Booting and Shutting Down Oracle® Solaris on SPARC Platforms

Part No: 821–2731–11
March 2012

120305@25097

Contents

Preface 7

1 **Booting and Shutting Down a SPARC Based System (Overview)** 11

What's New in Booting and Shutting Down a System 12

Administratively Provided `driver.conf` Files 12

Fast Reboot on SPARC Platforms 12

Booting and Shutting Down a SPARC Based System (Topic Map) 13

Guidelines for Booting a System 14

Reasons to Boot a System 14

Service Management Facility and Booting 15

Changes in Behavior When Using SMF 15

How Run Levels Work 16

What Happens When a System Is Booted to a Multiuser State (Run Level 3) 17

When to Use Run Levels or Milestones 18

Overview of the Oracle Solaris Boot Architecture 18

Description of the SPARC Boot Process 19

SPARC Boot Phases 19

2 **Booting a SPARC Based System to a Specified State (Tasks)** 21

Booting a SPARC Based System to a Specified State (Task Map) 21

Booting a SPARC Based System to a Specified State 22

Determining a System's Current Run Level 22

Booting a SPARC Based System to a Multiuser State (Run Level 3) 23

Booting a SPARC Based System to a Single-User State (Run Level S) 24

Booting a SPARC Based System Interactively 25

3 Shutting Down a System (Tasks) 27
Shutting Down a System (Task Map) 27
Overview of Shutting Down a System 28
Guidelines for Shutting Down a System 28
System Shutdown Commands 29
Shutting Down a System 30
▼ How to Determine Who Is Logged in to the System 30
▼ How to Shut Down a System by Using the `shutdown` Command 30
▼ How to Shut Down a System by Using the `init` Command 33
Turning Off Power to System Devices 34

4 Rebooting a SPARC Based System (Tasks) 35
Rebooting a SPARC Based System (Task Map) 35
Rebooting a SPARC Based System 36
▼ How to Reboot a System by Using the `init` Command 36
▼ How to Reboot a System by Using the `reboot` Command 37
Accelerating the Reboot Process on a SPARC Based System 38
▼ How to Initiate a Fast Reboot of a SPARC Based System 38
Changing the Default Behavior of the Fast Reboot Feature 39
Initiating a Standard Reboot of a System That Has Fast Reboot Enabled 39

5 Booting a SPARC Based System From the Network (Tasks) 41
Booting a SPARC Based System From the Network (Task Map) 41
Booting a SPARC Based System From the Network 42
SPARC Network Boot Processes 42
Requirements for Booting a SPARC Based System From the Network 42
Setting Network Boot Arguments in the OpenBoot PROM 43
Setting Up an NVRAM Alias to Automatically Boot by Using DHCP 45
▼ How to Boot a SPARC Based System From the Network 45

6 Modifying Boot Parameters on a SPARC Based System (Tasks) 47
Modifying Boot Parameters on a SPARC Based System (Task Map) 47
Modifying Boot Parameters on a SPARC Based System 48
▼ How to Identify the PROM Revision Number for a System 49

▼ How to Identify Devices on a System 49
▼ How to Determine the Default Boot Device 51
▼ How to Change the Default Boot Device by Using the Boot PROM 51
▼ How to Change the Default Boot Device by Using the `eeprom` Utility 53
▼ How to Change the Default Boot File by Using the Boot PROM 53
▼ How to Change the Default Boot File by Using the `eeprom` Utility 54

7 Creating, Administering, and Booting From ZFS Boot Environments on SPARC Platforms (Tasks) 55
Creating, Administering, and Booting From ZFS Boot Environments (Task Map) 55
Creating and Administering Boot Environments 57
▼ How to Create a New Boot Environment 57
▼ How to Create a Snapshot of a Boot Environment 59
▼ How to Create a Boot Environment From an Existing Snapshot 59
▼ How to Activate a Newly Created Boot Environment 59
▼ How to Display a List of Available Boot Environments, Snapshots, and Datasets 60
▼ How to Destroy a Boot Environment 61
Booting From a ZFS Boot Environment on SPARC Platforms 62
▼ SPARC: How to Display a List of Available Boot Environments During the Boot Sequence 62
▼ How to Boot From a ZFS Boot Environment or Root File System 63

8 Keeping a SPARC Based System Bootable (Tasks) 67
Keeping a SPARC Based System Bootable (Task Map) 67
Description of the Oracle Solaris Boot Archives 68
Obtaining Information About the Location and Contents of the SPARC Boot Archive 68
Managing the Boot Archive SMF Service 70
Determining Whether the `boot-archive` Service Is Running 70
▼ How to Enable or Disable the `boot-archive` SMF Service 70
Maintaining the Integrity of the Boot Archives 71
▼ How to Clear a Failed Automatic Boot Archive Update by Manually Updating the Boot Archive 71

9 Troubleshooting Booting a SPARC Based System (Tasks) 73
Troubleshooting Booting a SPARC Based System (Task Map) 73

Shutting Down and Booting a SPARC Based System for Recovery Purposes 74
Stopping and Booting for System Recovery Purposes 74
Forcing a Crash Dump and Reboot of a SPARC Based System 79
▼ How to Boot a System With the Kernel Debugger (`kmdb`) Enabled 80

Index 83

Preface

Booting and Shutting Down Oracle Solaris on SPARC Platforms is part of a documentation set that provides a significant portion of the Oracle Solaris system administration information. This guide contains information for SPARC platforms.

This book assumes you have completed the following tasks:

- Installed Oracle Solaris 11
- Set up all the networking software that you plan to use

Note – This Oracle Solaris release supports systems that use the SPARC and x86 families of processor architectures. The supported systems appear in the *Oracle Solaris OS: Hardware Compatibility Lists*. This document cites any implementation differences between the platform types.

For supported systems, see the *Oracle 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 Oracle Solaris 11 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
Booting and Shutting Down Oracle Solaris on SPARC Platforms	Booting and shutting down a system, managing boot services, modifying boot behavior, booting from ZFS, managing the boot archive, and troubleshooting booting on SPARC platforms

Book Title	Topics
Booting and Shutting Down Oracle Solaris on x86 Platforms	Booting and shutting down a system, managing boot services, modifying boot behavior, booting from ZFS, managing the boot archive, and troubleshooting booting on x86 platforms
Oracle Solaris Administration: Common Tasks	Using Oracle Solaris commands, booting and shutting down a system, managing user accounts and groups, managing services, hardware faults, system information, system resources, and system performance, managing software, printing, the console and terminals, and troubleshooting system and software problems
Oracle Solaris Administration: Devices and File Systems	Removable media, disks and devices, file systems, and backing up and restoring data
Oracle Solaris Administration: IP Services	TCP/IP network administration, IPv4 and IPv6 address administration, DHCP, IPsec, IKE, IP Filter, and IPQoS
Oracle Solaris Administration: Naming and Directory Services	DNS, NIS, and LDAP naming and directory services, including transitioning from NIS to LDAP
Oracle Solaris Administration: Network Interfaces and Network Virtualization	Automatic and manual IP interface configuration including WiFi wireless; administration of bridges, VLANs, aggregations, LLDP, and IPMP; virtual NICs and resource management.
Oracle Solaris Administration: Network Services	Web cache servers, time-related services, network file systems (NFS and autofs), mail, SLP, and PPP
Oracle Solaris Administration: Oracle Solaris Zones, Oracle Solaris 10 Zones, and Resource Management	Resource management features, which enable you to control how applications use available system resources; Oracle Solaris Zones software partitioning technology, which virtualizes operating system services to create an isolated environment for running applications; and Oracle Solaris 10 Zones, which host Oracle Solaris 10 environments running on the Oracle Solaris 11 kernel
Oracle Solaris Administration: Security Services	Auditing, device management, file security, BART, Kerberos services, PAM, Cryptographic Framework, Key Management Framework, privileges, RBAC, SASL, Secure Shell and virus scanning.
Oracle Solaris Administration: SMB and Windows Interoperability	SMB service, which enables you to configure an Oracle Solaris system to make SMB shares available to SMB clients; SMB client, which enables you to access SMB shares; and native identity mapping service, which enables you to map user and group identities between Oracle Solaris systems and Windows systems
Oracle Solaris Administration: ZFS File Systems	ZFS storage pool and file system creation and management, snapshots, clones, backups, using access control lists (ACLs) to protect ZFS files, using ZFS on an Oracle Solaris system with zones installed, emulated volumes, and troubleshooting and data recovery

Book Title	Topics
Trusted Extensions Configuration and Administration	System installation, configuration, and administration that is specific to Trusted Extensions
Oracle Solaris 11 Security Guidelines	Securing an Oracle Solaris system, as well as usage scenarios for its security features, such as zones, ZFS, and Trusted Extensions
Transitioning From Oracle Solaris 10 to Oracle Solaris 11	Provides system administration information and examples for transitioning from Oracle Solaris 10 to Oracle Solaris 11 in the areas of installation, device, disk, and file system management, software management, networking, system management, security, virtualization, desktop features, user account management, and user environments

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` or visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` if you are hearing impaired.

Typographic Conventions

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

TABLE P–1 Typographic Conventions

Typeface	Description	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 shells that are included in the Oracle Solaris OS. Note that the default system prompt that is displayed in command examples varies, depending on the Oracle Solaris release.

TABLE P–2 Shell Prompts

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

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 `/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.

CHAPTER 1

Booting and Shutting Down a SPARC Based System (Overview)

Oracle Solaris is designed to run continuously so that enterprise services, such as databases and web services, remain available as much as possible. This chapter provides guidelines for shutting down and booting a SPARC based system.

Note – This guide focuses primarily on booting and shutting down a single Oracle Solaris instance on servers and workstations. Information about booting and shutting down Oracle Solaris on systems that have service processors and systems that have multiple physical domains is not covered in detail in this document. For more information, see the product documentation for your specific hardware at `http://www.oracle.com/technetwork/indexes/documentation/index.html`.

The following is a list of the information that is in this chapter:

- "What's New in Booting and Shutting Down a System" on page 12
- "Booting and Shutting Down a SPARC Based System (Topic Map)" on page 13
- "Guidelines for Booting a System" on page 14
- "Service Management Facility and Booting" on page 15
- "How Run Levels Work" on page 16
- "Overview of the Oracle Solaris Boot Architecture" on page 18

For information about booting an x86 based system, see *Booting and Shutting Down Oracle Solaris on x86 Platforms*.

What's New in Booting and Shutting Down a System

The following boot features are new the Oracle Solaris 11 release:

Administratively Provided driver.conf Files

Driver configuration files (`driver.conf`) can be supplemented with local, administrative changes without modifying the original vendor provided files in the `/kernel` and `/platform` directories. This enhancement provides better preservation of local configuration during a system upgrade. You can now provide local changes to driver configuration by adding `driver.conf` files to the new `/etc/driver/drv` directory. At boot time, the system checks for a configuration file in `/etc/driver/drv` for that driver. If found, the system automatically merges the vendor-provided configuration with the administratively provided changes.

To display these merged properties, use the `prtconf` command with the new `-u` option. The `-u` option enables you to display both the original and updated property values for a specified driver. For more information, see the `prtconf`(1M) man page. For instructions, see "How to Display Default and Customized Property Values for a Device" in *Oracle Solaris Administration: Common Tasks*.

Note – Do not edit vendor-provided `driver.conf` files in the `/kernel` and `/platform` directories. If you need to supplement a driver's configuration, the preferred method is to add a corresponding `driver.conf` file to the local `/etc/driver/drv` directory, and then customize that file. For instructions, see Chapter 5, "Managing Devices (Overview/Tasks)," in *Oracle Solaris Administration: Devices and File Systems*.

