZFS-BOOTFS [*]
module$ /platform/i86pc/boot_archive
```

In this example, the `menu.lst` file has been modified to direct the hypervisor to use the serial console. Note that the serial console is also shared by the Solaris OS dom0 console.

```
title Solaris on xVM
kernel$ /boot/$ISADIR/xen.gz console=com1 com1=9600,8n1
module$ /platform/i86xpv/kernel/$ISADIR/unix /platform/i86xpv/kernel/$ISADIR/unix \
-B $ZFS-BOOTFS console=hypervisor
module$ /platform/i86pc/$ISADIR/boot_archive
```

x86: Locating the Active GRUB `menu.lst` File

On systems that have a ZFS root, the active `menu.lst` file is typically located in /*pool-name*/`boot/grub/menu.lst`.

On systems that have a UFS root, the active `menu.lst` file is typically located in `/boot/grub/menu.lst`.

To locate the active GRUB menu, use the `bootadm` command with the `list-menu` subcommand:

```
# bootadm list-menu
```

This command also lists the contents of the active `menu.lst` file:

```
# bootadm list-menu
The location for the active GRUB menu is: /pool-name/boot/grub/menu.lst
default 0
timeout 10
0 be1
1 be1 failsafe
2 be3
3 be3 failsafe
4 be2
5 be2 failsafe
```

For further instructions on using the `bootadm` command, see “Using the `bootadm` Command to Manage the Boot Archives” on page 299.

x86: Implementation of the `findroot` Command

All Solaris installation methods, including Solaris Live Upgrade, now use the `findroot` command for specifying which disk slice on an x86 based system to boot. This implementation supports booting systems with ZFS roots, as well as UFS roots. This information is located in the `menu.lst` file that is used by GRUB. Previously, the `root` command, `root (hd0.0.a)`, was used to explicitly specify which disk slice to boot.

The installation methods include Solaris Live Upgrade, JumpStart, and the installation GUI program.

In addition to the `findroot` command, is the additional of a signature file on the slice, (*mysign*, `0, a`), where *mysign* is the name of a signature file that is located in the `/boot/grub/bootsign` directory. When booting a system from a ZFS root, the ZFS GRUB plug-in looks for and tries to mount a ZFS file system in slice `a` of `fdisk` partition `0`.

The name of the signature file varies, depending on the type of installation that was used. For more information about the naming convention that is used by the `findroot` command, see "Naming Conventions That Are Used by the `findroot` Command" on page 313.

Additional menu entries, which also use the `findroot` command, can be added to the GRUB menu after an installation or upgrade. For instructions, see "x86: How to Add GRUB Menu Entries That Use the `findroot` Command" on page 230.

Caution – The boot signature must be unique. Do not use or remove system generated signatures or user signatures that are duplicated across multiple instances of the Solaris software. Doing so might result in booting an incorrect OS instance or prevent the system from booting.

Note that the `root` command can still be used in the `menu.lst` file in certain instances, for example to boot Windows. However, use of the `root` command in cases where the `findroot` command is preferred is discouraged.

EXAMPLE 11–9 x86: Default `menu.lst` file on a System That Uses the `findroot` Command

The following example shows the format of a `menu.lst` file entry that implements the `findroot` command:

```
# title Solaris version
    findroot  (foosignpartition-no,slice-no,x) --x = Solaris root slice
    kernel$ /platform/i86pc/kernel/$ISADIR/unix
    module$ /platform/i86pc/$ISADIR/boot_archive
```

EXAMPLE 11–10 x86: Default `menu.lst` file That Supports ZFS Boot Loader

This is an example of a `menu.lst` file on system that supports a ZFS boot loader. The information for booting from a ZFS root file system is automatically added to the file when a Solaris Live Upgrade is performed.

```
title be1
findroot (BE_be1,0,a)
kernel$ /platform/i86pc/kernel/$ISADIR/unix -B $ZFS-BOOTFS
module$ /platform/i86pc/$ISADIR/boot_archive

title be1 failsafe
findroot (BE_be1,0,a)
```

EXAMPLE 11–10 x86: Default `menu.lst` file That Supports ZFS Boot Loader *(Continued)*

```
bootfs rpool/ROOT/szboot_0508
kernel /boot/platform/i86pc/kernel/unix -s -B console=ttyb
module /boot/x86.miniroot-safe
```

▼ x86: How to Add GRUB Menu Entries That Use the `findroot` Command

This procedure shows how to manually update the `menu.lst` file with user defined entries that use the `findroot` command. Typically, these entries are added after an installation or an upgrade. For guidelines on adding user&hyphen;defined entries that use the `findroot` command, see "x86: Implementation of the `findroot` Command" on page 228.

1. **Become superuser or assume an equivalent role.**

2. **Create a boot signature file on the root (`/`) file system or root pool that is booted.**

   - **For a ZFS pool,** *my-pool*, **create the boot signature file in the** /*my-pool*/`boot/grub/bootsign` **directory.**

     ```
     # touch /my-pool/boot/grub/bootsign/user-sign
     ```

   - **For a UFS file system, create the boot signature file in the** `/boot/grub/bootsign` **directory of the root file system to be booted.**

     ```
     # touch /boot/grub/bootsign/user-sign
     ```

Note – Make sure the file name that you choose for the boot signature is unique. Do not use system generated signature names or user signature names that are duplicated across multiple instances of the Solaris software. Doing so might prevent the system from booting or cause the wrong Solaris instance to boot.

---

3. **Add a menu entry that contains the `findroot` command.**

   a. **Locate the active `menu.lst` file:**

   ```
   # bootadm list-menu
   ```

   b. **Using a text editor, edit the active `menu.lst` file, adding the following entry:**

   ```
   title    User Solaris boot entry
   findroot  (user-sign, 3, c)
   kernel$   /platform/i86pc/kernel/$ISADIR/unix
   module$  /platform/i86pc/$ISADIR/boot_archive
   ```

In the preceding example, the `3` represents the 4th `fdisk` partition (partitions start at 0). The `c` represents the slice within a Solaris `fdisk` partition (slices start with `a`).

4 **Reboot the system.**

The new entry appears in the GRUB menu and can be selected to boot the specified Solaris OS instance.

◆ ◆ ◆ CHAPTER 12

Booting a Solaris System (Tasks)

This chapter describes the procedures for booting the Solaris release on SPARC and x86 based systems.

The following is a list of information that is in this chapter:

- "Booting a SPARC Based System (Task Map)" on page 233
- "Booting an x86 Based System by Using GRUB (Task Map)" on page 252
- "Using Fast Reboot on the x86 Platform (Task Map)" on page 268

For overview information about the boot process, see Chapter 9, "Shutting Down and Booting a System (Overview)."

Booting a SPARC Based System (Task Map)

Task	Description	For Instructions
Boot a SPARC based system to run level 3.	Use this boot method after shutting down the system or performing a system hardware maintenance task.	"SPARC: How to Boot a System to Run Level 3 (Multiuser Level)" on page 235
Boot a SPARC based system to run level S.	Use this boot method to boot the system after performing a system maintenance task such as backing up a file system. At this level, only local file systems are mounted and users cannot log in to the system.	"SPARC: How to Boot a System to Run Level S (Single-User Level)" on page 236
Boot a SPARC based system interactively.	Use this boot method after making temporary changes to a system file or the kernel for testing purposes.	"SPARC: How to Boot a System Interactively" on page 237

Task	Description	For Instructions
Boot a Solaris kernel other than default.	Use this procedure to boot a Solaris kernel other than the default kernel. Alternately, you can obtain a copy of an alternate boot file, change the default kernel to the new kernel, then set the `boot-file` parameter to boot the new default boot device.	"SPARC: How to Boot a Solaris Kernel Other Than the Default Kernel" on page 239
Display a list of the available ZFS bootable datasets on a SPARC based system.	Use the `boot -L` command to display a list of the available BEs within a ZFS pool on a system. **Note –** This option is only supported for boot devices that contain a ZFS pool.	"SPARC: How to List Available Bootable Datasets Within a ZFS Root Pool" on page 241
Boot a SPARC based system from a ZFS root file system.	Use the `boot -Z` option to boot a specified ZFS dataset. **Note –** This option is only supported for boot devices that contain a ZFS pool.	"SPARC: How to Boot From a ZFS Root File System" on page 243
Boot the failsafe archive on a SPARC based system.	Use this procedure to boot the failsafe archive on a SPARC based system. Then, run the `bootadm` command to update the boot archive.	"How to Boot the Failsafe Archive on a SPARC Based System" on page 246
Boot a SPARC based system from the network.	Use this boot method to boot a system from the network. Note that this method is also used for booting a diskless client.	"SPARC: How to Boot a System From the Network" on page 250

Booting a SPARC Based System

If a system is turned off, turning it on starts the multiuser boot sequence. The following procedures show how to boot to different run levels from the `ok` PROM prompt. These procedures assume that the system has been cleanly shut down, unless stated otherwise.

Use the `who -r` command to verify that the system is brought to the specified run level. For a description of run levels, see Chapter 16, "Managing Services (Overview)."

▼ SPARC: How to Boot a System to Run Level 3 (Multiuser Level)

Use this procedure to boot a system that is currently at run level 0 to run level 3.

1. **Boot the system to run level 3.**

   ```
   ok boot
   ```

   The automatic boot procedure displays a series of startup messages, and brings the system to run level 3. For more information, see the `boot`(1M) man page.

2. **Verify that the system has booted to run level 3.**

   The login prompt is displayed when the boot process has finished successfully.

   hostname `console login:`

Example 12–1 SPARC: Booting a System to Run Level 3 (Multiuser Level)

The following example displays the messages from booting a system to run level 3.

```
ok boot
Resetting ...

Sun Ultra 2 UPA/SBus (2 X UltraSPARC-II 296MHz), No Keyboard
OpenBoot 3.25, 512 MB memory installed, Serial #10342381.
Ethernet address 8:0:20:xx:cf:ed, Host ID: 80xxcfed.


Rebooting with command: boot
Boot device: /sbus@1f,0/SUNW,fas@e,8800000/sd@a,0:a  File and args: kadb
Loading kmdb...
SunOS Release 5.10        64-bit
Copyright 1983-2007 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms
WARNING: consconfig: cannot find driver for screen device /SUNW,ffb@1e,0
Hostname: dancehallgirl
NIS domain name is boulder.Central.Sun.COM
/dev/rdsk/c0t10d0s7 is clean
Reading ZFS config: done.

dancehallgirl console login:
```

▼ SPARC: How to Boot a System to Run Level S (Single-User Level)

Use this procedure to boot a system that is currently at run level 0 to run level S. This run level is used for system maintenance tasks, such as backing up a file system.

1. **Boot the system to run level S.**

   ```
   ok boot -s
   ```

2. **Type the superuser password when the following message is displayed:**

   ```
   SINGLE USER MODE

   Root password for system maintenance (control-d to bypass): xxxxxx
   ```

3. **Verify that the system is at run level S.**

   ```
   # who -r
   ```

4. **Perform the maintenance task that required the run level change to S.**

5. **After you complete the system maintenance task, type Control-D to bring the system to the multiuser state.**

Example 12–2 SPARC: Booting a System to Run Level S (Single-User Level)

The following example displays the messages from booting a system to run level S.

```
ok boot -s
Resetting ...

Sun Ultra 2 UPA/SBus (2 X UltraSPARC-II 296MHz), No Keyboard
OpenBoot 3.25, 512 MB memory installed, Serial #10342381.
Ethernet address 8:0:20:xx:cf:ed, Host ID: 80xxcfed.

Rebooting with command: boot -s
Boot device: /sbus@1f,0/SUNW,fas@e,8800000/sd@a,0:a  File and args: -s
SunOS Release 5.11
Copyright 1983-2007 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms
WARNING: consconfig: cannot find driver for screen device /SUNW,ffb@1e,0

Root password for system maintenance (control-d to bypass):
svc.startd: Returning to milestone all.
NIS domain name is boulder.Central.Sun.COM
/dev/rdsk/c0t10d0s7 is clean
```

```
Reading ZFS config: done.
dancehallgirl console login:
```

▼ SPARC: How to Boot a System Interactively

Use this boot option when you need to specify an alternate kernel or `/etc/system` file.

Before You Begin To specify an alternate `/etc/system` file when booting a SPARC based system interactively by using the `boot -a` command, you must perform the following steps before the system is booted.

- 1. Make backup copies of the `/etc/system` and `boot/solaris/filelist.ramdisk` files.

  ```
  # cp /etc/system /etc/system.bak
  # cp /boot/solaris/filelist.ramdisk /boot/solaris/filelist.ramdisk.orig
  ```

- 2. Add the `etc/system.bak` file name to the `/boot/solaris/filelist.ramdisk` file

  ```
  # echo "etc/system.bak" >> /boot/solaris/filelist.ramdisk
  ```

- 3. Update the boot archive.

  ```
  # bootadm update-archive -v
  ```

1 **Boot the system interactively.**

```
ok boot -a
```

2 **Answer the following system prompts:**

a. **When prompted, enter the name of the kernel to use for booting.**

Press enter to use the default kernel file name. Otherwise, provide the name of an alternate kernel, press Enter.

b. **When prompted, provide an alternate path for the `modules` directories.**

Press enter to use the default module directories. Otherwise, provide the alternate paths to module directories, press Enter.

c. **When prompted, provide the name of an alternate system file.**

Type `/dev/null` if your `/etc/system` file has been damaged.

d. **When prompted, enter the `root` filesystem type.**

Press enter to select UFS for local disk booting, which is the default, or enter `NFS` for network booting.

e. **When prompted, enter the physical name of `root` device.**

   Provide an alternate device name or press return to use the default.

3 **If you are not prompted to answer these questions, verify that you typed the `boot -a` command correctly.**

Example 12–3 SPARC: Booting a System Interactively

In this example, the default choices (shown in square brackets `[]`) are accepted. For instructions and an example of booting an alternate file system by using the `boot -a` command, see "SPARC: How to Boot a System Interactively" on page 237.

```
ok boot -a
Resetting ...

Sun Ultra 2 UPA/SBus (2 X UltraSPARC-II 296MHz), No Keyboard
OpenBoot 3.25, 512 MB memory installed, Serial #10342381.
Ethernet address 8:0:20:9d:cf:ed, Host ID: 809dcfed.



Rebooting with command: boot -a
Boot device: /sbus@1f,0/SUNW,fas@e,8800000/sd@a,0:a  File and args: -a

Boot device: /sbus@1f,0/SUNW,fas@e,8800000/sd@a,0:a  File and args: -a
Name of system file [/etc/system]:
SunOS Release 5.11 Version zwicky:nbsclean-build:12/04/2007 64-bit
Copyright 1983-2007 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.
Retire store [/etc/devices/retire_store] (/dev/null to bypass):
root filesystem type [ufs]:
Enter physical name of root device
[/sbus@1f,0/SUNW,fas@e,8800000/sd@a,0:a]:
WARNING: consconfig: cannot find driver for screen device /SUNW,ffb@1e,0
Hostname: dancehallgirl
NIS domain name is boulder.Central.Sun.COM
/dev/rdsk/c0t10d0s7 is clean
Reading ZFS config: done.
dancehallgirl login:
```

▼ SPARC: How to Boot a Solaris Kernel Other Than the Default Kernel

1. **Become superuser or assume an equivalent role.**

2. **Obtain a copy of an existing Solaris kernel and rename it.**

3. **Add the kernel that you copied and renamed in Step 2 to the `/etc/boot/solaris/filelist.ramdisk` file.**

   ```
   # echo "kernel.name" >> /boot/solaris/filelist.ramdisk
   ```

4. **Verify that the alternate kernel has been added the `/etc/boot/solaris/filelist.ramdisk` file.**

   ```
   # cat > /etc/boot/solaris/filelist.ramdisk
   ```

5. **Update the boot archive by using the `bootadm` command.**

   ```
   # bootadm update-archive
   ```

6. **Change to run level 0.**

   ```
   # init 0
   ```

   The `ok` PROM prompt is displayed.

7. **Boot the alternate kernel.**

   ```
   ok boot alternate-kernel
   ```

   For example:

   ```
   ok boot kernel.myname/sparcv9/unix
   ```

   - **To boot the alternate kernel by default, follow these steps:**

     a. **Set the boot-file parameter to the new kernel.**

     ```
     ok setenv boot-file kernel.name/sparc9/unix
     ```

     b. **Verify that the boot-file property has been changed.**

     ```
     ok printenv boot-file
     ```

     c. **Reboot the system.**

     ```
     ok boot
     ```

8. **After the system has booted, verify that the alternate kernel that was booted.**

   ```
   # prtconf -vp | grep whoami
   ```

Example 12–4 Booting an Alternate Solaris Kernel by Changing the Default Boot File

```
# cp -r /platform/sun4v/kernel /platform/sun4vu/kernel.caiobella
# echo "kernel.caiobela" >> /boot/solaris/filelist.ramdisk

# cat > /etc/boot/solaris/filelist.ramdisk
/platform/sun4v/kernel.caiobella
^D (control D)

ok setenv boot-file kernel.caiobells/sparcv9/unix
ok printenv boot-file
boot-file = kernel.caiobella/sparcv9/unix

ok boot

SC Alert: Host System has Reset

SC Alert: Host system has shut down.


Sun Fire T200, No KeyboardCopyright 2006 Sun Microsystems, Inc.  All rights reserved.
OpenBoot 4.25.0.build_01***PROTOTYPE BUILD***, 32760 MB memory available, Serial
#69060038.
Ethernet address 0:x:4f:x:c5:c6, Host ID: 8xxc5c6.



Rebooting with command: boot
Boot device: /pci@7c0/pci@0/pci@1/pci@0,2/LSILogic,sas@2/disk@0,0:a  File and
args: kernel.caiobella/sparcv9/unix
SunOS Release 5.10
Copyright 1983-2007 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.
DEBUG enabled
misc/forthdebug (176650 bytes) loaded
Hostname: seasonz
NIS domain name is lab.domain.sun.com
Reading ZFS config: done.

seasonz console login:
Password:
Last login: Mon Nov 12 18:02:00 on console
Sun Microsystems Inc.   SunOS 5.11
.
.
.
You have new mail.
#
#
```

```
# prtconf -vp | grep whoami
        whoami:  '/platform/sun4v/kernel.caiobella/sparcv9/unix'
```

Booting From a ZFS Root File System on a SPARC Based System

To support booting from ZFS on the Solaris SPARC platform, two new boot options have been added:

`-L`	Displays a list of available bootable datasets within a ZFS pool. **Note –** The boot `-L` command is executed from the OBP, *not* from the command line.
`-Z` *dataset*	Boots the root file system for the specified ZFS bootable dataset.

If you are booting a system from a ZFS root file system, first use the boot command with the `-L` option from the OBP to print a list of the available BEs on the system. Then, use the `-Z` option to boot the specified BE.

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

▼ SPARC: How to List Available Bootable Datasets Within a ZFS Root Pool

On SPARC based systems, the `menu.lst` file contains the following two GRUB commands:

- `title` – Provides a title for a boot environment
- `bootfs` – Specifies the full name of the bootable dataset

To display a list of bootable datasets within a ZFS pool, choose from the following methods:

- Use the `lustatus` command. This command lists all of the BEs in a given ZFS pool.

  Note that the `lustatus` command can also be used on x86 based systems.
- Use the `boot -L` command. This command displays a list of available BEs in a given ZFS pool and provides instructions for booting the system.

The following procedure shows how to use the `boot -L` command to list available BEs on a system. To boot a specified BE after running this command, follow the instructions that are printed on the screen.

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 **Bring the system to the `ok` prompt.**

```
# init 0
```

3 **List the available BEs in a ZFS pool:**

```
ok boot device-specifier -L
```

4 **(Optional) To boot one of the entries that is displayed, type the number of the entry. To boot the specified BE, follow the directions that are printed to the screen.**

For instructions, see "SPARC: How to Boot From a ZFS Root File System" on page 243.

Example 12–5 SPARC: Displaying a List of Available BEs on a System by Using `boot -L`

```
# init 0
# svc.startd: The system is coming down. Please wait.
svc.startd: 94 system services are now being stopped.
svc.startd: The system is down.
syncing file systems... done
Program terminated
ok boot -L
.
.
.
Boot device: /pci@1f,0/pci@1/scsi@8/disk@0,0 File and args: -L
zfs-file-system
Loading: /platformsun4u/bootlst
1.s10s_nbu6wos
2 zfs2BE
Select environment to boot: [ 1 - 2 ]: 2

to boot the selected entry, invoke:
boot [<root-device] -Z rpool/ROOT/zfs2BE
```

See Also For more information, see Chapter 4, "Installing and Booting a ZFS Root File System," in *Solaris ZFS Administration Guide*.

▼ SPARC: How to Boot From a ZFS Root File System

Booting from ZFS differs from booting from UFS. When booting from ZFS, a device specifier identifies a storage pool, *not* a single root file system. A storage pool can contain multiple bootable datasets, or root file systems. Therefore, when booting from ZFS, you must also identify a root file system within the pool that is identified by the boot device as the default. By default, the default boot device is identified by the pool's `bootfs` property. This procedure shows how to boot the system by specifying a ZFS bootable dataset. See the `boot`(1M) man page for a complete description of all the boot options that are available.

Note – If the `bootfs` property was previously set up correctly, for example, if you used the `luactivate` command to activate a BE, the system boots a ZFS root automatically.

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

1. **Become superuser or assume an equivalent role.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2. **Bring the system to the `ok` prompt.**

   ```
   # init 0
   ```

3. **(Optional) To display a list of available BEs, use the `boot` command with the `-L` option.**

   For instructions, see "SPARC: How to List Available Bootable Datasets Within a ZFS Root Pool" on page 241.

4. **To boot a specified entry, type the number of the entry and press Return:**

   ```
   Select environment to boot: [1 - 2]:
   ```

5. **To boot the system, follow the instructions that are printed to the screen:**

   ```
   To boot the selected entry, invoke:
   boot [<root-device>] -Z rpool/ROOT/dataset

   ok boot -Z rpool/ROOT/dataset
   ```

   For example:

   ```
   # boot -Z rpool/ROOT/zfs2BE
   ```

6 **After the system has booted, type the following command to verify the active BE:**

```
# prtconf -vp | grep whoami
```

- **To display the boot path for the active BE, type:**

  ```
  # prtconf -vp | grep bootpath
  ```

- **Alternately, you can type the following command to determine whether the corrected BE was booted:**

  ```
  # df -lk
  ```

Example 12–6 SPARC: Booting From a ZFS Root File System

This example shows how to use the `boot -Z` command to boot a ZFS dataset on a SPARC based system.

```
# init 0
# svc.startd: The system is coming down. Please wait.
svc.startd: 79 system services are now being stopped.
svc.startd: The system is down.
syncing file systems... done
Program terminated
ok boot -Z rpool/ROOT/zfs2BEe
Resetting
LOM event: =44d+21h38m12s host reset
g ...

rProcessor Speed = 648 MHz
Baud rate is 9600
8 Data bits, 1 stop bits, no parity (configured from lom)

Firmware CORE Sun Microsystems, Inc.
@(#) core 1.0.12 2002/01/08 13:00
software Power ON
Verifying nVRAM...Done
Bootmode is 0
[New I2C DIMM address]
.
.
.
Environment monitoring: disabled
Executng last command: boot -Z rpool/ROOT/zfs2BE
Boot device: /pci@1f,0/pci@1/scsi@8/disk@0,0 File and args: -Z rpool/ROOT/zfs2Be
zfs-file-system
Loading: /platform/SUNW,UltraAX-i2/boot_archive
Loading: /platform/sun4u/boot_archive
```

```
ramdisk-root hsfs-file-system
Loading: /platform/SUNW,UltraAX-i2/kernel/sparcv9/unix
Loading: /platform/sun4u/kernel/sparcv9/unix
.
.
.
Hostname: mallory
NIS domainname is boulder.Central.Sun.COM
Reading ZFS config: done.
Mounting ZFS filesytems: (6/6)

mallory console login:
```

See Also For information about booting the failsafe archive for a specified ZFS bootable dataset, see "How to Boot the Failsafe Archive on a SPARC Based System" on page 246.

Booting the Failsafe Archive on a SPARC Based System

Booting a system from a root (/) file system image that is a boot archive, and then remounting this file system on the actual root device can sometimes result in a boot archive and root file system that do not match, or are *inconsistent*. Under these conditions, the proper operation and integrity of the system is compromised. After the root (/) file system is mounted, and before relinquishing the in-memory file system, the system performs a consistency verification against the two files systems. If an inconsistency is detected, the normal boot sequence is suspended and the system reverts to *failsafe mode*.

Also, if a system failure, a power failure, or a kernel panic occurs immediately following a kernel file update, the boot archives and the root (/) file system might not be synchronized. Although the system might still boot with the inconsistent boot archives, it is recommended that you boot the failsafe archive to update the boot archives. You can also use the `bootadm` command to manually update the boot archives. For more information, see "Using the `bootadm` Command to Manage the Boot Archives" on page 299.

The failsafe archive can be booted for recovery purposes or to update the boot archive on both the SPARC and x86 platforms.

On the SPARC platform the failsafe archive is:

```
/platform/‘uname -m‘/failsafe
```

You would boot the failsafe archive by using the following syntax:

```
ok boot  -F failsafe
```

Failsafe booting is also supported on systems that are booted from ZFS. When booting from a ZFS-rooted BE, each BE has its own failsafe archive. The failsafe archive is located where the root (`/`) file system is located, as is the case with a UFS-rooted BE. The default failsafe archive is the archive that is in the default bootable file system. The default bootable file system (dataset) is indicated by the value of the pool's `bootfs` property.

For information about booting an x86 based failsafe archive, see "Booting the Failsafe Archive on an x86 Based System" on page 264.

Another method that can be used to update the boot archives is to clear the boot-archive service. However, the preferred methods for updating the boot archives are to boot the failsafe archive or use the `bootadm` command. For more information, see "How to Update an Inconsistent Boot Archive by Clearing the `boot-archive` Service" on page 296.

▼ How to Boot the Failsafe Archive on a SPARC Based System

Use this procedure to boot the failsafe archive on a SPARC based system. If the system does not boot after the boot archive is updated, you might need to boot the system in single-user mode. For more information, see "SPARC: How to Boot a System to Run Level S (Single-User Level)" on page 236.

Note – This procedures also includes instructions for booting the failsafe archive for a specific ZFS dataset.

---

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 **Bring the system to the `ok` prompt:**

```
# init 0
```

3 **Boot the failsafe archive.**

- **To boot the default failsafe archive, type:**

```
ok boot -F failsafe
```

- **To boot the failsafe archive of a specific ZFS dataset:**

```
ok boot -F failsafe -Z dataset
```

For example:

```
ok  boot -F failsafe -Z rpool/ROOT/zfsBE2
```

Note – To determine the name of the dataset to boot, first use the `boot -L` command to display a list of the available BEs on the system. For more information, see "SPARC: How to List Available Bootable Datasets Within a ZFS Root Pool" on page 241.

If an inconsistent boot archive is detected a message is displayed.

4 **To update the boot archive, type `y` and press Return.**

```
An out of sync boot archive was detected on rpool.
The boot archive is a cache of files used during boot
and should be kept in sync to ensure proper system operation.

Do you wish to automatically update this boot archive? [y,n,?] y
```

If the archive was updated successfully, a message is displayed:

```
The boot archive on rpool was updated successfully.
```

Example 12–7 SPARC: Booting the Failsafe Archive

This example shows how to boot the failsafe archive on a SPARC based system. If no device is specified, the failsafe archive for the default boot device is booted.

```
ok boot -F failsafe
Resetting ...
screen not found.
Can't open input device. Keyboard not present.  Using ttya for input and output.

Sun Enterprise 220R (2 X UltraSPARC-II 450MHz), No Keyboard
OpenBoot 3.23, 1024 MB memory installed, Serial #13116682.
Ethernet address 8:0:20:c8:25:a, Host ID: 80c8250a.

Rebooting with command: boot -F failsafe
Boot device: /pci@1f,4000/scsi@3/disk@1,0:a  File and args: -F failsafe
SunOS Release 5.10t
Copyright 1983-2007 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.
Configuring /dev Searching for installed OS instances...

An out of sync boot archive was detected on /dev/dsk/c0t1d0s0.
The boot archive is a cache of files used during boot and
should be kept in syncto ensure proper system operation.
```

```
Do you wish to automatically update this boot archive? [y,n,?] y
Updating boot archive on /dev/dsk/c0t1d0s0.
The boot archive on /dev/dsk/c0t1d0s0 was updated successfully.

Solaris 5.10 was found on /dev/dsk/c0t1d0s0.
Do you wish to have it mounted read-write on /a? [y,n,?] n
Starting shell.
#
```

Example 12–8 SPARC: Booting the Failsafe Archive for a Specified ZFS Dataset

This example shows how to boot the failsafe archive of a ZFS dataset. Note that the `boot -L` command is first used to display a list of available boot environments. This command must be run at the `ok` prompt.

```
ok boot -L
Rebooting with command: boot -L
Boot device: /pci@1f,4000/scsi@3/disk@1,0  File and args: -L
1 zfsBE2
Select environment to boot: [ 1 - 1 ]: 1

To boot the selected entry, invoke:
boot [<root-device>] -Z rpool/ROOT/zfsBE2

Program terminated
{0} ok






Resetting ...

screen not found.
Can’t open input device.
Keyboard not present.  Using ttya for input and output.

Sun Enterprise 220R (2 X UltraSPARC-II 450MHz), No Keyboard
OpenBoot 3.23, 1024 MB memory installed, Serial #13116682.
Ethernet address 8:0:20:c8:25:a, Host ID: 80c8250a.





{0} ok  boot -F failsafe -Z rpool/ROOT/zfsBE2
Boot device: /pci@1f,4000/scsi@3/disk@1,0  File and args: -F failsafe -Z
```

```
rpool/ROOT/zfsBE2
SunOS Release 5.10
Copyright 1983-2008 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.
Configuring /dev
Searching for installed OS instances...

ROOT/zfsBE2 was found on rpool.
Do you wish to have it mounted read-write on /a? [y,n,?] y
mounting rpool on /a

Starting shell.
#
#
#
# zpool list
NAME    SIZE   USED  AVAIL    CAP  HEALTH  ALTROOT
rpool  16.8G  6.26G  10.5G    37%  ONLINE  /a
#
# zpool status
  pool: rpool
 state: ONLINE
 scrub: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        rpool       ONLINE       0     0     0
          c0t1d0s0  ONLINE       0     0     0

errors: No known data errors
#
# df -h
Filesystem             size   used  avail capacity  Mounted on
/ramdisk-root:a        163M   153M     0K   100%    /
/devices                 0K     0K     0K     0%    /devices
/dev                     0K     0K     0K     0%    /dev
ctfs                     0K     0K     0K     0%    /system/contract
proc                     0K     0K     0K     0%    /proc
mnttab                   0K     0K     0K     0%    /etc/mnttab
swap                   601M   344K   601M     1%    /etc/svc/volatile
objfs                    0K     0K     0K     0%    /system/object
sharefs                  0K     0K     0K     0%    /etc/dfs/sharetab
swap                   602M   1.4M   601M     1%    /tmp
/tmp/root/etc          602M   1.4M   601M     1%    /.tmp_proto/root/etc
fd                       0K     0K     0K     0%    /dev/fd
rpool/ROOT/zfsBE2       16G   5.7G   9.8G    37%    /a
rpool/export            16G    20K   9.8G     1%    /a/export
```

```
rpool/export/home       16G    18K   9.8G     1%    /a/export/home
rpool                   16G    63K   9.8G     1%    /a/rpool
```

Booting a SPARC Based System From the Network

You might need to boot a system from the network under the following conditions:

- When the system is first installed
- If the system won't boot from the local disk
- If the system is a diskless client

Two network configuration boot strategies are available:

- Reverse Address Resolution Protocol (RARP) and ONC+™ RPC Bootparams Protocol
- Dynamic Host Configuration Protocol (DHCP)
- For network devices, the process for booting over a local area network (LAN) and booting over a wide area network (WAN) is slightly different. In both network boot scenarios, the PROM downloads the booter from a boot server or an installation server, which is `inetboot` in this case.

  When booting over a (LAN), the firmware uses RARP and BOOTP or DHCP to discover the boot or installation server. TFTP is then used to download the booter, which is `inetboot` in this case.

  When booting over a WAN, the firmware uses either DHCP or NVRAM properties to discover the installation server, the router, and the proxies that are required for the system to boot from the network. The protocol that is used to download the booter is HTTP. In addition, the booter's signature might be checked with a predefined private key.

▼ SPARC: How to Boot a System From the Network

Any system can boot from the network if a boot server is available. You might want to boot a stand-alone system from the network if the system cannot boot from the local disk. For information on changing or resetting the default boot device, see "SPARC: How to Change the Default Boot Device by Using the Boot PROM" on page 213.

Two network configuration boot strategies are available on sun–4u systems:

- RARP – Reverse Address Resolution Protocol and ONC+ RPC Bootparams Protocol
- DHCP – Dynamic Host Configuration Protocol

The default network boot strategy is set to RARP. You can use either protocol, depending on whether a RARP boot server or a DHCP boot server is available on your network.

Note – Sun Ultra systems must have at least PROM version 3.25.*nn* to use the DHCP network boot strategy. For information on determining your PROM version, see "SPARC: How to Find the PROM Revision Number for a System" on page 210.

If both protocols are available, you can temporarily specify which protocol to use in the `boot` command. Or, you can save the network boot strategy across system reboots at the PROM level by setting up an NVRAM alias. The following example uses the `nvalias` command to set up a network device alias for booting DHCP by default on a Sun Ultra 10 system.

```
ok nvalias net    /pci@1f,4000/network@1,1:dhcp
```

As a result, when you type `boot net`, the system boots by using the DHCP network book strategy.

Note – You should not use the `nvalias` command to modify the `NVRAMRC` file, unless you are very familiar with the syntax of this command and the `nvunalias` command. For information on using these commands, see the *OpenBoot 3.x Command Reference Manual*.

Before You Begin You must have already set up a RARP or DHCP boot server in your network to use either protocol to boot successfully.

1. **If necessary, shut down the system.**

2. **Determine the method for booting from the network, and select one of the following:**

   a. Boot the system from the network by using the DHCP strategy.

   ```
   ok boot net[:dhcp]
   ```

   If you have changed the PROM setting to boot DHCP by default, as in the preceding `nvalias` example, you only have to specify `boot net`.

   b. Boot the system from the network by using the RARP strategy.

   ```
   ok boot net[:rarp]
   ```

   Because RARP is the default network boot strategy, you only have to specify `boot net:rarp` if you have changed the PROM value to boot DHCP.

Booting an x86 Based System by Using GRUB (Task Map)

Task	Description	For Instructions
Boot an x86 based system to run level 3, multiuser level.	Use this boot method to bring the system back to multiuser level after shutting down the system or performing a system hardware maintenance task.	"x86: How to Boot a System to Run Level 3 (Multiuser)" on page 253
Boot an x86 based system the in single-user mode.	Use this boot method to perform a system maintenance task, such as backing up a file system.	"x86: How to Boot a System to Run Level S (Single-User Level)" on page 254
Boot an x86 based system interactively.	Use this boot method after making temporary changes to a system file or the kernel for testing purposes.	"x86: How to Boot a System Interactively" on page 257
Display a list a ZFS bootable datasets on an x86 based system.	Use one of the following methods to display the available BEs on an x86 based system that has a ZFS root file system: ■ `lustatus` ■ `bootadm list-menu`	"How to Display a List of the Available ZFS Boot Environments on an x86 Based System" on page 260
Boot an x86 based system from a ZFS root file system.	If you install or upgrade your system to a Solaris release that supports a ZFS boot loader, the GRUB menu entry for the default ZFS BE contains the `-B $ZFS-BOOTFS` boot argument by default. The system boots automatically from ZFS. **Note –** This option is supported *only* for boot devices that contain a ZFS pool.	"How to Boot From a ZFS Root File System on an x86 Based System" on page 261
Boot the failsafe archive on an x86 based system.	Use this procedure to boot the failsafe archive on an x86 based system. Then, run the `bootadm` command to update the boot archive.	"How to Boot the Failsafe Archive on an x86 Based System by Using GRUB" on page 264
Boot an x86 based failsafe archive to forcibly update a corrupt boot archive.	Use this procedure in cases where the boot archive is corrupt, and the system refuses to boot normally, or you are not prompted to update an inconsistent boot archive.	"x86: How to Boot the Failsafe Archive to Forcibly Update a Corrupt Boot Archive" on page 266
Boot an x86 based system from the network by using GRUB.	Use this method to boot a PXE or non-PXE device from the network with the default network configuration strategy. This method is also used for booting a diskless client.	"x86: How to Perform a GRUB Based Boot From the Network" on page 278

x86: Error Messages Upon System Boot

The Solaris installation software and utilities, including the `bootadm` command, use the presence of the `/boot/multiboot` and `/platform/i86pc/multiboot` files to determine if the system's running OS or the Solaris installation software implements the GRUB boot method or the Solaris Device Configuration Assistant boot method.

If the multiboot module from the previous GRUB implementation is loaded by GRUB, the console displays an error message that says multiboot is no longer support and to manually update the entries in the `menu.lst` file to successfully boot the system. For more information, see `http://www.sun.com/msg/SUNOS-8000-AK` and the `boot`(1M) man page.

For instructions on booting a system interactively modifying the GRUB kernel line at boot time, see "x86: How to Modify Boot Behavior by Editing the GRUB Menu at Boot Time" on page 221. For instructions on modifying the `menu.lst` file permanently after the system has booted, see "x86: How to Modify Boot Behavior by Editing the `menu.lst` File" on page 224.

▼ x86: How to Boot a System to Run Level 3 (Multiuser)

Use this procedure to boot a system that is currently at run level 0 to run level 3.

1 **Reboot the system.**

```
# reboot
```

If the system displays the `Press any key to reboot` prompt, press any key to reboot the system.

You can also use the Reset button at this prompt. If the system is shut down, turn the system on with the power switch.

When the boot sequence begins, the GRUB menu is displayed.

2 **When the GRUB menu is displayed, press Enter to boot the default OS instance.**

If you do not choose an entry within 10 seconds, the system automatically boots to run level 3.

The login prompt is displayed when the boot process has finished successfully.

3 **Log in to the system.**

```
hostname console login:
```

4 **Verify that the system booted to run level 3.**

```
# who -r
system% who -r
   .       run-level 3  Mar  2 09:44     3      0  S
```

Example 12–9 x86: Booting a System To Run Level 3 (Multiuser Level)

```
# reboot

Jul 24 11:29:52 bearskin reboot: rebooted by root
```

```
syncing file systems... done
rebooting...

