Solaris 10 5/09 Installation Guide: Custom JumpStart and Advanced Installations

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

Part No: 820–7014–10
April 2009

090304@21990

Contents

Preface

This book describes how to install and upgrade the Solaris™ Operating System (OS) on both networked and nonnetworked SPARC® and x86 architecture based systems. This book covers using the custom JumpStart installation method and the creation of RAID-1 volumes during installation.

This book does not include instructions about how to set up system hardware or other peripherals.

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

In this document these x86 related terms mean the following:

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

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

Who Should Use This Book

This book is intended for system administrators responsible for installing the Solaris OS. This book provides both of the following types of information.

- Advanced Solaris installation information for enterprise system administrators who manage multiple Solaris machines in a networked environment
- Basic Solaris installation information for system administrators who perform infrequent Solaris installations or upgrades

Related Books

Table P–1 lists documentation for system administrators.

TABLE P–1 Are You a System Administrator Who is Installing Solaris?

Description	Information
Do you need system requirements or high-level planning information? Or want a high-level overview of Solaris ZFS™ installations, booting, Solaris Zones partitioning technology, or creating RAID-1 volumes?	*Solaris 10 5/09 Installation Guide: Planning for Installation and Upgrade*
Do you need to install a single system from DVD or CD media? The Solaris installation program steps you through an installation.	*Solaris 10 5/09 Installation Guide: Basic Installations*
Do you need to upgrade or patch your system with almost no downtime? Save system downtime when upgrading by using Solaris Live Upgrade.	*Solaris 10 5/09 Installation Guide: Solaris Live Upgrade and Upgrade Planning*
Do you need to install a secure installation over the network or Internet? Use WAN boot to install a remote client. Or, do you need to install over the network from a network installation image? The Solaris installation program steps you through an installation.	*Solaris 10 5/09 Installation Guide: Network-Based Installations*
Do you need to install or patch multiple systems quickly? Use Solaris Flash™ software to create a Solaris Flash archive and install a copy of the OS on clone systems.	*Solaris 10 5/09 Installation Guide: Solaris Flash Archives (Creation and Installation)*
Do you need to back up your system?	Chapter 23, "Backing Up and Restoring UFS File Systems (Overview)," in *System Administration Guide: Devices and File Systems*
Do you need troubleshooting information, a list of known problems, or a list of patches for this release?	*Solaris Release Notes*
Do you need to verify that your system works on Solaris?	SPARC: *Solaris Sun Hardware Platform Guide*
Do you need to check on which packages have been added, removed, or changed in this release?	*Solaris Package List*
Do you need to verify that your system and devices work with Solaris SPARC and x86 based systems and other third-party vendors.	Solaris Hardware Compatibility List for x86 Platforms

Documentation, Support, and Training

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

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

Sun Welcomes Your Comments

Sun is interested in improving its documentation and welcomes your comments and suggestions. To share your comments, go to `http://docs.sun.com` and click Feedback.

Typographic Conventions

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

TABLE P–2 Typographic Conventions

Typeface	Meaning	Example
`AaBbCc123`	The names of commands, files, and directories, and onscreen computer output	Edit your `.login` file. Use `ls -a` to list all files. `machine_name% you have mail.`
`AaBbCc123`	What you type, contrasted with onscreen computer output	`machine_name%` **`su`** `Password:`
aabbcc123	Placeholder: replace with a real name or value	The command to remove a file is `rm` *filename*.
AaBbCc123	Book titles, new terms, and terms to be emphasized	Read Chapter 6 in the *User's Guide*. A *cache* is a copy that is stored locally. Do *not* save the file. **Note:** Some emphasized items appear bold online.

Shell Prompts in Command Examples

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

TABLE P–3 Shell Prompts

Shell	Prompt
C shell	`machine_name%`
C shell for superuser	`machine_name#`

TABLE P–3 Shell Prompts *(Continued)*

Shell	Prompt
Bourne shell and Korn shell	$
Bourne shell and Korn shell for superuser	#

PART I

Using Custom JumpStart

This part provides instructions for creating, preparing, and performing custom JumpStart installations.