See also the following additional references:

- `driver.conf`(4)
- `driver`(4)
- *Writing Device Drivers*
- `ddi_prop_exists`(9F)
- `ddi_prop_lookup`(9F)

Fast Reboot on SPARC Platforms

The integration of Fast Reboot on the SPARC platform enables the `-f` option to be used with the `reboot` command to accelerate the boot process by skipping certain POST tests.

The Fast Reboot feature is managed by the Service Management Facility (SMF) feature of Oracle Solaris and implemented through a boot configuration service, `svc:/system/boot-config`. The `boot-config` service provides a means for setting or changing the default boot configuration parameters. When the `config/fastreboot_default` property is set to `true`, the system performs a fast reboot automatically, without the need to use the `reboot -f` command. This property's value is set to `false` on the SPARC platform. For task-related information, see "Accelerating the Reboot Process on a SPARC Based System" on page 38.

Note – Fast reboot behavior on SPARC is applicable only to certain systems. On `sun4v` systems, fast reboot is unnecessary because the reboot is actually a hypervisor restart that does not involve POST.

Booting and Shutting Down a SPARC Based System (Topic Map)

Use the following references to find step-by-step instructions on various boot-related topics within this document.

TABLE 1–1 Booting and Shutting Down a SPARC Based System: Topic Map

Task	For More Information
Bring a SPARC based system to a specified state (run level booting).	Chapter 2, "Booting a SPARC Based System to a Specified State (Tasks)"
Shut down a SPARC based system.	Chapter 3, "Shutting Down a System (Tasks)"
Reboot a SPARC based system.	Chapter 4, "Rebooting a SPARC Based System (Tasks)"
Boot a SPARC based system from the network.	Chapter 5, "Booting a SPARC Based System From the Network (Tasks)"
Change the default boot behavior on a SPARC based system.	Chapter 6, "Modifying Boot Parameters on a SPARC Based System (Tasks)"
Boot from a ZFS boot environment, snapshot, or dataset on a SPARC based system.	Chapter 7, "Creating, Administering, and Booting From ZFS Boot Environments on SPARC Platforms (Tasks)"
Keep a SPARC based system bootable by using the boot administration interface (`bootadm`).	Chapter 8, "Keeping a SPARC Based System Bootable (Tasks)"
Troubleshoot booting a SPARC based system.	Chapter 9, "Troubleshooting Booting a SPARC Based System (Tasks)"

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.
- A system can be rebooted by turning the power off and then back on.

Caution – This method is not considered a clean shutdown. 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.

Reasons to Boot a System

The following table lists system administration tasks and the corresponding boot option that is used to complete the task.

TABLE 1–2 Reasons for Booting a System

Reason for a System Boot	Appropriate Boot Option	For More Information
Turn off system power due to anticipated power outage.	Turn system power back on	Chapter 3, "Shutting Down a System (Tasks)"
Change kernel parameters in the `/etc/system` file.	Reboot the system to a multiuser state (run level 3 with SMB or NFS resources shared)	"How to Boot a System to a Multiuser State (Run Level 3)" on page 23
Perform file system maintenance, such as backing up or restoring system data.	Press Control-D from a single-user state (run level S) to bring the system back to a multiuser state (run level 3)	"How to Boot a System to a Single-User State (Run Level S)" on page 24
Repair a system configuration file such as `/etc/system`.	Interactive boot	"How to Boot a System Interactively" on page 25
Add or remove hardware from the system.	Reconfiguration boot (turn on system power after adding or removing devices, if devices are not hot-pluggable)	"Setting up Disks for ZFS File Systems (Task Map)" in *Oracle Solaris Administration: Devices and File Systems*
Recover from a hung system and force a crash dump.	Recovery boot	"How to Force a Crash Dump and Reboot of the System" on page 79
Boot the system by using the kernel debugger (`kmdb`) to track down a system problem.	Booting `kmdb`	"How to Boot a System With the Kernel Debugger (`kmdb`) Enabled" on page 80

Service Management Facility and Booting

SMF provides an infrastructure that augments the traditional UNIX startup scripts, init run levels, and configuration files. With the introduction of SMF, the boot process creates 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. To generate a message when each service is started during the boot process, use the `-v` option with the `boot` command.

When a system is being booted you can select the milestone to boot to or select the level of error messages to be recorded. For instance:

- You can choose a specific milestone to boot to using this command:

  ```
  ok boot -m milestone=milestone
  ```

  The default milestone is `all` which starts all enabled services. Another useful milestone is `none` which starts only `init`, `svc.startd` and `svc.configd`. This milestone provides a very useful debugging environment where services can be started manually. See "How to Boot a System Without Starting Any Services" on page 78 for instructions on how to use the `none` milestone.

  The run-level equivalents `single-user`, `multi-user`, and `multi-user-server` are also available, but are not commonly used. The `multi-user-server` milestone, in particular does not start any services which are not a dependency of that milestone, so may not include important services.
- You can choose which level of logging for `svc.startd` using the following command:

  ```
  ok boot -m logging_level
  ```

  The logging levels that you can select are `quiet`, `verbose` and `debug`. See "SMF Service Error Logging" in *Oracle Solaris Administration: Common Tasks* for specific information about the logging levels.

Changes in Behavior When Using SMF

Most of the features that are provided by SMF occur behind the scenes, so users are not typically aware of these features. 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.
- Because services are automatically restarted if possible, it might seem that a process fails to terminate. If the service is defective, the service is placed in maintenance mode, but normally a service is restarted if the process for the service is terminated. 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 by using SMF. Scripts and `inittab` entries that are provided by an ISV or are locally developed will continue to run. The services might not start at exactly the same point in the boot process, but they are not started before the SMF services..

How Run Levels Work

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.

Oracle Solaris 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 1–3 Oracle 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 systems. All daemons are running except for the NFS server daemons.
3	Multiuser level with NFS resources shared	Multiuser	For normal operations with NFS resources shared. This is the default run level.

TABLE 1–3 Oracle Solaris Run Levels *(Continued)*

Run Level	Init State	Type	Purpose
4	Alternative multiuser state	Multiuser	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 a multiuser level with NFS resources shared (or whatever run 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 1–4 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`

What Happens When a System Is Booted to a Multiuser State (Run Level 3)

1. The `init` process is started and reads the properties defined in the `svc:/system/environment:init` SMF service 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 log in to the system.
   b. Passes the startup activities to `svc.startd`.

   For a detailed description of how the `init` process uses the `inittab` file, see the `init`(1M) man page.

When to Use Run Levels or Milestones

In general, changing milestones or run levels is an uncommon procedure. If it is necessary, using the `init` command to change to a run level will change the milestone as well and is the appropriate command to use. The `init` command is also good for shutting down a system.

However, booting a system using the `none` milestone can be very useful for debugging startup problems. There is no equivalent run level to the `none` milestone. For more information, see "How to Boot a System Without Starting Any Services" on page 78.

Overview of the Oracle Solaris Boot Architecture

The Oracle Solaris SPARC boot architecture includes the following fundamental 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.
- **Use of a boot administrative interface to maintain the integrity of the Oracle Solaris boot archives**

  The `bootadm` command handles the details of boot archive update and verification. During an installation or upgrade, the `bootadm` command creates an initial boot archive. During the process of a normal system shutdown, the shutdown process compares the boot archive's contents with the root file system. If there have been updates to the system such as drivers or configuration files, the boot archive is rebuilt to include these changes so that upon reboot, the boot archive and root file system are synchronized. You can use the `bootadm` command to manually update the boot archive. For instructions, see "Maintaining the Integrity of the Boot Archives" on page 71.

  Note – Some `bootadm` command options do not apply to SPARC platforms.

  For more information, see the `bootadm`(1M) and `boot`(1M) man pages.
- **Use of a ramdisk image as the root file system during installation**

  This process is the same on the SPARC and x86 platforms. The ramdisk image is derived from the boot archive and then transferred to the system from the boot device.

  Note – On SPARC platforms, the OpenBoot PROM continues to be used to access the boot device and to transfer the boot archive to the system's memory.

  In the case of a software installation, the ramdisk image is the root file system that is used for the entire installation process. Use of a ramdisk image speeds up the boot process because

Oracle Solaris and any drivers and necessary applications are read one time from the removable media and placed into memory. The system then executes the installation process based on the RAM disk. The ramdisk file system type can be a High Sierra File System (HSFS).

Description of the SPARC Boot Process

This section describes the basic boot process on Oracle Solaris SPARC platforms. For more information about boot processes on specific hardware types, including systems that have service processors and system that have multiple physical domains, see the product documentation for your specific hardware at `http://www.oracle.com/technetwork/indexes/documentation/index.html`.

The process of loading and executing a stand-alone program is called *bootstrapping*. Typically, the stand-alone program is the operating system kernel. However, any stand-alone program can be booted instead of the kernel.

On SPARC platforms, the bootstrap process consists of the following basic phases:

- After you turn on a system, the system firmware (PROM) executes a power-on self-test (POST).
- After the test has been successfully completed, the firmware attempts to autoboot, if the appropriate flag has been set in the non-volatile storage area that is used by the machine's firmware.
- The second-level program is either a file system-specific boot block, when you booting from a disk, or `inetboot` or `wanboot`, when you are booting across the network or using the Automated Installer (AI) utility.

The network booting process is as follows:

- First, the client obtains an IP address and any other parameters that are required to load the second-stage booter.
- Next, the second-stage booter loads the boot archive from the boot device.

For more information about booting a SPARC based system from the network, see Chapter 5, “Booting a SPARC Based System From the Network (Tasks).”

SPARC Boot Phases

Starting with the Oracle Solaris 10 release, boot processes on SPARC platforms have been modified and enhanced to increase commonality with x86 platforms.

The following four boot phases are now independent of each other:

1. **Open Boot PROM phase**

The Open Boot PROM (OBP) phase of the boot process on SPARC platforms 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. 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 and any other components that are required to boot an instance of Oracle Solaris.

4. **Kernel phase**

   The kernel phase is the final stage of the boot process. During this phase, Oracle Solaris is initialized and a minimal root file system is mounted on the ramdisk that was constructed from the boot archive. In some environments, such as an installation, the ramdisk is used as the root file system and remains mounted. 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 remaining primary modules from the boot archive, initializes itself, mounts the real root file system, then discards the boot archive.

◆ ◆ ◆ CHAPTER 2

Booting a SPARC Based System to a Specified State (Tasks)

This chapter provides task-related information for booting a SPARC based system to various system states, also known as *run levels*.

The following is a list of the information that is in this chapter:

- "Booting a SPARC Based System to a Specified State (Task Map)" on page 21
- "Booting a SPARC Based System to a Specified State" on page 22

For overview information about booting a SPARC based system, see Chapter 1, "Booting and Shutting Down a SPARC Based System (Overview)."

For information about booting an x86 based system to a specified state, see Chapter 2, "Booting an x86 Based System to a Specified State (Tasks)," in *Booting and Shutting Down Oracle Solaris on x86 Platforms*.

Booting a SPARC Based System to a Specified State (Task Map)

TABLE 2–1 Booting a SPARC Based System to a Specified State: Task Map

Task	Description	For Instructions
Determine the current run level of a system.	Use the `who` command with the `-r` option to determine a system's current run level.	"Determining a System's Current Run Level" on page 22
Boot a SPARC based system to a multiuser state.	Use this boot method to bring the system back to a multiuser state (run level 3) after shutting down or performing a system hardware maintenance task.	"Booting a SPARC Based System to a Multiuser State (Run Level 3)" on page 23

TABLE 2–1 Booting a SPARC Based System to a Specified State: Task Map *(Continued)*

Task	Description	For Instructions
Boot a SPARC based system to a single-user state.	Use this boot method to perform a system maintenance task, such as backing up a file system.	"Booting a SPARC Based System to a Single-User State (Run Level S)" on page 24
Boot a SPARC based system interactively.	Use this boot method after making temporary changes to a system file or the kernel for testing purposes.	"Booting a SPARC Based System Interactively" on page 25

Booting a SPARC Based System to a Specified State

The following procedures describe how to boot a SPARC based system to a specified state, also called *run level booting*, from the ok PROM prompt. These procedures assume that the system has been cleanly shut down, unless stated otherwise.