Adaptec AIC-7899 SCSI BIOS v2.57S4
(c) 2000 Adaptec, Inc. All Rights Reserved.

 Press <Ctrl><A> for SCSISelect(TM) Utility!

Ch B,  SCSI ID: 0 SEAGATE  ST336607LSUN36G   160

GNU GRUB  version 0.95  (637K lower / 2096064K upper memory)
==============================================================
Solaris 10 10/08 s10x_u6wos_03 X86
Solaris failsafe

==============================================================
        Use the  and  keys to select which entry is highlighted.
        Press enter to boot the selected OS, 'e' to edit the
        commands before booting, or 'c' for a command-line.


SunOS Release 5.10 Version Generic_137138-04 32-bit
Copyright 1983-2008 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.
Hostname: pups
NIS domain name is ....sfbay.sun.com
Reading ZFS config: done.
Mounting ZFS filesystems: (5/5)

pups console login:

# who -r
   .       run-level 3  Jul 24 11:31     3      0  S
```

▼ x86: How to Boot a System to Run Level S (Single-User Level)

Use this procedure to boot a system that is at run level 0 to run level S. The single-user level is used for performing system maintenance.

Note – This procedure can be used for all GRUB implementations. However, the boot entries in the GRUB main menu vary, depending on the Solaris release you are running.

For a description of all the kernel options that you can specify in the GRUB menu at boot time, see "x86: Modifying Boot Behavior by Editing the GRUB Menu at Boot Time" on page 219.

1 **Reboot the system.**

```
# reboot
```

If the system displays the `Press any key to reboot` prompt, press any key to reboot the system.

You can also use the Reset button at this prompt. If the system is shut down, turn the system on with the power switch.

When the boot sequence begins, the GRUB menu is displayed.

2 **When the GRUB main menu is displayed, type `e` to edit the GRUB menu.**

3 **Depending on the release you are running, use the arrow keys to choose the `kernel` or `kernel$` line.**

If you cannot use the arrow keys, use the caret key (`^`) key to scroll up and the letter `v` key to scroll down.

4 **Type `e` again to edit the boot entry.**

From here, you can add options and arguments to the `kernel` or `kernel$` line.

5 **To boot the system in single-user mode, type `-s` at the end of the boot entry line. Then, press Return to go back to the previous screen.**

- **To specify other boot behaviors, replace the `-s` option with the appropriate boot option.**

  The following alternate boot behaviors can be specified in this manner.

  - Perform a reconfiguration boot.
  - Boot a 64-bit capable system in 32-bit mode.
  - Boot the system with the kernel debugger.
  - Redirect the console.

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

6 **To boot the system in single-user mode, type `b`.**

7 **When prompted, type the root password.**

Note – If you are running the OpenSolaris 2008.11 release, you need to also enter an account name *before* entering the root password. The account name can be root or any other privileged account, such as “jack” on the Live CD, or an account that you created during the installation.

---

8 **Verify that the system is at run level S.**

```
# who -r
.       run-level S  Jun 13 11:07     S      0  0
```

9 **Perform the system maintenance task that required the run level change to S.**

10 **After you complete the system maintenance task, reboot the system.**

Example 12–10 x86: Booting a System in Single-User Mode

```
# reboot
Jul  2 14:30:01 pups reboot: initiated by root on /dev/console
syncing files...

Press <Ctrl><A> forPSCSISelect(TM) Utility!


GNU GRUB  version 0.95  (637K lower / 2096064K upper memory)

===================================================
Solaris 10 10/08 s10x_u6wos_03 X86
Solaris failsafe

=====================================================
        Use the  and  keys to select which entry is highlighted.
        Press enter to boot the selected OS, 'e' to edit the
        commands before booting, or 'c' for a command-line.
=====================================================

GNU GRUB  version 0.95  (637K lower / 2096064K upper memory)

=====================================================
findroot (pool_rpool,0,a)
kernel$ /platform/i86pc/multiboot -B $ZFS-BOOTFS
module /platform/i86pc/boot_archive
================================================
        Use the  and  keys to select which entry is highlighted.
        Press 'b' to boot, 'e' to edit the selected command in the
        boot sequence, 'c' for a command-line, 'o' to open a new line
        after ('O' for before) the selected line, 'd' to remove the
        selected line, or escape to go back to the main menu.

[ Minimal BASH-like line editing is supported.  For the first word, TAB
lists possible command completions.  Anywhere else TAB lists the possible
completions of a device/filename.  ESC at any time exits. ]

grub edit> kernel$ /platform/i86pc/multiboot -B $ZFS-BOOTFS -s

 GNU GRUB  version 0.95  (637K lower / 2096064K upper memory)

=======================================================
findroot (pool_rpool,0,a)
```

```
kernel$ /platform/i86pc/multiboot -B $ZFS-BOOTFS -s
module /platform/i86pc/boot_archive
======================================
        Use the  and  keys to select which entry is highlighted.
        Press 'b' to boot, 'e' to edit the selected command in the
     boot sequence, 'c' for a command-line, 'o' to open a new line
        after ('O' for before) the selected line, 'd' to remove the
   selected line, or escape to go back to the main menu.
.
.
.
SunOS Release 5.10
Copyright 1983-2008 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.
Booting to milestone "milestone/single-user:default".
Hostname: pups Requesting System Maintenance Mode SINGLE USER MODE
Root password for system maintenance (control-d to bypass):
single-user privilege assigned to /dev/console.
Entering System Maintenance Mode
Jul  2 14:41:48 su: 'su root' succeeded for root on /dev/console Sun Microsystems Inc.
# who -r
who -r    .       run-level S  Jul  2 14:39     S      0  0 #
```

x86: How to Boot a System Interactively

Use this procedure to boot a system if you need to specify an alternate kernel or an alternate `/etc/system` file.

Before You Begin To specify an alternate `/etc/system` file when booting an x86 based system interactively by using the `boot -a` command, you must first perform the following steps:

- 1. Make backup copies of the `/etc/system` and the `boot/solaris/filelist.ramdisk` files.

  ```
  # cp /etc/system /etc/system.bak
  # cp /boot/solaris/filelist.ramdisk /boot/solaris/filelist.ramdisk.orig
  ```

- 2. Add the `/etc/system.bak` file name to the `/boot/solaris/filelist.ramdisk` file

  ```
  # echo "etc/system.bak" >> /boot/solaris/filelist.ramdisk
  ```

- 3. Update the boot archive.

  ```
  # bootadm update-archive -v
  ```

1 **Reboot the system.**

```
# reboot
```

If the system displays the `Press any key to reboot` prompt, press any key to reboot the system.

You can also use the Reset button at this prompt. If the system is shut down, turn the system on with the power switch.

When the boot sequence begins, the GRUB main menu is displayed.

2 **To access the GRUB edit menu, type `e`.**

3 **Use the arrow keys to select the `kernel` or `kernel$` line.**

4 **Type `e` to edit the boot entry line.**

5 **Type `-a` to boot the system interactively. Then, press Enter to return to the GRUB main menu.**

6 **To boot the system interactively, type `b`.**

7 **Type a default directory for modules, or press Enter to accept the default.**

```
Enter default directory for modules [/platform/i86pc/kernel /kernel /usr/kernel]:
```

8 **Type an alternate system file name,** ***alternate-file*****.**

```
Name of system file [etc/system]: /etc/system.bak
```

Pressing Enter without providing an alternate file accepts the default.

Repair the damaged `/etc/system` file.

9 **Reboot the system to run level 3.**

Example 12–11 x86: Booting a System Interactively

```
# reboot
syncing file systems... done
rebooting...


GNU GRUB  version 0.95  (637K lower / 2096064K upper memory)
===================================================
Solaris 10 10/08 s10x_u6wos_03 X86
Solaris failsafe
=====================================================
        Use the  and  keys to select which entry is highlighted.
        Press enter to boot the selected OS, ’e’ to edit the
        commands before booting, or ’c’ for a command-line.
=====================================================
GNU GRUB  version 0.95  (637K lower / 2096064K upper memory)
=====================================================
findroot (pool_rpool,0,a)
kernel$ /platform/i86pc/multiboot -B $ZFS-BOOTFS
module /platform/i86pc/boot_archive
=====================================================
        Use the  and  keys to select which entry is highlighted.
        Press 'b' to boot, 'e' to edit the selected command in the
        boot sequence, 'c' for a command-line, 'o' to open a new line
        after ('O' for before) the selected line, 'd' to remove the
        selected line, or escape to go back to the main menu.

[ Minimal BASH-like line editing is supported.  For the first word, TAB
lists possible command completions.  Anywhere else TAB lists the possible
completions of a device/filename.  ESC at any time exits. ]

grub edit> kernel$ /platform/i86pc/multiboot -B $ZFS-BOOTFS -a
GNU GRUB  version 0.95  (637K lower / 2096064K upper memory)

====================================================
findroot (pool_rpool,0,a)
kernel$ /platform/i86pc/multiboot -B $ZFS-BOOTFS -a
module /platform/i86pc/boot_archive
====================================================
.
.
.
Enter default directory for modules [/platform/i86pc/kernel /kernel /usr/kernel]:
Name of system file [/etc/system]:  /etc/system.bak
SunOS Release 5.10 Version Generic_137138-04 32-bit
Copyright 1983-2008 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.
Hostname: pups
NIS domain name is ....sfbay.sun.com
Reading ZFS config: done.
Mounting ZFS filesystems: (5/5)
pups console login:
```

Booting From a ZFS Root File System on an x86 Based System

To support booting a ZFS root file system on the x86 platform, a new GRUB keyword, `$ZFS-BOOTFS`, has been introduced. If a root device contains a ZFS pool, this keyword is assigned a value, which is then passed to the kernel by using the `-B` option to identify the dataset to boot. If you install or upgrade your system with a Solaris release that supports a ZFS boot loader, the GRUB `menu.lst` file, as well as the GRUB boot menu, contains this information by default.

▼ How to Display a List of the Available ZFS Boot Environments on an x86 Based System

1. **Become superuser or assume an equivalent role.**

2. **To display a list of available BEs on the system, type the following command:**

   ```
   # lustatus
   ```

   Note that the `lustatus` command can also be used on SPARC based systems.

Note – If the following error is displayed when you run the `lustatus` command, it is an indication that a new installation was performed and that Solaris Live Upgrade was not used. Before any BEs can be acknowledged in the `lustatus` output, a new BE must be first created on the system.

```
# lustatus
ERROR: No boot environments are configured on this system
ERROR: cannot determine list of all boot environment names
```

For more information about using Solaris Live Upgrade to migrate a UFS root file system to a ZFS root file system, see "Migrating a UFS Root File System to a ZFS Root File System (Solaris Live Upgrade)" in *Solaris ZFS Administration Guide*.

Example 12–12 Displaying a List of Available ZFS Bootable Datasets by Using the `lustatus` Command

In this example, the output of the `lustatus` command shows the status of three ZFS bootable datasets. The default boot environment is `be1` and therefore cannot be deleted.

```
# lustatus
Boot Environment           Is       Active Active    Can    Copy
Name                       Complete Now    On Reboot Delete Status
-------------------------- -------- ------ --------- ------ ----------
s10s_nbu6wos               yes      no     no        yes    -
zfs2BE                     yes      yes    yes       no     -
zfsbe3                     no       no     no        yes    -
#
```

If the BE has been created and is bootable, a "yes" appears in the `Is Complete` column. If a BE has been created, but is not yet activated, a 'no" appears in this column. To activate a BE, use the `luactivate` command. Run the `lustatus` command afterwards to verify that the BE was successfully activated.

For more information see the `lustatus`(1M) and the `luactivate`(1M)man pages.

▼ How to Boot From a ZFS Root File System on an x86 Based System

This procedure describes how to boot from a ZFS root file system on an x86 system that supports a ZFS boot loader.

Note that if you install or upgrade your system to a Solaris release that supports a ZFS boot loader, the GRUB menu entry contains the `-B $ZFS-BOOTFS` boot argument by default, so the system boots from ZFS without requiring any additional boot arguments.

1 **Reboot the system.**

```
# reboot
```

If the system displays the `Press any key to reboot` prompt, press any key to reboot the system.

You can also use the Reset button at this prompt. If the system is shut down, turn the system on with the power switch.

When the boot sequence begins, the GRUB main menu is displayed. If the default boot entry is a ZFS file system menu is similar to the following:

```
GNU GRUB  version 0.95  (637K lower / 3144640K upper memory)
 +-----------------------------------------------------------------+
| be1
| be1 failsafe
| be3
| be3 failsafe
| be2
| be2 failfafe
  +----------------------------------------------------------------+
      Use the ^ and v keys to select which entry is highlighted.
      Press enter to boot the selected OS, 'e' to edit the
      commands before booting, or 'c' for a command-line.
```

2 **When the GRUB menu is displayed, press Enter to boot the default OS instance.**

If you do not choose an entry within 10 seconds, the system automatically boots to run level 3.

3 **To boot another BE, use the arrow keys to highlight the specified boot entry.**

4 **Type `b` to boot this entry or `e` to edit the entry.**

If you type `e` to edit the entry, the default menu for booting a system with a ZFS root would appear as follows:

```
findroot (BE_be10,0,a)
kernel$ /platform/i86pc/kernel/$ISADIR/unix -B $ZFS-BOOTFS
module$ /platform/i86pc/$ISADIR/boot-archive
```

For more information about GRUB menu entries at boot time, see"x86: How to Modify Boot Behavior by Editing the GRUB Menu at Boot Time" on page 221.

Example 12–13 x86: Activating a New Boot Environment on an x86 Based System

This example shows the steps that are followed to activate a boot environment, `be10`, on a system. Note that the `lustatus` command is run first, to determine which BEs on the system are active and which BEs require activation.

```
# lustatus
Boot Environment           Is       Active Active    Can    Copy
Name                     Complete Now    On Reboot Delete Status
-----------------------------------------------------------------
be1                        yes      yes    yes       no
be10                       yes      no     no        yes



# luactivate be10
System has findroot enabled GRUB Generating boot-sign, partition and slice
information for PBE <be1>
WARNING: The following file s have change on both the current boot environment
<be1> zone <global> and the boot environment to be activitate <be10>
        /etc/zfs/zpool.cache
INFORMATION: The files listed above are in conflict between the current
boot environment <be1> zone <global> and the boot environment to be
activated <be10>. These files will not be automatically synchronized from
the current boot environment <be1> when boot environment <be10> is activated.

Setting failsafe console to <ttyb>
Generating boot-sign for ABE <be10>
Generating partition and slice information for ABE <be10>
Copied boot menu from top level dataset.
Generating direct boot menu entries for PBE.
Generating direct boot menu entries for ABE.
Disabling splashimage
Current GRUB menu default setting is not valid
title Solaris bootenv rc
No more bootadm entries. Deletion of bootadm entries is complete.
GRUB menu default setting is unchanged
Done eliding bootadm entries.
**************************************************************
The target boot environment has been activated. It will be used when you
reboot. NOTE: You MUST NOT USE the reboot, halt, or uadmin commands. You
MUST USE either the init or the shutdown command when you reboot. If you
do not use either init or shutdown, the system will not boot using the
target BE.
```

```
***************************************************************
,,,


# reboot
May 30 09:52:32 pups reboot: initiated by root on /dev/console
syncing file systems... done
rebooting...

CE SDRAM BIOS P/N GR-xlint.007-4.330
*

BIOS Lan-Console 2.0
Copyright (C) 1999-2001 Intel Corporation
.
.
.
GNU GRUB  version 0.95  (637K lower / 3144640K upper memory)
 +-------------------------------------------------------------------+
| be1
| be1 failsafe
| be10
| be10 failsafe
+-------------------------------------------------------------------+
      Use the ^ and v keys to select which entry is highlighted.
      Press enter to boot the selected OS, 'e' to edit the
      commands before booting, or 'c' for a command-line.

SunOS Release 5.10 32-bit
Copyright 1983-2008 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.

Hostname: pups
NIS domain name is sunsoft.eng.sun.com
Reading ZFS config: done.
Mounting ZFS filesystems: (8/8)

pups console login:
# lustatus
Boot Environment           Is       Active Active    Can    Copy
Name                      Complete Now    On Reboot Delete Status
-----------------------------------------------------------------
be1                        yes      yes    yes       no
be10                       yes      yes    yes       no
#
```

Booting the Failsafe Archive on an x86 Based System

To boot the failsafe archive on a x86 based system, select the failsafe boot entry when the GRUB menu is displayed during a system boot. During the failsafe boot procedure, when prompted by the system, type `y` to update the primary boot archive.

Failsafe booting is also supported on systems that are booted from ZFS. When booting from a UFS-rooted BE, each BE has its own failsafe archive. The failsafe archive is located where the root file system is located, as is the case with a ZFS-rooted BE. On x86 based systems, each failsafe archive has an entry in the pool-wide GRUB menu. The default failsafe archive is the archive that is in the default bootable file system. The default bootable file system (dataset) is indicated by the value of the pool's `bootfs` property.

Another method that can be used to update the boot archives is to clear the boot-archive service. See "How to Update an Inconsistent Boot Archive by Clearing the `boot-archive` Service" on page 296. However, the preferred methods for updating the boot archives are to boot the failsafe archive or use the `bootadm` command. For more information, see the Chapter 14, "Managing the Solaris Boot Archives (Tasks)."

▼ How to Boot the Failsafe Archive on an x86 Based System by Using GRUB

Note – The GRUB failsafe interaction in some Solaris releases prompts you to update the boot archives, regardless of whether any inconsistent boot archive are detected. In this Solaris release, the system only prompts you to update the boot archives if an inconsistent boot archive is detected.

1 **Stop the system by using one of the methods described in the procedure, "x86: How to Stop a System for Recovery Purposes" on page 288.**

2 **If the system displays the `Press any key to reboot` prompt, press any key to reboot the system.**

You can also use the Reset button at this prompt. Or, you can use the power switch to reboot the system.

When the boot sequence begins, the GRUB menu is displayed.

```
GNU GRUB  version 0.95  (637K lower / 3144640K upper memory)
 +-------------------------------------------------------------------+
| be1
| be1 failsafe
| be3
| be3 failsafe
| be2
| be2 failfafe
 +----------------------------------------------------------------+
     Use the ^ and v keys to select which entry is highlighted.
     Press enter to boot the selected OS, 'e' to edit the
     commands before booting, or 'c' for a command-line.
```

Note – The GRUB menu that is displayed may vary, depending on the Solaris release you are running.

---

3 **Use the arrow keys to navigate the GRUB menu to select a failsafe entry.**

4 **Press Return to boot the failsafe archive.**

The system searches for installed OS instances. If an inconsistent boot archive is detected, a message similar to the following is displayed:

```
Searching for installed OS instances...

    An out of sync boot archive was detected on /dev/dsk/c0t0d0s0.
    The boot archive is a cache of files used during boot and
    should be kept in sync to ensure proper system operation.

    Do you wish to automatically update this boot archive? [y,n,?]
```

5 **Type** y **to update the boot archive.**

If multiple inconsistent boot archives are detected, the system will prompt you to type y to update each inconsistent boot archive.

For each archive that is updated successfully, the following message is displayed:

```
Updating boot archive on /dev/dsk/c0t0d0s0.
    The boot archive on /dev/dsk/c0t0d0s0 was updated successfully.
```

After the boot archive is updated, the system searches again for all installed OS instances, then prompts you to select a device to mount on /a. Note that this same message is displayed when the system first boots if no inconsistent boot archives are detected.

```
Searching for installed OS instances...

Multiple OS instances were found. To check and mount one of them
read-write under /a, select it from the following list. To not mount
any, select 'q'.

  1  pool10:13292304648356142148     ROOT/be10
  2  rpool:14465159259155950256      ROOT/be01
```

```
Please select a device to be mounted (q for none) [?,??,q]:
```

- **If you choose not to mount a device, type `q` to continue to boot process.**
- **If you choose to mount a device, follow these steps:**

  a. **Type the number of the device and press Return.**

     The system mounts the device on `/a`, and returns you to a shell prompt.

  b. **Repair the critical system resource.**

  c. **When you are done repairing the critical system resource, unmount the device.**

     ```
     # umount /a
     ```

  d. **Reboot the system.**

     ```
     # reboot
     ```

▼ x86: How to Boot the Failsafe Archive to Forcibly Update a Corrupt Boot Archive

This procedure shows how to rebuild an inconsistent or corrupt boot archive in the event you are not prompted by the system to update the boot archive the system, or in the event of a system hang or looping sequence occurs.

1 **Stop the system by using one of the methods that are described in the procedure, "x86: How to Stop a System for Recovery Purposes" on page 288.**

2 **Reboot the system.**

```
# reboot
```

If the system displays the `Press any key to reboot` prompt, press any key to reboot the system.

You can also use the Reset button at this prompt.

When the boot sequence begins, the GRUB menu is displayed.

```
+-----------------------------------------------------------------+
 | Solaris 10.1... X86                                              |
 | Solaris failsafe                                                 |
 |                                                                  |
 |                                                                  |
 +-------------------------------------------------------------------+
      Use the  and  keys to select which entry is highlighted.
```

```
      Press enter to boot the selected OS, 'e' to edit the
      commands before booting, or 'c' for a command-line.
```

Note – The contents of the GRUB menus vary, depending on the Solaris release you are running.

3 **Use the arrow keys to navigate the GRUB menu, then select the failsafe entry. Press Return to boot the failsafe archive.**

If any boot archives are out of date, a message that is similar to the following is displayed:

```
Searching for installed OS instances...

    An out of sync boot archive was detected on /dev/dsk/c0t0d0s0.
    The boot archive is a cache of files used during boot and
    should be kept in sync to ensure proper system operation.

    Do you wish to automatically update this boot archive? [y,n,?]
```

4 **Type `y`, then press Enter to update the inconsistent boot archive.**

The system displays the following message:

```
Updating boot archive on /dev/dsk/c0t0d0s0.
    The boot archive on /dev/dsk/c0t0d0s0 was updated successfully.
```

If no inconsistent boot archives are found, a message that is similar to the following is displayed:

```
Searching for installed OS instances...

    Solaris 10.1... X86 was found on /dev/dsk/c0t0d0s0.
    Do you wish to have it mounted read-write on /a? [y,n,?]
```

This message is also displayed after any inconsistent boot archives are updated successfully.

5 **Mount the device that contains the corrupt boot archive on `/a` by typing the corresponding number of the device, then press Enter.**

Note – If any inconsistent boot archives were updated in the previous step, the device is already mounted on `/a`. Proceed to Step 6.

6 **To forcibly update the corrupt boot archive, type:**

```
# bootadm update-archive -f -R /a
```

7 **Unmount the device.**

```
# umount /a
```

8 Reboot the system.

```
# reboot
```

Using Fast Reboot on the x86 Platform (Task Map)

Task	Description	For Instructions
Initiate a fast reboot of the system.	Use the `reboot` command with `-f` option to initiate a fast reboot of the system.	"x86: How to Initiate a Fast Reboot of the System" on page 269
Use Fast Reboot to reboot to a specific UFS boot disk or a ZFS root pool.	The fast reboot capability can be used to reboot to a specific UFS boot disk or a specific ZFS root pool.	"x86: Initiating a Fast Reboot to a Specific UFS Boot Disk or a ZFS Root Pool" on page 271
Use Fast Reboot to initiate a reboot of a directly mounted root (`/`) disk or root dataset.	After mounting a root (`/`) disk or root dataset, you can initiate a fast reboot of the system.	"x86: How to Initiate a Fast Reboot of a Directly Mounted Root Disk or Root Dataset" on page 271
Initiate a fast reboot to an alternate BE.	Use the `reboot`command with the `-f` and `-e` options to fast reboot to an alternate BE. **Note –** Because the `-e` option of the reboot command has dependencies on Solaris Live Upgrade, this option is not currently supported in the OpenSolaris 2008.11 release.	"x86: Initiating a Fast Reboot to an Alternate Boot Environment" on page 272
Initiate a fast reboot by directly specifying an alternate dataset.	If you are running the OpenSolaris 2008.11 release, you cannot use the reboot `-f -e` command to reboot to an alternate BE. Instead, use the `reboot` command with the just the `-f` option, directly specifying which dataset to boot.	"x86: Initiating a Fast Reboot to an Alternate Boot Environment in the OpenSolaris 2008.11 OS" on page 273
Facilitate a fast reboot of the system by using the `uadmin` command.	The `uadmin` command has been modified to support the Fast Reboot feature. You can facilitate a fast reboot by using this method.	"x86: Facilitating a Fast Reboot by Using the `uadmin` Command" on page 273
Change the behavior of the `reboot` command to make Fast Reboot the default.	Adding the `/etc/fastreboot` file to a system enables the Fast Reboot feature by default.	"x86: Making Fast Reboot the Default Behavior of the `reboot` Command" on page 274
Troubleshoot issues and conditions that might prevent the Fast Reboot feature from working.	Under certain conditions, the fast reboot capability does not work. In some of these situations, a workaround is available.	"x86: Troubleshooting Conditions That Might Prevent Fast Reboot From Working" on page 275

x86: Fast Reboot Implementation

The following are key components of the Fast Reboot implementation:

- Two new options for the `reboot` command

`-f`	Initiates the fast reboot process, when used with the `reboot` command.
`-e`	Reboots the system to an alternate BE, when used in conjunction with the `reboot`command and the `-f` option.

  Note – The `-e` option cannot be used to fast reboot to an alternate BE in the OpenSolaris 2008.11 release. For instructions on rebooting to an alternate BE in this release, see "x86: Initiating a Fast Reboot to an Alternate Boot Environment in the OpenSolaris 2008.11 OS" on page 273.

- New `quiesce` Function

  The drivers' implementation of this function enables the driver to `quiesce` a device, so that at completion of the function, the driver no longer generates interrupts or access memory.

  See the `quiesce`(9E) and `dev_ops`(9S) man pages.

- New `uadmin` Function

  Fast Reboot also includes support for a new `uadmin` function, `AD_FASTREBOOT`. This function resets the system, enabling the `reboot` command to bypass both the BIOS and boot loader phases.

  For more information, see the `uadmin`(2)man page.

The following procedures and examples describe how to use the fast reboot capability on an x86 based system. For overview information, see the section, "x86: Introducing Fast Reboot" on page 194.

▼ x86: How to Initiate a Fast Reboot of the System

This procedure describes how to use the `reboot` command with `-f` option to initiate a fast reboot of an x86 based system.

1 **Become superuser or assume an equivalent role.**

2 **Initiate a fast reboot of the system:**

- **To reboot to a new kernel, you would type:**

```
# reboot -f -- '/platform/i86pc/new&hyphen;kernel&hyphen;name/amd64/unix -k'
```

- **To initiate a fast reboot using boot arguments from the previous boot, you would type:**

  ```
  # reboot -f
  ```

Note – The boot archive is derived from the kernel argument. In the event of a failure in the fast reboot path, such as insufficient memory, the normal reset path is used.

Example 12–14 x86: Using Fast Reboot to Reboot a 64-Bit Kernel

```
# reboot -f -- '/platform/i86pc/kernel/amd64/unix'
Oct 21 15:06:35 tonyspizza reboot: initiated by ... on /dev/console
Oct 21 15:06:36 /usr/lib/snmp/snmpdx: received signal 15
Fast reboot.
syncing file systems... done
SunOS Release 5.11 Version onnv-gate:2008-10-20 64-bit
Copyright 1983-2008 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.
DEBUG enabled
Hostname: tonyspizza
NIS domain name is lab.sfbay.sun.com
/dev/rdsk/c1d0s7 is clean
Reading ZFS config: done.
```

Example 12–15 x86: Using Fast Reboot Without Additional Boot Arguments

This example fast reboots a system using the boot arguments that were used for the previous boot.

```
# reboot -f
Oct 21 15:02:38 tonyspizza reboot: initiated by ... on /dev/console
Oct 21 15:02:38 tonyspizza rpcbind: rpcbind terminating on signal.
Oct 21 15:02:38 tonyspizza syslogd: going down on signal 15
Fast reboot.
syncing file systems... done
Loading kmdb...
SunOS Release 5.11 Version onnv-gate:2008-10-20 64-bit
Copyright 1983-2008 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.
DEBUG enabled
Hostname: tonyspizza
NIS domain name is mpklab.sfbay.sun.com
/dev/rdsk/c1d0s7 is clean
Reading ZFS config: done.
```

x86: Initiating a Fast Reboot to a Specific UFS Boot Disk or a ZFS Root Pool

You can specify an alternate UFS boot disk in any of the following ways:

```
# reboot -f -- '/dev/dsk/c0t0s3'
# reboot -f -- '/dev/dsk/c0t0s3 -k'
# reboot -f -- '/dev/dsk/c0t0s3
# /platform/i86pc/mykernel/amd64/unix -k'
```

You can specify a ZFS root dataset in any of the following ways:

```
# reboot -f -- 'rpool/zfsbe1'
# reboot -f -- 'rpool/zfsbe2 -k'
# reboot -f -- 'rpool/zfsbe3 /platform/i86pc/mykernel/amd64/unix -k'
```

Note – When rebooting to a different root (`/`) disk or root dataset by using a mount point or a boot environment, be aware that no transient menu entry is added to the `menu.lst` file.

▼ x86: How to Initiate a Fast Reboot of a Directly Mounted Root Disk or Root Dataset

You can use Fast Reboot to directly mount a root (`/`) disk or root dataset, then reboot to it:

1 **Become superuser or assume an equivalent role.**

2 **Mount the root (`/`) disk.**

For example:

```
# mount /dev/dsk/c1d0s0 /mnt
```

- **To mount a root dataset, type:**

  ```
  # zfs mount rpool/dataset
  ```

3 **Reboot the mounted disk or mounted dataset.**

For example:

```
# reboot -f -- '/mnt/platform/i86pc/kernel/amd64/unix'
```

x86: Initiating a Fast Reboot to an Alternate Boot Environment

You can optionally use the `reboot` command with the `-f` and `-e` options to specify an alternate BE.

```
# reboot -f -e alternate-be-name
```

Note – The `-e` option has dependencies on Solaris Live upgrade packages, in particular the `lumount` and `luumount` commands. Because Solaris Live Upgrade is not supported in the OpenSolaris release, you cannot use this option to specify an alternate BE. Instead, use the `-f` option by itself to directly specifying the alternate dataset. See "x86: Initiating a Fast Reboot to an Alternate Boot Environment in the OpenSolaris 2008.11 OS" on page 273.

EXAMPLE 12–16 x86: Using Fast Reboot to Reboot to an Alternate Boot Environment

This example shows how to fast reboot to an alternate BE by using the `reboot` command with the `-f` and the`-e` options. Note that in this example, the `bootadm list&hyphen;menu` command is used to display a list of the bootable environments that are available on a system. A fast reboot of the `s3` BE is then initiated.

```
# bootadm list-menu
The location for the active GRUB menu is: /boot/grub/menu.lst
default 0
timeout 10
0 Solaris Express Community Edition snv_82 X86
1 Solaris xVM
2 Solaris failsafe
3 s0
4 s0 Solaris xVM
5 s0 failsafe
6 s4
7 s4 Solaris xVM
8 s4 failsafe
9 s3
10 s3 Solaris xVM
11 s3 failsafe

# reboot -f -e s3

reboot: Halting 1 zone.
Oct 21 15:16:51 tonyspizza reboot: initiated by ... on /dev/console
reboot: Completing system halt.
Oct 21 15:16:57 tonyspizza syslogd: going down on signal 15
```

EXAMPLE 12–16 x86: Using Fast Reboot to Reboot to an Alternate Boot Environment *(Continued)*

```
Fast reboot.
syncing file systems... done
SunOS Release 5.11 Version snv_99 64-bit
Copyright 1983-2008 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.
DEBUG enabled
Configuring devices.
Hostname: tonyspizza
NIS domain name is lab.sfbay.sun.com
Loading smf(5) service descriptions: 2/2
/dev/rdsk/c1d0s7 is clean
Reading ZFS config: done.
```

x86: Initiating a Fast Reboot to an Alternate Boot Environment in the OpenSolaris 2008.11 OS

The `-e` option of the `reboot` command is not supported in the OpenSolaris 2008.11 release. To fast reboot to an alternate BE in this release, use the `reboot -f` command, directly specifying which dataset to boot. The following examples show how to fast reboot to an alternate BE by using this method.

For example to fast reboot to the `zfsbe1` boot environment, you would type:

```
# reboot -f -- 'rpool/zfsbe1'
```

To fast reboot to the `zfsbe3` boot environment, in 64-bit mode, with the kernel debugger enabled, you would type:

```
# reboot -f -- 'rpool/zfsbe3 /platform/i86pc/kernel/amd64/unix -k'
```

x86: Facilitating a Fast Reboot by Using the `uadmin` Command

To facilitate the use of the new `-f` option of the `reboot` command, the `AD_FASTREBOOT` function has been added to the current function list for the `uadmin` command. This function is recognized by commands that utilize these function numbers.

For example, to reset the system using the current boot arguments by using the fast reboot path, you would type:

```
# uadmin 2 8
```

Caution – Using the `uadmin` command to fast reboot a system does not update the boot archive or the `menu.lst` file.

For more information about this function, see the `uadmin`(1M) man page.

x86: Making Fast Reboot the Default Behavior of the `reboot` Command

To make a fast reboot the default behavior on your system, create a `fastreboot` file in the `/etc` directory.

```
# touch /etc/fastreboot
```

The addition of the `fastreboot` file on the system changes the default behavior of the `reboot` command, thereby eliminating the need to use the `-f` option to initiate a fast reboot.

To revert to the original behavior of the `reboot` command, remove the file.

```
# rm /etc/fastreboot
```

Note that removing this file does not remove fast reboot capability from the system.

x86: Troubleshooting Conditions That Might Prevent Fast Reboot From Working

The following are possible conditions under which the Fast Reboot feature might not work:

- Driver does not implement the `quiesce` function

  If you attempt a fast reboot of a system with an unsupported driver, a message similar to the following is displayed:

  ```
  Sep 18 13:19:12 too-cool genunix: WARNING: nvidia has no quiesce()
  reboot: not all drivers have implemented quiesce(9E)
  ```

  If the graphics drivers are the only drivers that do not support the `quiesce` function, you can attempt to force a fast reboot by running the following commands:

  ```
  # echo "force_fastreboot/W 1" | mdb -kw
  # echo "set force_fastreboot = 1" >> /etc/system
  ```

  If the driver for the Network Interface Card (NIC) does not implement the `quiesce` function, you can try to unplumb the interface first, then attempt a fast reboot of the system.

  ```
  # ifconfig your-nic-interface unplumb
  # reboot -f
  ```

- Insufficient memory

  If there is not enough memory on the system, below 1G (0x40000000) for building the page tables, or not enough free memory to load the new kernel and the boot archive, the fast reboot attempt fails with the following messages, then falls back to a regular reboot.