◆ ◆ ◆ CHAPTER 1

Where to Find Solaris Installation Planning Information

This book provides information on how to use the automated JumpStart installation program to install the Solaris operating system. This book provides all you need to know about installing with the JumpStart program, but a planning book in our collection of installation documentation might be useful to read before you begin preparing for a JumpStart installation. The following references provide useful information before you install your system.

Where to Find Planning and System Requirement Information

The *Solaris 10 5/09 Installation Guide: Planning For Installation and Upgrade* provides system requirements and high-level planning information, such as planning guidelines for file systems, and upgrade planning and much more. This section provides an overview of the chapters for this book.

Chapter Descriptions From the Planning Guide	Reference
This chapter describes new features in the Solaris installation programs.	Chapter 2, "What's New in Solaris Installation," in *Solaris 10 5/09 Installation Guide: Planning for Installation and Upgrade*
This chapter provides you with information about decisions you need to make before you install or upgrade the Solaris OS. Examples are deciding when to use a network installation image or DVD media and descriptions of all the Solaris installation programs.	Chapter 3, "Solaris Installation and Upgrade (Roadmap)," in *Solaris 10 5/09 Installation Guide: Planning for Installation and Upgrade*
This chapter describes system requirements to install or upgrade to the Solaris OS. General guidelines for planning the disk space and default swap space allocation are also provided. Upgrade limitations are also described.	Chapter 4, "System Requirements, Guidelines, and Upgrade (Planning)," in *Solaris 10 5/09 Installation Guide: Planning for Installation and Upgrade*

Chapter Descriptions From the Planning Guide	Reference
This chapter contains checklists to help you gather all of the information that you need to install or upgrade your system. This information is useful, for example, if you are performing an interactive installation. You'll have all the information in the checklist that you'll need to do an interactive installation.	Chapter 5, "Gathering Information Before Installation or Upgrade (Planning)," in *Solaris 10 5/09 Installation Guide: Planning for Installation and Upgrade*
These chapters provide overviews of several technologies that relate to a Solaris OS installation or upgrade. Guidelines and requirements related to these technologies are also included. These chapters include information about ZFS installations, booting, Solaris Zones partitioning technology, and RAID-1 volumes that can be created at installation.	Part II, "Understanding Installations That Relate to ZFS, Booting, Solaris Zones, and RAID-1 Volumes," in *Solaris 10 5/09 Installation Guide: Planning for Installation and Upgrade*

◆ ◆ ◆ CHAPTER 2

Custom JumpStart (Overview)

This chapter provides an introduction and overview to the custom JumpStart installation process.

Note – If you are installing a Solaris ZFS™ root pool, see Chapter 9, "Installing a ZFS Root Pool With JumpStart," for limitations and profile examples.

Custom JumpStart Introduction

The custom JumpStart installation method is a command–line interface that enables you to automatically install or upgrade several systems, based on profiles that you create. The profiles define specific software installation requirements. You can also incorporate shell scripts to include preinstallation and postinstallation tasks. You choose which profile and scripts to use for installation or upgrade. The custom JumpStart installation method installs or upgrades the system, based on the profile and scripts that you select. Also, you can use a `sysidcfg` file to specify configuration information so that the custom JumpStart installation is completely hands-off.

Custom JumpStart Example Scenario

The custom JumpStart process can be described by using an example scenario. In this example scenario, the systems need to be set up with the following parameters:

- Install Solaris on 100 new systems.
- Seventy of the systems are SPARC based systems that are owned by the engineering group and need to be installed as standalone systems with the Solaris OS software group for developers.

- The remaining 30 systems are x86 based, owned by the marketing group and need to be installed as standalone systems with the Solaris OS software group for end users.

First, the system administrator must create a `rules` file and a profile for each group of systems. The `rules` file is a text file that contains a rule for each group of systems or single systems on which you want to install the Solaris software. Each rule distinguishes a group of systems that are based on one or more system attributes. Each rule also links each group to a profile.

A profile is a text file that defines how the Solaris software is to be installed on each system in the group. Both the `rules` file and profile must be located in a JumpStart directory.

For the example scenario, the system administrator creates a `rules` file that contains two different rules, one for the engineering group and another for the marketing group. For each rule, the system's network number is used to distinguish the engineering group from the marketing group.