Determining a System's Current Run Level

To determine the current run level on a running system, use the who -r command.

EXAMPLE 2–1 Determining a System's Run Level

The output of the who -r command displays information about a system's current run level, as well as 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

Booting a SPARC Based System to a Multiuser State (Run Level 3)

If a system is turned off, turning it on starts the multiuser boot sequence.

Use the `who -r` command to verify that the system is brought to the specified run level. See "Determining a System's Current Run Level" on page 22.

▼ How to Boot a System to a Multiuser State (Run Level 3)

Use this procedure to boot a SPARC based system that is currently at run level 0 to run level 3.

1. **Bring the system to the `ok` PROM prompt.**

2. **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.

3. **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 2–2 Booting a System to a Multiuser State (Run Level 3)

The following example shows the messages from booting a system to run level 3.

```
ok boot
Probing system devices
Probing memory
ChassisSerialNumber FN62030249
Probing I/O buses

.
.
.
.
OpenBoot 4.30.4.a, 8192 MB memory installed, Serial #51944031.
Ethernet address 0:3:ba:18:9a:5f, Host ID: 83189a5f.
Rebooting with command: boot
Boot device: /pci@1c,600000/scsi@2/disk@0,0:a  File and args:
SunOS Release 5.11 Version fips_checksum_nightly 64-bit
Copyright (c) 1983, 2011, Oracle and/or its affiliates. All rights reserved.
DEBUG enabled
misc/forthdebug (455673 bytes) loaded
Hardware watchdog enabled
Hostname: portia-123
NIS domain name is solaris.us.oracle.com
```

```
portia-123 console login: NIS domain name is solaris.us.oracle.com
```

Booting a SPARC Based System to a Single-User State (Run Level S)

Booting a system to a single-user state is used for system maintenance, such as backing up a file system or troubleshooting other system issues.

▼ How to Boot a System to a Single-User State (Run Level S)

1. **Bring the system to the ok PROM prompt.**

2. **Boot the system to run level S.**

   ```
   ok boot -s
   ```

3. **Type the `root` password when the following message is displayed:**

   ```
   SINGLE USER MODE

   Root password for system maintenance (control-d to bypass): xxxxxx
   ```

4. **Verify that the system is at run level S.**

   ```
   # who -r
   ```

5. **Perform the maintenance task that required the change to run level S.**

6. **After you complete the system maintenance task, type Control-D to bring the system to the multiuser state.**

Example 2–3 SPARC: Booting a System to a Single-User State (Run Level S)

The following example shows the messages from booting a system to run level S.

```
ok boot -s
SC Alert: Host System has Reset
Enter #. to return to ALOM.
cpu Device: pci
Device: ebus
/ebus@800: serial
Device: pci
/pci@780: Device 0 Nothing there
/pci@7c0: Device 0 pci
/pci@7c0/pci@0: Device 4 network network
/pci@7c0/pci@0: Device 8 pci
/pci@7c0/pci@0/pci@8: Device 1 network network
/pci@7c0/pci@0/pci@8: Device 2 scsi tape disk
```

```
Sun Fire(TM) T1000, No Keyboard
Copyright 2008 ...  All rights reserved.
OpenBoot 4.30.0.build_12***PROTOTYPE BUILD***, 2000 MB memory available,
Serial #69312178.
Ethernet address 0:14:4f:21:9e:b2, Host ID: 84219eb2.

Boot device: /pci@7c0/pci@0/pci@8/scsi@2/disk@0,0:a  File and args:
zfs-file-system
Loading: /platform/SUNW,Sun-Fire-T1000/boot_archive
ramdisk-root hsfs-file-system
Loading: /platform/SUNW,Sun-Fire-T1000/kernel/sparcv9/unix
SunOS Release 5.11 64-bit
Copyright (c) 1983, 2011, Oracle and/or its affiliates. All rights reserved.
OpenBoot 4.30.0.build_12***PROTOTYPE BUILD***, 2000 MB memory available,
Serial #69312178.
Ethernet address 0:14:4f:21:9e:b2, Host ID: 84219eb2.



Boot device: /pci@7c0/pci@0/pci@8/scsi@2/disk@0,0:a  File and args:
zfs-file-system
Loading: /platform/SUNW,Sun-Fire-T1000/boot_archive
ramdisk-root hsfs-file-system
Loading: /platform/SUNW,Sun-Fire-T1000/kernel/sparcv9/unix
SunOS Release 5.11  64-bit
Copyright (c) 1983, 2011, Oracle and/or its affiliates. All rights reserved.
os-io Hostname: t1000


t1000 console login:
```

Booting a SPARC Based System Interactively

Booting a system interactively is useful if you need to specify an alternate kernel or the `/etc/system` file during the boot process. Use the following procedure to boot a system interactively.

▼ How to Boot a System Interactively

To specify an alternate `/etc/system` file when booting a SPARC based system that has only one boot environment, you can boot the system interactively by using the `boot -a` command. Alternatively, you can resolve a problem with the `/etc/system` file by creating and booting an alternative boot environment. See "Booting From a ZFS Boot Environment on SPARC Platforms" on page 62.

1 **Make backup copies of the `/etc/system` and `boot/solaris/filelist.ramdisk` files. For example:**

```
# 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
```

4 **Bring the system to the `ok` PROM prompt.**

5 **Boot the system interactively.**

```
ok boot -a
```

6 **Respond to the system prompts as follows:**

a. **Specify an alternate system file, then press Return. For example:**

```
Name of system file [etc/system]: /etc/system.bak
```

b. **Specify the root file system, then press Return.**

c. **When prompted, specify the physical name of the root device, then press Return.**

Pressing Return without providing any information accepts the system defaults.

7 **If you are not prompted by the system for information, verify that you typed the `boot -a` command correctly.**

Example 2–4 Booting a System Interactively

In the following 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 "How to Boot a System Interactively" on page 25.

```
ok boot -a
Boot device: /pci@7c0/pci@0/pci@1/pci@0,2/LSILogic,sas@2/disk@0,0:a  File and args: -a
Name of system file [/etc/system]:
SunOS Release 5.11 Version ... 64-bit
Copyright (c) 1983, 2011, Oracle and/or its affiliates. All rights reserved.
Retire store [/etc/devices/retire_store] (/dev/null to bypass):
root filesystem type [zfs]:
Enter physical name of root device
[/pci@7c0/pci@0/pci@1/pci@0,2/LSILogic,sas@2/disk@0,0:a]:
Hostname: system1
Mar 11 17:15:20 svc.startd[9]: svc:/system/filesystem/local:default: \
 Method "/lib/svc/method/fs-local" failed with exit status 95.
system1 console login: NIS domain name is solaris.us.oracle.com
NIS domain name is solaris.us.oracle.com

system1 console login:
```

◆ ◆ ◆ CHAPTER 3

Shutting Down a System (Tasks)

This chapter provides overview and task-related information for shutting down a system. The procedures for shutting down a SPARC based system are identical to the procedures for shutting down an x86 based system. However, output for certain examples might vary.

The following is a list of the information that is in this chapter:

- “Shutting Down a System (Task Map)” on page 27
- “Overview of Shutting Down a System” on page 28
- “Guidelines for Shutting Down a System” on page 28
- “Shutting Down a System” on page 30
- “Turning Off Power to System Devices” on page 34

For overview information about booting a SPARC based system, see Chapter 1, “Booting and Shutting Down a SPARC Based System (Overview).”

For information about booting and shutting down an x86 based system, see *Booting and Shutting Down Oracle Solaris on x86 Platforms*.

Shutting Down a System (Task Map)

TABLE 3–1 Shutting Down a System: Task Map

Task	Description	For Instructions
Determine who is logged in to a system.	If the system is a server that is used by multiple users, use the `who` command to determine who is logged in to a system.	“How to Determine Who Is Logged in to the System” on page 30

TABLE 3–1 Shutting Down a System: Task Map *(Continued)*

Task	Description	For Instructions
Shut down a system by using the `shutdown` command.	Use the `shutdown` command with the appropriate options to shut down a system. This method is preferred for shutting down a server.	"How to Shut Down a System by Using the `shutdown` Command" on page 30
Shut down a system by using the `init` command.	Use the `init` command and indicate the appropriate run level to shut down a system.	"How to Shut Down a System by Using the `init` Command" on page 33

Overview of Shutting Down a System

Oracle Solaris 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 be 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 information about using your system's power management features, see the `poweradm`(1M) man page.

Guidelines for Shutting Down a System

Keep the following in mind when you shut down a system:

- Use either the `shutdown` or the `init` command to shut down a system. Both commands perform a clean system shutdown, which means that all system processes and services are terminated normally.
- You need to be the `root` role to use the `shutdown` and `init` commands.
- Both the `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 the multiuser state with NFS resources shared.
- **Run level 6** – Shuts down the system to run level 0, and then reboots the system to a multiuser level with SMB or NFS resources shared (or whatever run level is the default in the `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 "How Run Levels Work" on page 16.

System Shutdown Commands

The `shutdown` and `init` commands are the primary commands that are used to shut down a system. Both commands perform a *clean shutdown* of the system. As such, all file system changes are written to disk, and all system services, processes, and the operating system are terminated normally.

Turning a system off and then on is not a clean shutdown because system services are terminated abruptly. However, sometimes these actions are needed in emergency situations.

The following table describes the various shutdown commands and provides recommendations for using them.

TABLE 3–2 Shutdown Commands

Command	Description	When to Use
`shutdown`	An executable that calls the `init` program to shut down the system. The system is brought to run level S by default.	Use this command to shut down servers that are operating at run level 3.
`init`	An executable that terminates all active processes and synchronizes the disks before changing run levels.	This command provides a faster system shutdown. The command is preferred for shutting down stand-alone systems when other users will not be affected.
`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.

TABLE 3–2 Shutdown Commands *(Continued)*

Command	Description	When to Use
`halt`, `poweroff`	An executable that synchronizes the disks and stops the processor.	Not recommended because it does not shut down all processes or 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.

Shutting Down a System

The following procedures and examples describe how to shut down a system by using the `shutdown` and `init` commands.

▼ How to Determine Who Is Logged in to the System

For Oracle Solaris systems that are used as multiuser timesharing systems, you might need to determine if any users are logged into the system before shutting it down. Use the following procedure in these instances.

- **To determine who is logged in to a system, use the `who` command, as follows:**

```
$ 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 fourth column, if present, identifies the host name if the user is logged in from a remote system.

▼ How to Shut Down a System by Using the shutdown Command

1 **Become the `root` role.**

2 **For a multiuser server shutdown, find out if any users are logged in to the system.**

```
# who
```

Note – This step is conditional and *only* required if the system is a multiuser timesharing system and not typically used when shutting down newer Oracle Solaris servers and processors.

---

3 **Shut down the system.**

```
# shutdown -iinit-state -ggrace-period -y
```

-i*init-state*	Brings the system to an init state that is different from the default of S. The choices are 0, 1, 2, 5, and 6. Run levels 0 and 5 are states reserved for shutting the system down. Run level 6 reboots the system. Run level 2 is available as a multiuser 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 `root` password, if prompted.**

```
Type Ctrl-d to proceed with normal startup,
(or give root password for system maintenance): xxxxxx
```

6 **After you have finished performing any 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
S (single-user state)	#
0 (power-down state)	`ok` or >
Run level 3 (multiuser state with remote resources shared)	*hostname* `console login:`

Example 3–1 Bringing a Multiuser Server to a Single-User State (Run Level S) by Using the shutdown Command

In the following example, the shutdown command is used to bring a SPARC based system to run level S (the single-user state) in three minutes.