  ```
  Fastboot: Couldn’t allocate size below PA 1G to do fast reboot
  Fastboot: Couldn’t allocate size below PA 64G to do fast reboot
  ```

- Unsupported environment

  Fast reboot functionality is not currently supported in the following environments:

  - Solaris xVM dom0 domains
  - Solaris xVM PV domU domains
  - Non&hyphen;global zones

Booting an x86 Based System from the Network

This section describes the requirements and warnings for performing a GRUB based boot from the network.

Any system can boot from the network, if a boot server is available. You might need to boot a stand-alone system from the network for recovery purposes if the system cannot boot from the local disk. You can boot a Solaris OS x86 based system directly from a network that supports the PXE network boot protocol.

Note – The PXE network boot is available only for devices that implement the Intel Preboot Execution Environment specification.

The default network boot strategy that is used for a GRUB based PXE network boot is DHCP. For non-PXE devices, you can use either the DHCP or the RARP boot strategy. The strategy that you use depends on which type of boot server is available on your network. If no PXE or DHCP server is available, you can load GRUB from a diskette, a CD-ROM, or a local disk.

To perform a GRUB based network boot, a DHCP server that is configured for PXE clients is required. A boot server that provides `tftp` service is also required. The DHCP server supplies the information that the client needs to configure its network interface.

The DHCP server must be able to respond to the DHCP classes, `PXEClient` and `GRUBClient` with the following information:

- IP address of the file server
- Name of the boot file (`pxegrub`)

The sequence for performing a PXE network boot of the Solaris OS is as follows:

1. The BIOS is configured to boot from a network interface.
2. The BIOS sends a DHCP request.
3. The DHCP server replies with the server address and the name of the boot file.
4. The BIOS downloads `pxegrub` by using `tftp` and executes `pxegrub`.
5. The system downloads a GRUB menu file by using `tftp`.

   This file displays the boot menu entries that are available.
6. After you select a menu entry, the system begins to load the Solaris OS.

See "How to Set Up a Network Configuration Server" in *System Administration Guide: IP Services* for more information.

Running the `add_install_client` command creates the `/tftpboot_01`*ethernet-address* file. This file is linked to `pxegrub` and the`/tftpboot/menu.lst.01`*ethernet-address* file. The `/tftpboot/menu.lst.01`*ethernet-address* file is the GRUB menu file. If this file does not exist, then `pxegrub` reverts to using DHCP Option 150, if this option is specified, or the `/tftpboot/boot/grub/menu.lst` file. Typically, a single system is set up to serve both functions. In this instance, the `add_install_client` command sets up the `/tftpboot` file with the correct `pxegrub` menu file and the Solaris files. DHCP service is handled separately by using the `add_install_client` command. The setup only needs to be completed once per client. See "x86: About DHCP Macros" on page 277 and "x86: How to Perform a GRUB Based Boot From the Network" on page 278 for more information.

x86: About DHCP Macros

When you add clients with the `add_install_client -d` script on the install server, the script reports DHCP configuration information to standard output. You can use this information when you create the options and macros that are needed to pass network installation information to clients.

To install DHCP clients with a DHCP server over the network, you must create DHCP options. This information is needed to install the Solaris OS.

When a client sends a DHCP request, the server must have the following client information:

- Client's ID, which is typically the Ethernet address
- Class of the client request
- Subnet the client resides on

The Solaris DHCP server forms a response. This response is based on the following *macros*, which matches the client request:

class macro	The class macro is based on a *class string* that is contained in the DHCP request. On x86 based systems, the BIOS already makes a DHCP request with the class `PXEClient:Arch:00000:UNDI:002001`. If a macro by this name is defined in the DHCP server configuration, then the macro content is sent to the x86 based clients.
network macro	The network macro is named by the IP address of the subnet that the client resides on. If the macro `129.146.87.0` is defined on the DHPC server, the macro content is sent to all clients on that subnet. The macro content is sent, regardless of the class of the request. If an option is defined in both the class macro and the network macro, the network macro takes precedence.
IP macro	The IP macro is named by an IP address. This macro is rarely used
client macro	The client macro is named by the client type (01 for Ethernet) and the mac address of the client, in uppercase letters. For a client with the Ethernet

address 0:0:39:fc:f2:ef, the corresponding macro name is 01000039FCEF. Note the absence of colons in the client macro.

For example, for a client on the subnet 192.168.100.0, with the Ethernet address 0:0:39:fc:f2:ef, making a DHCP request of class PXEClient, the DHCP server has the following matching macro:

```
PXEClient
    BootSrvA:  192.168.100.0
    BootFile:  pxegrub
  129.146.87.0
    Router:    129.146.87.1
    NISdmain:  sunsoft.eng.sun.com
  01000039FCEF
    BootFile:  01000039FCEF
The actual DHCP response will be
    BootSrvA:  192.168.100.0
    BootFile:  01000039FCEF
    Router:    129.146.87.1
    NISdmain:  sunsoft.eng.sun.com
```

Note that the BootFile in the client macro overrides the BootFile in the class macro.

For more detailed information, see "Preconfiguring System Configuration Information With the DHCP Service (Tasks)" in *Solaris Express Installation Guide: Network-Based Installations*.

▼ x86: How to Perform a GRUB Based Boot From the Network

To perform a GRUB based network boot a DHCP server that is configured for PXE clients is required. A boot server that provides tftp service is also required. The DHCP server must be able respond to the DHCP classes, PXEClient and GRUBClient to obtain the IP address of the file server and the boot file (pxegrub). By default, the menu file is /tftpboot/menu.lst.01*ethernet-address*. If this file does not exist, then pxegrub reverts to DHCP Option 150, if this option is specified, or the /tftpboot/boot/grub/menu.lst file.

If you are booting the system from the Solaris Software 1 CD or DVD, the system boots automatically.

Before You Begin Before performing a network boot on an x86 based system with GRUB, do the following:

- Run the appropriate commands on the installation server to enable the system to boot from the network.
- Add the client system as an install client.

See Chapter 4, "Installing From the Network (Overview)," in *Solaris Express Installation Guide: Network-Based Installations* for more information.

1 **On the DHCP server, create a client macro for the DHCP service with the following two options:**

- `BootSrvA`: *svr-addr*
- `BootFile`: *client-macro*

  Note that you must have superuser privileges on the DHCP server to run the `dhtadm` command.

  where *svr-addr* is the IP address of the server, and *client-macro* is named by the client's Ethernet type (01) and the mac address, in uppercase letters. This number is also the name of the file that is used in the `/tftpboot` directory on the installation server.

  Note – The notation for the *client-macro* should not contain any colons.

  You can create the client macro from the DHCP GUI or from command-line interface.

  To create the client macro from the command-line, type:

```
# dhtadm -[MA] -m client macro -d
":BootFile=client-macro:BootSrvA=svr-addr:"
```

2 **Reboot the system.**

3 **Instruct the BIOS to boot from the network.**

- If your system uses a specific keystroke sequence to boot from the network, type the keystrokes when the BIOS screen is displayed.
- If you need to manually modify the BIOS settings to boot from the network, type the keystroke sequence to access the BIOS setup utility. Then, modify the boot priority to boot from the network.

4 **When the GRUB menu is displayed, select the network installation image that you want to install.**

◆ ◆ ◆ CHAPTER 13

Troubleshooting Booting a Solaris System (Tasks)

This chapter describes the procedures for booting the Solaris release on SPARC and x86 based systems.

The following is a list of information that is in this chapter:

- "Troubleshooting Booting on the SPARC Platform (Task Map)" on page 281
- "Troubleshooting Booting on the x86 Platform (Task Map)" on page 287

Troubleshooting Booting on the SPARC Platform (Task Map)

Task	Description	For Instructions
Stop a system for recovery purposes.	If a damaged file is preventing the system from booting normally, first stop the system to attempt recovery	"SPARC: How to Stop the System for Recovery Purposes" on page 282
Force a crash dump of and reboot of the system.	You can force a crash dump and reboot of the system as a troubleshooting measure.	"SPARC: How to Force a Crash Dump and Reboot of the System" on page 283
Boot a SPARC based system for recovery purposes.	Boot to repair an important system file that is preventing the system from booting successfully.	"SPARC: How to Boot a System for Recovery Purposes" on page 284
Boot a system with the kernel debugger.	You can the system with the kernel debugger to troubleshoot booting problems. Use the `kmdb` command to boot the system.	"SPARC: How to Boot the System With the Kernel Debugger (`kmdb`)" on page 286

You might need to use one or more of the following methods to troubleshoot problems that prevent the system from booting successfully.

- Troubleshoot error messages when the system boots.
- Stop the system to attempt recovery.
- Boot a system for recovery purposes.

- Force a crash dump and reboot of the system.
- Boot the system with the kernel debugger by using the kmdb command.

▼ SPARC: How to Stop the System for Recovery Purposes

1 **Type the Stop key sequence for your system.**

The monitor displays the ok PROM prompt.

```
ok
```

The specific Stop key sequence depends on your keyboard type. For example, you can press Stop-A or L1-A. On terminals, press the Break key.

2 **Synchronize the file systems.**

```
ok sync
```

3 **When you see the `syncing file systems...` message, press the Stop key sequence again.**

4 **Type the appropriate `boot` command to start the boot process.**

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

5 **Verify that the system was booted to the specified run level.**

```
# who -r
 .       run-level s  May  2 07:39     3      0  S
```

Example 13–1 SPARC: Stopping the System for Recovery Purposes

```
    Press Stop-A
ok sync
syncing file systems...
    Press Stop-A
ok boot
```

SPARC: Forcing a Crash Dump and Reboot of the System

Forcing a crash dump and reboot of the system are sometimes necessary for troubleshooting purposes. The savecore feature is enabled by default.

For more information about system crash dumps, see Chapter 17, "Managing System Crash Information (Tasks)," in *System Administration Guide: Advanced Administration*.

▼ SPARC: How to Force a Crash Dump and Reboot of the System

Use this procedure to force a crash dump of the system. The example that follows this procedure shows how to use the `halt -d` command to force a crash dump of the system. You will need to manually reboot the system after running this command.

1. **Type the stop key sequence for your system.**

   The specific stop key sequence depends on your keyboard type. For example, you can press Stop-A or L1-A. On terminals, press the Break key.

   The PROM displays the `ok` prompt.

2. **Synchronize the file systems and write the crash dump.**

   ```
   > n
   ok sync
   ```

   After the crash dump is written to disk, the system will continue to reboot.

3. **Verify the system boots to run level 3.**

   The login prompt is displayed when the boot process has finished successfully.

   hostname `console login:`

Example 13–2 SPARC: Forcing a Crash Dump and Reboot of the System by Using the `halt -d` Command

This example shows how to force a crash dump and reboot of the system `jupiter` by using the `halt -d` and `boot` command. Use this method to force a crash dump and reboot of the system.

```
# halt -d
Jul 21 14:13:37 jupiter halt: halted by root

panic[cpu0]/thread=30001193b20: forced crash dump initiated at user request

000002a1008f7860 genunix:kadmin+438 (b4, 0, 0, 0, 5, 0)
  %l0-3: 0000000000000000 0000000000000000 0000000000000004 0000000000000004
  %l4-7: 00000000000003cc 0000000000000010 0000000000000004 0000000000000004
000002a1008f7920 genunix:uadmin+110 (5, 0, 0, 6d7000, ff00, 4)
  %l0-3: 0000030002216938 0000000000000000 0000000000000001 0000004237922872
  %l4-7: 000000423791e770 0000000000004102 0000030000449308 0000000000000005

syncing file systems... 1 1 done
dumping to /dev/dsk/c0t0d0s1, offset 107413504, content: kernel
100% done: 5339 pages dumped, compression ratio 2.68, dump succeeded
Program terminated
ok boot
Resetting ...
```

```
Sun Ultra 5/10 UPA/PCI (UltraSPARC-IIi 333MHz), No Keyboard
OpenBoot 3.15, 128 MB memory installed, Serial #10933339.
Ethernet address 8:0:20:a6:d4:5b, Host ID: 80a6d45b.

Rebooting with command: boot
Boot device: /pci@1f,0/pci@1,1/ide@3/disk@0,0:a
File and args: kernel/sparcv9/unix
SunOS Release 5.10 Version s10_60 64-bit
Copyright 1983-2004 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.
configuring IPv4 interfaces: hme0.
add net default: gateway 172.20.27.248
Hostname: jupiter
The system is coming up.  Please wait.
NIS domain name is example.com
.
.
.
System dump time: Wed Jul 21 14:13:41 2004
Jul 21 14:15:23 jupiter savecore: saving system crash dump
in /var/crash/jupiter/*.0
Constructing namelist /var/crash/jupiter/unix.0
Constructing corefile /var/crash/jupiter/vmcore.0
100% done: 5339 of 5339 pages saved

Starting Sun(TM) Web Console Version 2.1-dev...
.
.
.
```

▼ SPARC: How to Boot a System for Recovery Purposes

Use this procedure when an important file, such as `/etc/passwd`, has an invalid entry and causes the boot process to fail.

Use the stop sequence described in this procedure if you do not know the `root` password or if you can't log in to the system. For more information, see "SPARC: How to Stop the System for Recovery Purposes" on page 282.

Substitute the device name of the file system to be repaired for the *device-name* variable in the following procedure. If you need help identifying a system's device names, refer to "Displaying Device Configuration Information" in *System Administration Guide: Devices and File Systems*.

1 **Stop the system by using the system's Stop key sequence.**

2 **Boot the system in single-user mode.**

- Boot the system from the Solaris Software 1 CD or DVD,
  - Insert the Solaris installation media into the drive.
  - Boot from the installation media in single-user mode.

    ```
    ok boot cdrom -s
    ```
- Boot the system from the network if an installation server or remote CD or DVD drive is not available.

  ```
  ok boot net -s
  ```

3 **Mount the file system that contains the file with an invalid entry.**

```
# mount /dev/dsk/device-name /a
```

4 **Change to the newly mounted file system.**

```
# cd /a/file-system
```

5 **Set the terminal type.**

```
# TERM=sun
# export TERM
```

6 **Remove the invalid entry from the file by using an editor.**

```
# vi filename
```

7 **Change to the root (`/`) directory.**

```
# cd /
```

8 **Unmount the `/a` directory.**

```
# umount /a
```

9 **Reboot the system.**

```
# init 6
```

10 **Verify that the system booted to run level 3.**

The login prompt is displayed when the boot process has finished successfully.

```
hostname console login:
```

Example 13–3 SPARC: Booting a System for Recovery Purposes (Damaged Password File)

The following example shows how to repair an important system file (in this case, `/etc/passwd`) after booting from a local CD-ROM.

```
ok boot cdrom -s
# mount /dev/dsk/c0t3d0s0 /a
# cd /a/etc
# TERM=vt100
# export TERM
# vi passwd
    (Remove invalid entry)
# cd /
# umount /a
# init 6
```

Example 13–4 SPARC: Booting a System if You Forgot the `root` Password

The following example shows how to boot the system from the network when you have forgotten the `root` password. This example assumes that the network boot server is already available. Be sure to apply a new `root` password after the system has rebooted.

```
ok boot net -s
# mount /dev/dsk/c0t3d0s0 /a
# cd /a/etc
# TERM=vt100
# export TERM
# vi shadow
    (Remove root's encrypted password string)
# cd /
# umount /a
# init 6
```

▼ SPARC: How to Boot the System With the Kernel Debugger (`kmdb`)

This procedure shows you the basics for loading the kernel debugger (`kmdb`). For more detailed information, see the *Solaris Modular Debugger Guide*.

Note – Use the `reboot` and `halt` command with the `-d` option if you do not have time to debug the system interactively. To run the `halt` command with the `-d` option requires a manual reboot of the system afterwards. Whereas, if you use the `reboot` command, the system boots automatically. See the `reboot(1M)` for more information.

---

1 **Halt the system, causing it to display the `ok` prompt.**

To halt the system gracefully, use the `/usr/sbin/halt` command.

2 **Type either `boot kmdb` or `boot -k` to request the loading of the kernel debugger. Press return.**

3 Enter the kernel debugger.

The method used to enter the debugger is dependent upon the type of console that is used to access the system:

- If a locally attached keyboard is being used, press Stop-A or L1–A, depending upon the type of keyboard.
- If a serial console is being used, send a break by using the method that is appropriate for the type of serial console that is being used.

A welcome message is displayed when you enter the kernel debugger for the first time.

```
Rebooting with command: kadb
Boot device: /iommu/sbus/espdma@4,800000/esp@4,8800000/sd@3,0
.
.
.
```

Example 13–5 SPARC: Booting a System With the Kernel Debugger (`kmdb`)

```
ok boot kmdb
Resetting...

Executing last command: boot kmdb -d
Boot device: /pci@1f,0/ide@d/disk@0,0:a File and args: kmdb -d
Loading kmdb...
```

Troubleshooting Booting on the x86 Platform (Task Map)

Task	Description	For Instructions
Stop a system for recovery purposes.	If a damaged file is preventing the system from booting normally, first stop the system to attempt recovery	"x86: How to Stop a System for Recovery Purposes" on page 288
Force a crash dump of and reboot of the system.	You can force a crash dump and reboot of the system as a troubleshooting measure.	"x86: How to Force a Crash Dump and Reboot of the System" on page 289
Boot a system with the kernel debugger.	You can the system with the kernel debugger to troubleshoot booting problems. Use the `kmdb` command to boot the system.	"x86: How to Boot a System With the Kernel Debugger in the GRUB Boot Environment (`kmdb`)" on page 290

x86: Troubleshooting Error Messages Upon System Boot

If any multiboot entries that support directly booting the unix kernel are encountered by GRUB, the following message is displayed:

```
multiboot is no longer used to boot the Solaris Operating System.
The grub entry should be changed to:
kernel$ /boot/$ISADIR/xen.gz
module$ /platform/i86xpv/kernel/$ISADIR/unix /platform/i86xpv/kernel/$ISADIR/unix -B $ZFS-BOOTFS
module$ /platform/i86pc/$ISADIR/boot_archive
See http://www.sun.com/msg/SUNOS-8000-AK for details.
Press any key to reboot.
```

If the preceding message is displayed, you will need to update the entries in the GRUB `menu.lst` manually to successfully boot the system. More information can be found at `http://www.sun.com/msg/SUNOS-8000-AK` and the `boot`(1M) man page. For further information and instructions, see "x86: Error Messages Upon System Boot" on page 252.

▼ x86: How to Stop a System for Recovery Purposes

1. **Stop the system by using one of the following commands, if possible:**
   - If the keyboard and mouse are functional, become superuser. Then, type `init 0` to stop the system. After the `Press any key to reboot` prompt appears, press any key to reboot the system.
   - If the keyboard and mouse are functional, become superuser. then, type `init 6` to reboot the system.

2. **If the system does not respond to any input from the mouse or the keyboard, press the Reset key, if it exists, to reboot the system.**

   Or, you can use the power switch to reboot the system.

x86: Forcing a Crash Dump and Reboot of the System

Forcing a crash dump and reboot of the system are sometimes necessary for troubleshooting purposes. The `savecore` feature is enabled by default.

For more information about system crash dumps, see Chapter 17, "Managing System Crash Information (Tasks)," in *System Administration Guide: Advanced Administration*.

▼ x86: How to Force a Crash Dump and Reboot of the System

If you cannot use the `reboot -d` or the `halt -d` command, you can use the kernel debugger, `kmdb`, to force a crash dump. The kernel debugger must have been loaded, either at boot, or with the `mdb -k` command, for the following procedure to work.

Note – You must be in text mode to access the kernel debugger (`kmdb`). So, first exit any window system.

---

1 **Access the kernel debugger.**

The method used to access the debugger is dependent upon the type of console that you are using to access the system.

- If you are using a locally attached keyboard, press F1–A.
- If you are using a serial console, send a break by using the method appropriate to that type of serial console.

The `kmdb` prompt is displayed.

2 **To induce a crash, use the `systemdump` macro.**

```
[0]> $<systemdump
```

Panic messages are displayed, the crash dump is saved, and the system reboots.

3 **Verify that the system has rebooted by logging in at the console login prompt.**

Example 13–6 x86: Forcing a Crash Dump and Reboot of the System by Using `halt -d`

This example shows how to force a crash dump and reboot of the x86 based system `neptune` by using the `halt -d` and `boot` commands. Use this method to force a crash dump of the system. Reboot the system afterwards manually.

```
# halt -d
4ay 30 15:35:15 wacked.Central.Sun.COM halt: halted by user

panic[cpu0]/thread=ffffffff83246ec0: forced crash dump initiated at user request

fffffe80006bbd60 genunix:kadmin+4c1 ()
fffffe80006bbec0 genunix:uadmin+93 ()
fffffe80006bbf10 unix:sys_syscall32+101 ()

syncing file systems... done
dumping to /dev/dsk/c1t0d0s1, offset 107675648, content: kernel
NOTICE: adpu320: bus reset
100% done: 38438 pages dumped, compression ratio 4.29, dump succeeded
```

```
Welcome to kmdb
Loaded modules: [ audiosup crypto ufs unix krtld s1394 sppp nca uhci lofs
genunix ip usba specfs nfs md random sctp ]
[0]>
kmdb: Do you really want to reboot? (y/n) y
```

▼ x86: How to Boot a System With the Kernel Debugger in the GRUB Boot Environment (`kmdb`)

This procedure shows the basics for loading the kernel debugger (`kmdb`). The `savecore` feature is enabled by default. For more detailed information about using the kernel debugger, see the *Solaris Modular Debugger Guide*.

1 **Boot the system.**

The GRUB menu is displayed when the system is booted.

2 **When the GRUB menu is displayed, type `e` to access the GRUB edit menu.**

3 **Use the arrow keys to select the `kernel$` line.**

If you cannot use the arrow keys, use the ^ key to scroll up and the v key to scroll down.

4 **Type `e` to edit the line.**

The boot entry menu is displayed. In this menu, you can modify Solaris boot behavior by adding additional boot arguments to the end of the `kernel$` line.

5 **Type `-k` at the end of the line.**

6 **Press enter to return to the GRUB main menu.**

7 **Type `b` to boot the system with the kernel debugger enabled.**

8 **Access the kernel debugger.**

The method used to access the debugger is dependent upon the type of console that you are using to access the system:

- If you are using a locally attached keyboard, press F1–A.
- If you are using a serial console, send a break by using the method appropriate to that type of serial console.

A welcome message is displayed when you access the kernel debugger for the first time.

Example 13–7 x86: Booting a System With the Kernel Debugger (Hypervisor Support)

This example shows how to boot a 64-bit capable x86 based system, with the kernel debugger enabled:

```
kernel$ /boot/$ISADIR/xen.gz
module$ /platform/i86xpv/kernel/$ISADIR/unix /platform/i86xpv/kernel/$ISADIR/unix -B $ZFS-BOOTFS -k
```

This example shows how to boot a 64-bit capable x86 based system in 32-bit mode, with the kernel debugger enabled:

```
kernel$ /boot/xen.gz
module$ /platform/i86xpv/kernel/unix /platform/i86xpv/kernel/unix -B $ZFS-BOOTFS -k
```

Example 13–8 x86: Booting a System With the Kernel Debugger (Support for Directly Loading and Booting the `unix` Kernel)

The following examples apply to a Solaris release with GRUB support for directly loading and booting the `unix` kernel.

This example shows how to boot a 64-bit capable x86 based system, with the kernel debugger enabled.

```
kernel$ /platform/i86pc/kernel/$ISADIR/unix -B $ZFS-BOOTFS -k
```

This example shows how to boot a 64-bit capable x86 based system in 32-bit mode, with the kernel debugger enabled.

```
kernel$ /platform/i86pc/kernel/unix -B $ZFS-BOOTFS -k
```

Example 13–9 x86: Booting a System With the Kernel Debugger (GRUB Multiboot Implementation)

This example shows how to manually boot a 64-bit capable x86 based system with the kernel debugger enabled.

```
kernel$ /platform/i86pc/multiboot kernel/amd64/unix -k -B $ZFS-BOOTFS
```

This example shows how to boot a 64-bit capable x86 based system 32-bit mode with the kernel debugger enabled.

```
kernel$ /platform/i86pc/multiboot kernel/unix -k -B $ZFS-BOOTFS
```

CHAPTER 14

Managing the Solaris Boot Archives (Tasks)

This chapter describes boot archive management in the Solaris OS. Procedures for using the `bootadm` command are described in detail.

The following is a list of the information in this chapter:

- "Managing the Solaris Boot Archives (Task Map)" on page 293
- "Description of the Solaris Boot Archives" on page 294
- "Managing the `boot-archive` Service" on page 296
- "Using the `bootadm` Command to Manage the Boot Archives" on page 299

For overview information about the boot process, see Chapter 9, "Shutting Down and Booting a System (Overview)." For step-by-step instructions on booting a system, see Chapter 12, "Booting a Solaris System (Tasks)."

Managing the Solaris Boot Archives (Task Map)

TABLE 14–1 Solaris Boot Archive Management: Task Map

Task	Description	For Information
Manage the `boot-archive` service.	The `boot-archive` service is controlled by the Service Management Facilty (SMF). Use the `svcadm` command to enable and disable services. Use the `svcs` command to verify whether the `boot-archive` service is running.	"Managing the `boot-archive` Service" on page 296

TABLE 14–1 Solaris Boot Archive Management: Task Map *(Continued)*

Task	Description	For Information
Clear the `boot-archive` service.	Use this procedure as an alternate to booting the failsafe archive. After the `boot-archive` service is cleared, the `bootadm` command runs silently to update the boot archives.	"How to Update an Inconsistent Boot Archive by Clearing the `boot-archive` Service" on page 296
Update the boot archives by using the `bootadm` command.	Use the `bootadm update-archive` command to manually update the boot archive.	"How to Manually Update the Boot Archive" on page 299
Manually update the boot archive for a mirrored root (`/`) partition.	On systems that use a metadevice mirror for the root (`/`) partition, booting the failsafe archive and running the `bootadm update-archive` command to manually update the boot archive fails. This problem occurs because the mirror is a metadevice. Consequently, you must manually update the boot archive.	"How to Manually Update the Boot Archive on a RAID-1 (Mirror) Volume" on page 300
List the contents of the boot archives by using the `bootadm` command.	Use the `bootadm list-archive` command to list the contents of the boot archive.	"How to List Contents of the Boot Archive" on page 306
x86 only: Locate the active GRUB menu by using the `bootadm` command.	Use the `bootadm list-menu` command to determine the location of the active GRUB menu.	"x86: How to Locate the Active GRUB Menu and List Current Menu Entries" on page 306
x86 only: Set the default boot entry in the GRUB menu by using the `bootadm` command.	Use the `bootadm set-menu` command to set the default boot entry in the GRUB menu.	"x86: How to Set the Default Boot Entry for the Active GRUB Menu" on page 307

Description of the Solaris Boot Archives

When you install the Solaris OS on a system, the `bootadm` command creates one primary boot archive and one failsafe archive.

A *primary boot archive* is a subset of a root (`/`) file system. This boot archive contains all of the kernel modules, `driver.conf` files, in addition to a few configuration files. These files are located in the `/etc` directory. The files in the boot archive are read by the kernel before the root (`/`) file system is mounted. After the root (`/`) file system is mounted, the boot archive is discarded by the kernel from memory. Then, file I/O is performed against the root device.

Note – If you are running a Solaris Express release on an x86 based system, two primary boot archives (one 32-bit archive and one 64-bit archive) are created at installation time. The 32-bit archive is located in `/platform/i86pc/boot_archive`. The 64-bit archive is located in `/platform/i86pc/amd64/boot_archive`.

The files that make up the SPARC boot archives are located in the `/platform` directory.

The contents of this directory are divided into three groups of files:

- Files that are required for a `sun4u` boot archive
- Files that are required for a `sun4v` boot archive
- Files that are required for a `sun4us` boot archive

The files that make up the x86 boot archives are located in the `/platform/i86pc` directory.

If you are running the Solaris Express release, the contents of this directory are divided into two groups of files:

- 32-bit boot archive files, which are located in `/platform/i86pc/boot_archive`
- 64-bit boot archive files, which are located in `/platform/i86pc/amd64/boot_archive`

To list the files and directories that are included in the boot archives, use the `bootadm list-archive` command.

If any files in the archive are updated, the boot archive must be rebuilt. For modifications to take effect, the rebuild of the archive must take place before the next system reboot

The *failsafe* boot archive is the second type of archive that is created when you install the Solaris OS.

A failsafe boot archive has the following benefits and characteristics:

- Is self-sufficient
- Can boot on its own
- Is created by default during installation of the OS
- Requires no maintenance

For more information about booting a system in failsafe mode, see "Booting the Failsafe Archive on a SPARC Based System" on page 245 and "Booting the Failsafe Archive on an x86 Based System" on page 264.

Managing the boot-archive Service

The boot-archive service is controlled by the Service Management Facility (SMF). The boot-archive service instance is svc:/system/boot-archive:default. The svcadm command is used to enable and disable services.

To verify whether the boot-archive service is running, use the svcs command.

For more information, see the svcadm(1M) and the svcs(1) man pages.

▼ How to Enable or Disable the boot-archive Service

1 **Become superuser or assume an equivalent role.**

2 **To enable or disable the boot-archive service, type:**

```
# svcadm enable | disable system/boot-archive
```

3 **To verify the state of the boot-archive service, type:**

```
% svcs boot-archive
```

If the service is running, the output displays an online service state.

```
STATE          STIME    FMRI
online          9:02:38 svc:/system/boot-archive:default
```

If the service is not running, the output indicates the service is offline.

Troubleshooting For information about updating the boot archive by clearing the boot-archive service, see "How to Update an Inconsistent Boot Archive by Clearing the boot-archive Service" on page 296.

▼ How to Update an Inconsistent Boot Archive by Clearing the boot-archive Service

The boot-archive service, svc:/system/boot-archive, is managed by SMF. This procedure shows how to update the boot archive when an inconsistent archive is detected during the boot process. Clearing the service works the same as running the boot -F failsafe command. Note that when you use this method to update the boot archives, there is no need to boot the failsafe archive or run the bootadm update-archive command. This command runs silently after the boot-archive service has been cleared.

Caution – The preferred method for correcting an inconsistent boot archive is to boot the system in failsafe mode. See the following references for instructions on booting the failsafe archive:

For SPARC based systems, see "Booting a SPARC Based System From the Network" on page 250.

For x86 based systems, see "Booting the Failsafe Archive on an x86 Based System" on page 264.

1. **During the process of booting the system, if a warning similar to the following is displayed, ignore the warning.**

```
WARNING: The following files in / differ from the boot archive:

    changed file-name
```

The system will enter system maintenance mode.

2. **Clear the boot-archive service by typing the following command:**

```
# svcadm clear system/boot-archive
```

After this command is run, the `bootadm update-archive` command runs silently. If the boot archive is updated successfully, the system is rebooted.

3. **Verify the service is running.**

```
# svcs boot-archive
STATE          STIME    FMRI
online          9:02:38 svc:/system/boot-archive:default
```

Example 14–1 SPARC: Updating an inconsistent Boot Archive by Clearing the Boot-Archive Service

```
screen not found.
Can't open input device.
Keyboard not present.  Using ttya for input and output.

Sun Enterprise 220R (2 X UltraSPARC-II 450MHz), No Keyboard
OpenBoot 3.23, 1024 MB memory installed, Serial #13116682.
Ethernet address 8:0:20:c8:25:a, Host ID: 80c8250a.



Rebooting with command: boot
Boot device: /pci@1f,4000/scsi@3/disk@1,0:a  File and args:
SunOS Release 5.10 64-bit
Copyright 1983-2007 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.
DEBUG enabled
misc/forthdebug (507204 bytes) loaded
```

```
Hostname: marnie

WARNING: The following files in / differ from the boot archive:

    changed /kernel/drv/sd.conf

The recommended action is to reboot to the failsafe archive to correct
the above inconsistency. To accomplish this, on a GRUB-based platform,
reboot and select the "Solaris failsafe" option from the boot menu.
On an OBP-based platform, reboot then type "boot -F failsafe". Then
follow the prompts to update the boot archive. Alternately, to continue
booting at your own risk, you may clear the service by running:
"svcadm clear system/boot-archive"

Nov 21 15:47:20 svc.startd[100004]: svc:/system/boot-archive:default: Method
"/lib/svc/method/boot-archive" failed with exit status 95.
Nov 21 15:47:20 svc.startd[100004]: system/boot-archive:default failed fatally:
transitioned to maintenance (see 'svcs -xv' for details)
Requesting System Maintenance Mode
(See /lib/svc/share/README for more information.)
Console login service(s) cannot run

Root password for system maintenance (control-d to bypass):
single-user privilege assigned to /dev/console.
Entering System Maintenance Mode

Nov 21 15:48:36 su: 'su root' succeeded for root on /dev/console
Sun Microsystems Inc.   SunOS 5.10,
2007
.
.
.#
#
# svcadm clear system/boot-archive
#
# NIS domain name is mpklab.sfbay.sun.com
/dev/rdsk/c0t1d0s5 is clean
Reading ZFS config: done.
#
# bootadm update-archive
# svcs boot-archive
STATE          STIME    FMRI
online          9:02:38 svc:/system/boot-archive:default
```

Using the `bootadm` Command to Manage the Boot Archives

The `/sbin/bootadm` command enables you to perform the following tasks:

- Manually update the current boot archives on a system.
- List the files and directories that are included in the boot archives on a system.
- **x86 only:** Maintain the GRUB menu.
- **x86 only:** Locate the active GRUB menu, as well as the current GRUB menu entries.

The syntax of the command is as follows:

```
/sbin/bootadm [subcommand] [-option] [-R altroot]
```

For more information about the `bootadm` command, see the `bootadm`(1M) man page.

▼ How to Manually Update the Boot Archive

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 **To update the current boot archive, type:**

```
# bootadm update-archive
```

`bootadm`	Manages the boot archives on a system.
`update-archive`	Updates the current boot archive, if required. Applies to both SPARC and x86 based systems.

- **To update the boot archive on an alternate root, type:**

```
# bootadm update-archive -R /a
```

`-R` *altroot*	Specifies an alternate `root` path to apply to the `update-archive` subcommand.

Note – The root (`/`) file system of any non-global zone must not be referenced with the `-R` option. Doing so might damage the global zone's file system, compromise the security of the global zone, or damage the non-global zone's file system. See the `zones`(5) man page.

▼ How to Manually Update the Boot Archive on a RAID-1 (Mirror) Volume

---

Note – This procedure applies to updating the boot archive on RAID-1 (mirror) volumes that are created and maintained by using Solaris Volume Manager (SVM).

If the boot archive and the root (`/`) file system become inconsistent, an error message is displayed when you boot the system. Typically, the recommended action is to boot the system in failsafe mode, then run the `bootadm update-archive` command to update the boot archives. However, if the root (`/`) file system is a mirrored metadevice (RAID-1 volume), this method fails to successfully update the boot archive.

When you boot the system in failsafe mode, a message similar to the following is displayed:

```
Searching for installed OS instances...
/dev/dsk/c0t0d0s0 is under md control, skipping.
/dev/dsk/c1t3d0s0 is under md control, skipping.
No installed OS instance found.
```

This message indicates the metadevice was skipped. To manually update the boot archives, follow the steps that are described in the following procedure.

1 **On the system that has an inconsistent boot archive, become superuser or assume an equivalent role.**

2 **Boot the failsafe archive.**

- **On a SPARC based system, type:**

  ```
  # reboot -- "-F failsafe"
  ```

  To boot the failsafe archive from the `ok` prompt, type:

  ```
  ok boot -F failsafe
  ```

  For more information, see "How to Boot the Failsafe Archive on a SPARC Based System" on page 246.

- **On an x86 based system, boot the system, then select the failsafe boot entry in the GRUB menu.**

  For more information, see "How to Boot the Failsafe Archive on an x86 Based System by Using GRUB" on page 264.

The system boots in failsafe mode, searches for installed OS instances, then returns the message previously described, "No installed OS instance found". After the boot sequence completes, the command prompt is displayed.

3 **Use the `metastat` command to determine the primary submirror.**

```
# metastat -p
```

`-p`	Displays a list of active metadevices and hot spare pools. The `-p` output is designed for taking a snapshot of the configuration for later recovery or setup.

For example:

```
# metastat -p
d10 -m d0 d1 1
d0 1 1 c0t0d0s0
d1 1 1 c1t3d0s0
```

In the previous output, `d0` and `d1` are submirrors of `d10`. The primary submirror, which is typically listed first, is `d0`.

4 **Mount the primary submirror.**

For example:

```
# mount /dev/dsk/c0t0d0s0 /a
```

5 **Temporarily update the `/etc/vfstab` file to use a single root (`/`) partition.**

a. **Make a copy of the original `vfstab` file.**

```
# cp /a/etc/vfstab /a/etc/vfstab.orig
```

b. **Using a text editor, edit the `vfstab` file as follows:**

i. **Comment out the line for the root (`/`) mirror metadevice.**

```
#device        device        mount        FS    fsck    mount    mount
#to mount    to fsck        point        type    pass    at boot    options
#
.
.
.
#/dev/md/dsk/d10        /dev/md/rdsk/d10    /     ufs    1    no    -
```

In the previous example, the line, `/dev/md/dsk/d10`, was commented out.

ii. **Add a new line for the disk device of the primary submirror.**

```
#device        device        mount        FS    fsck    mount    mount
#to mount    to fsck        point        type    pass    at boot    options
#
.
.
.
#/dev/md/dsk/d10    /dev/md/rdsk/d10    /     ufs    1    no    -
/dev/dsk/c0t0d0s0    /dev/rdsk/c0t0d0s0  /     ufs    1    no    -
.
.
.
```

In the previous example, a new line for the disk device of the primary submirror, /dev/dsk/c0t0d0s0, was added.

c. **Save the changes.**

6 **To prevent the system from attempting to boot from the metadevice, temporarily update the /etc/system file as follows:**

a. **Make a copy of the original /etc/system file.**

```
# cp /a/etc/system /a/etc/system.orig
```

b. **Using a text editor, edit the /etc/system file, commenting out the rootdev line. This line is located between the Begin MDD root and the End MDD root lines.**

```
* Begin MDD root info (do not edit)
# rootdev:/pseudo/md@0:0,0,blk
* End MDD root info (do not edit)
```

c. **Save the changes.**

7 **Run the command to update the boot archive.**

```
# bootadm update-archive -R /a
```

8 **Unmount the primary submirror, then reboot the system.**

```
# umount /a
# shutdown -i 6
```

- **If the system still does not boot normally, reboot the failsafe archive and check the /etc/vfstab and the /etc/system files to make sure the information is correct.**

9 **After the system has successfully rebooted, rebuild the metadevice:**

a. **Identify the name of the root (/) mirror metadevice from the vfstab file.**

The name of the metadevice is the line that was commented out in Step 5.

b. **Display the components of the mirror by using the metastat command.**

For example:

```
# metastat -p
d10 -m d0 d1 1
d0 1 1 c0t0d0s0
d1 1 1 c1t3d0s0
```

c. **Detach the faulty submirror.**

```
# metadetach mirror submirror
```

For example:

```
# metadetach d10 d1
```

d. **Replace the existing copy of the /etc/vfstab file with the original file.**

```
# cp /a/etc/vfstab.orig /a/etc/vfstab
```

e. **Replace the existing copy of the /etc/system file with the original file.**

```
# cp /a/etc/system.orig /a/etc/system
```

f. **Reboot the system.**

```
# shutdown -i 6
```

After the system reboots, the mirrored root (/) partition is restored on the metadevice.

10 **Reattach the submirror that was detached in the previous step.**

```
# metattach mirror submirror
```

For example:

```
# metattach d10 d1
```

The mirror resynchronization begins.

11 **To check the status of the resynchronization process, use the metastat command:**

```
# metastat | grep 'Resync in progress'
```

When no output is returned, the process is finished.

Example 14–2 SPARC: Manually Updating the Boot Archive on a RAID-1 (Mirror) Volume

This example shows the steps for manually updating the boot archive on a system with an SVM root (/) mirrored metadevice. The system that was used for this example is a SPARC based system running the Solaris 10 10/08 FCS release.