Each rule also contains a link to an appropriate profile. For example, in the rule for the engineering group, a link is added to the profile, `eng_profile`, which was created for the engineering group. In the rule for the marketing group, a link is added to the profile, `market_profile`, which was created for the marketing group.

You can save the `rules` file and the profiles on a diskette or on a server.

- A profile diskette is required when you want to perform custom JumpStart installations on nonnetworked, standalone systems.
- A profile server is used when you want to perform custom JumpStart installations on networked systems that have access to a server.

After creating the `rules` file and profiles, validate the files with the `check` script. If the `check` script runs successfully, the `rules.ok` file is created. The `rules.ok` is a generated version of the `rules` file that the JumpStart program uses to install the Solaris software.

How the JumpStart Program Installs Solaris Software

After you validate the `rules` file and the profiles, you can begin a custom JumpStart installation. The JumpStart program reads the `rules.ok` file. Then, the JumpStart program searches for the first rule with defined system attributes that match the system on which the JumpStart program is attempting to install the Solaris software. If a match occurs, the JumpStart program uses the profile that is specified in the rule to install the Solaris software on the system.

Figure 2–1 illustrates how a custom JumpStart installation works on a standalone, nonnetworked system. The system administrator initiates the custom JumpStart installation on Pete's system. The JumpStart program accesses the rules files on the diskette in the system's diskette drive. The JumpStart program matches `rule 2` to the system. `rule 2` specifies that the JumpStart program use `Pete's profile` to install the Solaris software. The JumpStart program

reads `Pete's profile` and installs the Solaris software, based on the instructions that the system administrator specified in `Pete's profile`.

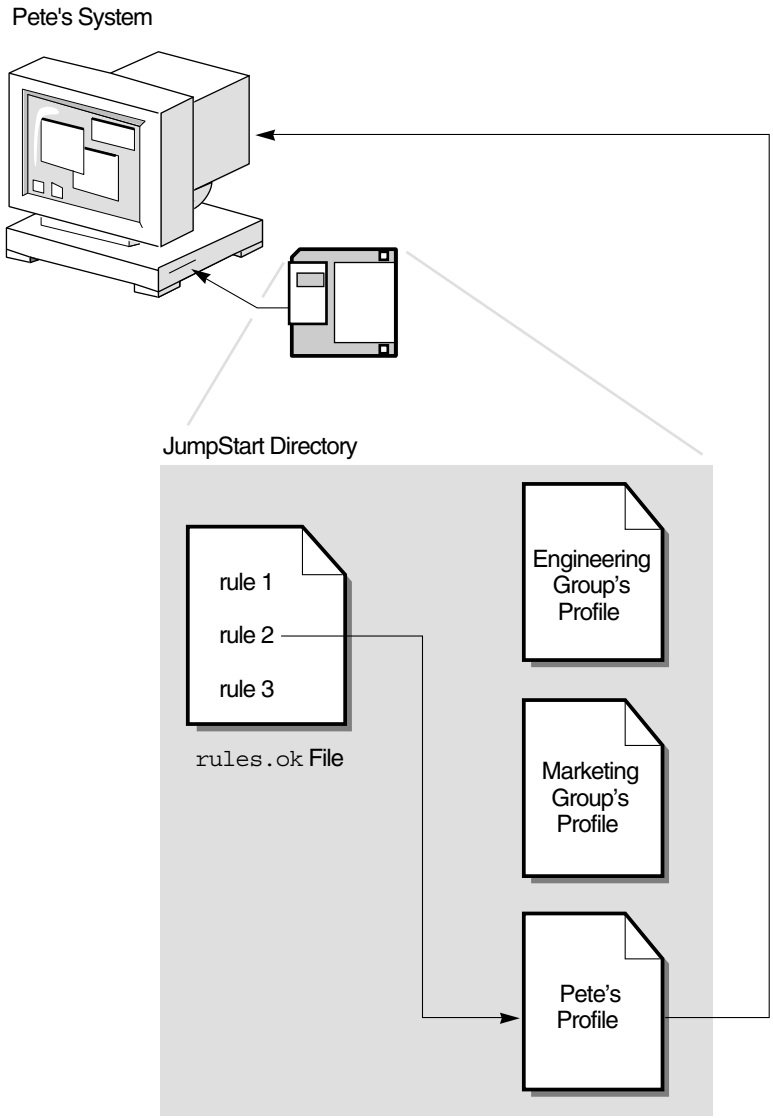

FIGURE 2–1 How a Custom JumpStart Installation Works: nonnetworked Example

Figure 2–2 illustrates how a custom JumpStart installation works with more than one system on a network. Previously, the system administrator set up different profiles and saved the profiles