```
# who
root   console      Jun 14 15:49    (:0)

# shutdown -g180 -y

Shutdown started.    Mon Jun 14 15:46:16...

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.
.
.
.
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 3–2 Bringing a System to a Shutdown State (Run Level 0) by Using the shutdown Command

In the following example, the shutdown command is used to bring a SPARC based system to run level 0 in five minutes without requiring additional confirmation.

```
# shutdown
Shutdown started.    Thu Jun 17 12:40:25...

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
```

See Also Regardless of why you shut down a system, you will 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 state, see "Booting a SPARC Based System to a Multiuser State (Run Level 3)" on page 23.

▼ How to Shut Down a System by Using the init Command

Use this procedure when you need to shut down a stand-alone system.

1. **Become the `root` role.**

2. **Shut down the system.**

   `# **init 5**`

   For more information, see the `init(1M)` man page.

Example 3–3 Bringing a System to a Shutdown State (Run Level 0) by Using the init Command

In this example, the `init` command is used to bring a system to the run 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
```

See Also Regardless of why you shut down the system, you will 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 state, see Booting a SPARC Based System to a Multiuser State (Run Level 3).

Turning Off Power to System Devices

You might need to turn off power to system devices to do the following:

- Replace or add hardware.
- Move the system from one location to another location.
- Prepare for an expected power outage or natural disaster such as an approaching electrical storm.

For information about turning off power to devices, see the instructions for the specified hardware in the product documentation at `http://www.oracle.com/technetwork/indexes/documentation/index.html`.

◆ ◆ ◆ CHAPTER 4

Rebooting a SPARC Based System (Tasks)

This chapter describes the various methods for rebooting a SPARC based system, including information about the Fast Reboot feature of Oracle Solaris.

The following is a list of the information that is in this chapter:

- "Rebooting a SPARC Based System (Task Map)" on page 35
- "Rebooting a SPARC Based System" on page 36
- "Accelerating the Reboot Process on a SPARC Based System" on page 38

For overview information about booting a SPARC based system, see Chapter 1, "Booting and Shutting Down a SPARC Based System (Overview)."

For information about rebooting an x86 based system, see Chapter 4, "Rebooting an x86 Based System (Tasks)," in *Booting and Shutting Down Oracle Solaris on x86 Platforms*.

Rebooting a SPARC Based System (Task Map)

TABLE 4–1 Rebooting a SPARC Based System: Task Map

Task	Description	For Instructions
Reboot a SPARC based system by using the `init` command.	Use the `init` command to initiate a run level transition. When using the `init` command to reboot a system, run levels 2, 3, and 4 are available as multiuser system states.	"How to Reboot a System by Using the `init` Command" on page 36
Reboot a SPARC based system by using the `reboot` command.	Use the `reboot` command to restart the kernel and bring the system to a multiuser state.	"How to Reboot a System by Using the `reboot` Command" on page 37

TABLE 4–1 Rebooting a SPARC Based System: Task Map *(Continued)*

Task	Description	For Instructions
Initiate a fast reboot of a SPARC based system.	If the Fast Reboot feature is not enabled, use the `reboot` command with the `-f` option to initiate a fast reboot of a SPARC based system. If the Fast Reboot feature has been enabled, you can use either the `reboot` or the `init 6` command to automatically initiate a fast reboot of a SPARC based system.	"How to Initiate a Fast Reboot of a SPARC Based System" on page 38
Make a fast reboot the default behavior on a SPARC based system.	On SPARC based systems, the Fast Reboot feature is supported, but disabled by default. You can configure the `boot-config` service to perform a fast reboot of a SPARC based system by default.	"Changing the Default Behavior of the Fast Reboot Feature" on page 39
Initiate a standard reboot of a system that has Fast Reboot enabled.	Use the `reboot` command with the `-p` option to perform a standard reboot of the system that has the Fast Reboot feature enabled.	"Initiating a Standard Reboot of a System That Has Fast Reboot Enabled" on page 39

Rebooting a SPARC Based System

You can reboot a system by using either the `init` command or the `reboot` command.

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. The `init` command can be used to initiate a run level transition. When using the `init` command to reboot a system, run levels 2, 3, and 4 are available as multiuser system states.

The `reboot` command restarts the kernel. The kernel is loaded into memory by the PROM monitor, which transfers control to the loaded kernel. Although the `reboot` command can be used by the `root` user at anytime, in certain cases, as with the reboot of a server, the `shutdown` command is normally used first to warn all users who are logged in to the system of the impending loss of service. For more information, see Chapter 3, "Shutting Down a System (Tasks)."

▼ How to Reboot a System by Using the init Command

The `init` command is an executable shell script that terminates all active processes on a system and then synchronizes the disks before changing run levels.

1 **Become the `root` role.**

2 **Reboot the system.**

- **To reboot the system to the state that is defined by the `initdefault` entry in the `/etc/inittab` file, type the following command:**

  ```
  # init 6
  ```

- **To reboot the system to a multiuser state, type the following command:**

  ```
  # init 2
  ```

Example 4–1 Bringing a System to a Single-User State (Run Level S) by Using the init Command

In this example, the `init` command is used to bring a system to a single-user state (run level S).

```
# 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
#
```

▼ How to Reboot a System by Using the reboot Command

1 **Become the `root` role.**

2 **Reboot the system.**

```
# reboot
```

Accelerating the Reboot Process on a SPARC Based System

The Fast Reboot feature of Oracle Solaris is now supported on SPARC platforms, which means that you can use the `-f` option with the `reboot` command to accelerate the boot process by skipping certain POST tests.

The Fast Reboot feature behaves differently on SPARC based systems than it does on an x86 based systems. On x86 based systems, Fast Reboot is the default. Whereas, on SPARC based systems, the behavior is enabled, but you must use the `-f` option with the `reboot` command to initiate a fast reboot. Also, fast reboot on SPARC is applicable only to certain SPARC based systems. On `sun4v` systems fast reboot is unnecessary because the reboot is actually a hypervisor restart that does not involve POST.

The Fast Reboot feature is managed through SMF and implemented through a boot configuration service, `svc:/system/boot-config`. The `boot-config` service provides a means for setting or changing default boot configuration properties. When the `config/fastreboot_default property` is set to `true`, the system performs a fast reboot automatically, without the need to use the `reboot -f` command. By default, this property value is set to `false` on SPARC platforms.

To make a fast reboot the default behavior on a SPARC based system, use the `svccfg` and `svcadm` commands. For instructions, see "Changing the Default Behavior of the Fast Reboot Feature" on page 39.

Note – On SPARC based systems the `boot-config` service also requires the `solaris.system.shutdown` authorization as the `action_authorization` and `value_authorization`.

▼ How to Initiate a Fast Reboot of a SPARC Based System

Use the following procedure to initiate a fast reboot of a SPARC based system when the `config/fastreboot_default` property of the `boot-config` service is set to `false`, which is the default behavior. To change the default behavior of the Fast Reboot feature so that a fast reboot is automatically performed when the system reboots, see "Changing the Default Behavior of the Fast Reboot Feature" on page 39.

1. **Become the `root` role.**

2. **Initiate a fast reboot of the system by typing the following command:**

```
# reboot -f
```

Changing the Default Behavior of the Fast Reboot Feature

The config/fastreboot_default property of the boot-config service enables an automatic fast reboot of the system when either the reboot or the init 6 command is used. When the config/fastreboot_default property is set to true, the system automatically performs a fast reboot, without the need to use the reboot -f command. By default, this property's value is set to false on a SPARC based system.

EXAMPLE 4–2 SPARC: Configuring Properties of the boot-config Service

To configure the properties that are part of the boot-config service use the svccfg and svcadm commands.

To set the property's value to true on a SPARC based system, type the following commands:

```
# svccfg -s "system/boot-config:default" setprop config/fastreboot_default=true
# svcadm refresh svc:/system/boot-config:default
```

Setting the property's value to true accelerates the reboot process, bypassing certain POST tests. When this property is set to true, you no longer have to use the -f option with the reboot command to initiate a fast reboot.

For information about managing the boot configuration service through SMF, see the svcadm(1M) and svccfg(1M) man pages.

Initiating a Standard Reboot of a System That Has Fast Reboot Enabled

To reboot a SPARC based system that has the Fast Reboot feature of Oracle Solaris enabled, without having to reconfigure the properties of the boot-config service, use the -p option with the reboot command, as follows:

```
# reboot -p
```

CHAPTER 5

Booting a SPARC Based System From the Network (Tasks)

This chapter provides overview information, guidelines, and tasks for booting a SPARC based system from the network.

The following is a list of the information that is in this chapter:

- "Booting a SPARC Based System From the Network (Task Map)" on page 41
- "Booting a SPARC Based System From the Network" on page 42

For overview information about booting a SPARC based system, see Chapter 1, "Booting and Shutting Down a SPARC Based System (Overview)."

For information about booting an x86 based system from the network, see Chapter 5, "Booting an x86 Based System From the Network (Tasks)," in *Booting and Shutting Down Oracle Solaris on x86 Platforms*.

Booting a SPARC Based System From the Network (Task Map)

TABLE 5–1 Booting a SPARC Based System From the Network: Task Map

Task	Description	For Instructions
Add network boot arguments in the OBP PROM to enable a wide area network (WAN) boot.	Save the information about which network boot protocol to use when performing a WAN boot by setting the `network-boot-arguments` parameter of the `eeprom` utility.	"How to Specify Network Boot Arguments in the OpenBoot PROM" on page 44
Set up an NVRAM alias to automatically boot a SPARC based system by using the DHCP network protocol.	Save the information about which network boot protocol to use across system reboots by setting up an NVRAM alias.	"Setting Up an NVRAM Alias to Automatically Boot by Using DHCP" on page 45

TABLE 5–1 Booting a SPARC Based System From the Network: Task Map *(Continued)*

Task	Description	For Instructions
Boot a SPARC based system from the network.	After performing any preliminary tasks, use the `boot` command to boot a SPARC based system from the network.	"How to Boot a SPARC Based System From the Network" on page 45

Booting a SPARC Based System From the Network

You might need to boot a system from the network for the following reasons:

- To install Oracle Solaris
- For recovery purposes

The network configuration boot strategy that is used in Oracle Solaris is the Dynamic Host Configuration Protocol (DHCP).

For general information about how DHCP works in this Oracle Solaris release and specific information about setting up a DHCP server, see Part II, "DHCP," in *Oracle Solaris Administration: IP Services*.

SPARC Network Boot Processes

For network devices, the process of booting over a local area network (LAN) and booting over a WAN is slightly different. In both network boot scenarios, the PROM downloads the booter from a boot server or an install server, which is `inetboot` in this case.

When booting over a LAN, the firmware uses DHCP to discover either the boot server or the install server. The Trivial File Transfer Protocol (TFTP) is then used to download the booter, which is `inetboot` in this case.

When you are booting over a WAN, the firmware uses either DHCP or NVRAM properties to discover the install 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.

Requirements for Booting a SPARC Based System 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.

- To perform a network boot of a SPARC based system to install Oracle Solaris for recovery purposes, a DHCP server is required.

  The DHCP server supplies the information that the client needs to configure its network interface. If you are setting up an Automated Installer (AI) server, that server can also be the DHCP server. Or, you can set up a separate DHCP server. For more information, see Part II, "DHCP," in *Oracle Solaris Administration: IP Services*.
- A boot server that provides `tftp` service is also required.

Setting Network Boot Arguments in the OpenBoot PROM

The `network-boot-arguments` parameter of the `eeprom` utility enables you to set configuration parameters to be used by the PROM when you perform a WAN boot. Setting network boot arguments in the PROM takes precedence over any default values. If you are using DHCP, these arguments also take precedence over configuration information that is provided by the DHCP server for the given parameter.

If you are manually configuring an Oracle Solaris system to boot from the network, you must provide the client system with all of the necessary information for the system to boot.

Information that is required by the PROM includes the following:

- IP address of the booting client
- Name of the boot file
- IP address of the server that is providing the boot file image

In addition, you might be required to provide the subnet mask and IP address of the default router to be used.

The syntax to use for network booting is as follows:

```
[protocol,] [key=value,]*
```

protocol	Specifies the address discovery protocol that is to be used.
key=value	Specifies configuration parameters as attribute pairs.

The following table lists the configuration parameters that you can specify for the `network-boot-arguments` parameter.

Parameter	Description
`tftp-server`	IP address of the TFTP server
`file`	File to download by using TFTP or URL for WAN boot

Parameter	Description
`host-ip`	IP address of the client (in dotted-decimal notation)
`router-ip`	IP address of the default router (in dotted-decimal notation)
`subnet-mask`	Subnet mask (in dotted-decimal notation)
`client-id`	DHCP client identifier
`hostname`	Host name to use in the DHCP transaction
`http-proxy`	HTTP proxy server specification (*IPADDR*[:*PORT*])
`tftp-retries`	Maximum number of TFTP retries
`dhcp-retries`	Maximum number of DHCP retries

▼ How to Specify Network Boot Arguments in the OpenBoot PROM

Before You Begin Complete any preliminary tasks that are required for booting a system from the network. For more information, see "Requirements for Booting a SPARC Based System From the Network" on page 42.

1. **On the system that is to be booted from the network, become the `root` role.**

2. **Specify the appropriate values for the `network-boot-arguments` parameter.**

   ```
   # eeprom network-boot-arguments="protocol,hostname=hostname"
   ```

   For example, to use DHCP as the boot protocol and a host name of `mysystem.example.com`, you would set the values for the `network-boot-arguments` parameter as follows:

   ```
   # eeprom network-boot-arguments="DHCP,hostname=mysystem.example.com"
   ```

3. **Bring the system to the ok PROM prompt.**

   ```
   # init 0
   ```

4. **Boot the system from the network.**

   ```
   ok boot net
   ```

Note – When you specify the `network-boot-arguments` parameter in this way, there is no need to specify the arguments from the PROM command line. Doing so will ignore any other values set for the `network-boot-arguments` parameter that you have might have specified.

Setting Up an NVRAM Alias to Automatically Boot by Using DHCP

In Oracle Solaris 11, DHCP is the network configuration boot strategy that is used when booting from the network to install Oracle Solaris. To boot a system from the network with DHCP, a DHCP boot server must be available on your network.

You can specify that a SPARC based system boot by using the DHCP protocol when you run the `boot` command. Or, you can save the information 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 with DHCP by default:

```
ok nvalias net    /pci@1f,4000/network@1,1:dhcp
```

As a result, when you type `boot net`, the system boots by using DHCP.

Caution – Do not use the `nvalias` command to modify the `NVRAMRC` file unless you are very familiar with the syntax of this command and also the `nvunalias` command.

▼ How to Boot a SPARC Based System From the Network

Before You Begin

- Perform any prerequisite tasks for setting up DHCP configuration. See "Requirements for Booting a SPARC Based System From the Network" on page 42.
- If you booting the system over the network to install Oracle Solaris, first download the AI client image and create an install service based on that image. For instructions, see Part III, "Installing Using an Install Server," in *Installing Oracle Solaris 11 Systems*.

1 Become the `root` role.

2 If necessary, bring the system to the ok PROM prompt.

```
# init 0
```

3 Boot the system from the network without using the ”install ”flag.

```
ok boot net:dhcp
```

Note – If you have changed the PROM setting to boot with DHCP by default, you only have to specify `boot net`, as shown here:

```
ok boot net
```

◆ ◆ ◆ CHAPTER 6

Modifying Boot Parameters on a SPARC Based System (Tasks)

This chapter provides task-related information about modifying the default boot behavior on a SPARC based system.

The following is a list of the information that is in this chapter:

- "Modifying Boot Parameters on a SPARC Based System (Task Map)" on page 47
- "Modifying Boot Parameters on a SPARC Based System" on page 48

If you need to configure SPARC boot mode properties on an Oracle Integrated Lights Out Manager (ILOM) service processor, see the hardware documentation at `http://download.oracle.com/docs/cd/E19166-01/E20792/z40003d6165586.html#scrolltoc`.

For overview information about booting a SPARC based system, see Chapter 1, "Booting and Shutting Down a SPARC Based System (Overview)."

For information about modifying boot parameters on an x86 based system, see Chapter 6, "Modifying Boot Parameters on an x86 Based System (Tasks)," in *Booting and Shutting Down Oracle Solaris on x86 Platforms*.

Modifying Boot Parameters on a SPARC Based System (Task Map)

TABLE 6–1 Modifying Boot Parameters on a SPARC Based System: Task Map

Task	Description	For Instructions
Identify the PROM revision number for a SPARC based system.	To display PROM revision number for a system, use the `banner` command at the `ok` PROM prompt.	"How to Identify the PROM Revision Number for a System" on page 49

TABLE 6–1 Modifying Boot Parameters on a SPARC Based System: Task Map *(Continued)*

Task	Description	For Instructions
Identify devices on a SPARC based system that can be booted.	Before modifying boot behavior by using the boot PROM, identify the devices on a system.	"How to Identify Devices on a System" on page 49
Display the current boot device for a SPARC based system.	Use this procedure to determine the current default boot device from which the system will boot.	"How to Determine the Default Boot Device" on page 51
Change the default boot device on a SPARC based system.	To change the default boot device, use one of the following methods: ■ Change the `boot-device` parameter at `ok` PROM prompt. ■ Change the `boot-device` parameter by using the `eeprom` command.	"How to Change the Default Boot Device by Using the Boot PROM" on page 51 "How to Change the Default Boot Device by Using the `eeprom` Utility" on page 53
Change the default boot file or kernel on a SPARC based system.	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.	"How to Change the Default Boot File by Using the Boot PROM" on page 53 "How to Change the Default Boot File by Using the `eeprom` Utility" on page 54

Modifying Boot Parameters on a SPARC Based System

The boot PROM is used to boot a SPARC based system and to modify boot parameters. For example, you might want to reset the device from which to boot, change the default boot file or kernel, or run hardware diagnostics before bringing the system to a multiuser state.

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 the `monitor(1M)` and `eeprom(1M)` man pages.

▼ How to Identify the PROM Revision Number for a System

1 **Bring the system to the ok PROM prompt.**

For more information, see "How to Shut Down a System by Using the `init` Command" on page 33.

2 **Display a system's PROM revision number by using the `banner` command.**

```
ok banner
```

▼ How to Identify Devices on a System

You might need to identify the devices on a system to determine the appropriate devices from which to boot.

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 you use the `reset` command, then reset the `auto-boot?` parameter to `true`.**

```
ok setenv auto-boot? true
auto-boot? =          true
```

3 Boot the system to a multiuser state.

```
ok reset-all
```

Example 6–1 Identifying the Devices on a System

The following example shows how to identify the devices connected to a system.

```
ok setenv auto-boot? false
auto-boot? =          false
ok reset-all
SC Alert: Host System has Reset



Sun Fire T200, No Keyboard
.
.
.
OpenBoot 4.30.4.a, 16256 MB memory available, Serial #69069018.
Ethernet address 0:14:4f:1d:e8:da, Host ID: 841de8da.
ok probe-ide
   Device 0  ( Primary Master )
         Removable ATAPI Model: MATSHITACD-RW  CW-8124

  Device 1  ( Primary Slave )
         Not Present

  Device 2  ( Secondary Master )
         Not Present

  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
ttya                     /pci@7c0/pci@0/pci@1/pci@0/isa@2/serial@0,3f8
nvram                    /virtual-devices/nvram@3
net3                     /pci@7c0/pci@0/pci@2/network@0,1
net2                     /pci@7c0/pci@0/pci@2/network@0
net1                     /pci@780/pci@0/pci@1/network@0,1
net0                     /pci@780/pci@0/pci@1/network@0
net                      /pci@780/pci@0/pci@1/network@0
ide                      /pci@7c0/pci@0/pci@1/pci@0/ide@8
cdrom                    /pci@7c0/pci@0/pci@1/pci@0/ide@8/cdrom@0,0:f
disk3                    /pci@7c0/pci@0/pci@1/pci@0,2/LSILogic,sas@2/disk@3
disk2                    /pci@7c0/pci@0/pci@1/pci@0,2/LSILogic,sas@2/disk@2
disk1                    /pci@7c0/pci@0/pci@1/pci@0,2/LSILogic,sas@2/disk@1
disk0                    /pci@7c0/pci@0/pci@1/pci@0,2/LSILogic,sas@2/disk@0
disk                     /pci@7c0/pci@0/pci@1/pci@0,2/LSILogic,sas@2/disk@0
scsi                     /pci@7c0/pci@0/pci@1/pci@0,2/LSILogic,sas@2
```

```
virtual-console          /virtual-devices/console@1
name                     aliases
```

▼ 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 System by Using the `init` Command" on page 33.

2 Determine the default boot device.

```
ok printenv boot-device
```

`boot-device`	Identifies the parameter for setting the device from which to boot. For more information, see the `printenv`(1B) man page.

The default `boot-device` is displayed in a format that is similar to the following:

```
boot-device =  /pci@7c0/pci@0/pci@1/pci@0,2/LSILogic,sas@2/disk@0,0:a
```

If the `boot-device` parameter specifies 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
```

▼ How to Change the Default Boot Device by Using the Boot PROM

Before You Begin You might need to identify the devices on the system before you can change the default boot device to some other device. For information about identifying devices on the system, see "How to Identify Devices on a System" on page 49.

1 Bring the system to the ok PROM prompt.

```
# init 0
```

2 Change the value of the `boot-device` parameter.

```
ok setenv boot-device device[n]
```

`device[`*n*`]`	Identifies the `boot-device` value, such as `disk` or `network`. The *n* can be specified as a disk number. 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 6–2 Changing the Default Boot Device by Using the Boot PROM

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.
.
.
.
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
.
```