```
SunOS Release 5.10 Version Generic_137137-09 64-bit
Copyright 1983-2008 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.
WARNING: Unexpected EOF on line 14 of /kernel/drv/md.conf
Hostname: pilgrim1

WARNING: The following files in / differ from the boot archive:

    changed /kernel/drv/md.conf

The recommended action is to reboot to the failsafe archive to correct
the above inconsistency. To accomplish this, on a GRUB-based platform,
reboot and select the "Solaris failsafe" option from the boot menu.
On an OBP-based platform, reboot then type "boot -F failsafe". Then
follow the prompts to update the boot archive. Alternately, to continue
booting at your own risk, you may clear the service by running:
"svcadm clear system/boot-archive"

Sep 18 15:22:06 svc.startd[7]: svc:/system/boot-archive:default:
Method "/lib/svc/method/boot-archive" failed with exit status 95.
Sep 18 15:22:06 svc.startd[7]: system/boot-archive:default
failed fatally: transitioned to maintenance (see 'svcs -xv' for details)
Requesting System Maintenance Mode
(See /lib/svc/share/README for more information.)
Console login service(s) cannot run

Root password for system maintenance (control-d to bypass):
single-user privilege assigned to /dev/console.
Entering System Maintenance Mode

Sep 18 15:22:18 su: 'su root' succeeded for root on /dev/console
Sun Microsystems Inc.    SunOS 5.10  Generic  January 2005
# reboot -- "-F failsafe"
syncing file systems... done
rebooting...
Resetting ...
Rebooting with command: boot -F failsafe
Boot device: /pci@1f,4000/scsi@3/disk@0,0:a  File and args: -F failsafe
SunOS Release 5.10 Version Generic_137137-08     64-bit
Copyright 1983-2008 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.
```

```
Configuring devices.
Searching for installed OS instances...
/dev/dsk/c0t0d0s0 is under md control, skipping.
/dev/dsk/c1t3d0s0 is under md control, skipping.
No installed OS instance found.

Starting shell.

pilgrim1# metastat -p
d10 -m d0 d1 1
d0 1 1 c0t0d0s0
d1 1 1 c1t3d0s0

# mount /dev/dsk/c0t0d0s0 /a
# cp /a/etc/vfstab /a/etc/vfstab.orig
# vi /a/etc/vfstab

<< input changes to vfstab file, then save changes >>

# cp /a/etc/system /a/etc/system.orig
# vi /a/etc/system

<< input changes to /etc/system file, then save changes >>

# bootadm update-archive -R /a
Creating boot_archive for /a
updating /a/platform/sun4u/boot_archive
15+0 records in
15+0 records out
# umount /a
# shutdown -i 6

<< reboot the system >>

Rebooting with command: boot
Boot device: /pci@1f,4000/scsi@3/disk@0,0:a  File and args:
SunOS Release 5.10 Version Generic_137137-08  64-bit
Copyright 1983-2008 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.
[...]
# metastat -p d10
# metadetach d10 d1
# cp /a/etc/vfstab.orig /a/etc/vfstab
# cp /a/etc/system.orig /a/etc/system
# shutdown -i 6

<< reboot the system >>
```

```
Rebooting with command: boot
Boot device: /pci@1f,4000/scsi@3/disk@0,0:a  File and args:
SunOS Release 5.10 Version Generic_137137-08 64-bit
Copyright 1983-2008 Sun Microsystems, Inc.  All rights reserved.
Use is subject to license terms.
[...]
# metattach d10 d1
# metastat | grep 'Resync in progress'
    Resync in progress: 4 % done
# metastat | grep 'Resync in progress'
```

▼ How to List Contents of the Boot Archive

1. **Become superuser or assume an equivalent role.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2. **To list the files and directories that are included in the boot archive, type:**

   ```
   # bootadm list-archive
   ```

`list-archive`	Lists the files and directories that are included in the boot archive or archives. Applies to both SPARC and x86 based systems.

▼ x86: How to Locate the Active GRUB Menu and List Current Menu Entries

Use this procedure to determine the location of the active GRUB menu and to list current GRUB menu entries.

1. **Become superuser or assume an equivalent role.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2. **To list the location of the active GRUB menu and current GRUB menu entries, type:**

   ```
   # bootadm list-menu
   ```

`list-menu`	Lists the location of the active GRUB menu, as well as the current GRUB menu entries. Information about the `autoboot-timeout`, the default entry number, and the title of each entry is included in this listing. Applies to x86 based systems *only*.

Example 14–3 Listing the Location of the Active GRUB Menu and Current GRUB Menu Entries

```
# bootadm list-menu
The location for the active GRUB menu is: /stubboot/boot/grub/menu.lst
default=0
timeout=10
(0) Solaris10
(1) Solaris10 Failsafe
(2) Linux
```

▼ x86: How to Set the Default Boot Entry for the Active GRUB Menu

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 **To set the default boot entry in the active GRUB menu, type:**

```
# bootadm set-menu menu-entry
```

`set-menu`	Maintains the GRUB menu. The location of the active GRUB menu is `boot/grub/menu.lst`. Applies to x86 bases systems *only*.
menu-entry	Specifies the GRUB menu entry to set as the default.

3 **To verify default menu entry has been changed, type:**

```
# bootadm list-menu
```

The new default menu entry should be displayed.

Example 14–4 Switching the GRUB Default Menu Entry

This example shows how to switch the default GRUB menu to one of the menu entries that is displayed in the previous example. The menu entry that is selected is The Linux, menu entry 2.

```
# bootadm set-menu default=2
```

See Also For a description of the `menu.lst` file in each GRUB implementation, see "x86: Supported GRUB Implementations" on page 315.

CHAPTER 15

x86: GRUB Based Booting (Reference)

This chapter contains information about x86 boot processes, including GRUB implementation details and additional GRUB reference information.

For overview information, see Chapter 9, "Shutting Down and Booting a System (Overview)."

For step-by-step instructions on booting a system, see Chapter 12, "Booting a Solaris System (Tasks)."

x86: Boot Processes

This section includes information about boot processes that are unique to booting an x86 based system.

x86: System BIOS

When a system is powered on, the system is controlled by the read-only-memory (ROM) Basic Input/Output System (BIOS). The BIOS is the firmware interface on Solaris Operating Systems that have x86 64-bit and 32-bit support.

Hardware adapters usually have an on-board BIOS that displays the physical characteristics of the device. The BIOS is used to access the device. During the startup process, the system BIOS checks for the presence of any adapter BIOS. If any adapters are found, the system then loads and executes each adapter BIOS. Each adapter's BIOS runs self-test diagnostics and then displays device information.

The BIOS on most systems has a user interface, where you can select an ordered list of boot devices that consists of the following selections:

- Diskette
- CD or DVD

- Hard disk
- Network

The BIOS attempts to boot from each device, in turn, until a valid device with a bootable program is found.

x86: Kernel Initialization Process

The `/platform/i86pc/multiboot` program is an `ELF32` executable that contains a header which is defined in the Multiboot Specification.

The multiboot program is responsible for performing the following tasks:

- Interpreting the content of boot archive
- Autodetection of systems that are 64-bit capable
- Selecting the best kernel mode for booting the system
- Assembling core kernel modules in memory
- Handing control of the system to the Solaris kernel

x86: Solaris Support for the GRUB Bootloader

The following sections contain additional reference information for administering GRUB in the Solaris OS

x86: GRUB Terminology

To thoroughly grasp GRUB concepts, an understanding of the following terms is essential.

Note – Some of the terms that are described in this list are not exclusive to GRUB based booting.

---

boot archive

A collection of critical files that is used to boot the Solaris OS. These files are needed during system startup before the root file system is mounted. Multiple boot archives are maintained on a system:

- A *primary boot archive* is used to boot the Solaris OS on an x86 based system.

Note – On the x86 platform, when you install the Solaris OS, two primary boot archives are created, one 32-bit archive and one 64-bit archive.

- A *failsafe boot archive* that is used for recovery when a primary boot archive is damaged. This boot archive starts the system without mounting the root file system. On the GRUB menu, this boot archive is called *failsafe*. The archive's primary purpose is to regenerate the primary boot archives, which are usually used to boot the system.

boot loader	The first software program that runs after you power on a system. This program begins the booting process.
failsafe archive	See boot archive.
GRUB	GNU GRand Unified Bootloader (GRUB) is an open-source boot loader with a menu interface. The menu displays a list of the operating systems that are installed on a system. GRUB enables you to easily boot these various operating systems, such as the Solaris OS, Linux, or Windows.
GRUB main menu	A boot menu that lists the operating systems that are installed on a system. From this menu, you can easily boot an operating system without modifying the BIOS or `fdisk` partition settings.
GRUB edit menu	A submenu of the GRUB main menu. GRUB commands are displayed on this submenu. These commands can be edited to change boot behavior.
`menu.lst` **file**	A configuration file that lists all the operating systems that are installed on a system. The contents of this file dictate the list of operating systems that is displayed in the GRUB menu. From the GRUB menu, you can easily boot an operating system without modifying the BIOS or `fdisk` partition settings.
miniroot	A minimal, bootable root (`/`) file system that resides on the Solaris installation media. A miniroot consists of the Solaris software that is required to install and upgrade systems. On x86 based systems, the miniroot is copied to the system to be used as the failsafe boot archive. See boot archive for details about the failsafe boot archive.
primary boot archive	See boot archive.

x86: Functional Components of GRUB

GRUB consists of the following functional components:

- `stage1` – Is an image that is installed on the first sector of the Solaris `fdisk` partition. You can optionally install `stage1` on the master boot sector by specifying the `-m` option with the `installgrub` command. See the `installgrub`(1M) man page and "Disk Management in the GRUB Boot Environment" in *System Administration Guide: Devices and File Systems* for more information.
- `stage2` – Is an image that is installed in a reserved area in the Solaris `fdisk` partition. The `stage2` image is the core image of GRUB.
- `menu.lst` file – Is typically located in the `/boot/grub` directory on systems with a UFS root and in the `/`*pool-name*`/boot/grub` directory on systems with a ZFS root. This file is read by the GRUB `stage2` file. For more information, see the section, "x86: Modifying Boot Behavior by Editing the `menu.lst` File" on page 223.

You cannot use the `dd` command to write `stage1` and `stage2` images to disk. The `stage1` image must be able to receive information about the location of the `stage2` image that is on the disk. Use the `installgrub` command, which is the supported method for installing GRUB boot blocks.

Naming Conventions That Are Used for Configuring GRUB

GRUB uses conventions that are slightly different from previous Solaris releases. Understanding the GRUB device-naming conventions can assist you in correctly specifying drive and partition information when you configure GRUB on your system.

The following table describes the GRUB device-naming conventions for this Solaris release.

TABLE 15–1 Conventions for GRUB Devices

Device Name	Description
`(fd0)`	First diskette
`(fd1)`	Second diskette
`(nd)`	Network device
`(hd0,0)`	First `fdisk` partition on first hard disk
`(hd0,1)`	Second `fdisk` partition on first hard disk
`(hd0,0,a)`,	Slice `a` on first `fdisk` partition on first hard disk
`(hd0,0,b)`	Slice `b` on first `fdisk` partition on first hard disk

Note – All GRUB device names must be enclosed in parentheses.

For more information about `fdisk` partitions, see "Guidelines for Creating an fdisk Partition" in *System Administration Guide: Devices and File Systems*.

Naming Conventions That Are Used by the `findroot` Command

Starting with the Solaris 10 10/08 release, the `findroot` command replaces the `root` command that was previously used by GRUB. The `findroot` command provides enhanced capabilities for discovering a targeted disk, regardless of the boot device. The `findroot` command also supports booting from a ZFS root file system This command replaces the `root` command that was formerly used by GRUB.

The following is a description of the device naming convention that is used by the `findroot` command for various GRUB implementations:

- Solaris Live Upgrade:

  ```
  findroot (BE_x,0,a)
  ```

  The *x* variable is the name of the boot environment.
- Standard system upgrades and new installations for systems with ZFS support:

  ```
  findroot(pool_p,0,a)
  ```

  The *p* variable is the name of the root pool.
- Standard system upgrades and new installations for systems with UFS support:

  ```
  findroot (rootfsN,0,a)
  ```

  The *N* variable is an integer number that starts at 0.

How Multiple Operating Systems Are Supported by GRUB

This section describes how multiple operating systems that are on the same disk are supported with GRUB. The following is an example of an x86 based system that has the Solaris 10 10/08 OS, the Solaris 9 OS, Linux, and Windows installed on the same disk.

TABLE 15–2 Sample GRUB Menu Configuration

Operating System	Location on Disk
Windows	`fdisk` partition 0
Linux	`fdisk` partition 1
Solaris	`fdisk` partition 2
Solaris 9 OS	Slice `0`
Solaris 10 10/08 OS	Slice 3

Based on the preceding information, the GRUB menu would look like the following:

```
title Solaris 10
            findroot (pool_rpool,0,a)
            kernel$ /platform/i86pc/multiboot -B $ZFS-BOOTFS
            module /platform/i86pc/boot_archive
title Solaris 9 OS (pre-GRUB)
            root (hd0,2,a)
            chainloader +1
            makeactive
title Linux
            root (hd0,1)
            kernel <from Linux GRUB menu...>
            initrd <from Linux GRUB menu...>
title Windows
            root (hd0,0)
            chainloader +1
```

Note – The Solaris slice must be the active partition. Also, do not indicate `makeactive` under the Windows menu. Doing so causes the system to boot Windows every time. Note that if Linux has installed GRUB on the master boot block, you cannot access the Solaris boot option. The inability to access the Solaris boot option occurs whether or not you designate it as the active partition.

In this case, you can do one of the following:

- Chain-load from the Linux GRUB by modifying the menu on Linux.

  Chain-loading is a mechanism for loading unsupported operating systems by using another boot loader.
- Replace the master boot block with the Solaris GRUB by running the `installgrub` command with the `-m` option:

  ```
  # installgrub -m /boot/grub/stage1 /boot/grub/stage2 /dev/rdsk/root-slice
  ```

  See the `installgrub`(1M) man page for more information.

For information about the Solaris Live Upgrade boot environment, see *Solaris Express Installation Guide: Solaris Live Upgrade and Upgrade Planning*.

x86: Supported GRUB Implementations

In the Solaris Express release, GRUB uses the direct boot implementation. The contents of the `menu.lst` file varies, depending on the Solaris release you are running, the installation method that is used, and whether you are booting the system from a ZFS root or a UFS root.

- **GRUB ZFS boot support**

  For a description of the `menu.lst` file and an example, see "Description of the `menu.lst` File (ZFS Support)" on page 315.
- **GRUB UFS boot support**

  For a description of the `menu.lst` file and an example, see "Description of the `menu.lst` File (UFS Support)" on page 316.
- **GRUB hypervisor support**

  For a description of the `menu.lst` file and instructions on booting an x86 based system with this implementation of GRUB, see "Description of a `menu.lst` File That Supports Hypervisor Technology" on page 317.

Note – In this implementation of GRUB, the `multiboot` module is no longer used.

Description of the `menu.lst` File (ZFS Support)

The following are various examples of a `menu.lst` file for a boot environment that contains a ZFS boot loader:

Note – Because the miniroot is mounted as the real root file system, the entry for failsafe booting in the `menu.lst` file does *not* change to the ZFS `bootfs` property, even if the failsafe archive is read from a ZFS dataset. The ZFS dataset is not accessed after the boot loader reads the miniroot.

EXAMPLE 15–1 Default `menu.lst` File (New Installation or Standard Upgrade)

```
title Solaris 11 s10x_90 X86
findroot (pool_rpool,0,a)
kernel$ /platform/i86pc/kernel/$ISADIR/unix -B $ZFS-BOOTFS
module$ /platform/i86pc/$ISADIR/boot_archive

title Solaris 11 failsafe
findroot (pool_rpool,0,a)
kernel /boot/platform/i86pc/kernel/unix -s -B console=ttyb
module /boot/x86.miniroot-safe
```

EXAMPLE 15–2 Default `menu.lst` File (Solaris Live Upgrade)

```
title be1
findroot (BE_be1,0,a)
bootfs rpool/ROOT/szboot_0508
kernel$ /platform/i86pc/kernel/$ISADIR/unix -B $ZFS-BOOTFS
module$ /platform/i86pc/$ISADIR/boot_archive

title be1 failsafe
findroot (BE_be1,0,a)
kernel /boot/platform/i86pc/kernel/unix -s -B console=ttyb
module /boot/x86.miniroot-safe
```

Description of the `menu.lst` File (UFS Support)

The following are examples of a `menu.lst` file on a system that supports booting from UFS.

EXAMPLE 15–3 Default GRUB `menu.lst` File (New Installation or Standard Upgrade)

```
title Solaris 10 5/08 s10x_nbu6wos_nightly X86
findroot (pool_rpool,0,a)
kernel /platform/i86pc/multiboot
module /platform/i86pc/boot_archive

title Solaris failsafe
findroot (rootfs0,0,a)
kernel /boot/multiboot kernel/unix -s -B console-ttyb
module /boot/x86.miniroot-safe
```

EXAMPLE 15–4 Default GRUB `menu.lst` File (Solaris Live Upgrade)

```
title be1
findroot (BE_be1,0,a)
kernel$ /platform/i86pc/kernel/$ISADIR/unix
module$ /platform/i86pc/$ISADIR/boot_archive

title be1 failsafe
findroot (BE_be1,0,a)
kernel /boot/platform/i86pc/kernel/unix -s -B console=ttyb
module /boot/x86.miniroot-safe
```

Description of a `menu.lst` File That Supports Hypervisor Technology

You can run the Solaris OS as a virtualized control domain, with the hypervisor. To run the Solaris release with this support, there must be an entry in `menu.lst` file that specifies the hypervisor. This entry can either be the default boot menu item, or you can select this entry manually at boot time. After you upgrade your system for the first time to a Solaris release that includes this support, the `bootadm` command automatically adds a GRUB `menu.lst` entry for the hypervisor.

The following are `menu.lst` entries for this GRUB implementation:

```
title Solaris on xVM
kernel$ /boot/$ISADIR/xen.gz
module$ /platform/i86xpv/kernel/$ISADIR/unix /platform/i86xpv/kernel/$ISADIR/unix -B $ZFS-BOOTFS
module$ /platform/i86pc/$ISADIR/boot_archive
```

- The `kernel$` line specifies a path to `xen.gz` file, followed by optional hypervisor arguments.
- The first `module$` line includes the path to UNIX twice, followed by any arguments for the Solaris dom0 kernel.
- The second `module$` line provides the path to the boot archive.

Note that the path to UNIX in the `menu.lst` entry for the hypervisor uses `i86xpv`, *not* `i86pc`. The options that are interpreted by the hypervisor are added to end of the `kernel$` line, after the `xen.gz` file information.

If you choose to run the Solaris release as a stand-alone OS, you can continue to use the same GRUB menu entries that you used previously.

For example:

```
title Solaris Nevada ... X86
kernel$ /platform/i86pc/kernel/$ISADIR/unix -B $ZFS-BOOTFS
module$ /platform/i86pc/$ISADIR/boot_archive
```

For more information about how to modify GRUB `menu.lst` entries, see “x86: How to Modify Boot Behavior by Editing the `menu.lst` File” on page 224.

◆ ◆ ◆ CHAPTER 16

Managing Services (Overview)

This chapter provides an overview of the Service Management Facility (SMF). In addition, information that is related to run levels is provided.

This is a list of the overview information in this chapter.

- "Introduction to SMF" on page 319
- "SMF Concepts" on page 321
- "SMF Administrative and Programming Interfaces" on page 325
- "SMF Components" on page 326
- "SMF Compatibility" on page 328
- "Run Levels" on page 328
- "`/etc/inittab` File" on page 330

For information on the procedures associated with SMF, see "Managing Services (Task Map)" on page 333. For information on the procedures associated with run levels, see "Using Run Control Scripts (Task Map)" on page 349.

Introduction to SMF

SMF provides an infrastructure that augments the traditional UNIX start-up scripts, `init` run levels, and configuration files. SMF provides the following functions:

- Automatically restarts failed services in dependency order, whether they failed as the result of administrator error, software bug, or were affected by an uncorrectable hardware error. The dependency order is defined by dependency statements.
- Makes services objects that can be viewed, with the new `svcs` command, and managed, with `svcadm` and `svccfg` commands. You can also view the relationships between services and processes using `svcs -p`, for both SMF services and legacy `init.d` scripts.
- Makes it easy to backup, restore, and undo changes to services by taking automatic snapshots of service configurations.

- Makes it easy to debug and ask questions about services by providing an explanation of why a service isn't running by using `svcs -x`. Also, this process is eased by individual and persistent log files for each service.
- Allows for services to be enabled and disabled using `svcadm`. These changes can persist through upgrades and reboots. If the `-t` option is used, the changes are temporary.
- Enhances the ability of administrators to securely delegate tasks to non-root users, including the ability to modify properties and enable, disable, or restart services on the system.
- Boots faster on large systems by starting services in parallel according to the dependencies of the services. The opposite process occurs during shutdown.
- Allows you to customize the boot console output to either be as quiet as possible, which is the default, or to be verbose by using `boot -m verbose`.
- Preserves compatibility with existing administrative practices wherever possible. For example, most customer and ISV-supplied rc scripts still work as usual.

Dependency statements define the relationships between services. These relationships can be used to provide precise fault containment by restarting only those services that are directly affected by a fault, rather than restarting all of the services. Another advantage of dependency statements is that the statements allow for scalable and reproducible initialization processes. In addition, by defining all of the dependencies, you can take advantage of modern, highly parallel machines, because all independent services can be started in parallel.

SMF defines a set of actions that can be invoked on a service by an administrator. These actions include enable, disable, refresh, restart, and maintain. Each service is managed by a service restarter which carries out the administrative actions. In general, the restarters carry out actions by executing methods for a service. Methods for each service are defined in the service configuration repository. These methods allow the restarter to move the service from one state to another state.

The service configuration repository provides a per-service snapshot at the time that each service is successfully started so that fallback is possible. In addition, the repository provides a consistent and persistent way to enable or disable a service, as well as a consistent view of service state. This capability helps you debug service configuration problems.

Changes in Behavior When Using SMF

Most of the features that are provided by SMF happen behind the scenes, so users are not aware of them. Other features are accessed by new commands. Here is a list of the behavior changes that are most visible.

- The boot process creates many fewer messages now. Services do not display a message by default when they are started. All of the information that was provided by the boot messages can now be found in a log file for each service that is in `/var/svc/log`. You can use the `svcs` command to help diagnose boot problems. In addition, you can use the `-v` option to the `boot` command, which generates a message when each service is started during the boot process.

- Since services are automatically restarted if possible, it may seem that a process refuses to die. If the service is defective, the service will be placed in maintenance mode, but normally a service is restarted if the process for the service is killed. The `svcadm` command should be used to stop the processes of any SMF service that should not be running.
- Many of the scripts in `/etc/init.d` and /etc/rc*.d have been removed. The scripts are no longer needed to enable or disable a service. Entries from `/etc/inittab` have also been removed, so that the services can be administered using SMF. Scripts and `inittab` entries that are provided by an ISV or are locally developed will continue to run. The services may not start at exactly the same point in the boot process, but they are not started before the SMF services, so that any service dependencies should be OK.

SMF Concepts

This section presents terms and their definitions within the SMF framework. These terms are used throughout the documentation. To grasp SMF concepts, an understanding of these terms is essential.

SMF Service

The fundamental unit of administration in the SMF framework is the *service instance*. Each SMF service has the potential to have multiple versions of it configured. As well, multiple instances of the same version can run on a single Solaris system. An *instance* is a specific configuration of a service. A web server is a service. A specific web server daemon that is configured to listen on port 80 is an instance. Each instance of the web server service could have different configuration requirements. The service has system-wide configuration requirements, but each instance can override specific requirements, as needed. Multiple instances of a single service are managed as child objects of the service object.

Services are not just the representation for standard long-running system services such as `in.dhcpd` or `nfsd`. Services also represent varied system entities that include ISV applications such as Oracle software. In addition, a service can include less traditional entities such as the following:

- A physical network device
- A configured IP address
- Kernel configuration information
- Milestones that correspond to system init state, such as the multiuser run level

Generically, a service is an entity that provides a list of capabilities to applications and other services, local and remote. A service is dependent on an implicitly declared list of local services.

A *milestone* is a special type of service. Milestone services represent high-level attributes of the system. For example, the services which constitute run levels S, 2, and 3 are each represented by milestone services.

Service Identifiers

Each service instance is named with a Fault Management Resource Identifier or FMRI. The FMRI includes the service name and the instance name. For example, the FMRI for the `rlogin` service is `svc:/network/login:rlogin`, where `network/login` identifies the service and `rlogin` identifies the service instance.

Equivalent formats for an FMRI are as follows:

- `svc://localhost/system/system-log:default`
- `svc:/system/system-log:default`
- `system/system-log:default`

In addition, some SMF commands can use the following FMRI format: `svc:/system/system-log`. Some commands infer what instance to use, when there is no ambiguity. See the SMF command man pages, such as `svcadm`(1M) or `svcs`(1), for instructions about which FMRI formats are appropriate.

The service names usually include a general functional category. The categories include the following:

- `application`
- `device`
- `milestone`
- `network`
- `platform`
- `site`
- `system`

Legacy `init.d` scripts are also represented with FMRIs that start with `lrc` instead of `svc`, for example: `lrc:/etc/rcS_d/S35cacheos_sh`. The legacy services can be monitored using SMF. However, you cannot administer these services.

When booting a system for the first time with SMF, services listed in `/etc/inetd.conf` are automatically converted into SMF services. The FMRIs for these services are slightly different. The syntax for a converted `inetd` services is:

`network/`*<service-name>*`/`*<protocol>*

In addition, the syntax for a converted service that uses the RPC protocol is:

`network/rpc-`*<service-name>*`/rpc_`*<protocol>*

Where *<service-name>* is the name defined in `/etc/inetd.conf` and *<protocol>* is the protocol for the service. For instance, the FMRI for the `rpc.cmsd` service is `network/rpc-100068_2-5/rpc_udp`.

Service States

The `svcs` command displays the state, start time, and FMRI of service instances. The state of each service is one of the following:

- `degraded` – The service instance is enabled, but is running at a limited capacity.
- `disabled` – The service instance is not enabled and is not running.
- `legacy_run` – The legacy service is not managed by SMF, but the service can be observed. This state is only used by legacy services.
- `maintenance` – The service instance has encountered an error that must be resolved by the administrator.
- `offline` – The service instance is enabled, but the service is not yet running or available to run.
- `online` – The service instance is enabled and has successfully started.
- `uninitialized` – This state is the initial state for all services before their configuration has been read.

SMF Manifests

An SMF *manifest* is an XML file that contains a complete set of properties that are associated with a service or a service instance. The files are stored in `/var/svc/manifest`. Manifests should not be used to modify the properties of a service. The service configuration repository is the authoritative source of configuration information. To incorporate information from the manifest into the repository, you must either run `svccfg import` or allow the service to import the information during a system boot.

See the `service_bundle`(4) man page for a complete description of the contents of the SMF manifests. If you need to change the properties of a service, see the `svccfg`(1M) or `inetadm`(1M) man pages.

SMF Profiles

An SMF *profile* is an XML file that lists a set of service instances and whether each should be enabled or disabled. Some profiles which are delivered with the Solaris release include:

- `/var/svc/profile/generic_open.xml` – This profile enables the standard services that have been started by default in earlier Solaris releases.
- `/var/svc/profile/generic_limited_net.xml` – This profile disables many of the internet services that have be started by default in earlier Solaris releases. The `network/ssh` service is enabled to provide network connectivity.
- `/var/svc/profile/ns_*.xml` – These profiles enable services associated with the name service that is configured to run on the system.
- `/var/svc/profile/platform_*.xml` – These profiles enable services associated with particular hardware platforms.

During the first boot after a new installation or an upgrade to the Solaris 10 release or any of the subsequent Solaris Express releases, some Solaris profiles are automatically applied. To be specific, the `/var/svc/profile/generic.xml` profile is applied. This file is usually symbolically linked to `generic_open.xml` or `generic_limited_net.xml`. Also, if a profile called `site.xml` is in `/var/svc/profile` during the first boot or is added between boots, the contents of this profile are applied. By using the `site.xml` profile, the initial set of enabled services may be customized by the administrator.

For more information about using profiles, see "How to Apply an SMF Profile" on page 343.

Service Configuration Repository

The *service configuration repository* stores persistent configuration information as well as SMF runtime data for services. The repository is distributed among local memory and local files. SMF is designed so that eventually, service data can be represented in the network directory service. The network directory service is not yet available. The data in the service configuration repository allows for the sharing of configuration information and administrative simplicity across many Solaris instances. The service configuration repository can only be manipulated or queried using SMF interfaces. For more information about manipulating and accessing the repository, see the `svccfg`(1M) and `svcprop`(1) man pages. The service configuration repository daemon is covered in the `svc.configd`(1M) man page. The service configuration library is documented in the `libscf`(3LIB) man page.

SMF Repository Backups

SMF automatically takes the following backups of the repository:

- The boot backup is taken immediately before the first change to the repository is made during each system startup.
- The `manifest_import` backup occurs after `svc:/system/manifest-import:default` completes, if it imported any new manifests or ran any upgrade scripts.

Four backups of each type are maintained by the system. The system deletes the oldest backup, when necessary. The backups are stored as `/etc/svc/repository-`*type*-*YYYYMMDD_HHMMSWS*, where *YYYYMMDD* (year, month, day) and *HHMMSS* (hour, minute, second), are the date and time when the backup was taken. Note that the hour format is based on a 24–hour clock.

You can restore the repository from these backups, if an error occurs. To do so, use the `/lib/svc/bin/restore_repository` command. For more information, see "How to Repair a Corrupt Repository" on page 352.

SMF Snapshots

The data in the service configuration repository includes *snapshots*, as well as a configuration that can be edited. Data about each service instance is stored in the snapshots. The standard snapshots are as follows:

- `initial` – Taken on the first import of the manifest
- `running` – Used when the service methods are executed
- `start` – Taken at the last successful start

The SMF service always executes with the `running` snapshot. This snapshot is automatically created if it does not exist.

The `svcadm refresh` command, sometimes followed by the `svcadm restart` command, makes a snapshot active. The `svccfg` command is used to view or revert to instance configurations in a previous snapshot. See "How to Revert to Another SMF Snapshot" on page 340 for more information.

SMF Administrative and Programming Interfaces

This section introduces the interfaces that are available when you use SMF.

SMF Command-Line Administrative Utilities

SMF provides a set of command-line utilities that interact with SMF and accomplish standard administrative tasks. The following utilities can be used to administer SMF.

TABLE 16–1 Service Management Facility Utilities

Command Name	Function
`inetadm`	Provides the ability to observe or configure services controlled by `inetd`
`svcadm`	Provides the ability to perform common service management tasks, such as enabling, disabling, or restarting service instances
`svccfg`	Provides the ability to display and manipulate the contents of the service configuration repository
`svcprop`	Retrieves property values from the service configuration repository with a output format appropriate for use in shell scripts
`svcs`	Gives detailed views of the service state of all service instances in the service configuration repository

Service Management Configuration Library Interfaces

SMF provides a set of programming interfaces that are used to interact with the service configuration repository through the `svc.configd` daemon. This daemon is the arbiter of all requests to the local repository datastores. A set of fundamental interfaces is defined as the lowest level of interaction possible with services in the service configuration repository. The interfaces provide access to all service configuration repository features such as transactions and snapshots.

Many developers only need a set of common tasks to interact with SMF. These tasks are implemented as convenience functions on top of the fundamental services to ease the implementation burden.

SMF Components

SMF includes a master restarter daemon and delegated restarters.

SMF Master Restarter Daemon

The `svc.startd` daemon is the master process starter and restarter for the Solaris OS. The daemon is responsible for managing service dependencies for the entire system. The daemon takes on the previous responsibility that `init` held of starting the appropriate `/etc/rc*.d` scripts at the appropriate run levels. First, `svc.startd` retrieves the information in the service configuration repository. Next, the daemon starts services when their dependencies are met. The daemon is also responsible for restarting services that have failed and for shutting down services whose dependencies are no longer satisfied. The daemon keeps track of service state through an operating system view of availability through events such as process death.

SMF Delegated Restarters

Some services have a set of common behaviors on startup. To provide commonality among these services, a delegated restarter might take responsibility for these services. In addition, a delegated restarter can be used to provide more complex or application-specific restarting behavior. The delegated restarter can support a different set of methods, but exports the same service states as the master restarter. The restarter's name is stored with the service. A current example of a delegated restarter is `inetd`, which can start Internet services on demand, rather than having the services always running.

SMF and Booting

SMF provides new methods for booting a system. For instance:

- There is a additional system state which is associated with the `all` milestone. With the `all` milestone, all of the services with a defined dependency on the `multi-user-server` milestone are started, as well as any services that do not have a defined dependency. If you have added services, such as third party products, they may not be started automatically unless you use the following command:

  ```
  ok boot -m milestone=all
  ```
- When booting a system, you can choose to use the verbose option to see more messages. By default, the system will not display these messages. To boot in the verbose mode, use the following command:

  ```
  ok boot -mverbose
  ```
- There is a new system state which is associated with the `none` milestone. Only `init`, `svc.startd` and `svc.configd` are started if you boot a system using this milestone. This state can be very useful for debugging booting problems. In particular, debugging any problems with the configuration of SMF services is made simpler, because none of the services are started. See "How to Boot Without Starting Any Services" on page 355 for instructions on how to use the `none` milestone.

SMF Compatibility

While many standard Solaris services are now managed by SMF, the scripts placed in `/etc/rc*.d` continue to be executed on run-level transitions. Most of the `/etc/rc*.d` scripts that were included in previous Solaris releases have been removed as part of SMF. The ability to continue to run the remaining scripts allows for third-party applications to be added without having to convert the services to use SMF.

In addition, `/etc/inittab` and `/etc/inetd.conf` must be available for packages to amend with postinstall scripts. These are called legacy-run services. The `inetconv` command is run to add these legacy-run services to the service configuration repository. The status of these services can be viewed, but no other changes are supported through SMF. Applications that use this feature will not benefit from the precise fault containment provided by SMF.

Applications converted to utilize SMF should no longer make modifications to the `/etc/inittab` and `/etc/inetd.conf` files. The converted applications will not use the `/etc/rc*.d` scripts. Also, the new version of `inetd` does not look for entries in `/etc/inetd.conf`.

Run Levels

A system's *run level* (also known as an *init state*) defines what services and resources are available to users. A system can be in only one run level at a time.

The Solaris OS has eight run levels, which are described in the following table. The default run level is specified in the `/etc/inittab` file as run level 3.

TABLE 16–2 Solaris Run Levels

Run Level	Init State	Type	Purpose
0	Power-down state	Power-down	To shut down the operating system so that it is safe to turn off power to the system.
s or S	Single-user state	Single-user	To run as a single user with some file systems mounted and accessible.
1	Administrative state	Single-user	To access all available file systems. User logins are disabled.
2	Multiuser state	Multiuser	For normal operations. Multiple users can access the system and all file system. All daemons are running except for the NFS server daemons.

TABLE 16–2 Solaris Run Levels *(Continued)*

Run Level	Init State	Type	Purpose
3	Multiuser level with NFS resources shared	Multiuser	For normal operations with NFS resources shared. This is the default run level for the Solaris OS.
4	Alternative multiuser state		Not configured by default, but available for customer use.
5	Power-down state	Power-down	To shut down the operating system so that it is safe to turn off power to the system. If possible, automatically turns off power on systems that support this feature.
6	Reboot state	Reboot	To shut down the system to run level 0, and then reboot to multiuser level with NFS resources shared (or whatever level is the default in the `inittab` file).

In addition, the `svcadm` command can be used to change the run level of a system, by selecting a milestone at which to run. The following table shows which run level corresponds to each milestone.

TABLE 16–3 Solaris Run Levels and SMF Milestones

Run Level	SMF Milestone FMRI
S	`milestone/single-user:default`
2	`milestone/multi-user:default`
3	`milestone/multi-user-server:default`

When to Use Run Levels or Milestones

Under most circumstances, using the `init` command with a run level to change the system state is sufficient. Using milestones to change system state can be confusing and can lead to unexpected behavior. In addition, the `init` command allows for the system to be shutdown, so `init` is the best command for changing system state.

However, booting a system using the none milestone, can be very useful when debugging startup problems. There is no equivalent run level to the none milestone. See "How to Boot Without Starting Any Services" on page 355 for specific instructions.

Determining a System's Run Level

Display run level information by using the `who -r` command.

```
$ who -r
```

Use the `who -r` command to determine a system's current run level for any level.

EXAMPLE 16–1 Determining a System's Run Level

This example displays information about a system's current run level and previous run levels.