```
.
.
Boot device: net  File and args:

pluto console login:
```

▼ How to Change the Default Boot Device by Using the eeprom Utility

1. **Become the `root` role.**

2. **Specify the alternate device from which to boot.**

   `#` **`eeprom boot-device`** *new-boot-device*

3. **Verify that the new boot parameter has been set.**

   `#` **`eeprom boot-device`**

   The output should display the new `eeprom` value for the `boot-device` parameter.

▼ How to Change the Default Boot File by Using the Boot PROM

1. **Bring the system to run level 0.**

   `#` **`init 0`**

   The `ok` PROM prompt is displayed. For more information, see the `init`(1M) man page.

2. **Set the `boot-file` property to an alternate boot file or kernel.**

   `ok` **`setenv boot-file`** *boot-file*

3. **Verify that the default boot file or kernel 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.

▼ How to Change the Default Boot File by Using the eeprom Utility

1. **Become the `root` role.**

2. **Specify the alternate boot file or kernel to boot.**

   ```
   # eeprom boot-file new boot-file
   ```

   For example:

   ```
   # eeprom boot-file=kernel.name/sparcv9/unix
   ```

3. **Verify that the default boot file has been changed.**

   ```
   # eeprom boot-file
   ```

   The output should display the new `eeprom` value for the specified parameter.

◆ ◆ ◆ CHAPTER 7

Creating, Administering, and Booting From ZFS Boot Environments on SPARC Platforms (Tasks)

This chapter describes how to create, administer, and boot from a ZFS boot environment, also called a *BE*, on a SPARC based system.

The following is a list of the information that is in this chapter:

- "Creating, Administering, and Booting From ZFS Boot Environments (Task Map)" on page 55
- "Creating and Administering Boot Environments" on page 57
- "Booting From a ZFS Boot Environment on SPARC Platforms" on page 62

For overview information about booting a SPARC based system, see Chapter 1, "Booting and Shutting Down a SPARC Based System (Overview)."

For information about booting from a ZFS boot environment on an x86 based system, see Chapter 7, "Creating, Administering, and Booting From ZFS Boot Environments on x86 Platforms (Tasks)," in *Booting and Shutting Down Oracle Solaris on x86 Platforms*.

For detailed information about managing boot environments, see *Creating and Administering Oracle Solaris 11 Boot Environments*.

Creating, Administering, and Booting From ZFS Boot Environments (Task Map)

TABLE 7–1 Creating, Administering, and Booting From ZFS Boot Environments: Task Map

Task	Description	For Instructions
Display a list of boot environments, snapshots, and datasets.	To display a list of boot environments, snapshots, and datasets, use the `beadm list` command.	"How to Display a List of Available Boot Environments, Snapshots, and Datasets" on page 60

TABLE 7–1 Creating, Administering, and Booting From ZFS Boot Environments: Task Map *(Continued)*

Task	Description	For Instructions
Create a new boot environment.	Create a new boot environment by using the `beadm create` command.	"How to Create a New Boot Environment" on page 57
Create a snapshot of a boot environment.	Create a snapshot of an existing boot environment by using the `beadm create` *beName@snapshot* command.	"How to Create a Snapshot of a Boot Environment" on page 59
Create a boot environment from an existing snapshot.	Create a new boot environment from an existing snapshot by using the `beadm` command.	"How to Create a Boot Environment From an Existing Snapshot" on page 59
Activate a newly created boot environment.	Activate a newly created boot environment by using the `beadm activate` command.	"How to Activate a Newly Created Boot Environment" on page 59
Display a list of boot environments and datasets during the boot sequence on a SPARC based system.	To display a list of boot environments that are on a system during the boot sequence, specify the `-L` option with the `boot` command.	"SPARC: How to Display a List of Available Boot Environments During the Boot Sequence" on page 62
Destroy a boot environment.	Destroy a boot environment by using the `beadm destroy` command.	"How to Destroy a Boot Environment" on page 61
Boot from a specified boot environment, dataset, or root file system on a SPARC based system.	Use the `boot -Z` option to boot a specified ZFS boot environment, snapshot, or dataset. **Note –** This option is only supported for boot devices that contain a ZFS pool.	"How to Boot From a ZFS Boot Environment or Root File System" on page 63

Creating and Administering Boot Environments

The following tasks describe how to create and administer boot environments, snapshots, and datasets by using the `beadm` utility.

- A *boot environment* (BE) is a ZFS file system that is designated for booting. A boot environment is essentially a bootable instance of the Oracle Solaris OS image, plus any other software packages that are installed into that image. You can maintain multiple boot environments on a single system. Each boot environment can have different OS versions installed. When you install Oracle Solaris, a new boot environment is automatically created during the installation.
- A *snapshot* is a read-only image of a dataset or boot environment that is taken at a given point in time. Note that a snapshot is not bootable. However, you can create a boot environment that is based on a particular snapshot and then activate that new boot environment so that it becomes the default boot environment upon the next system reboot.
- A *dataset* is a generic term that is used to identify a ZFS file system, clone, snapshot, or volume.
- *Shared datasets* are user-defined directories, such as `/export`, that contain the same mount point in both the active and inactive boot environments. Shared datasets are located outside the root dataset area of each boot environment.
- A boot environment's *critical datasets* are included within the root dataset area for that environment.

For more information about the `beadm` command, see the `beadm`(1M) man page. For more information about managing boot environments, see *Creating and Administering Oracle Solaris 11 Boot Environments*. For specific information about using the `beadm` command in a global or non-global zones environment, see Chapter 2, “beadm Zones Support,” in *Creating and Administering Oracle Solaris 11 Boot Environments*.

▼ How to Create a New Boot Environment

1 **Become the `root` role.**

2 **Create a boot environment by using the `beadm create` command.**

```
# beadm create beName
```

where *beName* is a variable for the name of the new boot environment. This new boot environment is inactive.

Note – The `beadm create` command does not create a partial boot environment. Either a new, full boot environment is successfully created, or the command fails.

---

3 **(Optional) Mount the new boot environment.**

```
# beadm mount beName mountpoint
```

If the directory for the mount point does not exist, the `beadm` command creates the directory, then mounts the boot environment on that directory. If the boot environment is already mounted, the `beadm mount` command fails and does not remount the boot environment at the new location.

The boot environment is mounted, but remains inactive. Note that you can upgrade a mounted, inactive boot environment. Also, remember to unmount the boot environment before rebooting your system.

4 **(Optional) To boot from the new boot environment, first activate the boot environment.**

```
# beadm activate beName
```

where *beName* is a variable for the name of the boot environment to be activated. Upon reboot, the newly active boot environment becomes the default boot entry that is listed in the GRUB menu.

Example 7–1 Creating a Cloned Boot Environment With Shared Datasets

The following example shows the datasets in a newly created boot environment named `BE2`. The original boot environment in this example is `BE1`. The new boot environment, `BE2`, contains separate datasets that were cloned from `BE1`. If `BE1` contains separate datasets for traditional file systems, such as `/opt`, then those datasets are also cloned.

```
# beadm create BE2
# beadm list -a BE2
BE/Dataset/Snapshot Active Mountpoint Space Policy Created
------------------- ------ ---------- ----- ------ -------
BE2
   rpool/ROOT/BE2   -      -          42.0K static 2011-04-07 10:56
```

As shown in the previous output, the name of the storage pool is `rpool`. The pool already exists on the system, as it was previously set up by the initial installation or an upgrade. `ROOT` is a special dataset that was also created previously by the initial installation or upgrade. `ROOT` is reserved exclusively for use by boot environment roots.

▼ How to Create a Snapshot of a Boot Environment

1. **Become the `root` role.**

2. **Create the snapshot of the boot environment.**

   ```
   # beadm create beName@snapshot
   ```

   Example snapshot names include the following:

   - `BE@0312200.12:15pm`
   - `BE2@backup`
   - `BE1@march132008`

▼ How to Create a Boot Environment From an Existing Snapshot

1. **Become the `root` role.**

2. **Create a new boot environment from a snapshot by typing the following command:**

   ```
   # beadm create -e BEname@snapshotdescription beName
   ```

   Replace *BEname@snapshotdescription* with the name of an existing snapshot and *beName* with a custom name for the new boot environment.

   For example:

   ```
   # beadm create -e BE1@now BE2
   ```

   This command creates a new boot environment named `BE2` from the existing snapshot named `BE1@now`. You can then active the boot environment. For instructions, see "How to Activate a Newly Created Boot Environment" on page 59.

▼ How to Activate a Newly Created Boot Environment

You can activate a newly created boot environment so that upon reboot it is the default boot environment that is booted. Note that only one boot environment can be active at any given time.

1. **Become the `root` role.**

2. **Activate a newly created boot environment by using the following command:**

   ```
   # beadm activate beName
   ```

   where *beName* is a variable for the boot environment to be activated.

Note the following:

- The `beadm activate` *beName* command activates the boot environment by setting the `bootfs` bootable pool property to the value of the `ROOT` dataset of the boot environment that is being activated.
- The `beadm activate` command sets the newly activated boot environment as the default in the `menu.lst` file.

3 **Reboot the system.**

The newly activated boot environment is now the default entry in the SPARC boot menu.

▼ How to Display a List of Available Boot Environments, Snapshots, and Datasets

To display available boot environments, snapshots, and datasets that were created by using the `beadm` command, use the `beadm list` command.

1 **Become the `root` role.**

2 **To list all of the available datasets on the system that were created by using the `beadm` command, type the following command:**

```
# beadm list option
```

`-a`	Lists all available information about the boot environment. This option includes subordinate datasets and snapshots.
`-d`	Lists information about a boot environment's datasets.
`-s`	Lists information about a boot environment's snapshots. This option is used in conjunction with the `-d` option.
`-H`	Omits the header information from the display. Choosing this option results in a display that can be more easily parsed for scripts or other programs.

3 **To list the available datasets for a specific boot environment, include the boot environment name in the `beadm list` command syntax.**

For example, to list all of the available datasets in the `oracle-solaris` boot environment, you would type the following command:

```
# beadm list -a oracle-solaris
 BE/Dataset/Snapshot   Active Mountpoint Space  Policy Created
-------------------   ------ ---------- -----  ------ -------
oracle-solaris
   rpool/ROOT/solaris -      -          14.33M static 2011-01-20 07:45
```

Example 7–2 Viewing Snapshot Specifications

The following `beadm list` example includes the `-s` option, which displays information for any snapshots that exist on the current image. The status of those snapshots can also be displayed by using the `zfs` command.

In the following sample results, each snapshot title includes a time stamp, indicating when that snapshot was taken.

```
# beadm list -s test-2
```

The sample results are displayed.

```
BE/Snapshot     Space Policy Created
----------- ----- ------ -------
test-2
test-2@2010-04-12-22:29:27 264.02M static 2010-04-12 16:29
test-2@2010-06-02-20:28:51 32.50M static 2010-06-02 14:28
test-2@2010-06-03-16:51:01 16.66M static 2010-06-03 10:51
test-2@2010-07-13-22:01:56 25.93M static 2010-07-13 16:01
test-2@2010-07-21-17:15:15 26.00M static 2010-07-21 11:15
test-2@2010-07-25-19:07:03 13.75M static 2010-07-25 13:07
test-2@2010-07-25-20:33:41 12.32M static 2010-07-25 14:33
test-2@2010-07-25-20:41:23 30.60M static 2010-07-25 14:41
test-2@2010-08-06-15:53:15 8.92M static 2010-08-06 09:53
test-2@2010-08-06-16:00:37 8.92M static 2010-08-06 10:00
test-2@2010-08-09-16:06:11 193.72M static 2010-08-09 10:06
test-2@2010-08-09-20:28:59 102.69M static 2010-08-09 14:28
test-2@install 205.10M static 2010-03-16 19:04
```

▼ How to Destroy a Boot Environment

If you want to make more disk space available on your system, you can use the `beadm` command to destroy (remove) an existing boot environment.

Note the following:

- You cannot destroy a boot environment that is currently booted.
- The `beadm destroy` command automatically removes the destroyed boot environment's entry from the SPARC boot menu.
- The `beadm destroy` command destroys only the critical or nonshared datasets of the boot environment. Shared datasets are located outside of the boot environment root dataset area and are not affected when a boot environment is destroyed.

1 Become the `root` role.

2 **To destroy a boot environment, type the following command:**

```
# beadm destroy beName
```

You are prompted for confirmation before destroying the boot environment.

`beadm destroy`	Destroys the boot environment that is specified by *beName*.
`-F`	Forces the destruction of the boot environment without a confirmation request.
`-f`	Forces the destruction of the boot environment, even if it is mounted.

Booting From a ZFS Boot Environment on SPARC Platforms

The following two options of the `boot` command support booting from a ZFS root file system on SPARC based systems:

`-L` Displays a list of available boot environments 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 boot environment.

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 boot environments on the system. Then, use the `-Z` option to boot the specified boot environment.

For more information, see the `boot`(1M) man page.

▼ SPARC: How to Display a List of Available Boot Environments During the Boot Sequence

On SPARC based systems, the `menu.lst` file contains the following two commands:

- `title` – Provides a title for a boot environment
- `bootfs` – Specifies the full name of the boot environment

As explained in the following procedure, to display a list of the boot environments within a ZFS pool, use the `boot -L` command. This command displays a list of the available boot environments within a given ZFS root pool and provides instructions for booting the system.

1 **Become the `root` role.**

2 **Bring the system to the ok PROM prompt.**

```
# init 0
```

3 **List the available boot environments in a ZFS pool.**

```
ok boot device-specifier -L
```

where *device-specifier* identifies a storage pool, *not* a single root file system.

4 **To boot one of the entries that is displayed, type the number that corresponds to the entry.**

5 **Boot the specified boot environment by following the instructions that are displayed on the screen.**

For instructions, see "How to Boot From a ZFS Boot Environment or Root File System" on page 63.

Example 7–3 Displaying a List of Available Boot Environments by Using the boot -L Command

```
# 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: /platformsun4v/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 5, "Managing ZFS Root Pool Components," in *Oracle Solaris Administration: ZFS File Systems*.

▼ How to Boot From a ZFS Boot Environment or Root File System

When booting from ZFS, the *device-specifier* identifies a storage pool, *not* a single root file system. A storage pool can contain multiple boot environments, 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. The default boot device is identified by the pool's `bootfs` property. This procedure shows how to boot the system by specifying a ZFS boot environment. See the `boot`(1M) man page for a complete description of all the boot options that are available.

Note – In Oracle Solaris 11, a ZFS root file system is booted by default. Use this procedure to specify a ZFS root file system from which to boot.

For more information, see the `zpool`(1M) man page.

1 **Become the root role.**

2 **Bring the system to the `ok` PROM prompt.**

```
# init 0
```

3 **(Optional) Display a list of available boot environments by using the `boot` command with the `-L` option.**

For instructions, see "SPARC: How to Display a List of Available Boot Environments During the Boot Sequence" on page 62.

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 displayed on the screen.**

```
To boot the selected entry, invoke:
boot [<root-device>] -Z rpool/ROOT/boot-environment

ok boot -Z rpool/ROOT/boot-environment
```

For example:

```
# boot -Z rpool/ROOT/zfs2BE
```

6 **After the system has booted, verify the active boot environment.**

```
# prtconf -vp | grep whoami
```

7 **(Optional) To display the boot path for the active boot environment, type the following command:**

```
# prtconf -vp | grep bootpath
```

8 **(Optional) To determine whether the correct boot environment was booted, type the following command:**

```
# df -lk
```

Example 7–4 Booting From a ZFS Boot Environment

This example shows how to use the `boot -Z` command to boot a ZFS boot environment 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)

.
.
.
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
.
.
.
Hostname: mallory
NIS domainname is ...
Reading ZFS config: done.
Mounting ZFS filesytems: (6/6)

mallory console login:
```

See Also For more information about booting from a ZFS root file system, see "Booting From a ZFS Root File System" in *Oracle Solaris Administration: ZFS File Systems*.

◆ ◆ ◆ CHAPTER 8

Keeping a SPARC Based System Bootable (Tasks)

This chapter describes how the keep a SPARC based system bootable by using the boot administration interface (`bootadm`). Procedures for displaying information about the boot archive and for maintaining the integrity of boot archive, as well as troubleshooting boot archive issues, are described.

The following is a list of the information that is in this chapter:

- "Keeping a SPARC Based System Bootable (Task Map)" on page 67
- "Description of the Oracle Solaris Boot Archives" on page 68
- "Managing the Boot Archive SMF Service" on page 70
- "Maintaining the Integrity of the Boot Archives" on page 71

For overview information about booting a SPARC based system, see Chapter 1, "Booting and Shutting Down a SPARC Based System (Overview)."

For information about keeping an x86 based system bootable, see Chapter 8, "Keeping an x86 Based System Bootable (Tasks)," in *Booting and Shutting Down Oracle Solaris on x86 Platforms*.

Keeping a SPARC Based System Bootable (Task Map)

TABLE 8–1 Keeping a SPARC Based System Bootable: Task Map

Task	Description	For Instructions
List the contents of the boot archive by using the `bootadm` command.	Use the `bootadm list-archive` command to list the contents of the boot archive.	"How to List the Contents of the Boot Archive" on page 68

TABLE 8–1 Keeping a SPARC Based System Bootable: Task Map *(Continued)*

Task	Description	For Instructions
Manage the `boot-archive` service.	The `boot-archive` service is controlled by the SMF. Use the `svcadm` command to enable or disable the service. Use the `svcs` command to verify whether the `boot-archive` service is running.	"Managing the Boot Archive SMF Service" on page 70
Clear a boot archive update failure by using the `bootadm` command to manually update the boot archive.	Use this procedure to manually clear boot archive update failures on a SPARC based system.	"How to Clear a Failed Automatic Boot Archive Update by Manually Updating the Boot Archive" on page 71

Description of the Oracle Solaris Boot Archives

When you install Oracle Solaris, the `bootadm` command creates a boot archive on your system. A *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.

In addition, the `bootadm` command handles the details of boot archive update and verification. During the process of a normal system shutdown, the shutdown process compares the boot archive's contents with the root file system. If there have been updates to the system such as drivers or configuration files, the boot archive is rebuilt to include these changes so that upon reboot, the boot archive and root file system are synchronized.

Obtaining Information About the Location and Contents of the SPARC Boot Archive

The files in the SPARC boot archive are located in the `/platform` directory. You can list the contents of the boot archive by using the `bootadm list-archive` command, as described in the following procedure. If any files in the boot archive are updated, the archive must be rebuilt. For modifications to take effect, the rebuild of the archive must take place before the next system reboot.

▼ How to List the Contents of the Boot Archive

1 Become the `root` role.

2 To list the files and directories that are included in the boot archive, type:

```
# bootadm list-archive
```

Example 8–1 Listing the Contents of the SPARC Boot Archive

The following example shows the contents of the boot archive on a SPARC based system.

```
root@tsystem:~# bootadm list-archive
platform/SUNW,A70/kernel
platform/SUNW,Netra-210/kernel
platform/SUNW,Netra-240/kernel
platform/SUNW,Netra-440/kernel
platform/SUNW,Netra-CP2300/kernel
platform/SUNW,Netra-CP3010/kernel
platform/SUNW,Netra-CP3060/kernel
platform/SUNW,Netra-CP3260/kernel
platform/SUNW,Netra-T12/kernel
platform/SUNW,Netra-T2000/kernel
platform/SUNW,Netra-T4/kernel
platform/SUNW,Netra-T5220/kernel
platform/SUNW,Netra-T5440/kernel
platform/SUNW,SPARC-Enterprise-T1000/kernel
platform/SUNW,SPARC-Enterprise-T2000/kernel
platform/SUNW,SPARC-Enterprise-T5120/kernel
platform/SUNW,SPARC-Enterprise-T5220/kernel
platform/SUNW,SPARC-Enterprise/kernel
platform/SUNW,Serverblade1/kernel
platform/SUNW,Sun-Blade-100/kernel
platform/SUNW,Sun-Blade-1000/kernel
platform/SUNW,Sun-Blade-1500/kernel
platform/SUNW,Sun-Blade-2500/kernel
platform/SUNW,Sun-Blade-T6300/kernel
platform/SUNW,Sun-Blade-T6320/kernel
platform/SUNW,Sun-Blade-T6340/kernel
platform/SUNW,Sun-Fire-15000/kernel
platform/SUNW,Sun-Fire-280R/kernel
platform/SUNW,Sun-Fire-480R/kernel
platform/SUNW,Sun-Fire-880/kernel
platform/SUNW,Sun-Fire-T1000/kernel
platform/SUNW,Sun-Fire-T200/kernel
platform/SUNW,Sun-Fire-V210/kernel
platform/SUNW,Sun-Fire-V215/kernel
platform/SUNW,Sun-Fire-V240/kernel
platform/SUNW,Sun-Fire-V245/kernel
platform/SUNW,Sun-Fire-V250/kernel
platform/SUNW,Sun-Fire-V440/kernel
platform/SUNW,Sun-Fire-V445/kernel
platform/SUNW,Sun-Fire-V490/kernel
platform/SUNW,Sun-Fire-V890/kernel
platform/SUNW,Sun-Fire/kernel
platform/SUNW,T5140/kernel
platform/SUNW,T5240/kernel
platform/SUNW,T5440/kernel
platform/SUNW,USBRDT-5240/kernel
platform/SUNW,Ultra-250/kernel
platform/SUNW,Ultra-4/kernel
platform/SUNW,Ultra-5_10/kernel
platform/SUNW,Ultra-80/kernel
platform/SUNW,Ultra-Enterprise-10000/kernel
platform/SUNW,Ultra-Enterprise/kernel
platform/SUNW,UltraAX-i2/kernel
```

```
platform/SUNW,UltraSPARC-IIe-NetraCT-40/kernel
platform/SUNW,UltraSPARC-IIe-NetraCT-60/kernel
platform/SUNW,UltraSPARC-IIi-Netract/kernel
platform/sun4u-us3/kernel
platform/sun4v/kernel
etc/cluster/nodeid
etc/dacf.conf
etc/driver
etc/mach
kernel
root@tsystem:~#
```

Managing the Boot Archive SMF Service

The `boot-archive` service is controlled by SMF. The service instance is `svc:/system/boot-archive:default`. The `svcadm` command is used to enable and disable services.

Determining Whether the boot-archive Service Is Running

If the `boot-archive` service is disabled, automatic recovery of the boot archives upon a system reboot might not occur. As a result, the boot archives could become unsynchronized or corrupted, preventing the system from booting.

To determine whether the `boot-archive` service is running, use the `svcs` command, as follows:

```
$ svcs boot-archive
STATE          STIME    FMRI
online         Mar_31   svc:/system/boot-archive:default
```

In this example, the output of the `svcs` command indicates that the `boot-archive` service is online.

For more information, see the `svcadm`(1M) and `svcs`(1) man pages.

▼ How to Enable or Disable the boot-archive SMF Service

1. **Become the `root` role.**

   For more information, see "How to Obtain Administrative Rights" in *Oracle Solaris Administration: Security Services*.

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 that the service is offline.

Troubleshooting For more information troubleshooting boot archive update failures, see "Maintaining the Integrity of the Boot Archives" on page 71.

Maintaining the Integrity of the Boot Archives

The boot administration interface, `bootadm`, enables you to perform the follow tasks for maintaining the boot archives:

- List the files and directories that are included in a system's boot archive.
- Manually update the current boot archives on a system.

The syntax of the command is as follows:

`bootadm [`*subcommand***`] [-`***option***`] [-R`** ***altroot*****`]`**

For more information about the `bootadm` command, see the `bootadm`(1M) man page.

▼ How to Clear a Failed Automatic Boot Archive Update by Manually Updating the Boot Archive

During the process of booting the system, if a warning message that is similar to the following is displayed, take action accordingly:

```
WARNING: Automatic update of the boot archive failed.
Update the archives using 'bootadm update-archive'
command and then reboot the system from the same device that
was previously booted.
```

The following procedure describes how to manually update an out-of-date boot archive by using the `bootadm` command.

Note – The same procedure can also be used to manually update the boot archive.

1 **Become the `root` role.**

2 **To update the boot archive, type the following command:**

```
# bootadm update-archive
```

Note – To update the boot archive on an alternate root file system, type the following command:

```
# bootadm update-archive -R /a
```

`-R` *altroot*	Specifies an alternate root path to apply to the `update-archive` subcommand.

Caution – 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.

---

3 **Reboot the system.**