```
$ who -r
 .    run-level 3  Dec 13 10:10  3  0 S
$
```

Output of `who -r` command	Description
`run-level 3`	Identifies the current run level
`Dec 13 10:10`	Identifies the date of last run level change
`3`	Also identifies the current run level
`0`	Identifies the number of times the system has been at this run level since the last reboot
`S`	Identifies the previous run level

`/etc/inittab` File

When you boot the system or change run levels with the `init` or `shutdown` command, the `init` daemon starts processes by reading information from the `/etc/inittab` file. This file defines these important items for the `init` process:

- That the `init` process will restart
- What processes to start, monitor, and restart if they terminate
- What actions to take when the system enters a new run level

Each entry in the `/etc/inittab` file has the following fields:

id:*rstate*:*action*:*process*

The following table describes the fields in an `inittab` entry.

TABLE 16–4 Fields Descriptions for the inittab File

Field	Description
id	Is a unique identifier for the entry.
rstate	Lists the run levels to which this entry applies.
action	Identifies how the process that is specified in the process field is to be run. Possible values include: sysinit, boot, bootwait, wait, and respawn. For a description of the other action keywords, see inittab(4).
process	Defines the command or script to execute.

EXAMPLE 16–2 Default inittab File

The following example shows a default inittab file that is installed with the Solaris release. A description for each line of output in this example follows.

```
ap::sysinit:/sbin/autopush -f /etc/iu.ap      (1)
sp::sysinit:/sbin/soconfig -f /etc/sock2path                   (2)
smf::sysinit:/lib/svc/bin/svc.startd    >/dev/msglog 2<>/dev/msglog         (3)
p3:s1234:powerfail:/usr/sbin/shutdown -y -i5 -g0 >/dev/msglog 2<>/dev/...    (4)
```

1. Initializes STREAMS modules
2. Configures socket transport providers
3. Initializes the master restarter for SMF
4. Describes a power fail shutdown

What Happens When the System Is Brought to Run Level 3

1. The init process is started and reads the /etc/default/init file to set any environment variables. By default, only the TIMEZONE variable is set.
2. Then, init reads the inittab file and does the following:
   a. Executes any process entries that have sysinit in the action field so that any special initializations can take place before users login.
   b. Passes the startup activities to svc.startd.

   For a detailed description of how the init process uses the inittab file, see init(1M).

CHAPTER 17

Managing Services (Tasks)

This chapter covers the tasks required to manage and monitor the Service Management Facility (SMF). In addition, information that is related to managing run level scripts is provided. The following topics are covered:

- "Managing Services (Task Map)" on page 333
- "Monitoring SMF Services" on page 334
- "Managing SMF Services" on page 337
- "Configuring SMF Services" on page 344
- "Using Run Control Scripts" on page 349
- "Troubleshooting the Service Management Facility" on page 352

Managing Services (Task Map)

The following task map describes the procedures that are needed to use SMF.

Task	Description	For Instructions
Display the status of a service instance.	Displays the status of all running service instances.	"How to List the Status of a Service" on page 334
Display the service dependents.	Display the services that are dependent on the specified service.	"How to Show Which Services Are Dependent on a Service Instance" on page 336
Display the dependencies of a service.	Display the services that a specified service is dependent on. This information can be used to help identify what is preventing a service from starting.	"How to Show Which Services a Service Is Dependent On" on page 336

Task	Description	For Instructions
Disable a service instance.	Turns off a service that is not functioning properly or needs to be off to increase security.	"How to Disable a Service Instance" on page 338
Enable a service instance	Starts a service.	"How to Enable a Service Instance" on page 339
Restart a service instance.	Restart a service, without having to use separate commands to disable and then enable the service.	"How to Restart a Service" on page 339
Modify a service instance.	Modifies the configuration parameters of a specified service instance.	"How to Modify a Service" on page 344
	Changes a configuration property of a service controlled by `inetd`.	"How to Change a Property for an `inetd` Controlled Service" on page 345
	Changes the startup options of a service controlled by `inetd`.	"How to Modify a Command-Line Argument for an `inetd` Controlled Service" on page 347
Convert `inetd.conf` entries.	Converts `inetd` services into legacy-run services that can be monitored using SMF.	"How to Convert `inetd.conf` Entries" on page 348
Repair a corrupt service configuration repository.	Replaces a corrupt repository with a default version.	"How to Repair a Corrupt Repository" on page 352
Boot a system without starting any services.	Boots a system without starting any services so that configuration problems that prevent booting can be fixed.	"How to Boot Without Starting Any Services" on page 355

Monitoring SMF Services

The following tasks show how to monitor SMF services.

▼ How to List the Status of a Service

This procedure can be used to show what services are running.

● **Run the `svcs` command.**

Running this command without any options displays a status report of the service specified by the FMRI.

```
% svcs -l FMRI
```

Example 17–1 Showing the Status of the `rlogin` Service

This example shows the status of a service that includes many contracts.

```
% svcs -l network/login:rlogin
fmri         svc:/network/login:rlogin
enabled      true
state        online
next_state   none
restarter    svc:/network/inetd:/default
contract_id  42325 41441 40776 40348 40282 40197 39025 38381 38053\
 33697 28625 24652 23689 15352 9889 7194 6576 6360 5387 1475 3015\
 6545 6612 9302 9662 10484 16254 19850 22512 23394 25876 26113 27326\
 34284 37939 38405 38972 39200 40503 40579 41129 41194
```

Example 17–2 Showing the Status of the `sendmail` Service

This example shows the status of a service that includes dependencies.

```
% svcs -l network/smtp:sendmail
fmri         svc:/network/smtp:sendmail
enabled      true
state        online
next_state   none
restarter    svc:/system/svc/restarter:default
contract_id  29462
dependency   require_all/refresh file://localhost/etc/nsswitch.conf (-)
dependency   require_all/refresh file://localhost/etc/mail/sendmail.cf (-)
dependency   optional_all/none svc:/system/system-log (online)
dependency   require_all/refresh svc:/system/identity:domain (online)
dependency   require_all/refresh svc:/milestone/name-services (online)
dependency   require_all/none svc:/network/service (online)
dependency   require_all/none svc:/system/filesystem/local (online)
```

Example 17–3 Showing the Status of all Services

The following command lists all services that are installed on the system as well as the status of each service. The command displays those services that are disabled as well as those that are enabled.

```
% svcs -a
```

Example 17–4 Showing the Status of Services Controlled by `inetd`

The following command lists services that are controlled by `inetd`. Each service's FMRI is listed, along with the run state and whether the service is enabled or disabled.

```
% inetadm
```

▼ How to Show Which Services Are Dependent on a Service Instance

This procedure shows how to determine which service instances depend on the specified service.

● **Display the service dependents.**

```
% svcs -D FMRI
```

Example 17–5 Displaying the Service Instances That Are Dependent on the Multiuser Milestone

The following example shows how to determine which service instances are dependent on the multiuser milestone.

```
% svcs -D milestone/multi-user
STATE          STIME    FMRI
online         Apr_08   svc:/milestone/multi-user-server:default
```

▼ How to Show Which Services a Service Is Dependent On

This procedure shows how to determine which services a specified service instance is dependent on.

● **Display the service dependencies.**

```
% svcs -d FMRI
```

Example 17–6 Displaying the Service Instances That the Multiuser Milestone Is Dependent On

The following example shows the services instances that the multiuser milestone is dependent on.

```
% svcs -d milestone/multi-user:default
STATE          STIME    FMRI
disabled       Aug_24   svc:/platform/sun4u/sf880drd:default
online         Aug_24   svc:/milestone/single-user:default
online         Aug_24   svc:/system/utmp:default
online         Aug_24   svc:/system/system-log:default
online         Aug_24   svc:/system/system-log:default
online         Aug_24   svc:/system/rmtmpfiles:default
online         Aug_24   svc:/network/rpc/bind:default
online         Aug_24   svc:/milestone/name-services:default
online         Aug_24   svc:/system/filesystem/local:default
online         Aug_24   svc:/system/mdmonitor:default
```

Managing SMF Services (Task Map)

Task	Description	For Instructions
Disable a service instance.	Stops a running service and prevents the service from restarting.	"How to Disable a Service Instance" on page 338
Enable a service instance.	Starts a service. In addition, the service will be restarted during subsequent reboots.	"How to Enable a Service Instance" on page 339
Restarting a service.	Stops and starts a service with one command	"How to Restart a Service" on page 339
Restoring a service in maintenance state.	Shows how to clean up and restart a service that is in maintenance state.	"How to Restore a Service That Is in the Maintenance State" on page 340
Revert to a snapshot.	Uses a previous snapshot to correct problems with a service.	"How to Revert to Another SMF Snapshot" on page 340
Create an profile.	Create a profile to disable or enable services as needed.	"How to Create an SMF Profile" on page 341
Apply a profile.	Uses the information in a profile to disable or enable services as needed.	"How to Apply an SMF Profile" on page 343
Change the services and their configuration using the `netservices` command.	Uses the information in the `generic_limited.xml` or `generic_open.xml` profiles to disable or enable services and make configuration changes to those services, as well.	"Changing Services Offered to the Network with `generic*.xml`" on page 343

Managing SMF Services

This section includes information on managing SMF services.

Using RBAC Rights Profiles With SMF

You can use RBAC rights profiles to allow users to manage some of the SMF services, without having to give the user `root` access. The rights profiles define what commands the user can run. For SMF, the following profiles have been created:

- `Service Management`: User can add, delete or modify services.
- `Service Operator`: User can request state changes of any service instance, such as restart and refresh.

For specific information about the authorizations, see the `smf_security(5)` man page. For instructions to assign a rights profile, see "How to Change the RBAC Properties of a User" in *System Administration Guide: Security Services*.

▼ How to Disable a Service Instance

Use the following procedure to disable a service. The service status change is recorded in the service configuration repository. Once the service is disabled, the disabled state will persist across reboots. The only way to get the service running again is to enable it.

1. **Become superuser or assume a role that includes the `Service Management` rights profile.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC" in *System Administration Guide: Security Services*.

2. **Check the dependents of the service you want to disable.**

   If this service has dependents that you need, then you cannot disable this service.

   ```
   # svcs -D FMRI
   ```

3. **Disable the service.**

   ```
   # svcadm disable FMRI
   ```

Example 17–7 Disabling the `rlogin` Service

The output from the first command shows that the `rlogin` service has no dependents. The second command in this example disables the `rlogin` service. The third command shows that the state of the `rlogin` service instance is disabled.

```
# svcs -D network/login:rlogin
# svcadm disable network/login:rlogin
STATE          STIME    FMRI
# svcs network/login:rlogin
STATE          STIME    FMRI
disabled         11:17:24 svc:/network/login:rlogin
```

▼ How to Enable a Service Instance

Use the following procedure to enable a service. The service status change is recorded in the service configuration repository. Once the service is enabled, the enabled state will persist across system reboots if the service dependencies are met.

1. **Become superuser or assume a role that includes the `Service Management` rights profile.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC" in *System Administration Guide: Security Services*.

2. **Determine whether service dependencies are satisfied.**

   If the service is enabled, then the service dependencies are satisfied. If not, use `svcadm enable -r` FMRI to recursively enable all dependencies.

   ```
   # svcs -l FMRI|grep enabled
   ```

3. **Enable a service.**

   ```
   # svcadm enable FMRI
   ```

Example 17–8 Enabling the `rlogin` Service

The second command in this example enables the `rlogin` service. The third command shows that the state of the `rlogin` service instance is online.

```
# svcs -l network/login:rlogin|grep enabled
enabled      false
# svcadm enable network/login:rlogin
# svcs network/login:rlogin
STATE          STIME    FMRI
online         12:09:16 svc:/network/login:rlogin
```

Example 17–9 Enabling a Service in Single-user Mode

The following command enables rpcbind. The `-t` option starts the service in temporary mode which does not change the service repository. The repository is not writable in single-user mode. The `-r` option recursively starts all the dependencies of the named service.

```
# svcadm enable -rt rpc/bind
```

▼ How to Restart a Service

If a service is currently running but needs to be restarted due to a configuration change or some other reason, the service can be restarted without you having to type separate commands to stop and start the service. The only reason to specifically disable and then enable a service is if changes need to be made before the service is enabled, and after the service is disabled.

1. **Become superuser or assume a role that includes the `Service Management` rights profile.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC" in *System Administration Guide: Security Services*.

2. **Restart a service.**

   ```
   # svcadm restart FMRI
   ```

▼ How to Restore a Service That Is in the Maintenance State

1. **Become superuser or assume a role that includes the `Service Management` rights profile.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC" in *System Administration Guide: Security Services*.

2. **Determine if any process that are dependent to the service have not stopped.**

   Normally, when a service instance is in a maintenance state, all processes associated with that instance have stopped. However, you should make sure before you proceed. The following command lists all of the processes that are associated with a service instance as well as the PIDs for those processes.

   ```
   # svcs -p FMRI
   ```

3. **(Optional) Kill any remaining processes.**

   Repeat this step for all processes that are displayed by the `svcs` command.

   ```
   # pkill -9 PID
   ```

4. **If necessary, repair the service configuration.**

   Consult the appropriate service log files in `/var/svc/log` for a list of errors.

5. **Restore the service.**

   ```
   # svcadm clear FMRI
   ```

▼ How to Revert to Another SMF Snapshot

If the service configuration is wrong, the problem can be fixed by reverting to the last snapshot that started successfully. In this procedure, a previous snapshot of the `console-login` service is used.

1. **Become superuser or assume a role that includes the `Service Management` rights profile.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC" in *System Administration Guide: Security Services*.

2 **Run the `svccfg` command.**

```
# svccfg
svc:>
```

a. **Select the service instance that you want to fix.**

Note – You must use an FMRI that fully defines the instance. No shortcuts are allowed.

```
svc:> select system/console-login:default
svc:/system/console-login:default>
```

b. **Generate a list of available snapshots.**

```
svc:/system/console-login:default> listsnap
initial
running
start
svc:/system/console-login:default>
```

c. **Select to revert to the `start` snapshot.**

The `start` snapshot is the last snapshot in which the service successfully started.

```
svc:/system/console-login:default> revert start
svc:/system/console-login:default>
```

d. **Quit `svccfg`.**

```
svc:/system/console-login:default> quit
#
```

3 **Update the information in the service configuration repository.**

This step updates the repository with the configuration information from the `start` snapshot.

```
# svcadm refresh system/console-login
```

4 **Restart the service instance.**

```
# svcadm restart system/console-login
```

▼ How to Create an SMF Profile

A profile is an XML file which lists SMF services and whether each should be enabled or disabled. Profiles are used to enable or disable many services at once. Not all services need to be listed in a profile. Each profile only needs to include those services that need to be enabled or disabled to make the profile useful.

1 **Create a profile.**

In this example, the `svccfg` command is used to create a profile which reflects which services are enabled or disabled on the current system. Alternately, you could make a copy of an existing profile to edit.

```
# svccfg extract> profile.xml
```

If you are using JumpStart, if you have large numbers of identical systems, or if you want to archive the system configuration for later restoration, you may want to use this procedure to create a unique version of a SMF profile.

2 **Edit the `profile.xml` file to make any required changes.**

a. **Change the name of the profile in the `service_bundle` declaration.**

In this example the name is changed to `profile`.

```
# cat profile.xml
  ...
<service_bundle type=‘profile‘ name=‘profile‘
    xmIns::xi=’http://www.w3.org/2003/XInclude’
  ...
```

b. **Remove any services that should not be managed by this profile.**

For each service, remove the three lines that describe the service. Each service description starts with `<service` and ends with `</service`. This example shows the lines for the LDAP client service.

```
# cat profile.xml
 ...
 <service name=’network/ldap/client’ version=’1’ type=’service’>
         <instance  name=’default’ enabled=’true’/>
 </service>
```

c. **Add any services that should be managed by this profile.**

Each service needs to be defined using the three line syntax shown above.

d. **If necessary, change the enabled flag for selected services.**

In this example, the `sendmail` service is disabled.

```
# cat profile.xml
  ...
  <service  name=’network/smtp’ version=’1’ type=’service’>
    <instance  name=’sendmail’ enabled=’false’/>
  </service>
  ...
```

3 **When necessary, apply the new profile.**

See “How to Apply an SMF Profile” on page 343 for instructions.

▼ How to Apply an SMF Profile

1 **Become superuser or assume a role that includes the `Service Management` rights profile.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC" in *System Administration Guide: Security Services*.

2 **Apply an profile.**

In this example, the `profile.xml` profile is used.

```
# svccfg apply profile.xml
```

Note – For specific instructions for switching between the `generic_limited_net.xml` and `generic_open.xml` and the properties that need to be applied when making this switch, please see "Changing Services Offered to the Network with `generic*.xml`" on page 343

▼ Changing Services Offered to the Network with `generic*.xml`

The `netservices` command switches system services between minimal network exposure and the traditional network exposure (as in previous Solaris releases). The switch is done with the `generic_limited.xml` and `generic_open.xml` profiles. In addition, some services properties are changed by the command to limit some services to a local-only mode or to the traditional mode, as appropriate.

Note – In the Solaris Express 7/06 release, the `generic_limited_net` profile and the local-mode only service properties are applied by default.

---

1 **Become superuser or assume a role that includes the `Service Management` rights profile.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC" in *System Administration Guide: Security Services*.

2 **Run the `netservices` command.**

In this example, the open or traditional network exposure is selected.

```
# /usr/sbin/netservices open
```

Example 17–10 Limiting Network Service Exposure

This command changes properties to run some services in local mode, as well as restricts which services are enabled with the `generic_limited_net` profile. The command should only be used if the `generic_open.xml` profile had been applied.

```
# /usr/sbin/netservices limited
```

Configuring SMF Services

▼ How to Modify a Service

The following procedure shows how to change the configuration of a service that is not managed by the `inetd` service.

1. **Become superuser or assume a role that includes the `Service Management` rights profile.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC" in *System Administration Guide: Security Services*.

2. **Make changes to the configuration files, as needed.**

   Many of the services have one or more configuration files that are used to define the startup or other configuration information. These files can be changed while the service is running. The contents of the files is only checked when the service is started.

3. **Restart the service.**

   ```
   # svcadm restart FMRI
   ```

Example 17–11 Sharing an NFS File System

To share a file system using the NFS service, you must define the file system in the `/etc/dfs/dfstab` file and then restart the NFS service. This example shows you what the `dfstab` file could look like, as well as how to restart the service.

```
# cat /etc/dfs/dfstab
  .
  .
share -F nfs -o rw /export/home
# svcadm restart svc:/network/nfs/server
```

▼ How to Change an Environment Variable for a Service

This procedure shows how to modify `cron` environment variables to help with debugging.

1. **Become superuser or assume a role that includes the `Service Management` rights profile.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC" in *System Administration Guide: Security Services*.

2 **Verify that the service is running.**

```
# svcs system/cron
STATE          STIME    FMRI
online         Dec_04   svc:/system/cron:default
```

3 **Set environment variables.**

In this example the UMEM_DEBUG and LD_PRELOAD environment variables are set. For information about the setenv subcommand refer to the svccfg(1M) man page.

```
# svccfg -s system/cron:default setenv UMEM_DEBUG default
# svccfg -s system/cron:default setenv LD_PRELOAD libumem.so
```

4 **Refresh and restart the service.**

```
# svcadm refresh system/cron
# svcadm restart system/cron
```

5 **Verify that the change has been made.**

```
# pargs -e ‘pgrep -f /usr/sbin/cron‘
100657: /usr/sbin/cron
envp[0]: LOGNAME=root
envp[1]: LD_PRELOAD=libumem.so
envp[2]: PATH=/usr/sbin:/usr/bin
envp[3]: SMF_FMRI=svc:/system/cron:default
envp[4]: SMF_METHOD=/lib/svc/method/svc-cron
envp[5]: SMF_RESTARTER=svc:/system/svc/restarter:default
envp[6]: TZ=GB
envp[7]: UMEM_DEBUG=default
#
```

▼ How to Change a Property for an `inetd` Controlled Service

1 **Become superuser or assume a role that includes the `Service Management` rights profile.**

Roles contain authorizations and privileged commands. For more information about roles, see “Configuring RBAC” in *System Administration Guide: Security Services*.

2 **List the properties for the specific service.**

This command displays all of the properties for the service identified by the FMRI.

```
# inetadm -l FMRI
```

3 **Change the property for the service.**

Each property for an `inetd` controlled service is defined by a property name and an assigned value. Supplying the property name without a specified value resets the property to the default value. Specific information about the properties for a service should be covered in the man page associated with the service.

```
# inetadm -m FMRI property-name=value
```

4 **Verify that the property has changed.**

List the properties again to make sure that the appropriate change has occurred.

```
# inetadm -l FMRI
```

5 **Confirm that the change has taken effect.**

Confirm the property change that the change has the desired effect.

Example 17–12 Changing the `tcp_trace` Property for `telnet`

The following example shows how to set the `tcp_trace` property for `telnet` to `true`. Checking the `syslog` output after running a `telnet` command shows that the change has taken effect.

```
# inetadm -l svc:/network/telnet:default
SCOPE    NAME=VALUE
         name="telnet"
 .
 .
default  inherit_env=TRUE
default  tcp_trace=FALSE
default  tcp_wrappers=FALSE
# inetadm -m svc:/network/telnet:default tcp_trace=TRUE
# inetadm -l svc:/network/telnet:default
SCOPE    NAME=VALUE
         name="telnet"
 .
 .
default  inherit_env=TRUE
         tcp_trace=TRUE
default  tcp_wrappers=FALSE
# telnet localhost
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
login: root
Password:
Last login: Mon Jun 21 05:55:45 on console
Sun Microsystems Inc.   SunOS 5.10      s10_57  May 2004
# ^D
```

```
Connection to localhost closed by foreign host.
# tail -1 /var/adm/messages
Jun 21 06:04:57 yellow-19 inetd[100308]: [ID 317013 daemon.notice] telnet[100625]
    from 127.0.0.1 32802
```

▼ How to Modify a Command-Line Argument for an `inetd` Controlled Service

1 **Become superuser or assume a role that includes the `Service Management` rights profile.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC" in *System Administration Guide: Security Services*.

2 **List the `exec` property for the specific service.**

This command displays all the properties for the service identified by the FMRI. Adding the `grep` command restricts the output to the `exec` property for the service.

```
# inetadm -l FMRI|grep exec
```

3 **Change the `exec` property for the service.**

The *command-syntax* set with the `exec` property defines the command string that is run when the service is started.

```
# inetadm -m FMRI exec="command-syntax"
```

4 **Verify that the property has changed.**

List the properties again to make sure that the appropriate change has occurred.

```
# inetadm -l FMRI
```

Example 17–13 Adding the Connection Logging (`-l`) Option to the `ftp` Command

In this example, the `-l` option is added to the `ftp` daemon when it is started. The effect of this change can be seen by reviewing the `syslog` output after a `ftp` login session has been completed.

```
# inetadm -l svc:/network/ftp:default | grep exec
        exec="/usr/sbin/in.ftpd -a"
# inetadm -m svc:/network/ftp:default exec="/usr/sbin/in.ftpd -a -l"
# inetadm -l svc:/network/ftp:default
SCOPE    NAME=VALUE
         name="ftp"
         endpoint_type="stream"
         proto="tcp6"
         isrpc=FALSE
         wait=FALSE
         exec="/usr/sbin/in.ftpd -a -l"
  .
  .
# ftp localhost
Connected to localhost.
220 yellow-19 FTP server ready.
Name (localhost:root): mylogin
331 Password required for mylogin.
Password:
230 User mylogin logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> quit
221-You have transferred 0 bytes in 0 files.
221-Total traffic for this session was 236 bytes in 0 transfers.
221-Thank you for using the FTP service on yellow-19.
221 Goodbye.
# tail -2 /var/adm/messages
Jun 21 06:54:33 yellow-19 ftpd[100773]: [ID 124999 daemon.info] FTP LOGIN FROM localhost
     [127.0.0.1], mylogin
Jun 21 06:54:38 yellow-19 ftpd[100773]: [ID 528697 daemon.info] FTP session closed
```

▼ How to Convert `inetd.conf` Entries

The following procedure converts `inetd.conf` entries into SMF service manifests. This procedure needs to be run anytime a third-party application that depends on `inetd` is added to a system. Also run this procedure, if you need to make configuration changes to the entry in `/etc/inetd.conf`.

1. **Become superuser or assume a role that includes the `Service Management` rights profile.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC" in *System Administration Guide: Security Services*.

2. **Convert the `inetd.conf` entries.**

   The `inetconv` command converts each entry in the selected file into service manifests.

   ```
   # inetconv -i filename
   ```

Example 17–14 Converting `/etc/inet/inetd.conf` Entries into SMF Service Manifests

```
# inetconv -i /etc/inet/inetd.conf
```

Using Run Control Scripts (Task Map)

Task	Description	For Instructions
Stop or start a service.	Use a run control script to stop or start a service.	"How to Use a Run Control Script to Stop or Start a Legacy Service" on page 349
Add a run control script.	Create a run control script and add it to the `/etc/init.d` directory.	"How to Add a Run Control Script" on page 350
Disable a run control script.	Disable a run control script by renaming the file.	"How to Disable a Run Control Script" on page 351

Using Run Control Scripts

▼ How to Use a Run Control Script to Stop or Start a Legacy Service

One advantage of having individual scripts for each run level is that you can run scripts in the `/etc/init.d` directory individually to stop system services without changing a system's run level.

1 **Become superuser or assume a role that includes the `Service Management` rights profile.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC" in *System Administration Guide: Security Services*.

2 **Stop the system service.**

```
# /etc/init.d/filename stop
```

3 **Restart the system service.**

```
# /etc/init.d/filename start
```

4 **Verify that the service has been stopped or started.**

```
# pgrep -f service
```

Example 17–15 Using a Run Control Script to Stop or Start a Service

For example, you can stop the NFS server daemons by typing the following:

```
# /etc/init.d/nfs.server stop
# pgrep -f nfs
```

Then, you can restart the NFS server daemons by typing the following:

```
# /etc/init.d/nfs.server start
# pgrep -f nfs
101773
101750
102053
101748
101793
102114
# pgrep -f nfs -d, | xargs ps -fp
     UID    PID   PPID   C    STIME TTY         TIME CMD
  daemon 101748      1   0   Sep 01 ?           0:06 /usr/lib/nfs/nfsmapid
  daemon 101750      1   0   Sep 01 ?          26:27 /usr/lib/nfs/lockd
  daemon 101773      1   0   Sep 01 ?           5:27 /usr/lib/nfs/statd
    root 101793      1   0   Sep 01 ?          19:42 /usr/lib/nfs/mountd
  daemon 102053      1   0   Sep 01 ?        2270:37 /usr/lib/nfs/nfsd
  daemon 102114      1   0   Sep 01 ?           0:35 /usr/lib/nfs/nfs4cbd
```

▼ How to Add a Run Control Script

If you want to add a run control script to start and stop a service, copy the script into the `/etc/init.d` directory. Then, create links in the `rc`*n*`.d` directory where you want the service to start and stop.

See the `README` file in each `/etc/rc`*n*`.d` directory for more information on naming run control scripts. The following procedure describes how to add a run control script.

1 **Become superuser or assume a role that includes the `Service Management` rights profile.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC" in *System Administration Guide: Security Services*.

2 **Add the script to the `/etc/init.d` directory.**

```
# cp filename /etc/init.d
# chmod 0744 /etc/init.d/filename
# chown root:sys /etc/init.d/filename
```

3 **Create links to the appropriate `rc`*n*`.d` directory.**

```
# cd /etc/init.d
# ln filename /etc/rc2.d/Snnfilename
# ln filename /etc/rcn.d/Knnfilename
```

4 **Verify that the script has links in the specified directories.**

```
# ls /etc/init.d/*filename /etc/rc2.d/*filename /etc/rcn.d/*filename
```

Example 17–16 Adding a Run Control Script

The following example shows how to add a run control script for the `xyz` service.

```
# cp xyz /etc/init.d
# chmod 0744 /etc/init.d/xyz
# chown root:sys /etc/init.d/xyz
# cd /etc/init.d
# ln xyz /etc/rc2.d/S99xyz
# ln xyz /etc/rc0.d/K99xyz
# ls /etc/init.d/*xyz /etc/rc2.d/*xyz /etc/rc0.d/*xyz
```

▼ How to Disable a Run Control Script

You can disable a run control script by renaming it with an underscore (_) at the beginning of the file name. Files that begin with an underscore or dot are not executed. If you copy a file by adding a suffix to it, both files will be run.

1 **Become superuser or assume a role that includes the `Service Management` rights profile.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC" in *System Administration Guide: Security Services*.

2 **Rename the script by adding an underscore (_) to the beginning of the new file.**

```
# cd /etc/rcn.d
# mv filename _filename
```

3 **Verify that the script has been renamed.**

```
# ls _*
_filename
```

Example 17–17 Disabling a Run Control Script

The following example shows how to rename the `S99datainit` script.

```
# cd /etc/rc2.d
# mv S99datainit _S99datainit
# ls _*
_S99datainit
```

Troubleshooting the Service Management Facility

▼ Debugging a Service That Is Not Starting

In this procedure, the print service is disabled.

1 **Become superuser or assume a role that includes the `Service Management` rights profile.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC" in *System Administration Guide: Security Services*.

2 **Request information about the hung service.**

```
# svcs -xv
svc:/application/print/server:default (LP Print Service)
 State: disabled since Wed 13 Oct 2004 02:20:37 PM PDT
Reason: Disabled by an administrator.
   See: http://sun.com/msg/SMF-8000-05
   See: man -M /usr/share/man -s 1M lpsched
Impact: 2 services are not running:
        svc:/application/print/rfc1179:default
        svc:/application/print/ipp-listener:default
```

The `-x` option provides additional information about the service instances that are impacted.

3 **Enable the service.**

```
# svcadm enable application/print/server
```

▼ How to Repair a Corrupt Repository

This procedure shows how to replace a corrupt repository with a default copy of the repository. When the repository daemon, `svc.configd`, is started, it does an integrity check of the configuration repository. This repository is stored in `/etc/svc/repository.db`. The repository can become corrupted due to one of the following reasons:

- Disk failure
- Hardware bug
- Software bug
- Accidental overwrite of the file

If the integrity check fails, the `svc.configd` daemon writes a message to the console similar to the following:

```
svc.configd: smf(5) database integrity check of:

    /etc/svc/repository.db
```

```
  failed.  The database might be damaged or a media error might have
  prevented it from being verified.  Additional information useful to
  your service provider is in:

    /etc/svc/volatile/db_errors

  The system will not be able to boot until you have restored a working
  database.  svc.startd(1M) will provide a sulogin(1M) prompt for recovery
  purposes.  The command:

    /lib/svc/bin/restore_repository

  can be run to restore a backup version of your repository. See
  http://sun.com/msg/SMF-8000-MY for more information.
```

The `svc.startd` daemon then exits and starts `sulogin` to enable you to perform maintenance.

1 **Enter the `root` password at the `sulogin` prompt. `sulogin` enables the `root` user to enter system maintenance mode to repair the system.**

2 **Run the following command:**

```
# /lib/svc/bin/restore_repository
```

Running this command takes you through the necessary steps to restore a non-corrupt backup. SMF automatically takes backups of the repository at key system moments. For more information see "SMF Repository Backups" on page 325.

When started, the `/lib/svc/bin/restore_repository` command displays a message similar to the following:

```
Repository Restore utility
See http://sun.com/msg/SMF-8000-MY for more information on the use of
this script to restore backup copies of the smf(5) repository.

If there are any problems which need human intervention, this script
will give instructions and then exit back to your shell.

Note that upon full completion of this script, the system will be
rebooted using reboot(1M), which will interrupt any active services.
```

If the system that you are recovering is not a local zone, the script explains how to remount the `/` and `/usr` file systems with read and write permissions to recover the databases. The script exits after printing these instructions. Follow the instructions, paying special attention to any errors that might occur.

After the `root` (`/`) file system is mounted with write permissions, or if the system is a local zone, you are prompted to select the repository backup to restore:

```
The following backups of /etc/svc/repository.db exists, from
oldest to newest:
```

`...` *list of backups* `...`

Backups are given names, based on type and the time the backup was taken. Backups beginning with `boot` are completed before the first change is made to the repository after system boot. Backups beginning with `manifest_import` are completed after `svc:/system/manifest-import:default` finishes its process. The time of the backup is given in *YYYYMMDD_HHMMSS* format.

3 **Enter the appropriate response.**

Typically, the most recent backup option is selected.

```
Please enter one of:
        1) boot, for the most recent post-boot backup
        2) manifest_import, for the most recent manifest_import backup.
        3) a specific backup repository from the above list
        4) -seed-, the initial starting repository. (All customizations
           will be lost.)
        5) -quit-, to cancel.

Enter response [boot]:
```

If you press Enter without specifying a backup to restore, the default response, enclosed in `[]` is selected. Selecting `-quit-` exits the `restore_repository` script, returning you to your shell prompt.

Note – Selecting `-seed-` restores the `seed` repository. This repository is designed for use during initial installation and upgrades. Using the `seed` repository for recovery purposes should be a last resort.

After the backup to restore has been selected, it is validated and its integrity is checked. If there are any problems, the `restore_repository` command prints error messages and prompts you for another selection. Once a valid backup is selected, the following information is printed, and you are prompted for final confirmation.

```
After confirmation, the following steps will be taken:

svc.startd(1M) and svc.configd(1M) will be quiesced, if running.
/etc/svc/repository.db
    -- renamed --> /etc/svc/repository.db_old_YYYYMMDD_HHMMSS
/etc/svc/volatile/db_errors
    -- copied --> /etc/svc/repository.db_old_YYYYMMDD_HHMMSS_errors
```

```
repository_to_restore
    -- copied --> /etc/svc/repository.db
and the system will be rebooted with reboot(1M).

Proceed [yes/no]?
```

4 **Type `yes` to remedy the fault.**

The system reboots after the `restore_repository` command executes all of the listed actions.

▼ How to Boot Without Starting Any Services

If problems with starting services occur, sometimes a system will hang during the boot. This procedure shows how to troubleshoot this problem.

1 **Boot without starting any services.**

This command instructs the `svc.startd` daemon to temporarily disable all services and start `sulogin` on the console.

```
ok boot -m milestone=none
```

2 **Log in to the system as `root`.**

3 **Enable all services.**

```
# svcadm milestone all
```

4 **Determine where the boot process is hanging.**

When the boot process hangs, determine which services are not running by running `svcs -a`. Look for error messages in the log files in `/var/svc/log`.

5 **After fixing the problems, verify that all services have started.**

a. **Verify that all needed services are online.**

```
# svcs -x
```

b. **Verify that the `console-login` service dependencies are satisfied.**

This command verifies that the `login` process on the console will run.

```
# svcs -l system/console-login:default
```

6 **Continue the normal booting process.**

▼ How to Force a `sulogin` Prompt If the `system/filesystem/local:default` Service Fails During Boot

Local file systems that are not required to boot the Solaris OS are mounted by the `svc:/system/filesystem/local:default` service. When any of those file systems are unable to be mounted, the service enters a maintenance state. System startup continues, and any services which do not depend on `filesystem/local` are started. Services which require `filesystem/local` to be online before starting through dependencies are not started.

To change the configuration of the system so that a `sulogin` prompt appears immediately after the service fails instead of allowing system startup to continue, follow the procedure below.

1 **Modify the `system/console-login` service.**

```
# svccfg -s svc:/system/console-login
svc:/system/console-login> addpg site,filesystem-local dependency
svc:/system/console-login> setprop site,filesystem-local/entities = fmri: svc:/system/filesystem/local
svc:/system/console-login> setprop site,filesystem-local/grouping = astring: require_all
svc:/system/console-login> setprop site,filesystem-local/restart_on = astring: none
svc:/system/console-login> setprop site,filesystem-local/type = astring: service
svc:/system/console-login> end
```

2 **Refresh the service.**

```
# svcadm refresh console-login
```

Example 17–18 Forcing an `sulogin` Prompt Using Jumpstart

Save the following commands into a script and save it as `/etc/rcS.d/S01site-customfs`.

```
#!/bin/sh
#
# This script adds a dependency from console-login -> filesystem/local
# This forces the system to stop the boot process and drop to an sulogin prompt
# if any file system in filesystem/local fails to mount.