```
# reboot
```

◆ ◆ ◆ CHAPTER 9

Troubleshooting Booting a SPARC Based System (Tasks)

The following are procedures for troubleshooting booting an Oracle Solaris instance on a SPARC based system.

The following is a list of the information that is in this chapter:

- "Troubleshooting Booting a SPARC Based System (Task Map)" on page 73
- "Shutting Down and Booting a SPARC Based System for Recovery Purposes" on page 74

For information about stopping and starting Oracle Solaris for recovery purposes, if you are running a service processor, as well as instructions on controlling Oracle ILOM service processors, see the hardware documentation at `http://download.oracle.com/docs/cd/E19166-01/E20792/z400130a9112.html#scrolltoc`.

For overview information about booting a SPARC based system, see Chapter 1, "Booting and Shutting Down a SPARC Based System (Overview)."

For information about how to resolve problems with the Oracle Solaris boot archives, see "Maintaining the Integrity of the Boot Archives" on page 71.

Troubleshooting Booting a SPARC Based System (Task Map)

TABLE 9–1 Troubleshooting Booting a SPARC Based System: Task Map

Task	Description	For Instructions
Stop a SPARC based system for recovery purposes.	If a damaged file is preventing a SPARC based system from booting, first stop the system to attempt recovery.	"How to Stop a System for Recovery Purposes" on page 75

TABLE 9–1 Troubleshooting Booting a SPARC Based System: Task Map *(Continued)*

Task	Description	For Instructions
Boot a SPARC based system in single-user mode to resolve a minor booting problem, such as a bad `root` shell or incorrect password entry.	Boot a system in single-user mode to resolve an unknown `root` password or similar problem.	"How to Boot in Single-User Mode to Resolve a Bad `root` Shell or Password Problem" on page 76
Boot a SPARC based from media to resolve an unknown `root` password problem.	Boot a system from media, then import and mount the root pool to correct the problem.	"How to Boot From Media to Resolve an Unknown `root` Password" on page 76
Boot a SPARC based system without starting any services.	If a system hangs during the boot process, boot the system without starting any services to troubleshoot the problem.	"How to Boot a System Without Starting Any Services" on page 78
Force a crash dump and reboot of a SPARC based system.	Force a crash dump and reboot of a SPARC based system as a troubleshooting measure.	"How to Force a Crash Dump and Reboot of the System" on page 79
Boot a SPARC based system with the kernel debugger (`kmdb`) enabled.	Boot a SPARC based system with the kernel debugger enabled to interact with the kernel and troubleshoot system problems.	"How to Boot a System With the Kernel Debugger (`kmdb`) Enabled" on page 80

Shutting Down and Booting a SPARC Based System for Recovery Purposes

In the following instances, you must first shut down a system to analyze or troubleshoot booting and other system problems.

- 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.

The procedures that follow describe how to safely shut down and then boot a SPARC based system for recovery purposes.

Stopping and Booting for System Recovery Purposes

You might need to boot the system for recovery purposes. Some of the more common error and recovery scenarios include the following:

- Boot a system in single-user mode to resolve a minor problem, such as correcting the `root` shell entry in the `/etc/passwd` file or changing a NIS server.
- Boot from the installation media or from an install server on the network to recover from a problem that is preventing the system from booting or to recover from a lost `root` password. Resolving a boot configuration problem by importing the root pool, mounting the BE, and fixing the problem.

  On SPARC systems, the `boot net:dhcp` command replaces the `boot net` command that is used in Oracle Solaris 10 releases.

▼ How to Stop a System for Recovery Purposes

1 **Bring the system to `ok` PROM prompt by using the `shutdown` or `init 0` command.**

2 **Synchronize the file systems.**

```
ok sync
```

3 **Type the appropriate `boot` command to start the boot process.**

For more information, see the boot(1M) man page.

4 **Verify that the system was booted to the specified run level.**

```
# who -r
 .       run-level s  May  2 07:39     3      0  S
```

Example 9–1 Powering Off a Service Processor

If you are running Oracle Solaris 11 on an Oracle ILOM service processor, after shutting down the operating system, you must switch from the system console prompt to the service processor prompt. From there, you can stop the service processor, as shown in this example:

```
# shutdown -g0 -i0 -y
# svc.startd: The system is coming down. Please wait.
svc.startd: 91 system services are now being stopped.
Jun 12 19:46:57 wgs41-58 syslogd: going down on signal 15
svc.stard: The system is down.
syncing file systems...done
Program terminated
r)eboot o)k prompt, h)alt?
# o


ok #.
->


-> stop /SYS
Are you sure you want to stop /SYS (y/n)? y
Stopping /SYS

->
```

If you need to perform an immediate shutdown, use the `stop -force -script /SYS` command. Before you type this command, ensure that all data is saved.

Example 9–2 Powering On a Service Processor

The following example shows how to power on the server. You must first be logged in to Oracle ILOM. See http://download.oracle.com/docs/cd/E19166-01/E20792/z40002fe1296006.html#scrolltoc.

If you have a modular system, make sure you are logged into the desired server module.

```
-> start /SYS
Are you sure you want to start /SYS (y/n) ? y
Starting /SYS

->
```

If you do not want to be prompted for a confirmation, use the `start -script /SYS` command.

▼ How to Boot in Single-User Mode to Resolve a Bad root Shell or Password Problem

1. **Bring the system to the ok PROM prompt. See "How to Stop a System for Recovery Purposes" on page 75.**
2. **Boot the system in single-user mode.**

   ```
   ok boot -s
   ```
3. **Correct the shell entry in the `/etc/passwd` file.**

   ```
   # vi /etc/password
   ```
4. **Press `control-d` to reboot the system.**

▼ How to Boot From Media to Resolve an Unknown root Password

1. **Boot the system from the Oracle Solaris media.**
   - **Text installation – Boot from the install media or from the network, then select Option 3 Shell from the text installation screen.**
   - **Automated installation – Use the following command to boot directly from an installation menu that allows you to exit to a shell.**

     ```
     ok boot net:dhcp
     ```

2 **At the shell prompt, import the root pool.**

```
# zpool import -f rpool
```

3 **Create a mount point for the boot environment.**

```
# mkdir /a
```

4 **Mount the boot environment.**

```
# beadm mount solaris-instance|bename /a
```

5 **Set the `TERM` type.**

```
# TERM=vt100
# export TERM
```

6 **Carefully remove the unknown password entry.**

```
# cd /a/etc
# vi shadow
# cd /
```

Note – You must change directories after this step.

---

7 **Update the boot archive.**

```
# bootadm update-archive -R /a
```

8 **Unmount the boot environment.**

```
# beadm umount be-name
```

9 **Halt the system.**

```
# halt
```

10 **Reboot the system in single-user mode, and when prompted for the `root` password, press Return. For example:**

```
ok boot -s
Boot device: /pci@780/pci@0/pci@9/scsi@0/disk@0,0:a File and args: -s
SunOS Release 5.11 Version 11.0 64-bit
Copyright (c) 1983, 2011, Oracle and/or its affiliates. All rights
reserved.
Booting to milestone "milestone/single-user:default".
Hostname: tardis.central
Requesting System Maintenance Mode
SINGLE USER MODE
Enter user name for system maintenance (control-d to bypass): root
Enter root password (control-d to bypass): <Press return>
single-user privilege assigned to root on /dev/console.
Entering System Maintenance Mode
```

11 **Reset the `root` password.**

```
root@system:~# passwd -r files root
New Password: xxxxxx
```

```
Re-enter new Password: xxxxxx
passwd: password successfully changed for root
```

12 **Press `control-d` to reboot the system.**

See Also If the default OS on your system will not will not boot, and you need to boot from an alternate ZFS dataset, see "Booting From a ZFS Boot Environment on SPARC Platforms" on page 62 for further troubleshooting information.

▼ How to Boot a System Without Starting Any Services

If problems with starting services occur, sometimes a system hangs during the boot process. 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 and become the `root` role.**

3 **Enable all services.**

```
# svcadm milestone all
```

4 **When the boot process hangs, determine which services are not running and where the boot process is hanging.**

```
# svcs -a
```

5 **Check for error messages in the log files in `/var/svc/log`.**

6 **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
```

7 **Continue the normal boot process.**

Forcing a Crash Dump and Reboot of a SPARC Based 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 *Oracle Solaris Administration: Common Tasks*.

▼ 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 **Bring the system to the `ok` PROM 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 that the system boots to run level 3.**

The login prompt is displayed when the boot process has finished successfully.

```
hostname console login:
```

Example 9–3 SPARC: Forcing a Crash Dump and Reboot of a System by Using the halt -d Command

This example shows how to force a crash dump and reboot of the system by using the `halt -d` and `boot` commands.

```
# 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 ...

.
.
Rebooting with command: boot
Boot device: /pci@1f,0/pci@1,1/ide@3/disk@0,0:a
File and args: kernel/sparcv9/unix
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 2010
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
.
.
.
```

▼ How to Boot a System With the Kernel Debugger (kmdb) Enabled

This procedure shows how to load the kernel debugger (`kmdb`).

Note – Use the `reboot` command and the `halt` command with the `-d` option if you do not have time to debug the system interactively. Running the `halt` command with the `-d` option requires a manual reboot of the system afterward. However, 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 cleanly, use the `halt` command.

2 Type either `boot kmdb` or `boot -k` to request the loading of the kernel debugger. Press return.

3 Access the kernel debugger.

The method used to enter the debugger depends on the type of console that is used to access the system:

- **If you are using a locally attached keyboard, press Stop-A or L1–A, depending on the type of keyboard.**

- **If you are using a serial console, send a break by using the method that is appropriate for your type of serial console.**

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 9–4 SPARC: Booting a System With the Kernel Debugger (kmdb) Enabled

```
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...
```

Index

B

`banner` command (PROM), 49
boot archives
  managing, 67–72
boot behavior, managing, 47–54
`boot-file` parameter, changing, 53
boot options
  `-L`
    ZFS root file system, 62–65
  `-Z`
    ZFS root file system, 62–65
`bootadm` command, using to manage boot archives, 70–71
booting
  how to troubleshoot, 73–74
  interactively (how to), 26
  to run level S, 24
booting a system, guidelines, 14–15
booting a system from the network, (Task Map), 41–42
booting a system to a specified state, (Task Map), 21–22
booting from ZFS root file system, 62–65

C

changing boot parameters, 53
clean shutdown, 29
creating, administering, and booting from ZFS boot environments, (Task Map), 55–57

D

determining, system's run level, 22

F

fast reboot, how to initiate, 38

H

`halt` command, 30

I

`init` command, description, 29
init states, *See* run levels
initiating a fast reboot of the system, how to, 38

K

keeping a system bootable, 67–68

L

`-L` boot option, booting a ZFS root file system, 62–65

M

managing, boot behavior, 47–54
managing`boot-archive` service, 70–71
modifying boot parameters, (Task Map), 47–48
multiuser level, *See* run level 3

O

Oracle Solaris boot behavior, how to manage, 47–54

P

parameter, changing the `boot-file`, 53
`poweroff` command, 30

R

`reboot` command, 29
rebooting a system, (Task Map), 35–36
recovery shutdown, troubleshooting booting, 73–74
run level
  0 (power-down level), 16
  1 (single-user level), 16
  2 (multiuser level), 16
  3 (multiuser with NFS), 16
    what happens when system is brought to, 17
  6 (reboot level), 17
  default run level, 16
  definition, 16
  s or S (single-user level), 16
  s or S (single-user state)
    booting to, 24
run level 3
  multiuser with NFS
    booting to, 23

S

`shutdown` command
  description, 29
  shutting down a server (how to), 31
shutting down a system
  (Task Map), 27–28
  cleanly with `shutdown` and `init` commands, 29
  for recovery purposes, 73–74
  guidelines, 28–30
single-user level, *See* run level s or S
`sync` command, 79
synchronizing file systems with `sync` command, 79
system shutdown commands, 29
system state
  multiuser with NFS
    booting to, 23
  single-user
    booting to, 24

T

troubleshooting booting
  (Task Map), 73–74
  how to, 73–74

W

`who` command, 22

Z

`-Z` boot option, booting a ZFS root file system, 62–65
ZFS booting, boot options, 62–65
ZFS root file system, booting from, 62–65