PATH=/usr/sbin:/usr/bin
export PATH

    svccfg -s svc:/system/console-login << EOF
addpg site,filesystem-local dependency
setprop site,filesystem-local/entities = fmri: svc:/system/filesystem/local
setprop site,filesystem-local/grouping = astring: require_all
setprop site,filesystem-local/restart_on = astring: none
setprop site,filesystem-local/type = astring: service
EOF
```

```
svcadm refresh svc:/system/console-login

[ -f /etc/rcS.d/S01site-customfs ] &&
    rm -f /etc/rcS.d/S01site-customfs
```

Troubleshooting When a failure occurs with the `system/filesystem/local:default` service, the `svcs -vx` command should be used to identify the failure. After the failure has been fixed, the following command clears the error state and allows the system boot to continue: `svcadm clear filesystem/local`.

◆ ◆ ◆ CHAPTER 18

Managing Software (Overview)

Software management involves adding and removing software from stand-alone systems, servers, and their clients. This chapter describes the various tools that are available for installing and managing software.

This chapter does not describe installing the Solaris Operating System (Solaris OS) on a new system, nor does it describe installing or upgrading a new version of the Solaris OS. For information about installing or upgrading the Solaris OS, see *Solaris Express Installation Guide: Basic Installations*.

This is a list of the overview information in this chapter.

For the most up-to-date information for managing packages and patches in the Solaris OS, see `http://www.sun.com/bigadmin/patches/`.

For step-by-step instructions on managing software, see Chapter 19, "Managing Software With Solaris System Administration Tools (Tasks)," and Chapter 20, "Managing Software by Using Package Commands (Tasks)."

For information about managing software on Solaris systems with zones installed, see Chapter 25, "Adding and Removing Packages and Patches on a Solaris System With Zones Installed (Tasks)," in *System Administration Guide: Virtualization Using the Solaris Operating System*.

What's New in Software Management in the Solaris Operating System?

This section describes the new software management features in this Solaris release.

Deferred-Activation Patching

Patching tools have changed to handle larger patches. Starting with patch 119254-42 and 119255-42, the patch installation commands, `patchadd` and `patchrm`, have been modified to change the way in which certain patches that deliver new features are handled. This modification affects the installation of these patches on any Solaris 10 release. These *deferred-activation* patches are better equipped to handle the large scope of changes that are delivered in feature patches.

For more details, see `http://www.sun.com/bigadmin/sundocs/articles/patch-wn.jsp`.

Common Agent Container Included in the Solaris OS

The Common Agent Container is a stand-alone Java program that implements a container for Java management applications. This program provides a management infrastructure that is designed for Java Management Extensions (JMX) and Java Dynamic Management Kit (Java DMK) based management functionality. The software is installed by the `SUNWcacaort` package and resides in the `/usr/lib/cacao` directory.

Typically, the container is not visible.

However, there are two instances when you might need to interact with the container daemon:

- In the event that another application attempts to use a network port that is reserved for the Common Agent Container.
- In the event that a certificate store is compromised. If this conflict occurs, you might have to regenerate the Common Agent Container certificate keys.

For information about how to troubleshoot these problems, see "Troubleshooting Common Agent Container Problems in the Solaris OS" in *System Administration Guide: Advanced Administration*.

Improvements to How `patchadd -M` Command Handles Multiple Patches

Solaris 10: Starting with the Solaris 10 release, the functionality of the `patchadd -M` command is improved to enable more effective and efficient handling of multiple patches and dependencies between patches. As a result, you no longer have to specify patch IDs in numerical order when using this command.

Note that if you use the `patchadd -M` command without specifying a patch ID or patch IDs, all the patches in the directory are automatically installed on the system. To install a specific patch or patches, you must specify the patch ID when using the `patchadd -M` command.

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

Package and Patch Tool Enhancements

Solaris 10: The Solaris package and patch tools were enhanced in the Solaris 10 initial 3/05 release to provide improved performance and extended functionality.

As a part of these enhancements, the `pkgchk` command now provides a new option to assist you in mapping files to packages. To map files to packages, use the `pkgchk -P` option instead of `grep` *pattern*`/var/sadm/install/contents`. The `-P` option enables you to use a partial path. Use this option with the `-l` option to list the information about the files that contain the partial path. For more information see "How to Check the Integrity of Installed Objects (`pkgchk -p, pkgchk -P`)" on page 413 and the `pkgchk`(1M) man page.

Where to Find Software Management Tasks

Use this table to find step-by-step instructions for managing software.

Software Management Topics	For More Information
Installing Solaris software.	*Solaris Express Installation Guide: Basic Installations*
Adding or removing Solaris software packages after installation.	Chapter 19, "Managing Software With Solaris System Administration Tools (Tasks)," and Chapter 20, "Managing Software by Using Package Commands (Tasks)"
Adding or removing Solaris patches after installation.	"Managing Patches in the Solaris Operating System" on page 420

Software Management Topics	For More Information
Troubleshooting software package problems.	Chapter 21, "Troubleshooting Software Package Problems (Tasks)," in *System Administration Guide: Advanced Administration*

Overview of Software Packages

Software management involves installing or removing software products. Sun and its third-party ISVs deliver software as a collection of one or more *packages*.

The term *packaging* generically refers to the method for distributing and installing software products to systems where the products will be used. A package is a collection of files and directories in a defined format. This format conforms to the application binary interface (ABI), which is a supplement to the System V Interface Definition. The Solaris OS provides a set of utilities that interpret this format and provide the means to install a package, to remove a package, or to verify a package installation.

A *patch* is an accumulation of fixes for a known or potential problem within the Solaris OS or other supported software. A patch can also provide a new feature or an enhancement to a particular software release. A patch consists of files and directories that replace or update existing files and directories. Most Solaris patches are delivered as a set of sparse packages.

A *sparse package* contains only those objects that have been altered since the packages were first delivered as part of the Solaris distribution. Sparse packages accommodate patches that are smaller than if they were redistributed as complete packages to provide software updates. Delivering sparse packages also minimizes the changes that are made to the customer's environment. For more information about patches, see "Managing Patches in the Solaris Operating System" on page 420.

Signed Packages, Patches, and Software Updates

Packages can include a digital signature. A package with a valid digital signature ensures that the package has not been modified since the signature was applied to the package. Using signed packages is a secure method of downloading or adding packages because the digital signature can be verified before the package is added to your system.

The same holds true for signed patches. A patch with a valid digital signature ensures that the patch has not been modified since the signature was applied to the patch. Using signed patches is a secure method of downloading or applying patches because the digital signature can be verified before the patch is applied to your system.

For more information about *applying* signed patches to your system, see "Managing Solaris Patches by Using the `patchadd` Command (Task Map)" on page 422.

For information about *creating* signed packages, see *Application Packaging Developer's Guide*.

A signed package is identical to an unsigned package, except for the digital signature. The package can be installed, queried, or removed with existing Solaris packaging tools. A signed package is also binary-compatible with an unsigned package.

Before you can use `pkgadd` and `patchadd` to add a package or patch with a digital signature to your system, you must set up a package keystore with trusted certificates. These certificates are used to identify that the digital signature on the package or patch is valid.

The following describes the general terms associated with signed packages and patches.

Keystore

A repository of certificates and keys that is queried when needed.

- Java keystore – A repository of certificates that is installed by default with the Solaris release. The Java keystore is usually stored in the `/usr/j2se/jre/lib/security` directory.
- Package keystore – A repository of certificates that you import when adding signed packages and patches to your system.

  The package keystore is stored in the `/var/sadm/security` directory by default.

Trusted certificate

A certificate that holds a public key that belongs to another entity. The *trusted certificate* is named as such because the keystore owner trusts that the public key in the certificate indeed belongs to the identity identified by the subject or owner of the certificate. The issuer of the certificate vouches for this trust by signing the certificate.

Trusted certificates are used when verifying signatures, and when initiating a connection to a secure (SSL) server.

User key

Holds sensitive cryptographic key information. This information is stored in a protected format to prevent unauthorized access. A user key consists of both the user's private key and the public key certificate that corresponds to the private key.

The process of using the `pkgadd` or `patchadd` command to add a signed package or patch to your system involves three basic steps:

1. Adding the certificates to your system's package keystore by using the `pkgadm` command
2. (Optional) Listing the certificates by using the `pkgadm` command
3. Adding the package with the `pkgadd` command or applying the patch by using the `patchadd` command

For step-by-step instructions on adding signed packages to your system, see "Adding and Removing Signed Packages by Using the `pkgadd` Command (Task Map)" on page 399.

For step-by-step instructions on applying signed patches to your system with the `patchadd` command, see "Managing Solaris Patches by Using the `patchadd` Command (Task Map)" on page 422.

Using Sun's Certificates to Verify Signed Packages and Patches

Access to a package keystore is protected by a special password that you specify when you import the Sun certificates into your system's package keystore.

If you use the `pkgadm listcert` command, you can view information about your locally stored certificates in the package keystore. For example:

```
# pkgadm listcert -P pass:store-pass
    Keystore Alias: /C=US/O=VeriSign, Inc./OU=Class 2 Public Primary Certification Authority - G2/O
        Common Name: /C=US/O=VeriSign, Inc./OU=Class 2 Public Primary Certification Authority - G2/O
   Certificate Type: Trusted Certificate
Issuer Common Name: /C=US/O=VeriSign, Inc./OU=Class 2 Public Primary Certification Authority - G2/O
Validity Dates: <May 18 00:00:00 1998 GMT> - <Aug 1 23:59:59 2028 GMT>
 MD5 Fingerprint: 2D:BB:E5:25:D3:D1:65:82:3A:B7:0E:FA:E6:EB:E2:E1
   SHA1 Fingerprint: B3:EA:C4:47:76:C9:C8:1C:EA:F2:9D:95:B6:CC:A0:08:1B:67:EC:9D
```

The following describes the output of the `pkgadm listcert` command.

Keystore Alias	When you retrieve certificates for printing, signing, or removing, this name must be used to reference the certificate.
Command Name	The common name of the certificate. For trusted certificates, this name is the same as the keystore alias.
Certificate Type	Can be one of two types: ■ Trusted certificate – A certificate that can be used as a trust anchor when verifying other certificates. No private key is associated with a trusted certificate. ■ Signing certificate – A certificate that can be used when signing a package or patch. A private key is associated with a signing certificate.
Issuer Command Name	The name of the entity that issued, and therefore signed, this certificate. For trusted certificate authority (CA) certificates, the issuer common name and common name are the same.
Validity Dates	A date range that identifies when the certificate is valid.
MD5 Fingerprint	An MD5 digest of the certificate. This digest can be used to verify that the certificate has not been altered during transmission from the source of the certificate.

SHA1 Fingerprint	Similar to an MD5 fingerprint, except that it is calculated using a different algorithm.

Each certificate is authenticated by comparing its MD5 and SHA1 hashes, also called *fingerprints*, against the known correct fingerprints published by the issuer.

Importing Sun's Trusted Certificates

You can obtain Sun's trusted certificates for adding signed packages and patches in the following ways:

- **Java keystore** – Import Sun's Root CA certificate that is included by default in the Java keystore when you install the Solaris release.
- **Sun's Public Key Infrastructure (PKI) site** – If you do not have a Java keystore available on your system, you can import the certificates from this site.

Setting Up a Package Keystore

If your system already has a populated Java keystore, you can now export the Sun Microsystems root CA certificate from the Java keystore with the `keytool` command. Then, use the `pkgadm` command to import this certificate into the package keystore.

After the Root CA certificate is imported into the package keystore, you can use the `pkgadd` and `patchadd` commands to add signed packages and patches to your system.

Note – The Sun Microsystems root-level certificates are only required when adding Sun-signed patches and packages.

For step-by-step instructions on importing certificates into the package keystore, see "How to Import a Trusted Certificate From the Java Keystore (`pkgadm addcert`)" on page 400.

For complete instructions on adding signed packages with the `pkgadd` command, see "How to Add a Signed Package (`pkgadd`)" on page 404.

Tools for Managing Software Packages

The following table describes the tools for adding and removing software packages from a system after the Solaris release is installed on a system.

TABLE 18–1 Tools or Commands for Managing Software Packages

Tool or Command	Description	Man Page
`installer`	Launches an installer, such as Solaris installation GUI, to add software from the Solaris media. The installer must be available either locally or remotely.	`installer`(1M)
`prodreg (GUI)`	Launches an installer to add, remove, or display software product information. Use Solaris Product Registry to remove or display information about software products that were originally installed by using the Solaris installation GUI or the Solaris `pkgadd` command.	`prodreg`(1M)
Solaris Product Registry `prodreg` Viewer (CLI)	Use the `prodreg` command to remove or display information about software products that were originally installed by using the Solaris installation GUI or the Solaris `pkgadd` command.	`prodreg`(1M)
`pkgadd`	Installs a signed or unsigned software package.	`pkgadd`(1M)
`pkgadm`	Maintains the keys and certificates used to manage signed packages and signed patches.	`pkgadm`(1M)
`pkgchk`	Checks the installation of a software package.	`pkgchk`(1M)
`pkginfo`	Lists software package information.	`pkginfo`(1)
`pkgparam`	Displays software package parameter values.	`pkgparam`(1)
`pkgrm`	Removes a software package.	`pkgrm`(1M)
`pkgtrans`	Translates an installable package from one format to another format. The `-g` option instructs the `pkgtrans` command to generate and store a signature in the resulting data stream.	`pkgtrans`(1)

For more information about these commands, see Chapter 19, "Managing Software With Solaris System Administration Tools (Tasks)," and Chapter 20, "Managing Software by Using Package Commands (Tasks)."

Adding or Removing a Software Package (pkgadd)

All the software management tools that are listed in Table 18–1 are used to add, remove, or query information about installed software. The Solaris Product Registry `prodreg` viewer and the Solaris installation GUI both access install data that is stored in the Solaris Product Registry. The package tools, such as the `pkgadd` and `pkgrm` commands, also access or modify install data.

When you add a package, the `pkgadd` command uncompresses and copies files from the installation media to a local system's disk. When you remove a package, the `pkgrm` command deletes all files associated with that package, unless those files are also shared with other packages.

Package files are delivered in package format and are unusable as they are delivered. The `pkgadd` command interprets the software package's control files, and then uncompresses and installs the product files onto the system's local disk.

Although the `pkgadd` and `pkgrm` commands do not log their output to a standard location, they do keep track of the package that is installed or removed. The `pkgadd` and `pkgrm` commands store information about a package that has been installed or removed in a software product database.

By updating this database, the `pkgadd` and `pkgrm` commands keep a record of all software products installed on the system.

Key Points for Adding Software Packages (pkgadd)

Keep the following key points in mind before you install or remove packages on your system:

- **Package naming conventions** – Sun packages always begin with the prefix `SUNW`, as in `SUNWaccr`, `SUNWadmap`, and `SUNWcsu`. Third-party packages usually begin with a prefix that corresponds to the company's stock symbol.
- **What software is already installed** – You can use the Solaris installation GUI, Solaris Product Registry `prodreg` viewer (either GUI or CLI) or the `pkginfo` command to determine the software that is already installed on a system.
- **How servers and clients share software** – Clients might have software that resides partially on a server and partially on the client. In such cases, adding software for the client requires that you add packages to both the server and the client.

Guidelines for Removing Packages (pkgrm)

You should use one of the tools listed in Table 18–1 to remove a package, even though you might be tempted to use the rm command instead. For example, you could use the rm command to remove a binary executable file. However, doing so is not the same as using the pkgrm command to remove the software package that includes that binary executable. Using the rm command to remove a package's files will corrupt the software products database. If you really only want to remove one file, you can use the removef command. This command will update the software product database correctly so that the file is no longer a part of the package. For more information, see the removef(1M) man page.

If you intend to keep multiple versions of a package, install new versions into a different directory than the already installed package by using the pkgadd command. For example, if you intended to keep multiple versions of a document processing application. The directory where a package is installed is referred to as the base directory. You can manipulate the base directory by setting the basedir keyword in a special file called an administration file. For more information on using an *administration file* and on setting the base directory, see "Avoiding User Interaction When Adding Packages (pkgadd)" on page 369 and the admin(4) man page.

Note – If you use the upgrade option when installing Solaris software, the Solaris installation software checks the software product database to determine the products that are already installed on the system.

Restrictions on Adding and Removing Software Packages and Patches for Solaris Releases That are Not Zones Aware

On systems that are running a Solaris release that is not zones aware, using any command that accepts the -R option to specify an alternate root path for a global zone that has non-global zones installed, does not work.

These commands include:

- pkgadd
- pkgrm
- patchadd
- patchrm

See the pkgadd(1M), pkgrm(1M), patchadd(1M), and patchrm(1M) man pages.

For additional information, see "Restrictions on Using patchadd -R to Create an Alternate root Path" on page 425.

Avoiding User Interaction When Adding Packages (pkgadd)

This section provides information about avoiding user interaction when adding packages with the pkgadd command.

Using an Administration File

When you use the pkgadd -a command, the command consults a special administration file for information about how the installation should proceed. Normally, the pkgadd command performs several checks and prompts the user for confirmation before it actually adds the specified package. You can, however, create an administration file that indicates to the pkgadd command that it should bypass these checks and install the package without user confirmation.

The pkgadd command, by default, checks the current working directory for an administration file. If the pkgadd command doesn't find an administration file in the current working directory, it checks the /var/sadm/install/admin directory for the specified administration file. The pkgadd command also accepts an absolute path to the administration file.

Note – Use administration files judiciously. You should know where a package's files are installed and how a package's installation scripts run before using an administration file to avoid the checks and prompts that the pkgadd command normally provides.

The following example shows an administration file that prevents the pkgadd command from prompting the user for confirmation before installing the package.

```
mail=
instance=overwrite
partial=nocheck
runlevel=nocheck
idepend=nocheck
rdepend=nocheck
space=nocheck
setuid=nocheck
conflict=nocheck
action=nocheck
networktimeout=60
networkretries=3
authentication=quit
keystore=/var/sadm/security
proxy=
basedir=default
```

Besides using administration files to avoid user interaction when you add packages, you can use them in several other ways. For example, you can use an administration file to quit a package installation (without user interaction) if there's an error or to avoid interaction when you remove packages by using the `pkgrm` command.

You can also assign a special installation directory for a package, which you might do if you wanted to maintain multiple versions of a package on a system. To do so, set an alternate base directory in the administration file by using the `basedir` keyword. The keyword specifies where the package will be installed. For more information, see the `admin(4)` man page.

Using a Response File (`pkgadd`)

A response file contains your answers to specific questions that are asked by an *interactive package*. An interactive package includes a `request` script that asks you questions prior to package installation, such as whether optional pieces of the package should be installed.

If you know prior to installation that the package is an interactive package, and you want to store your answers to prevent user interaction during future installations, use the `pkgask` command to save your response. For more information on this command, see `pkgask(1M)`.

Once you have stored your responses to the questions asked by the `request` script, you can use the `pkgadd -r` command to install the package without user interaction.

CHAPTER 19

Managing Software With Solaris System Administration Tools (Tasks)

This chapter describes how to add, verify, and remove software packages by using the Solaris installation graphical user interface (GUI) and the Solaris Product Registry.

For information about software management features that are new in this release, see "What's New in Software Management in the Solaris Operating System?" on page 360.

For information about the procedures that are associated with performing software management tasks, see:

- "Adding Software With the Solaris Installation GUI" on page 372
- "Managing Software With the Solaris Product Registry GUI (Task Map)" on page 373
- "Managing Software With the Solaris Product Registry Command-Line Interface (Task Map)" on page 378

Solaris Product Registry and Solaris GUI Installation Tools for Managing Software

The following table lists the commands to use for adding, removing, and checking the installation of software packages the Solaris installation GUI and Solaris Package Registry tools.

TABLE 19–1 System Administration Tools for Managing Software Packages

Tool	Description	Man Page
`installer`	Installs or removes a software package with an installer	`installer`(1M)
`prodreg`	Enables you to browse, unregister, and uninstall software in the Solaris Product Registry	`prodreg`(1M)

Adding Software With the Solaris Installation GUI

This section describes how to use the Solaris installation GUI to add software to a system on which you have installed the Solaris Operating System (Solaris OS). The Solaris installation GUI installs only the components of the software groups that you skipped when you initially installed the Solaris OS. You cannot upgrade to another software group after installing or upgrading the OS. .

▼ How to Install Software With the Solaris Installation GUI Program

---

Note – This procedure assumes that the system is running volume management (`vold`). If your system is not running volume management, see Chapter 3, "Accessing Removable Media (Tasks)," in *System Administration Guide: Devices and File Systems*. This chapter provides information about accessing removable media without volume management.

---

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 **Decide to install software from a CD, a DVD, or from the network.**

Select one of the following:

- If you are installing from a CD, insert the CD into the CD-ROM drive.

  If you insert theSolaris Languages CDs , the Solaris installation GUI starts automatically. Proceed to Step 5.
- If you are installing from a DVD, insert the DVD into the DVD-ROM drive.
- If you are installing from the network, locate the net image of the software you want to install.

3 **Change directories to find the Solaris installation GUI installer.**

Solaris installation GUI installers are located in various directories on the CDs and on the DVD.

- Solaris 11 Software CDs or DVD.
- Solaris11 Documentation DVD.
- Solaris 11 Languages CDs. The Solaris installation GUI starts automatically when the CD is inserted.

4 **Follow the instructions to install the software.**

- From the command line, type the following command:

```
% ./installer [options]
```

`-nodisplay`	Runs the installer without a GUI.
`-noconsole`	Runs without any interactive text console device. Use this option with the `-nodisplay` option when you include the `installer` command in a UNIX script for installing software.

- From a file manager, double-click Installer or installer.

  An Installer window is displayed, followed by the Solaris installation GUI dialog box.

5 **Follow the directions on the screen to install the software.**

6 **When you have finished adding software, click Exit.**

The Solaris installation GUI exits.

Managing Software With the Solaris Product Registry GUI (Task Map)

The following task map describes the software management tasks that you can perform with the Solaris Product Registry.

Task	Description	For Instructions
View installed or uninstalled software with the Solaris Product Registry.	Used for learning about installed or uninstalled software.	"How to View Installed or Uninstalled Software Information With the Solaris Product Registry GUI" on page 375
Install software with the Solaris Product Registry.	You can use the Solaris Product Registry to find software and launch the Solaris installation GUI. This program takes you through the installation of that software.	"How to Install Software With the Solaris Product Registry GUI" on page 376
Uninstall software with the Solaris Product Registry.	Use tor uninstall software with the Solaris Product Registry.	"How to Uninstall Software With the Solaris Product Registry GUI" on page 377

The Solaris Product Registry is a tool to help you manage installed software. After you have installed the software, Product Registry provides a list of all the installed software by using the Solaris installation GUI or the Solaris `pkgadd` command.

You can use the Solaris Product Registry in a GUI or with a command-line interface (CLI). For more information on how to use the Solaris Product Registry CLI, see "Managing Software With the Solaris Product Registry Command-Line Interface (Task Map)" on page 378.

The Solaris Product Registry GUI interface enables you to do the following:

- View a list of installed and registered software and some software attributes.
- View all Solaris system products that you installed in their localized version in the System Software Localizations directory.
- Find and launch an installer.
- Install additional software products.
- Uninstall software and individual software packages.

The Solaris Product Registry GUI main window consists of three panes of information:

- Installed, registered, and removed software
- Standard attributes of the currently selected software
- Attributes that are customized and attributes that are internal to the registered software

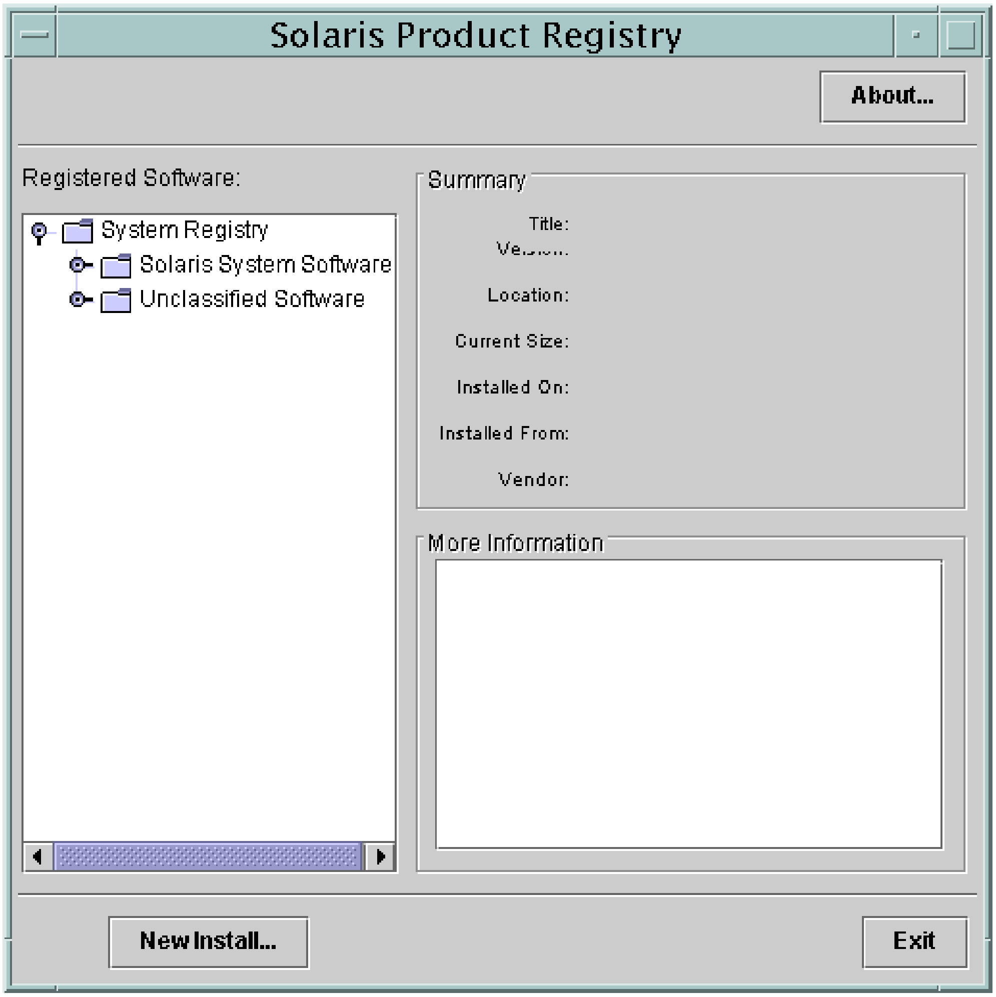


FIGURE 19–1 Solaris Product Registry Main Window

▼ How to View Installed or Uninstalled Software Information With the Solaris Product Registry GUI

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 **Start the Solaris Product Registry tool.**

```
# prodreg &
```

The Solaris Product Registry main window is displayed.

3 **Click the turner control to the left of the System Registry directory in the Registered Software box.**

The turner control changes from pointing to the right to pointing downward. You can expand or collapse any item in the registry, except an item that has a text file icon to its left.

The Software Installed in Registered Software box always contains the following components:

- The configuration software group that you chose when you installed the Solaris release. Software groups that can be displayed include Reduced Network Support, Core, End User System Support, Developer System Support, Entire Distribution, or Entire Distribution Plus OEM Support.
- Additional system software, which contains Solaris products that are not part of the software group you chose.
- Unclassified software that is not a Solaris product or part of the software group. This software includes any package that you installed by using the `pkgadd` command.

4 **Select directories until you find a software application to view.**

The list expands as you open directories.

5 **To view the attributes, select a directory or file.**

The Product Registry displays attribute information in the System Registry box.

- For software products that were installed with the Solaris installation GUI, the Solaris Product Registry contains values for at least the following: Title, Version, Location, and Installed on. Items in an expanded list under a product or software group inherit the version information of the product.
- If all or part of the product was removed with the `pkgrm` command, a cautionary icon appears next to the software product's name.

▼ How to Install Software With the Solaris Product Registry GUI

You can use Solaris Product Registry to find software and launch the Solaris installation GUI program. This program takes you through the installation of that software.

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 **Start the Solaris Product Registry tool.**

```
# prodreg
```

The Solaris Product Registry main window is displayed.

3 **Decide if you are installing from a CD, a DVD, or from the network. Select one of the following:**

- **If you are installing from a CD, insert the CD into the CD-ROM drive.**
- **If you are installing from a DVD, insert the DVD into the DVD-ROM drive.**
- **If you are installing from the network, locate the net image of the software that you want to install.**

4 **To view the list of installed and registered software, click the turner control.**

5 **Click the New Install button at the bottom of the Solaris Product Registry window.**

The Select Installer dialog box is displayed. This box initially points to the `/cdrom` directory or the directory you are in.

6 **Select directories to find the Solaris installation GUI installer.**

Solaris installation GUI installers are located in various directories on the CDs and on the DVD.

- Solaris 10 Software CDs or DVD.
- Solaris 10 Documentation DVD.
- Solaris 10 Languages CDs. The Solaris installation GUI automatically starts when the CD is inserted.

7 **When you find the installer you want, select its name in the Files box.**

8 **Click OK.**

The installer you selected is launched.

9 **Follow the directions that are displayed by the installer to install the software.**

▼ How to Uninstall Software With the Solaris Product Registry GUI

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see “Configuring RBAC (Task Map)” in *System Administration Guide: Security Services*.

2 **Start the Solaris Product Registry tool.**

```
# prodreg
```

The Solaris Product Registry main window is displayed.

3 **To view the list of installed and registered software, click the turner control.**

4 **Select directories until you find the name of the software that you want to uninstall.**

5 **Read the software attributes to make sure that this software is the software that you want to uninstall.**

6 **Click the Uninstall** *software-product-name* **button at the bottom of the Solaris Product Registry window.**

The software product you selected is uninstalled.

Managing Software With the Solaris Product Registry Command-Line Interface (Task Map)

The following task map describes the software management tasks that you cab perform with the Solaris Product Registry command-line interface.

Task	Description	For Instructions
View installed or uninstalled software.	You can view software information by using the `browse` subcommand.	"How to View Installed or Uninstalled Software Information (`prodreg`)" on page 379
View software attributes.	You can view specific software attributes by using the `info` subcommand.	"How to View Software Attributes (`prodreg`)" on page 382
Check dependencies between software components.	You can view the components that depend on a specific software component by using the `info` subcommand.	"How to Check for Software Dependencies (`prodreg`)" on page 384
Identify damaged software products.	If you remove installed software files or packages without using the appropriate uninstaller, you can damage the software on your system.	"How to Identify Damaged Software Products (`prodreg`)" on page 386
Uninstall software	You can remove software from your system by using the `uninstall` subcommand.	"How to Uninstall Software (`prodreg`)" on page 388

Task	Description	For Instructions
Uninstall damaged software.	Uninstalling a damaged software component might fail if the uninstaller program for the software component has been removed from the system.	"How to Uninstall Damaged Software (prodreg)" on page 392
Reinstall damaged software components.	If other software depends on a damaged software component, you might want to reinstall the damaged component, rather than uninstall the component and the other dependent software.	"How to Reinstall Damaged Software Components (prodreg)" on page 395

Managing Software With the Solaris Product Registry Command-Line Interface

The `prodreg` command is the command-line interface (CLI) to the Solaris Product Registry. The `prodreg` command supports several subcommands that enable you to manage the software on your system.

You can use the `prodreg` command in a terminal window to perform the following tasks:

- View a list of installed and registered software and software attributes.
- View all Solaris system products that you installed in their localized version in the System Software Localizations directory.
- Identify damaged software.
- Remove software entries from the Solaris Product Registry.
- Uninstall software and individual software packages.

For more information on how to manage the Solaris Product Registry by using the command-line interface, see the `prodreg`(1M) man page.

▼ How to View Installed or Uninstalled Software Information (`prodreg`)

You can view information about software in the Solaris Product Registry in a terminal window by using the `browse` subcommand to the `prodreg` command.

1 **Open a terminal window.**

2 Browse the Solaris Product Registry.

```
% prodreg browse
    BROWSE # +/-/.  UUID                                  #  NAME
    ======== =====  ====================================  =  ============
    1        -      root                                  1  System
                                                             Registry
    2         +     a01ee8dd-1dd1-11b2-a3f2-0800209a5b6b  1  Solaris 11
                                                             System
                                                             Software
    3         +     8f64eabf-1dd2-11b2-a3f1-0800209a5b6b  1  Unclassified
                                                             Software
```

The `browse` subcommand to the `prodreg` command displays the following information about registered software.

`BROWSE #` When you use the `prodreg browse` command, the Solaris Product Registry generates a *browse number* for each registered software component. This number can be used as an argument to either the `prodreg browse` command or the `info` subcommand to descend the hierarchy of specific registered components.

Note – Browse numbers might change when you reboot or reinstall your system. Do not store browse numbers in scripts or attempt to reuse them between separate login sessions.

`+/-/.` This field indicates if a software component has additional software component children registered in the Solaris Product Registry.

The following characters are displayed in this field:

- `+` indicates that the software component has additional children components that are not currently displayed.
- `-` indicates that the software component has additional children components that are currently displayed.
- `.` indicates that the software component does not have children components.

`UUID` This field lists the software's unique identifier in the Solaris Product Registry.

`#` This field indicates the *instance number* of the software component on the system. If the system contains multiple instances of a software component, the Solaris Product Registry assigns a separate instance number to each instance of the component.

`NAME` This field lists the localized name of the software. The name of the Solaris OS in this sample output is the Solaris 10 system software.

3 **Browse the information for one of the software components that are listed in the Solaris Product Registry.**

```
% prodreg browse  -m "name"
```

The `-m` "*name*" command displays information on the software component with the name *name*.

4 **If the system contains multiple instances of *name* software, type the following command to browse the Solaris Product Registry:**

```
% prodreg browse  -u name-UUID -i instance -n number
```

`-u` *name-UUID*	Displays information on the *name* software component with the unique identifier *name-UUID*.
`-i` *instance*	Displays information on the *name* software component with the instance number *instance*.
`-n` *number*	Displays software information by referencing the component's browse number *number*.

5 **Repeat Step 3 and Step 4 for each software component that you want to browse.**

Example 19–1 Viewing Software Information by Component Name (`prodreg`)

The following example shows how to view software information by referencing the component's name.

```
% prodreg browse
    BROWSE # +/-/.  UUID                                  #  NAME
    ======== =====  ====================================  =  ============
    1        -      root                                  1  System
                                                             Registry
    2         +     a01ee8dd-1dd1-11b2-a3f2-0800209a5b6b  1  Solaris 10
                                                             System
                                                             Software
    3         +     8f64eabf-1dd2-11b2-a3f1-0800209a5b6b  1  Unclassified
                                                             Software

% prodreg browse -m "Solaris 10 System Software"
```

Example 19–2 Viewing Software Information by Component Browse Number (`prodreg`)

The following example shows how to use the `-n` option with the `prodreg browse` command to view software information by referencing the component's browse number.

```
% prodreg browse
    BROWSE # +/-/.  UUID                                  #  NAME
    ======== =====  ====================================  =  ============
    1        -      root                                  1  System
                                                             Registry
    2         +     a01ee8dd-1dd1-11b2-a3f2-0800209a5b6b  1  Solaris 10
                                                             System
                                                             Software
    3         +     8f64eabf-1dd2-11b2-a3f1-0800209a5b6b  1  Unclassified
                                                             Software

% prodreg browse -n 2
```

Example 19–3 Viewing Software Information by Component UUID (`prodreg`)

The following example shows how to use the `-u` option with the `prodreg browse` command to view software information by referencing the component's UUID. The UUID is the software's unique identifier in the Solaris Product Registry.

```
% prodreg browse
    BROWSE # +/-/.  UUID                                  #  NAME
    ======== =====  ====================================  =  ============
    1        -      root                                  1  System
                                                             Registry
    2         +     a01ee8dd-1dd1-11b2-a3f2-0800209a5b6b  1  Solaris 10
                                                             System
                                                             Software
    3         +     8f64eabf-1dd2-11b2-a3f1-0800209a5b6b  1  Unclassified
                                                             Software

% prodreg browse -u a01ee8dd-1dd1-11b2-a3f2-0800209a5b6b
```

▼ How to View Software Attributes (`prodreg`)

You can view specific software attributes by using the `info` subcommand of the `prodreg` command.

The `prodreg info` command displays a variety of information about registered software, including the following items:

- Software component name
- Software component description
- Required components of the software
- Other components that require the software
- Base directory of the software

- Path to the software component

1 **Open a terminal window.**

2 **Browse the Solaris Product Registry.**

```
% prodreg browse
    BROWSE # +/-/.  UUID                                 #  NAME
    ======== =====  ==================================== =  ============
    1        -      root                                 1  System
                                                            Registry
    2         +     a01ee8dd-1dd1-11b2-a3f2-0800209a5b6b 1  Solaris 10
                                                            System
                                                            Software
    3         +     8f64eabf-1dd2-11b2-a3f1-0800209a5b6b 1  Unclassified
                                                            Software
```

3 **View the attributes for one of the listed software components.**

```
% prodreg info  -m "name"
```

The `-m` "*name*" command displays the attributes of the software component with the name *name*.

4 **Repeat Step 3 for each software component you want to view.**

Example 19–4 Viewing Software Attributes by Component Name (`prodreg`)

The following example shows how to view software attributes by referencing the component's name.

```
% prodreg browse
    BROWSE # +/-/.  UUID                                 #  NAME
    ======== =====  ==================================== =  ============
    1        -      root                                 1  System
                                                            Registry
    2         +     a01ee8dd-1dd1-11b2-a3f2-0800209a5b6b 1  Solaris 10
                                                            System
                                                            Software
    3         +     8f64eabf-1dd2-11b2-a3f1-0800209a5b6b 1  Unclassified
                                                            Software

% prodreg info -m "Solaris 10 System Software"
```

Example 19–5 Viewing Software Attributes by Component Browse Number (`prodreg`)

The following example shows how to use the `-n` option with the `prodreg info` command to view software attributes by referencing the component's browse number.

```
% prodreg browse
    BROWSE # +/-/.  UUID                                  #  NAME
    ======== =====  ====================================  =  ============
    1        -      root                                  1  System
                                                             Registry
    2         +     a01ee8dd-1dd1-11b2-a3f2-0800209a5b6b  1  Solaris 10
                                                             System
                                                             Software
    3         +     8f64eabf-1dd2-11b2-a3f1-0800209a5b6b  1  Unclassified
                                                             Software

% prodreg info -n 2
```

Example 19–6 Viewing Software Attributes by Component UUID (`prodreg`)

The following example shows how to use the `-u` option with the `prodreg info` command to view software attributes by referencing the component's UUID. The UUID is the software's unique identifier in the Solaris Product Registry.

```
% prodreg browse
    BROWSE # +/-/.  UUID                                  #  NAME
    ======== =====  ====================================  =  ============
    1        -      root                                  1  System
                                                             Registry
    2         +     a01ee8dd-1dd1-11b2-a3f2-0800209a5b6b  1  Solaris 10
                                                             System
                                                             Software
    3         +     8f64eabf-1dd2-11b2-a3f1-0800209a5b6b  1  Unclassified
                                                             Software

% prodreg info -u a01ee8dd-1dd1-11b2-a3f2-0800209a5b6b
```

▼ How to Check for Software Dependencies (`prodreg`)

You can use the `prodreg info` command to view components that depend on a specific software component. You might want to check dependencies between software products before you uninstall specific components.

1 **Open a terminal window.**

2 **Browse the Solaris Product Registry.**

```
% prodreg browse
    BROWSE # +/-/.  UUID                                  #  NAME
    ======== =====  ====================================  =  ============
    1        -      root                                  1  System
                                                             Registry
    2         +     a01ee8dd-1dd1-11b2-a3f2-0800209a5b6b  1  Solaris 10
                                                             System
                                                             Software
    3         +     8f64eabf-1dd2-11b2-a3f1-0800209a5b6b  1  Unclassified
                                                             Software
```

Repeat the `prodreg browse` command until the software component you want to check is displayed. See "How to View Installed or Uninstalled Software Information (`prodreg`)" on page 379 for more information on browsing the Solaris Product Registry by using the `prodreg browse` command.

3 **View the dependencies of a specific software component.**

```
% prodreg info -m "name" -a "Dependent Components"
```

`-m` "*name*"	Displays the attributes of the software component with the name *name*.
`-a` "Dependent Components"	Displays components that depend on *name* software by displaying the values of the Dependent Components attribute.

This command output lists the software components that depend on *name* software.

Example 19–7 Viewing Components That Depend on Other Software Products (`prodreg`)

The following example shows how to view the components that depend on the software product that is named `ExampleSoft`.

```
% prodreg -m "ExampleSoft" -a "Dependent Components"
Dependent Components:
Name                         UUID                                  #
---------------------------  ------------------------------------  -
ExampleSoftA                 7f49ecvb-1ii2-11b2-a3f1-0800119u7e8e  1
```

▼ How to Identify Damaged Software Products (`prodreg`)

If you remove installed software files or packages without using the appropriate uninstaller, you can damage the software on your system. If software is damaged, the software might not function properly. You can use the `info` subcommand of the `prodreg` command to help you determine if a software product is damaged.

1 View the Solaris Product Registry information on the software you want to check.

```
% prodreg browse -m name
BROWSE #  +/-/.  UUID                                  #  NAME
========  =====  ====================================  =  ============
1         -      root                                  1  System
                                                          Registry
2          +     a01ee8dd-1dd1-11b2-a3f2-0800209a5b6b  1  Solaris 10
                                                          System
                                                          Software
3          +     8f64eabf-1dd2-11b2-a3f1-0800209a5b6b  1  Unclassified
                                                          Software
4          -     name-UUID                               1  name
233         .    component-a-pkg                          1  component-a
234         .    component-b-pkg                          1
```

`-m` "*name*"	Displays information on the software component with the name *name*.
name-UUID	Specifies the UUID of the *name* software component.
component-a-pkg	Specifies the package name of the *component-a* component that depends on *name* software.
component-a	Specifies the name of a component that depends on *name* software.
component-b-pkg	Specifies the package name of the *component-b* component that depends on *name* software.

In the previous sample output, the *component-b-pkg* entry does not have an associated name in the `Name` field. If a software component name is not displayed in the Solaris Product Registry, the component might be damaged.

2 Verify that the software component is damaged.

```
% prodreg info -u name-UUID -i 1 -d
isDamaged=TRUE
```

`-u` *name-UUID*	Displays information on the *name* software component.
`-i 1`	Displays information on the first instance of the *name* software component.
`-d`	Displays the value of the `isDamaged` attribute of the *name* software component.

The output `isDamaged=TRUE` indicates that the *name* software component is damaged.

3 **Identify the packages that form the *name-UUID* software component.**

```
% prodreg info -u name-UUID -i 1 -a PKGS
pkgs:
component-a-pkg  component-b-pkg
```

4 **Verify that these packages are installed on the system.**

```
% pkginfo component-a-pkg
application  component-a-pkg  component-a

% pkginfo component-b-pkg
ERROR: information on "component-b-pkg" was not found
```

The error message output of the `pkginfo` *component-b-pkg* command indicates that the *component-b-pkg* package has been removed from the system. The *name* software component might not work without the *component-b-pkg* package.

Example 19–8 Identifying Damaged Software Components (`prodreg`)

The following example shows how to determine if the ExampleSoft software component is damaged.

```
% prodreg browse -m Examplesoft
BROWSE #  +/-/.  UUID                                  #  NAME
========  =====  ====================================  =  ============
1         -      root                                  1  System
                                                          Registry
2          +     a01ee8dd-1dd1-11b2-a3f2-0800209a5b6b  1  Solaris 10
                                                          System
                                                          Software
3          +     8f64eabf-1dd2-11b2-a3f1-0800209a5b6b  1  Unclassified
                                                          Software
4          -     95842091-725a-8501-ef29-0472985982be  1  ExampleSoft
233         .    90209809-9785-b89e-c821-0472985982be  1  Example Doc
234         .    EXSOzzt                               1
235         .    EXSOblob                              1  Example Data
```

The ExampleSoft child component EXSOzzt does not have an entry in the NAME field. The ExampleSoft software might be damaged. You would use the prodreg info command with the -u, -i, and -d options to determine if the ExampleSoft software is damaged.

```
% prodreg info -u 95842091-725a-8501-ef29-0472985982be -i 1 -d
isDamaged=TRUE
```

The output isDamaged=TRUE indicates that the ExampleSoft software is damaged. You would use the -a PKGS option of the prodreg info command to identify the ExampleSoft software packages.

```
% prodreg info
     -u 95842091-725a-8501-ef29-0472985982be
     -i 1 -a PKGS
pkgs:
EXSOzzt EXSOblob
```

To verify that the EXSOzzt and EXSOblob packages are installed on the system, you would use the pkginfo command.

```
% pkginfo EXSOzzt
ERROR: information for "EXSOzzt" was not found

% pkginfo EXSOblob
application EXSOblob       Example Data
```

The output of the pkginfo command indicates that the EXSOzzt package is not installed on the system. Thus, the ExampleSoft software is damaged.

▼ How to Uninstall Software (prodreg)

You can use the uninstall subcommand of the prodreg command to remove software from your system. When you uninstall software by using the prodreg uninstall command, you remove a specified software and all the child components associated with that software. Before you remove software, verify that other software does not depend on the software you want to uninstall. See "How to Check for Software Dependencies (prodreg)" on page 384.

After you uninstall software, you can remove that software and all the child components of that software from the Solaris Product Registry by using the prodreg unregister -r command.

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 **View the information on the software you want to uninstall.**

```
# prodreg browse -u name-UUID
BROWSE #  +/-/.  UUID                                  #  NAME
========  =====  ====================================  =  ============
1         -      root                                  1  System
                                                          Registry
2          +     a01ee8dd-1dd1-11b2-a3f2-0800209a5b6b  1  Solaris 10
                                                          System
                                                          Software
3          +     8f64eabf-1dd2-11b2-a3f1-0800209a5b6b  1  Unclassified
                                                          Software
1423       -     name-UUID                              1  name
1436        .    component-a-UUID                        1  component-a
1437        -    component-b-UUID                        1  component-b
1462         .   component-c-UUID                        1  component-c
```

`-u` *name-UUID*	Displays information on the software component with the unique identifier *name-UUID*.
name	Specifies the name of the software component you want to uninstall with the unique identifier *name-UUID*.
. *component-a-UUID*	Specifies the unique identifier of the *component-a* software component that is required by *name* software.
component-a	Specifies the name of a component that is required by *name* software.
- *component-b-UUID*	Specifies the unique identifier of the *component-b* component that is required by *name* software. The - symbol indicates that *component-b* requires an additional software component.
component-b	Specifies the name of a software component that is required by *name* software.
. *component-c-UUID*	Specifies the unique identifier of the *component-b* software component that is required by *component-b* software.
component-c	Specifies the name of a software component that is required by *component-b* software.

3 **Uninstall the software.**

```
# prodreg uninstall -u name-UUID
```

4 **Check the dependencies for the software that you want to uninstall.**

```
# prodreg info -u name-UUID
Title: name
.
.
.
Child Components:
Name                        UUID                                    #
-------------------------  ------------------------------------  -
component-a                    component-a-UUID                          1
component-b                    component-b-UUID                          1

Required Components:
Name                        UUID                                    #
-------------------------  ------------------------------------  -
component-a                    component-a-UUID                          1
component-b                    component-b-UUID                          1
```

Check the following information in the output of the `prodreg info` command.

- Child Components – Lists the software components that are associated with the *name* software component. When you unregister the *name* software, you also unregister the child components of *name* software. If the output of the previous `prodreg info` command lists any child components, verify that you want to unregister these child components.
- Required Components – Lists the software components that are required by the *name* software component. Software components might require other components that are not child components. When you uninstall and unregister a component, only child components are unregistered and uninstalled.
- Dependent Components – Lists the components that require *name* software to run. When you unregister the *name* software, you also unregister the dependent components of *name* software. If the output of the `prodreg info` command lists any dependent components, verify that you want to unregister these dependent components.

In the previous sample output, *name* software does not have any dependent components.

5 **Check the dependencies of *name* software's child components.**

```
# prodreg info -u component-a-UUID -i 1 -a "Dependent Components"
Dependent Components:
Name                         UUID                                     #
--------------------------  -------------------------------------  -
name                           name-UUID                                  1

# prodreg info -u component-b-UUID -i 1 -a "Dependent Components"
Dependent Components:
Name                         UUID                                     #
--------------------------  -------------------------------------  -
```

```
name                           name-UUID                     1

# prodreg info -u component-c-UUID -i 1 -a "Dependent Components"
Dependent Components:
Name                        UUID                                  #
--------------------------  ------------------------------------  -
component-b                    component-b-UUID                   1
```

The sample output shows that no other software depends on the child components of *name* software.

6 **Unregister the software and its child components.**

```
# prodreg unregister -r -u name-UUID -i 1
```

`-r`	Recursively unregisters software with the unique identifier *name-UUID* and all the child components of this software.
`-u` *name-UUID*	Specifies the unique identifier of the software you want to unregister.
`-i 1`	Specifies the instance of the software you want to unregister.

Example 19–9 Uninstalling Software Components (`prodreg`)

The following example shows how to uninstall ExampleSoft software and all the child components of ExampleSoft software.

```
# prodreg browse -m "ExampleSoft"
BROWSE #  +/-/.  UUID                                  #  NAME
========  =====  ====================================  =  ============
1           -    root                                  1  System
                                                          Registry
2            +   a01ee8dd-1dd1-11b2-a3f2-0800209a5b6b  1  Solaris 10
                                                          System
                                                          Software
3            +   8f64eabf-1dd2-11b2-a3f1-0800209a5b6b  1  Unclassified
                                                          Software
1423         -   95842091-725a-8501-ef29-0472985982be  1  ExampleSoft
1436          .  90209809-9785-b89e-c821-0472985982be  1  Example Doc
1437          -  EXSOzzt                               1  Example Data
1462           . EXSOblob                              1  Example Data

# prodreg uninstall -u 95842091-725a-8501-ef29-0472985982be -i 1

# prodreg info -u 95842091-725a-8501-ef29-0472985982be
```

```
Title: ExampleSoft Software
.
.
.
Child Components:
Name                        UUID                                  #
--------------------------  ------------------------------------  -
Example Doc                 90209809-9785-b89e-c821-0472985982be  1
Example Data                EXSOzzt                               1

Required Components:
Name                        UUID                                  #
--------------------------  ------------------------------------  -
Example Doc                 90209809-9785-b89e-c821-0472985982be  1
Example Data                EXSOzzt                               1

# prodreg info -u 90209809-9785-b89e-c821-0472985982be -i 1
    -a "Dependent Components"
Dependent Components:
Name                         UUID                                  #
---------------------------  ------------------------------------  -
ExampleSoft                  95842091-725a-8501-ef29-0472985982be  1

# prodreg info -u EXSOzzt -i 1 -a "Dependent Components"
Dependent Components:
Name                         UUID                                  #
---------------------------  ------------------------------------  -
ExampleSoft                  95842091-725a-8501-ef29-0472985982be  1

# prodreg info -u EXSOblob -i 1 -a "Dependent Components"
Dependent Components:
Name                         UUID                                  #
---------------------------  ------------------------------------  -
Example Data                 EXSOzzt                               1

# prodreg unregister -r -u 95842091-725a-8501-ef29-0472985982be -i 1
```

▼ How to Uninstall Damaged Software (`prodreg`)

If you try to uninstall a damaged software component by using the `prodreg uninstall` command, the command might fail. This failure can occur if the uninstaller program for the software component has been removed from the system.

Follow these steps to uninstall a software component with no associated uninstaller program on the system.

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 **View the information on the software you want to uninstall.**

```
# prodreg browse -m "name"
BROWSE #  +/-/.  UUID                                  #  NAME
========  =====  ====================================  =  ============
1         -      root                                  1  System
                                                          Registry
2          +     a01ee8dd-1dd1-11b2-a3f2-0800209a5b6b  1  Solaris 10
                                                          System
                                                          Software
3          +     8f64eabf-1dd2-11b2-a3f1-0800209a5b6b  1  Unclassified
                                                          Software
4           -    UUID                                   1  name
1436        .    component-a-UUID                         1  component-a
1437        .    component-b-UUID                         1
```

`-m` "*name*"	Displays information on the *name* software component you want to uninstall.
UUID	Specifies the UUID of the software component you want to uninstall.
. *component-a-UUID*	Specifies the UUID of the *component-a* software component.
component-a	Specifies the name of a child software component of *name* software.
. *component-b-UUID*	Specifies the UUID of a child software component of *name* software.

The *component-b-UUID* entry does not have an associated component name. The missing name value might indicate that this component is damaged.

3 **Uninstall the software.**

```
# prodreg uninstall -u UUID -i 1
The install program requested could not be found
```

`-u` *UUID*	Specifies the UUID of the software component you want to uninstall.
`-i` 1	Specifies the instance of the software you want to uninstall.

The error message indicates that the uninstaller program is not on the system.

4 **Identify the uninstaller program for the software component.**

```
# prodreg info -m "name" -a uninstallprogram
uninstallprogram: /usr/bin/java -mx64m -classpath
uninstaller-location uninstall_name
```

`-m` "*name*"	Displays information on the *name* software component.
`-a uninstallprogram`	Displays information on the uninstaller program that is associated with the *name* software component.
uninstaller-location	Specifies the registered location of the uninstaller program for the *name* software component.

5 **Determine if the uninstaller is in the registered location.**

```
# ls uninstaller-location
uninstaller-location:
No such file or directory
```

The output of the `ls` command indicates that the uninstaller program is not in the registered location.

6 **Remove the software from the system in one of the following ways:**

- **If you have a system backup available, follow these steps:**

  a. **Load the uninstaller program from the backup.**

  b. **Run the uninstaller program from a shell command-line interface such as a terminal window.**

- **If you do not have access to the uninstaller program on a backup, follow these steps:**

  a. **Unregister the software component.**

  ```
  # prodreg unregister -u UUID -i 1
  ```

  b. **Remove any remaining registered components that are required by the software you want to remove.**

  ```
  # pkgrm component-a-UUID
  ```

Example 19–10 Uninstalling Damaged Software (`prodreg`)

The following example shows how to uninstall the damaged ExampleSoft software. In this example, the uninstaller program is not readily available on a system backup.

```
# prodreg browse -m Examplesoft
BROWSE #  +/-/.  UUID                                  #  NAME
========  =====  ====================================  =  ============
1         -      root                                  1  System
                                                          Registry
2          +     a01ee8dd-1dd1-11b2-a3f2-0800209a5b6b  1  Solaris 10
                                                          System
                                                          Software
3          +     8f64eabf-1dd2-11b2-a3f1-0800209a5b6b  1  Unclassified
                                                          Software
4          -     95842091-725a-8501-ef29-0472985982be  1  ExampleSoft
233         .    90209809-9785-b89e-c821-0472985982be  1  Example Doc
234         .    EXSOzzt                               1
235         .    EXSOblob                              1  Example Data

# prodreg uninstall -u 95842091-725a-8501-ef29-0472985982be -i 1
The install program requested could not be found

# prodreg info -m "ExampleSoft" -a uninstallprogram
uninstallprogram: /usr/bin/java -mx64m -classpath
/var/sadm/prod/org.example.ExampleSoft/987573587 uninstall_ExampleSoft

# ls /var/sadm/prod/org.example.ExampleSoft/987573587
/var/sadm/prod/org.example.ExampleSoft/987573587:
No such file or directory

# prodreg unregister -u 95842091-725a-8501-ef29-0472985982be -i 1

# pkgrm EXSOblob
```

▼ How to Reinstall Damaged Software Components (prodreg)

If other software depends on a damaged software component, you might want to reinstall the damaged component, rather than uninstall the component and the other dependent software. You can use the `-f` option with the `prodreg unregister` command to forcibly the unregister the damaged component. Then, you can reinstall the component.

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see “Configuring RBAC (Task Map)” in *System Administration Guide: Security Services*.

2 **View the information on the software you want to reinstall.**

```
# prodreg browse -m "name"
BROWSE #  +/-/.  UUID                                  #  NAME
========  =====  ====================================  =  ============
1         -      root                                  1  System
                                                          Registry
2          +     a01ee8dd-1dd1-11b2-a3f2-0800209a5b6b  1  Solaris 10
                                                          System
                                                          Software
3          +     8f64eabf-1dd2-11b2-a3f1-0800209a5b6b  1  Unclassified
                                                          Software
4           .    UUID                                   1  name
```

`-m` "*name*"	Displays information on the *name* software component you want to reinstall.
UUID	Specifies the UUID of the software component you want to reinstall.

3 **Identify the software that depends on the software you want to reinstall.**

```
# prodreg info -m "name" -a "Dependent Components"
Dependent Components:
Name                          UUID                                  #
----------------------------  ------------------------------------  -
component-a                      component-a-UUID 1
```

`-m` "*name*"	Specifies the name of the software component you want to reinstall.
`-a` "`Dependent Components`"	Displays the components that depend on *name* software.
component-a	Specifies the name of a software component that depends on *name* software.
component-a-UUID	Specifies the UUID of the *component-a* software component.

The *component-a* software component depends on the software you want to reinstall. To reinstall *name* software and not unregister *component-a*, you must forcibly unregister the *name* software, then reinstall *name* software.

4 **Unregister the software component you want to reinstall.**

```
# prodreg unregister -f -u UUID
```

5 **Reinstall the software component.**

```
# /usr/bin/java -cp /usr/installers/installer
```

The *installer* option specifies the name of the installer program for *name* software.

Example 19–11 Reinstalling Damaged Software Components (`prodreg`)

The following example shows how to reinstall the damaged software component `ComponentSoft` without unregistering or uninstalling the dependent component ExampleSoft.

```
# prodreg browse -m "ComponentSoft"
BROWSE #  +/-/.  UUID                                  #  NAME
========  =====  ====================================  =  ============
1         -      root                                  1  System
                                                          Registry
2          +     a01ee8dd-1dd1-11b2-a3f2-0800209a5b6b  1  Solaris 10
                                                          System
                                                          Software
3          +     8f64eabf-1dd2-11b2-a3f1-0800209a5b6b  1  Unclassified
                                                          Software
4           .    86758449-554a-6531-fe90-4352678362fe  1  ComponentSoft

# prodreg info -m "ComponentSoft" -a "Dependent Components"
Dependent Components:
Name                         UUID                                  #
---------------------------  ------------------------------------  -
ExampleSoft                  95842091-725a-8501-ef29-0472985982be  1

# prodreg unregister -f -u 86758449-554a-6531-fe90-4352678362fe -i 1

# /usr/bin/java -cp /usr/installers/org.example.componentsoft
```

◆ ◆ ◆ CHAPTER 20

Managing Software by Using Package Commands (Tasks)

This chapter describes how to add, verify, and remove software packages by using the package commands.

For information on the procedures associated with performing these tasks, see:

- "Adding and Removing Signed Packages by Using the `pkgadd` Command (Task Map)" on page 399
- "Managing Software Packages by Using Package Commands (Task Map)" on page 405

Adding and Removing Signed Packages by Using the `pkgadd` Command (Task Map)

The following task map describes software management tasks that you can perform with signed package commands.

Task	Description	For Instructions
Import a certificate.	You can import a trusted certificate by using the `pkgadm addcert` command.	"How to Import a Trusted Certificate From the Java Keystore (`pkgadm addcert`)" on page 400
Print the details of one or more certificates.	You can print the details of a certificate by using the `pkgadm listcert` command.	"How to Display Certificate Information (`pkgadm listcert`)" on page 402
Remove a certificate.	You can remove a certificate by using the `pkgadm removecert` command.	"How to Remove a Certificate (`pkgadm removecert`)" on page 402

Task	Description	For Instructions
Set up a proxy server.	Use this procedures for systems that are set up behind a firewall with a proxy.	"How to Set Up a Proxy Server (pkgadd)" on page 403
Add a signed package.	After the root certificate is imported, you can add a signed package by using he `pkgadd` command.	"How to Add a Signed Package (pkgadd)" on page 404

Adding and Removing Signed Packages by Using the `pkgadd` Command

The following procedures explain how to add and remove signed packages by using the `pkgadd` command.

▼ How to Import a Trusted Certificate From the Java Keystore (`pkgadm addcert`)

1 Become superuser or assume an equivalent role.

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 Verify that the root certificate authority (CA) certificate exists in the Java ™ keystore.

```
# keytool -storepass storepass -list -keystore certfile
```

`keytool`	Manages a Java keystore (database) of private keys and their associated X.509 certificate chains that authenticate the corresponding public keys. Also manages certificates from trusted entities. For more information on the `keytool` utility, see keytool-Key and Certificate Management Tool.
`-storepass` *storepass*	Specifies the password that protects the integrity of the keystore.
`-list`	By default, prints the MD5 fingerprint of a certificate.
`-keystore` *certfile*	Specifies the name and location of the persistent keystore file.

3 Export the root CA certificate from the Java keystore to a temporary file.

```
# keytool -export -storepass storepass -alias verisignclass2g2ca -keystore
/usr/java/jre/lib/security/cacerts certfile -file filename
```

`-export`	Exports the trusted certificate.
`-storepass` *storepass*	Specifies the password that protects the integrity of the Java keystore.
`-alias verisignclass2g2ca`	Identifies the alias of the trusted certificate.
`-keystore` *certfile*	Specifies the name and location of the keystore file.
`-file` *filename*	Identifies the file to hold the exported certificate.

4 **Import a trusted certificate to the package keystore.**

```
# pkgadm addcert -t -f format  certfile
```

`-t`	Indicates that the certificate is a trusted CA certificate. The output includes the details of the certificate, which the user is asked to verify.
`-f` *format*	Specifies the format of certificates and private keys. When you import a certificate, it must be encoded using PEM or binary DER format.
certfile	Specifies the file that contains the certificate.

5 **Remove the temporary file.**

```
# rm /tmp/file-name
```

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

Example 20–1 Importing a Trusted Certificate From the Java Keystore

The following example shows how to import a trusted certificate. In this example, Sun's root CA certificate is imported from the Java keystore into the package keystore by using the `keytool` command.

```
# keytool -export -storepass changeit -alias verisignclass2g2ca \
-keystore /usr/java/jre/lib/security/cacerts -file /tmp/root.crt
Certificate stored in file </tmp/root.crt>

# pkgadm addcert -t -f der /tmp/root.crt
     Keystore Alias: /C=US/O=VeriSign, Inc./OU=Class 2 Public Primary Certification Authority - G2/O
        Common Name: /C=US/O=VeriSign, Inc./OU=Class 2 Public Primary Certification Authority - G2/O
   Certificate Type: Trusted Certificate
Issuer Common Name: /C=US/O=VeriSign, Inc./OU=Class 2 Public Primary Certification Authority - G2/O
     Validity Dates: <May 18 00:00:00 1998 GMT> - <Aug  1 23:59:59 2028 GMT>
MD5 Fingerprint: 2D:BB:E5:25:D3:D1:65:82:3A:B7:0E:FA:E6:EB:E2:E1
   SHA1 Fingerprint: B3:EA:C4:47:76:C9:C8:1C:EA:F2:9D:95:B6:CC:A0:08:1B:67:EC:9D

Are you sure you want to trust this certificate? yes
Trusting certificate </C=US/O=VeriSign, Inc./OU=Class 2 Public Primary Certification Authority - G2/O>
```

```
Type a Keystore protection Password. xxxxxx
Press ENTER for no protection password (not recommended):
For Verification: Type a Keystore protection Password.
Press ENTER for no protection password (not recommended):
Certificate(s) from </tmp/root.crt> are now trusted
```

▼ How to Display Certificate Information (`pkgadm listcert`)

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 **Display the contents of the package keystore.**

```
# pkgadm listcert -p passarg
```

Example 20–2 Displaying Certificate Information

The following example shows how to display the details of a locally stored certificate.

```
# pkgadm listcert -P pass:test123
Keystore Alias: /C=US/O=VeriSign, Inc./OU=Class 2 Public Primary Certification Authority - G2/O
        Common Name: /C=US/O=VeriSign, Inc./OU=Class 2 Public Primary Certification Authority - G2/O
   Certificate Type: Trusted Certificate
Issuer Common Name: /C=US/O=VeriSign, Inc./OU=Class 2 Public Primary Certification Authority - G2/O
Validity Dates: <May 18 00:00:00 1998 GMT> - <Aug 1 23:59:59 2028 GMT>
 MD5 Fingerprint: 2D:BB:E5:25:D3:D1:65:82:3A:B7:0E:FA:E6:EB:E2:E1
   SHA1 Fingerprint: B3:EA:C4:47:76:C9:C8:1C:EA:F2:9D:95:B6:CC:A0:08:1B:67:EC:9D
```

▼ How to Remove a Certificate (`pkgadm removecert`)

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 **Remove the trusted certificate from the package keystore.**

```
# pkgadm removecert -n "certfile"
```

The `removecert -n` "*certfile*" option specifies the alias of the user certificate/key pair or the alias of the trusted certificate.

Note – View the alias names for certificates by using the `pkgadm listcert` command.

Example 20–3 Removing a Certificate

The following example shows how to remove a certificate.

```
# pkgadm listcert
    Keystore Alias: /C=US/O=VeriSign, Inc./OU=Class 2 Public Primary Certification Authority - G2/O
        Common Name: /C=US/O=VeriSign, Inc./OU=Class 2 Public Primary Certification Authority - G2/O
   Certificate Type: Trusted Certificate
Issuer Common Name: /C=US/O=VeriSign, Inc./OU=Class 2 Public Primary Certification Authority - G2/O
Validity Dates: <May 18 00:00:00 1998 GMT> - <Aug 1 23:59:59 2028 GMT>
 MD5 Fingerprint: 2D:BB:E5:25:D3:D1:65:82:3A:B7:0E:FA:E6:EB:E2:E1
   SHA1 Fingerprint: B3:EA:C4:47:76:C9:C8:1C:EA:F2:9D:95:B6:CC:A0:08:1B:67:EC:9D
# pkgadm removecert -n "/C=US/O=VeriSign, Inc./OU=Class 2 Public Primary Certification Authority - G2/O"
Enter Keystore Password: storepass
Successfully removed Certificate(s) with alias \
</C=US/O=VeriSign, Inc./OU=Class 2 Public Primary Certification Authority - G2/O>
```

▼ How to Set Up a Proxy Server (pkgadd)

If your system is behind a firewall with a proxy, you will need to set up a proxy server before you can add a package from an HTTP server by using the `pkgadd` command.

1. **Become superuser or assume an equivalent role.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2. **Select one of the following methods to specify a proxy server.**

   a. **Specify the proxy server by using the `http_proxy`, `HTTPPROXY`, or `HTTPPROXYPORT` environment variable.**

      For example:

      ```
      # setenv http_proxy http://mycache.domain:8080
      ```

      Or, specify one of the following:

      ```
      # setenv HTTPPROXY mycache.domain
      # setenv HTTPPROXYPORT 8080
      ```

b. **Specify the proxy server on the pkgadd command line.**

For example:

```
# pkgadd -x mycache.domain:8080 -d http://myserver.com/pkg SUNWpkg
```

c. **Create an administration file that includes proxy server information.**

For example:

```
# cat /tmp/admin
mail=
instance=unique
partial=ask
runlevel=ask
idepend=ask
rdepend=ask
space=ask
setuid=ask
conflict=ask
action=ask
networktimeout=60
networkretries=3
authentication=quit
keystore=/var/sadm/security
basedir=default
proxy=mycache.domain:8080
```

Then, identify the administration file by using the pkgadd -a command. For example:

```
# pkgadd -a /tmp/admin -d http://myserver.com/pkg SUNWpkg
```

▼ How to Add a Signed Package (pkgadd)

This procedure assumes that you have imported Sun's root CA certificate. For more information, see "How to Import a Trusted Certificate From the Java Keystore (pkgadm addcert)" on page 400.

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 **Add a signed package.**

```
# pkgadd -d /pathname/device-name
```

The -d *device-name* option specifies the device from which the package is installed. The device can be a directory, tape, diskette, or removable disk. The device can also be a data stream created by the pkgtrans command.

Example 20–4 Adding a Signed Package

The following example shows how to add a signed package that is stored on the system.

```
# # pkgadd -d /tmp/signed_pppd
The following packages are available:
  1  SUNWpppd     Solaris PPP Device Drivers
                  (sparc) 11.10.0,REV=2003.05.08.12.24

Select package(s) you wish to process (or 'all' to process
all packages). (default: all) [?,??,q]: all
Enter keystore password:
## Verifying signature for signer <User Cert 0>

.
.
.
```

The following example shows how to install a signed package using an HTTP URL as the device name. The URL must point to a stream-formatted package.

```
# pkgadd -d http://install/signed-video.pkg

## Downloading...
..............25%..............50%..............75%..............100%
## Download Complete
.
.
.
```

Managing Software Packages by Using Package Commands (Task Map)

The following task map describes the software management tasks that you can perform with the package commands for both signed and unsigned packages.

Task	Description	For Instructions
Add software packages to the local system.	You can add software packages to the local system by using the pkgadd command.	"How to Add Software Packages (pkgadd)" on page 406

Task	Description	For Instructions
Add software packages to a spool directory.	You can add software packages to a spool directory without actually installing the software.	"Adding a Software Package to a Spool Directory" on page 409
List information about all installed software packages.	You can list information about installed packages by using the `pkginfo` command.	"How to List Information About All Installed Packages (`pkginfo`)" on page 411
Check the integrity of installed software packages.	You can verify the integrity of installed software packages by using the `pkgchk` command.	"How to Check the Integrity of Installed Software Packages (`pkgchk`)" on page 412
Check the integrity of an installed object.	You can verify the integrity of an installed object by using the `pkchk` command with the `-p` and `-P` options. The `-p` option specifies the full path name. The new `-P` option specifies a partial path name.	"How to Check the Integrity of Installed Objects (`pkgchk -p, pkgchk -P`)" on page 413
Remove software packages.	You can remove unneeded software packages by using the `pkgrm` command.	"How to Remove Software Packages (`pkgrm`)" on page 416

Using Package Commands to Manage Software Packages

The following procedures explain how to manage software packages by using package commands.

▼ How to Add Software Packages (`pkgadd`)

1 Become superuser or assume an equivalent role.

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 Remove any already installed packages with the same names as the packages you are adding.

This step ensures that the system keeps a proper record of software that has been added and removed. Sometimes, you might want to maintain multiple versions of the same application on the system. For strategies on maintaining multiple software copies, see "Guidelines for Removing Packages (`pkgrm`)" on page 368. For task information, see "How to Remove Software Packages (`pkgrm`)" on page 416.

3 Add a software package to the system.

```
# pkgadd -a admin-file -d device-name pkgid ...
```

`-a` *admin-file*	(Optional) Specifies an administration file that the `pkgadd` command should check during the installation. For details about using an administration file, see "Using an Administration File" on page 369.
`-d` *device-name*	Specifies the absolute path to the software packages. *device-name* can be the path to a device, a directory, or a spool directory. If you do not specify the path where the package resides, the `pkgadd` command checks the default spool directory (`/var/spool/pkg`). If the package is not there, the package installation fails.
pkgid	(Optional) Is the name of one or more packages, separated by spaces, to be installed. If omitted, the `pkgadd` command installs all available packages from the specified device, directory, or spool directory.

If the `pkgadd` command encounters a problem during installation of the package, it displays a message related to the problem, followed by this prompt:

```
Do you want to continue with this installation?
```

Respond with `yes`, `no`, or `quit`. If more than one package has been specified, type `no` to stop the installation of the package being installed. The `pkgadd` command continues to install the other packages. Type `quit` to stop the installation.

4 **Verify that the package has been installed successfully.**

```
# pkgchk -v pkgid
```

If no errors occur, a list of installed files is returned. Otherwise, the `pkgchk` command reports the error.

Example 20–5 Adding Software Packages From a Mounted CD

The following example shows how install the `SUNWpl5u` package from a mounted Solaris 10 CD. The example also shows how to verify that the package files were installed properly.

```
# pkgadd -d /media/Solaris_11/Product SUNWpl5u
    .
    .
    .
Installation of <SUNWpl5u> was successful.
# pkgchk -v SUNWpl5u
/usr
/usr/bin
/usr/bin/perl
/usr/perl5
/usr/perl5/5.8.4
.
```

```
.
.
```

Example 20–6 Installing Software Packages From a Remote Package Server

If the packages you want to install are available from a remote system, you can manually mount the directory that contains the packages (in package format) and install packages on the local system.

The following example shows how to install software packages from a remote system. In this example, assume that the remote system named `package-server` has software packages in the `/latest-packages` directory. The `mount` command mounts the packages locally on `/mnt`. The `pkgadd` command installs the `SUNWpl5u` package.

```
# mount -F nfs -o ro package-server:/latest-packages /mnt
# pkgadd -d /mnt SUNWpl5u
    .
    .
    .
Installation of <SUNWpl5u> was successful.
```

If the automounter is running at your site, you do not need to mount the remote package server manually. Instead, use the automounter path, in this case, `/net/package-server/latest-packages`, as the argument to the `-d` option.

```
# pkgadd -d /net/package-server/latest-packages SUNWpl5u
    .
    .
    .
Installation of <SUNWpl5u> was successful.
```

Example 20–7 Installing Software Packages From a Remote Package Server by Specifying an Administration File

This example is similar to the previous example, except that it uses the `-a` option and specifies an administration file named `noask-pkgadd`, which is illustrated in “Avoiding User Interaction When Adding Packages (`pkgadd`)” on page 369. In this example, assume that the `noask-pkgadd` administration file is in the default location, `/var/sadm/install/admin`.

```
# pkgadd -a noask-pkgadd -d /net/package-server/latest-packages SUNWpl5u
    .
    .
    .
Installation of <SUNWpl5u> was successful.
```

Example 20–8 Installing Software Packages From an HTTP URL

The following example shows how to install a package using an HTTP URL as the device name. The URL must point to a stream-formatted package.

```
# pkgadd -d http://install/xf86-4.3.0-video.pkg

## Downloading...
..............25%..............50%..............75%..............100%
## Download Complete


The following packages are available:
  1  SUNWxf86r     XFree86 Driver Porting Kit (Root)
                   (i386) 4.3.0,REV=0.2003.02.28
  2  SUNWxf86u     XFree86 Driver Porting Kit (User)
                   (i386) 4.3.0,REV=0.2003.02.28

.
.
.
```

Adding a Software Package to a Spool Directory

For convenience, you can copy frequently installed packages to a spool directory. If you copy packages to the default spool directory, `/var/spool/pkg`, you do not need to specify the source location of the package (`-d` *device-name* argument) when you use the `pkgadd` command. The `pkgadd` command, by default, checks the `/var/spool/pkg` directory for any packages that are specified on the command line. Note that copying packages to a spool directory is not the same as installing the packages on a system.

▼ How to Add Software Packages to a Spool Directory (`pkgadd`)

1. **Become superuser or assume an equivalent role.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2. **Remove any already spooled packages with the same names as the packages you are adding.**

   For information on removing spooled packages, see Example 20–20.

3. **Add a software package to a spool directory.**

   ```
   # pkgadd -d device-name -s spooldir pkgid ...
   ```

`-d` *device-name*	Specifies the absolute path to the software packages. *device-name* can be the path to a device, a directory, or a spool directory.
`-s` *spooldir*	Specifies the name of the spool directory where the package will be spooled. You must specify a *spooldir*.
pkgid	(Optional) Is the name of one or more packages, separated by spaces, to be added to the spool directory. If omitted, the `pkgadd` command copies all available packages.

4 **Verify that the package has been copied successfully to the spool directory.**

```
$ pkginfo -d spooldir| grep pkgid
```

If *pkgid* was copied correctly, the `pkginfo` command returns a line of information about the *pkgid*. Otherwise, the `pkginfo` command returns the system prompt.

Example 20–9 Setting Up a Spool Directory From a Mounted CD

The following example shows how to transfer the `SUNWman` package from a mounted SPARC based Solaris 10 CD to the default spool directory (`/var/spool/pkg`).

```
# pkgadd -d /media/Solaris_11/Product -s /var/spool/pkg SUNWman
Transferring <SUNWman> package instance
```

Example 20–10 Setting Up a Spool Directory From a Remote Software Package Server

If packages you want to copy are available from a remote system, you can manually mount the directory that contains the packages, in package format, and copy them to a local spool directory.

The following example shows the commands for this scenario. In this example, assume that the remote system named `package-server` has software packages in the `/latest-packages` directory. The `mount` command mounts the package directory locally on `/mnt`. The `pkgadd` command copies the `SUNWpl5p` package from `/mnt` to the default spool directory (`/var/spool/pkg`).

```
# mount -F nfs -o ro package-server:/latest-packages /mnt
# pkgadd -d /mnt -s /var/spool/pkg SUNWpl5p
Transferring <SUNWpl5p> package instance
```

If the automounter is running at your site, you do not have to mount the remote package server manually. Instead, use the automounter path, in this case, `/net/package-server/latest-packages`, as the argument to the `-d` option.

```
# pkgadd -d /net/package-server/latest-packages -s /var/spool/pkg SUNWpl5p
Transferring <SUNWpl5p> package instance
```

Example 20–11 Installing Software Packages From the Default Spool Directory

The following example shows how to install the `SUNWpl5p` package from the default spool directory. When no options are used, the `pkgadd` command searches the `/var/spool/pkg` directory for the named packages.

```
# pkgadd SUNWpl5p
    .
    .
    .
Installation of <SUNWpl5p> was successful.
```

▼ How to List Information About All Installed Packages (`pkginfo`)

● **List information about installed packages by using the `pkginfo` command.**

```
$ pkginfo
```

Example 20–12 Listing Installed Packages

This example shows how to list all packages installed on a local system, whether that system is a stand-alone system or a server. The output shows the primary category, package name, and the description of the package.

```
$ pkginfo
system      SUNWaccr       System Accounting, (Root)
system      SUNWaccu       System Accounting, (Usr)
system      SUNWadmap      System administration applications
system      SUNWadmc       System administration core libraries
.
.
.
```

Example 20–13 Displaying Detailed Information About Software Packages

This example shows how to list all packages installed on a system by specifying the long format, which includes all available information about the designated packages.

```
$ pkginfo -l SUNWcar
   PKGINST:  SUNWcar
      NAME:  Core Architecture, (Root)
  CATEGORY:  system
      ARCH:  sparc.sun4u
   VERSION:  11.9.0,REV=2002.04.06.15.27
   BASEDIR:  /
    VENDOR:  Sun Microsystems, Inc.
      DESC:  core software for a specific hardware platform group
    PSTAMP:  leo20031003183400
  INSTDATE:  Feb 20 2004 16:57
   HOTLINE:  Please contact your local service provider
    STATUS:  completely installed
     FILES:      114 installed pathnames
                  36 shared pathnames
                  40 directories
                  57 executables
               21469 blocks used (approx)
```

▼ How to Check the Integrity of Installed Software Packages (pkgchk)

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 **Check the status of an installed package.**

- To check the file attributes and contents, type the following:

```
# pkgchk -a| -c -v pkgid ...
```

- To specify the absolute path of the spool directory, type the following:

```
# pkgchk -d spooldir pkgid ...
```

`-a`	Specifies to audit only the file attributes (the permissions), rather than the file attributes and the contents, which is the default.
`-c`	Specifies to audit only the file contents, rather than the file contents and attributes, which is the default.
`-v`	Specifies verbose mode, which displays file names as they are processed.
`-d` *spooldir*	Specifies the absolute path of the spool directory.
pkgid	(Optional) Is the name of one or more packages, separated by spaces. If you do not specify a *pkgid*, all the software packages installed on the system are checked.

Example 20–14 Checking the Contents of Installed Software Packages

The following example shows how to check the contents of a package.

```
# pkgchk -c SUNWbash
```

If no errors occur, the system prompt is returned. Otherwise, the `pkgck` command reports the error.

Example 20–15 Checking the File Attributes of Installed Software Packages

The following example shows how to check the file attributes of a package.

```
# pkgchk -a SUNWbash
```

If no errors occur, the system prompt is returned. Otherwise, the `pkgck` command reports the error.

Example 20–16 Checking Software Packages Installed in a Spool Directory

The following example shows how to check a software package that was copied to a spool directory (`/export/install/packages`).

```
# pkgchk -d /export/install/packages
## checking spooled package <SUNWadmap>
## checking spooled package <SUNWadmfw>
## checking spooled package <SUNWadmc>
## checking spooled package <SUNWsadml>
```

The checks made on a spooled package are limited because not all information can be audited until a package is installed.

▼ How to Check the Integrity of Installed Objects (`pkgchk -p, pkgchk -P`)

This procedure explains how to use the `pkgchk` command to check the integrity of installed objects. The new `-P` option enables you to specify a partial path. This option has been added to assist you in mapping files to packages. Use this option with the `-l` option to list the information about the files that contain the partial path. Use the `-p` option to check the integrity of installed objects by specifying the full path. For more information, see the `pkgchk`(1M) man page.

1 **Become superuser or assume an equivalent role.**

Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2 **Check the integrity of an installed object.**

- To verify the integrity of an installed object for a full path name or path names, type the following:

  ```
  # pkgchk -lp path-name
  ```

- To verify the integrity of an installed object for a partial-path name or path names, type the following:

  ```
  # pkgchk -lP  partial-path-name
  ```

`-p` *path*	Checks the accuracy only of the path name or path names that are listed. Path can be one or more path names separated by commas. Specifies to audit only the file attributes (the permissions), rather than the file attributes and the contents, which is the default.
`-P` *partial-path*	Checks the accuracy of only the partial path name or path names that are listed. The partial-path can be one or more partial path names separated by commas. Matches any path name that contains the string contained in the partial path. Specifies to audit only the file contents, rather than the file contents and attributes, which is the default.
`-l`	Lists information about the selected files that make up a package. This option is not compatible with the `-a`, `-c`, `-f`, `-g`, and `-v` options. Specifies verbose mode, which displays file names as they are processed.

Example 20–17 Checking the Integrity of an Installed Object by Specifying a Full Path Name

This example shows you how to use the `pkgchk -lp` command to check the contents/attributes of an object on a file system by a specifying the full path name. The `-l` option lists information on the selected files that make up a package.

```
# pkgchk -lp /usr/sbin/pkgadd
Pathname: /usr/sbin/pkgadd
Type: regular file
Expected mode: 0555
Expected owner: root
Expected group: sys
Expected file size (bytes): 867152
Expected sum(1) of contents: 45580
Expected last modification: Jul 02 02:20:34 2004
Referenced by the following packages:
```

```
        SUNWpkgcmdsu
Current status: installed
```

Example 20–18 Checking the Integrity of an Installed Object by Specifying a Partial Path Name

This example shows you how to use the `pkgchk -lP` command to check the contents/attributes of an object on a file system by a specifying a partial path name, such as a file or directory name. The `-l` option lists information on the selected files that make up a package.

```
# pkgchk -lP /sbin/pkgadd
Pathname: /usr/sbin/pkgadd
Type: regular file
Expected mode: 0555
Expected owner: root
Expected group: sys
Expected file size (bytes): 867152
Expected sum(1) of contents: 45580
Expected last modification: Jul 02 02:20:34 2004
Referenced by the following packages:
        SUNWpkgcmdsu
Current status: installed

Pathname: /usr/sbin/pkgask
Type: linked file
Source of link: ../../usr/sbin/pkgadd
Referenced by the following packages:
        SUNWpkgcmdsu
Current status: installed
```

Removing Software Packages

To remove or uninstall a software package, use the associated tool that you used to add or install a software package. For example, if you used the Solaris installation GUI to install software, use the Solaris installation GUI to uninstall software.

Caution – Do no use the `rm` command to remove software packages. Doing so will result in inaccuracies in the database that keeps track of all installed packages on the system.

▼ How to Remove Software Packages (`pkgrm`)

1. **Become superuser or assume an equivalent role.**

   Roles contain authorizations and privileged commands. For more information about roles, see "Configuring RBAC (Task Map)" in *System Administration Guide: Security Services*.

2. **Remove an installed package.**

   ```
   # pkgrm pkgid ...
   ```

   pkgid identifies the name of one or more packages, separated by spaces, to be removed. If omitted, the `pkgrm`command removes all available packages.

Example 20–19 Removing Software Packages

This example shows how to remove a package.

```
# pkgrm SUNWctu

The following package is currently installed:
   SUNWctu         Netra ct usr/platform links (64-bit)
                   (sparc.sun4u) 11.9.0,REV=2001.07.24.15.53

Do you want to remove this package? y

## Removing installed package instance <SUNWctu>
## Verifying package dependencies.
## Processing package information.
## Removing pathnames in class <none>
.
.
.
```

Example 20–20 Removing a Spooled Software Package

This example shows how to remove a spooled package.

```
# pkgrm -s /export/pkg SUNWaudh
The following package is currently spooled:
   SUNWaudh        Audio Header Files
                   (sparc) 11.10.0,REV=2003.08.08.00.03
Do you want to remove this package? y
Removing spooled package instance <SUNWaudh>
```

◆ ◆ ◆ CHAPTER 21

Managing Solaris Patches by Using the `patchadd` Command (Tasks)

Patch management involves *applying* Solaris patches and software updates to a system. Patch management might also involve removing unwanted or faulty patches. Removing patches is also called *backing out* patches.

This chapter provides step-by-step instructions on how to manage Solaris patches by using the `patchadd` command. For additional information, see the `patchadd`(1M) man page.

The following overview information is in this chapter:

- "Types of Patches" on page 417
- "Accessing Solaris Patches" on page 418
- "Managing Patches in the Solaris Operating System" on page 420
- "Solaris Patch Management Terms and Definitions" on page 420
- "Managing Solaris Patches by Using the `patchadd` Command (Task Map)" on page 422

For information about applying patches to diskless client systems, see "Patching Diskless Client OS Services" on page 163.

For information about recommended strategies and practices for using Solaris patches, see *Solaris Patch Management: Recommended Strategies*.

Types of Patches

A *patch* is an accumulation of fixes for a known or potential problem within the Solaris OS or other supported software. A patch can also provide a new feature or an enhancement to a particular software release. A patch consists of files and directories that replace or update existing files and directories. Most Solaris patches are delivered as a set of sparse packages. For details about packages, see Chapter 18, "Managing Software (Overview)."

A software *update* is a change that you apply to software that corrects an existing problem or that introduces a feature. To update is also the process of applying software updates to a system.

You can manage patches on your Solaris system by using the `patchadd` command.

Signed and Unsigned Patches

A *signed patch* is one that has a *digital signature* applied to it. A patch that has its digital signature verified has not been modified since the signature was applied. The digital signature of a signed patch is verified after the patch is *downloaded* to your system.

Patches for the Solaris OS, starting with the Solaris 2.6 release, are available as signed patches and as *unsigned patches*. Unsigned patches do not have a digital signature.

Signed patches are stored in Java archive format (JAR) files and are available from the SunSolve Online^{SM} web site. Unsigned patches are stored in directory format and are also available from the SunSolve Online web site as `.zip` files.

For information about applying patches to your system by using the `patchadd` command, see "Managing Solaris Patches by Using the `patchadd` Command (Task Map)" on page 422.

For additional overview information about signed patches, see "Signed Packages, Patches, and Software Updates" on page 362.

Accessing Solaris Patches

Sun customers can access patches from the SunSolve Patch Portal web site. Although, some patches might only be accessible to customers with a service plan, such as a SunSpectrum^{SM} or a Solaris Service Plan customer. In *all* cases, you must be registered with Sun and have a Sun online ID to enter the SunSolve Patch Portal. These patches are updated nightly.

You can obtain Solaris patches from the `http://sunsolve.sun.com` web site. To access patches from the SunSolve Patch Portal web site, your system must be connected to the Internet and be capable of running a web browser, such as the Firefox browser.

You can access individual patches or a set of patches from a patch cluster, or refer to patch reports.

Each patch is associated with a README file that has information about the patch.

Solaris Patch Numbering

Patches are identified by unique *patch IDs*. A patch ID is an alphanumeric string that is a patch base code and a number that represents the patch revision number joined with a hyphen. For example, patch 118833-10 is the patch ID for the SunOS 5.10 kernel update patch, 10th revision.

Managing Solaris Patches

This section describes how to manage Solaris patches with the Solaris patch tools that are available.

The patch tools do the following:

- Determine the Solaris version number of the managing host and the target host
- Update the patch package's `pkginfo` file with this information:
  - Patches that have been *obsoleted* by the patch being applied
  - Other patches that are required by this patch
  - Patches that are *incompatible* with this patch

While you apply patches, the `patchadd` command logs information in the `/var/sadm/patch/`*patch-id*`/log` file.

Note – In this Solaris release, improvements have been made to the `patchadd -M` command. When you use this command to apply patches to your system, you are no longer required to specify patch IDs in numeric order. If you use the `patchadd -M` command without specifying a patch ID, all patches in the directory are installed on the system. For more information about these changes, see the `patchadd`(1M) man page.

The `patchadd` command cannot apply a patch or software update under the following conditions:

- The package is not fully installed on the system.
- The patch package's architecture differs from the system's architecture.
- The patch package's version does not match the installed package's version.
- A patch with the same base code and a higher revision number has already been applied.
- A patch that obsoletes this patch has already been applied.
- The patch is incompatible with a patch that has already been applied to the system. Each patch that has been applied keeps this information in its `pkginfo` file.
- The patch being applied depends on another patch that has not yet been applied.

Managing Patches in the Solaris Operating System

Use the following information to identify tasks for managing Solaris patches. Each task points to additional tasks, such as managing signed or unsigned patches.

Task	Description	For Instructions
Determine whether to apply signed or unsigned patches.	Determine whether applying signed or unsigned patches is best for your environment.	"Determining Whether to Apply Signed or Unsigned Patches to Your System" on page 420
Apply a patch to your system.	Use the `patchadd` command on Solaris 2.6, Solaris 7, Solaris 8, Solaris 9, Solaris 10 or Solaris Express systems to apply unsigned Solaris patches.	"Managing Solaris Patches by Using the `patchadd` Command (Task Map)" on page 422

Determining Whether to Apply Signed or Unsigned Patches to Your System

The key factor when determining whether to apply signed or unsigned patches to your system is whether you trust the source of patches.

If you trust the source of patches, for example, a patch CD from a known distributor or an `HTTPS` connection to a trusted web site, you can use unsigned patches. However, if you do not trust the source, use signed patches.

If you are unsure about whether to trust the source of patches, use signed patches.

Solaris Patch Management Terms and Definitions

The following terms are used throughout the patch management chapters.

apply	To install a patch on a system.
back out	To remove a patch from a system.
backout data	Data that is created when a patch is applied to enable the system to return to its previous state if the patch is removed (backed out).
backout directory	Directory in which backout data is stored. By default, this is the `save` directory of each package that was installed by the patch.

dependency	See **patch dependency**.
digital signature	An electronic signature that can be used to ensure that a document has not been modified since the signature was applied.
download	To copy one or more patches from a source of patches, such as the Sun patch server, to the system where the patches are to be applied.
download directory	Directory in which patches are stored when they are downloaded from the patch source. This is also the directory from which patches are applied. The default location is `/var/sadm/spool`.
keystore	A repository of certificates and keys that is queried when you attempt to apply a signed patch.
nonstandard patch	Nonstandard patches cannot be installed using the `patchadd` command. Nonstandard patches, those that are typically used to deliver firmware or software application fixes that are not delivered in package format, must be installed by using the instructions that are specified in the patch README file.
order	To sort a set of patches in an order suitable for applying patches.
package	The form in which software products are delivered for installation on a system. The package contains a collection of files and directories in a defined format.
patch	An update to software that corrects an existing problem or that introduces a feature.
patch analysis	A method of checking a system to determine which patches are appropriate for the system.
patch dependency	An instance where a patch depends on the existence of another patch on a system. A patch that depends on one or more patches can only be applied to a system when those other patches have already been applied.
patch ID	A unique alphanumeric string, with the patch base code first, a hyphen, and a number that represents the patch revision number.
patch incompatibility	A rare situation where two patches cannot be on the same system. Each patch in the relationship is incompatible with the other. If you want to apply a patch that is incompatible with a patch already on the system, you must first remove the patch that is already on the system. Then, you can apply the new patch.
patch list	A file that contains a list of patches, one patch ID per line. Such a list can be used to perform patch operations. The list can be generated based on the analysis of a system or on user input. Each line in a patch list has two columns. The first column is the patch ID, and the second column is a synopsis of that patch.
patch obsolescence	An instance where a patch replaces another patch, even if it has not already been applied to a system. A patch that obsoletes one or more patches replaces those patches entirely and does not require that the obsolete patches be applied before the replacement patch is applied.
patch server	A source of Solaris patches that can be used by your systems to perform patch analyses and from which to obtain the appropriate patches.
signed patch	A patch that is signed with a valid digital signature. A signed patch offers greater security than an unsigned patch. The digital signature of the patch can be verified before the patch is applied to your system. A valid digital signature ensures that the signed patch has not been modified since the signature was applied. Signed patches are stored in Java Archive (JAR) format files.

software update	A change to software that you apply that corrects an existing problem or that introduces a feature.
special handling	Patches with properties that indicate they must be installed in single-user mode. Also, patches that require you to restart the system after the patch has been applied are referred to as having *special handling requirements*.
standard patch	Standard patches are those that adhere to the Solaris patch specification and are installable by using the `patchadd` command. Note that nonstandard patches cannot be installed by using the `patchadd` command
Sun Alert	A notification to customers of a known product issue that might negatively impact customers' computing environments or productivity. A problem that warrants a Sun Alert notification meets the criteria for issues that are related to at least one of these concerns: availability, security, and data loss.
SunSolve Online	The Sun Microsystems patch portal web site that provides access to patch, patch information, and patch clusters. See `http://sunsolve.sun.com` for more information.
unsigned patch	A patch that is not signed with a digital signature.
web proxy	A system that is used to connect your system to the Internet. Your system cannot connect directly to the Internet, but must use the web proxy to establish the connection.

Managing Solaris Patches by Using the `patchadd` Command (Task Map)

Task	Description	For Instructions
1. (Optional) Set up the package keystore.	If you plan to apply signed patches to your system, you must first import Sun's Root CA certificate into your package keystore.	"How to Import a Trusted Certificate to Your Package Keystore" on page 423
2. (Optional) Specify a web proxy.	If your system is behind a firewall with a web proxy, you must specify the web proxy to obtain patches from the Sun patch server.	"How to Specify a Web Proxy" on page 425
3. Download and apply a patch.	You can download and apply a patch to your system by using the `patchadd` command.	"How to Download and Apply a Solaris Patch" on page 426
4. (Optional) Display information about patches that have been applied to your system.	If you want information about the patches that have already been applied to your system, use the `patchadd`, `showrev`, or `pkgparam` command.	"How to Display Information About Solaris Patches" on page 428
5. (Optional) Remove a patch from your system.	If necessary, remove a patch from your system by using the `patchrm` command.	"How to Remove a Solaris Patch by Using the `patchrm` Command" on page 428

▼ How to Import a Trusted Certificate to Your Package Keystore

To apply *signed patches* to your system by using the `patchadd` command, you must add Sun's Root CA certificate, at the very least, to verify the signature of your signed patch. You can import this certificate from the Java *keystore* to the package keystore.

1. **Become superuser or assume an equivalent role.**

2. **If you are using the `patchadd` command to install signed patches, add the new trusted Verisign certificate to the keystore.**

   a. **Download the Class 2 Public Primary Certification Authority - G2 trusted Verisign certificate from `http://www.sun.com/pki/certs/ca/`.**

      The Subject Name of this certificate is:

      ```
      C=US, O=VeriSign, Inc., OU=Class 2 Public Primary Certification
      Authority - G2, OU=(c) 1998 VeriSign, Inc. - For authorized use only,
      OU=VeriSign Trust Network
      ```

   b. **Select the binary format (DER encoded).**

   c. **Copy the certificate to the file, `/tmp/root.crt`.**

Note – In the event you are unable to download the trusted Verisign certificate, see "Exporting the Root CA Certificate From the Java Keystore" on page 424 for alternate instructions.

---

3. **Import the Root CA certificate from the temporary file to the package keystore.**

   Unless changed by the system administrator, the default Java keystore password is `changeit`.

   For example:

```
# pkgadm addcert -t -f der /tmp/root.crt
     Keystore Alias: /C=US/O=VeriSign, Inc./OU=Class 2 Public Primary Certification Authority - G2/O
        Common Name: /C=US/O=VeriSign, Inc./OU=Class 2 Public Primary Certification Authority - G2/O
   Certificate Type: Trusted Certificate
Issuer Common Name: /C=US/O=VeriSign, Inc./OU=Class 2 Public Primary Certification Authority - G2/O
     Validity Dates: <May 18 00:00:00 1998 GMT> - <Aug  1 23:59:59 2028 GMT>
MD5 Fingerprint: 2D:BB:E5:25:D3:D1:65:82:3A:B7:0E:FA:E6:EB:E2:E1
   SHA1 Fingerprint: B3:EA:C4:47:76:C9:C8:1C:EA:F2:9D:95:B6:CC:A0:08:1B:67:EC:9D

Are you sure you want to trust this certificate? yes
Trusting certificate </C=US/O=VeriSign, Inc./OU=Class 2 Public Primary Certification Authority - G2/O>
Type a Keystore protection Password. changeit
Press ENTER for no protection password (not recommended):
For Verification: Type a Keystore protection Password.
```

```
Press ENTER for no protection password (not recommended):
Certificate(s) from </tmp/root.crt> are now trusted
```

`-t`	Indicates that the certificate is a trusted CA certificate. The command output includes the certificate details, which you are asked to verify.
`-f` *format*	Specifies the format of the certificate or private key. When importing a certificate, it must be encoded using either the PEM (`pem`) or binary DER (`der`) format.
certfile	Specifies the file that contains the certificate.

4 **Display the certificate information.**

```
# pkgadm listcert
Enter Keystore Password: storepass
     Keystore Alias: /C=US/O=VeriSign, Inc./OU=Class 2 Public Primary Certification Authority - G2/O
        Common Name: /C=US/O=VeriSign, Inc./OU=Class 2 Public Primary Certification Authority - G2/O
   Certificate Type: Trusted Certificate
Issuer Common Name: /C=US/O=VeriSign, Inc./OU=Class 2 Public Primary Certification Authority - G2/O
Validity Dates: <May 18 00:00:00 1998 GMT> - <Aug 1 23:59:59 2028 GMT>
 MD5 Fingerprint: 2D:BB:E5:25:D3:D1:65:82:3A:B7:0E:FA:E6:EB:E2:E1
   SHA1 Fingerprint: B3:EA:C4:47:76:C9:C8:1C:EA:F2:9D:95:B6:CC:A0:08:1B:67:EC:9D
```

5 **Remove the temporary file.**

```
# rm /tmp/root.crt
```

Exporting the Root CA Certificate From the Java Keystore

If you are unable to download the trusted Verisign certificate from `http://www.sun.com/pki/certs/ca/`, as described in Step 2 of "How to Import a Trusted Certificate to Your Package Keystore" on page 423, you can export the Root CA certificate from the Java keystore to a temporary file.

For example:

```
# keytool -export -storepass changeit -alias verisignclass2g2ca \
-keystore /usr/java/jre/lib/security/cacerts -file /tmp/root.crt
Certificate stored in file </tmp/root.crt>
```

`-export`	Exports the trusted certificate.
`-storepass` *storepass*	Specifies the password that protects the integrity of the Java keystore.
`-alias verisignclass2g2ca`	Identifies the alias of the trusted certificate.

`-keystore` *certfile*	Specifies the name and location of the keystore file.
`-file` *filename*	Identifies the file in which to hold the exported certificate.

You are now ready to import the Root CA certificate from the temporary file to the package keystore. See the remaining steps in the section, "How to Import a Trusted Certificate to Your Package Keystore" on page 423, for instructions.

▼ How to Specify a Web Proxy

If your system is behind a firewall with a web proxy, you must specify the web proxy to use `patchadd` to *apply* a patch.

1 **Become superuser or assume an equivalent role.**

2 **Use one of the following methods to specify a web proxy:**

- Specify the web proxy by using the `http_proxy`, `HTTPPROXY`, or `HTTPPROXYPORT` environment variable.

  For example:

  ```
  # setenv http_proxy http://mycache.domain:8080
  ```

  Or, specify one of the following:

  ```
  # setenv HTTPPROXY mycache.domain
  # setenv HTTPPROXYPORT 8080
  ```

- Specify the web proxy on the `patchadd` command line.

  For example:

  ```
  # patchadd -x mycache.domain:8080 \
  -M http://www.sun.com/solaris/patches/latest 101223-02 102323-02
  ```

Restrictions on Using `patchadd -R` to Create an Alternate `root` Path

On systems that are running a Solaris release that is not zones aware, using the `patchadd` command, or any command that accepts the `-R` option to specify an alternate `root` path for a global zone that has non-global zones installed, does not work.

You can use of the `-R` option to add and remove software packages and patches, if the alternate boot environment has configured non-global zones, but no installed non-global zones.

To avoid a potential problem, restrict the use of the `-R` option for the creation of an alternate root path.

If you are running this Solaris release, you can alternately choose one of the following methods:

- Upgrade any systems that are not running the current Solaris release.
- Boot an alternate root as the active OS.

If you are running the Solaris 10 OS, you can alternately choose one of the following methods:

- Upgrade any systems that are not running at least the Solaris 10 1/06 OS to the Solaris 10 1/06 release.
- If you are running the Solaris 10 initial 3/05 release, you can install the following patch to enable the use of commands that accept the `-R` option for creation of an alternate root path.
  - **For SPARC based systems** – Install at least revision 19 of patch 119254.
  - **For x86 based systems** – Install at least revision 19 patch 119255.
- Boot an alternate root, for example the Solaris 10 release, as the active OS. You can then install and uninstall packages and patches without using the `-R` option.

For more information, see the `patchadd`(1M), `patchrm`(1M), `pkgadd`(1M), and `pkgrm`(1M) man pages.

▼ How to Download and Apply a Solaris Patch

Use this procedure to *download* either a signed or an *unsigned Solaris patch* and then apply it to your system.

If you want to apply signed patches, you must first set up the package keystore.

1. **Gain access to the system in one of the following ways:**

   - **Log in to the system where you want to apply the patch.**

   - **Download the patch and use the `ftp` command to copy the patch to the target system.**

2. **Start a web browser and go to the SunSolve Online Patch Portal at `http://sunsolve.Sun.COM`.**

3. **Determine whether to download a specific patch or a patch cluster, then do one of the following:**

   - **Type the patch number (*patch-id*) in the Find Patch search field, then click Find Patch.**

     Entering *patch-id* downloads the latest patch revision.

If this patch is freely available, the patch README appears. If this patch is not freely available, an `ACCESS DENIED` message appears.

Note that patch numbers for SPARC based and x86 based systems are different. The *patch IDs* are listed in the patch README. Ensure that you apply the patch that matches your system architecture.

- **Select the Recommended Patch Cluster that matches the Solaris release that is running on the system that you want to patch.**

4 **Download the patch by following these instructions:**

- **To download a copy of the signed patch, click the Download Signed Patch (*n* bytes) button.**
- **To download an unsigned patch, click the Download Patch (*n* bytes) button.**

When the patch or patches are successfully downloaded, close the web browser.

5 **Change to the directory that contains the downloaded patch.**

6 **Become superuser or assume an equivalent role.**

7 (*Unsigned patch*) **If you downloaded an unsigned patch, unzip the patch.**

```
# unzip patch-id
```

8 **Apply the signed or unsigned patch.**

- If you downloaded a signed patch, apply it.

  For example:

  ```
  # patchadd /tmp/111879-01.jar
  ```
- If you downloaded an unsigned patch, apply it.

  For example:

  ```
  # patchadd /tmp/111879-01
  ```

9 **Verify that the patch has been successfully applied.**

For example:

```
# patchadd -p | grep 111879
Patch: 111879-01 Obsoletes:  Requires:  Incompatibles:  Packages: SUNWwsr
```

▼ How to Display Information About Solaris Patches

Before applying patches, you might want to know more about patches that have been previously applied.

The following commands provide useful information about patches that are already applied to a system.

- `patchadd -p` or `showrev -p`

  Shows all patches that have been applied to the system.
- `pkgparam` *pkgid* `PATCHLIST`

  Shows all patches that have been applied to the package identified by *pkgid*, for example, `SUNWadmap`.
- `patchadd -S` *Solaris-OS* `-p`

  Shows all the `/usr` patches that have been applied to an OS server.

● **Use one of the following `patchadd` command lines to display information about patches that have been applied to your system.**

- To obtain information about all patches that have been applied to your system, type:

  ```
  $ patchadd -p
  ```
- To verify whether a particular patch has been applied to your system, type, for example:

  ```
  $ patchadd -p | grep 111879
  ```

▼ How to Remove a Solaris Patch by Using the `patchrm` Command

1 **Become superuser.**

2 **Remove the patch.**

```
# patchrm 111879-01
Checking installed patches...

Backing out patch 111879-01...

Patch 111879-01 has been backed out.
```

3 **Verify that the patch was removed.**

```
# patchadd -p | grep 111879
#
```

APPENDIX A

SMF Services

The following table lists some of the services that have been converted to use SMF. Each service includes the daemon or service name, the FMRIs for that service, the run script that is used to start the service, and whether the service is started by `inetd`.

TABLE A–1 SMF Services

Service Name	FMRI	Run Script	`inetd` Service
`automount`	`svc:/system/filesystem/autofs:default`	`autofs`	Not applicable
`consadmd`	`svc:/system/consadm:default`	`rootusr`	Not applicable
`coreadm`	`svc:/system/coreadm:default`	`coreadm`	Not applicable
`cron`	`svc:/system/cron:default`	`cron`	Not applicable
`cryptoadm`	`svc:/system/cryptosvc:default`	N/A	Not applicable
`cvcd`	`svc:/system/cvc:default`	`cvcd`	Not applicable
`dcs`	`svc:/platform/<arch>/dcs:default`	None	Applicable
`dtlogin`	`svc:/application/graphical-login/cde-login:default`	`dtlogin`	Not applicable
`dtprintinfo`	`svc:/application/cde-printinfo:default`	`dtlogin`	Not applicable
`dtspcd`	`svc:/network/cde-spc:default`	None	Applicable
`dumpadm`	`svc:/system/dumpadm:default`	`savecore`	Not applicable
`efdaemon`	`svc:/platform/<arch>/efdaemon:default`	`efcode`	Not applicable
`fmd`	`svc:/system/fmd:default`	N/A	Not applicable
`gssd`	`svc:/network/rpc/gss:default`	None	Applicable

TABLE A–1 SMF Services *(Continued)*

Service Name	FMRI	Run Script	`inetd` Service
`imapd`	`svc:/network/imap/tcp:default`	None	Applicable
	`svc:/network/imapnew/tcp:default`		
`in.chargend`	`svc:/network/chargen:dgram`	None	Applicable
	`svc:/network/chargen:stream`		
`in.comsat`	`svc:/network/comsat:default`	None	Applicable
`in.daytimed`	`svc:/network/daytime:dgram`	None	Applicable
	`svc:/network/daytime:stream`		
`in.dhcpd`	`svc:/network/dhcp-server:default`	`dhcp`	Not applicable
`in.discardd`	`svc:/network/discard:dgram`	None	Applicable
	`svc:/network/discard:stream`		
`in.echod`	`svc:/network/echo:dgram`	None	Applicable
	`svc:/network/echo:stream`		
`in.fingerd`	`svc:/network/finger:default`	None	Applicable
`in.ftpd`	`svc:/network/ftp:default`	None	Applicable
`in.named`	`svc:/network/dns/server:default`	`inetsvc`	Not applicable
`in.rarpd`	`svc:/network/rarp:default`	`boot.server`	Not applicable
`in.rdisc`	`svc:/network/initial:default`	`inetinit`	Not applicable
`in.rexecd`	`svc:/network/rexec:default`	None	Applicable
`in.rlogind`	`svc:/network/login:rlogin`	None	Applicable
	`svc:/network/login:eklogin`		
	`svc:/network/login:klogin`		
`in.routed`	`svc:/network/initial:default`	`inetinit`	Not applicable
`in.rshd`	`svc:/network/shell:default`	None	Applicable
	`svc:/network/kshell`		
`in.talkd`	`svc:/network/talk:default`	None	Applicable
`in.telnetd`	`svc:/network/telnet:default`	None	Applicable
`in.tftpd`	`svc:/network/tftp/udp6:default`	None	Applicable

TABLE A–1 SMF Services *(Continued)*

Service Name	FMRI	Run Script	inetd Service
in.timed	svc:/network/time:dgram svc:/network/time:stream	None	Applicable
in.tnamed	svc:/network/tname:default	None	Applicable
in.uucpd	svc:/network/uucp:default	None	Applicable
inetd-upgrade	svc:/network/inetd-upgrade:default	N/A	Not applicable
inetd	svc:/network/inetd:default	inetsvc	Not applicable
intrd	svc:/system/intrd:default	None	Not applicable
ipop3d	svc:/network/pop3/tcp:default	None	Applicable
kadmind	svc:/network/security/kadmin:default	kdc.master	Not applicable
kbd	svc:/system/keymap:default	keymap	Not applicable
keyserv	svc:/network/rpc/keyserv:default	rpc	Not applicable
kpropd	svc:/network/security/krb5_prop:default	None	Applicable
krb5kdc	svc:/network/security/krb5kdc:default	kdc	Not applicable
ktkt_warnd	svc:/network/security/ktkt_warn:default	None	Applicable
ldap_cachemgr	svc:/network/ldap/client:default	ldap.client	Not applicable
loadkeys	svc:/system/keymap:default	keymap	Not applicable
lockd	svc:/network/nfs/client:default svc:/network/nfs/server:default	nfs.server	Not applicable
lpsched and lpshut	svc:/application/print/server:default	lp	Not applicable
mdmonitord	svc:/system/mdmonitor:default	svm.sync	Not applicable
metainit	svc:/system/metainit:default	svm.init	Not applicable
metadevadm	svc:/platform/<arch>/mpxio-upgrade:default	N/A	Not applicable
mount	svc:/system/filesystem/local:default svc:/system/filesystem/minimal:default svc:/system/filesystem/root:default svc:/system/filesystem/usr:default	nfs.client, rootusr, standardmounts	Not applicable
mountd	svc:/network/nfs/server:default	nfs.server	Not applicable
nfsd	svc:/network/nfs/server:default	nfs.server	Not applicable

TABLE A–1 SMF Services *(Continued)*

Service Name	FMRI	Run Script	inetd Service
nfsmapid	svc:/network/nfs/client:default svc:/network/nfs/server:default	nfs.server	Not applicable
nis_cachemgr	svc:/network/rpc/nisplus:default	rpc	Not applicable
nscd	svc:/system/name-service-cache:default	nscd	Not applicable
ntpdate	svc:/network/ntp:default	xntpd	Not applicable
ocfserv	svc:/network/rpc/ocfserv:default	ocfserv	Not applicable
picld	svc:/system/picl:default	picld	Not applicable
pmconfig	svc:/system/power:default	power	Not applicable
printd	svc:/application/print/cleanup:default	spc	Not applicable
quotaon	svc:/system/filesystem/local:default	ufs_quota	Not applicable
rcapd	svc:/system/rcap:default	rcapd	Not applicable
rpcbind	svc:/network/rpc/bind:default	rpc	Not applicable
rpc.bootparamd	svc:/network/rpc/bootparams:default	boot.server	Not applicable
rpc.mdcomm	svc:/network/rpc/mdcomm:default	None	Applicable
rpc.metad	svc:/network/rpc/meta:default	None	Applicable
rpc.metamedd	svc:/network/rpc/metamed:default	None	Applicable
rpc.metamhd	svc:/network/rpc/metamh:default	None	Applicable
rpc.nisd	svc:/network/rpc/nisplus:default	rpc	Not applicable
rpc.nispasswdd	svc:/network/rpc/nisplus:default	rpc	Not applicable
rpc.rexd	svc:/network/rpc/rex:default	None	Applicable
rpc.rstatd	svc:/network/rpc/rstat:default	None	Applicable
rpc.rusersd	svc:/network/rpc/rusers:default	None	Applicable
rpc.smserverd	svc:/network/rpc/smserver:default	None	Applicable
rpc.sprayd	svc:/network/rpc/spray:default	None	Applicable
rpc.ttdbserverd	svc:/network/rpc/ttdbserver:tcp	None	Applicable
rpc.walld	svc:/network/rpc/wall:default	None	Applicable
rpc.yppasswdd and rpc.ypupdated	svc:/network/nis/server:default	rpc	Not applicable

TABLE A–1 SMF Services *(Continued)*

Service Name	FMRI	Run Script	inetd Service
rquotad	svc:/network/nfs/rquota:default	None	Applicable
sadc	svc:/system/sar:default	perf	Not applicable
savecore	svc:/system/dumpadm:default	savecore	Not applicable
sendmail	svc:/network/smtp:sendmail	sendmail	Not applicable
sf880drd	svc:/platform/<arch>/sf880drd:default	sf880dr	Not applicable
slpd	svc:/network/slp:default	slpd	Not applicable
sshd	svc:/network/ssh:default	sshd	Not applicable
statd	svc:/network/nfs/client:default svc:/network/nfs/server:default	nfs.server	Not applicable
svc.startd	svc:/system/svc/restarter:default	N/A	Not applicable
syseventd	svc:/system/sysevent:default	devfsadm	Not applicable
sysidpm, sysidns, sysidroot, sysidsys	svc:/system/sysidtool:system	sysid.sys	Not applicable
sysidnet	svc:/system/sysidtool:net	sysid.net	Not applicable
syslogd	svc:/system/system-log:default	syslog	Not applicable
ttymon	svc:/system/console-login:default	inittab	Not applicable
utmpd	svc:/system/utmp:default	utmpd	Not applicable
vold	svc:/system/filesystem/volfs:default	volmgt	Not applicable
xntpd	svc:/network/ntp:default	xntpd	Not applicable
ypbind	svc:/network/nis/client:default	rpc	Not applicable
ypserv	svc:/network/nis/server:default	rpc	Not applicable
ypxfrd	svc:/network/nis/server:default	rpc	Not applicable
zoneadm	svc:/system/zones:default	N/A	Not applicable
None	svc:/network/loopback:default	network	Not applicable
None	svc:/network/physical:default	network	Not applicable

Index