on a single server. The system administrator initiates the custom JumpStart installation on one of the engineering systems. The JumpStart program accesses the rules files in the `JumpStart/` directory on the server. The JumpStart program matches the engineering system to `rule 1`. `rule 1` specifies that the JumpStart program use `Engineering Group's Profile` to install the Solaris software. The JumpStart program reads `Engineering Group's Profile` and installs the Solaris software, based on the instructions that the system administrator specified in `Engineering Group's Profile`.

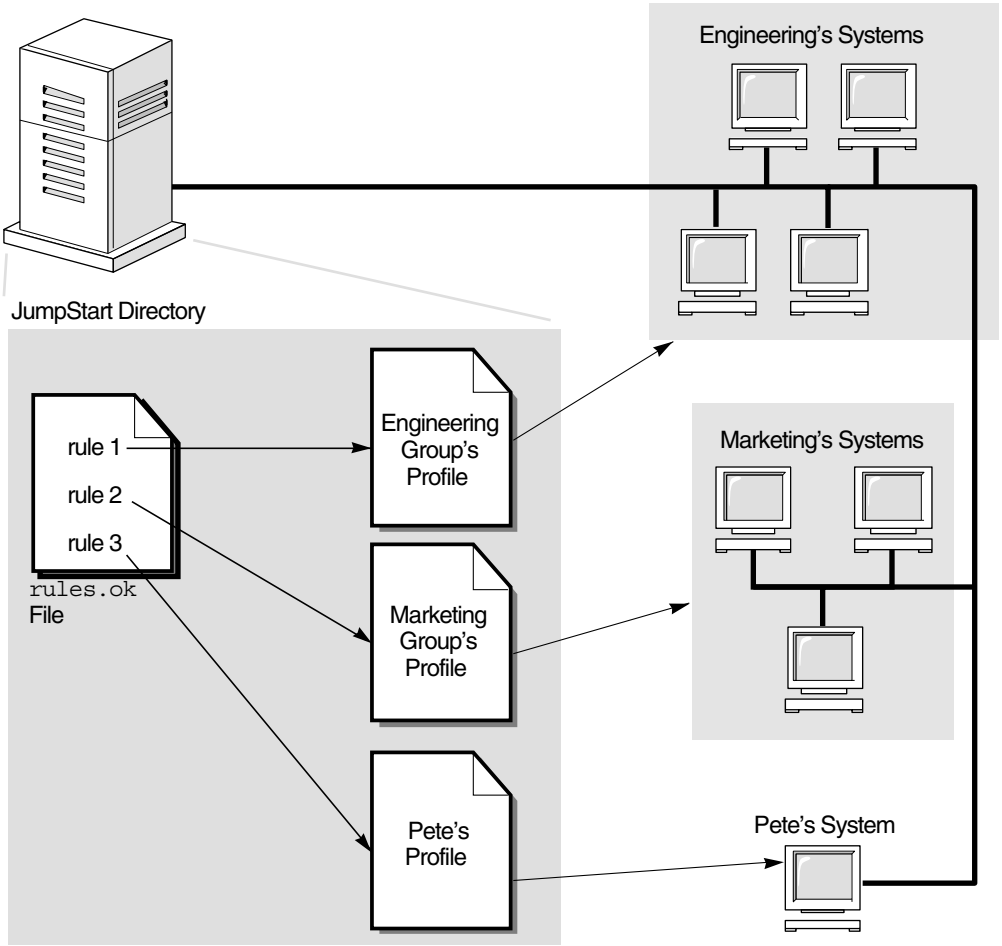

FIGURE 2–2 How a Custom JumpStart Installation Works: Networked Example

Figure 2–3 describes the order in which the JumpStart program searches for custom JumpStart files.

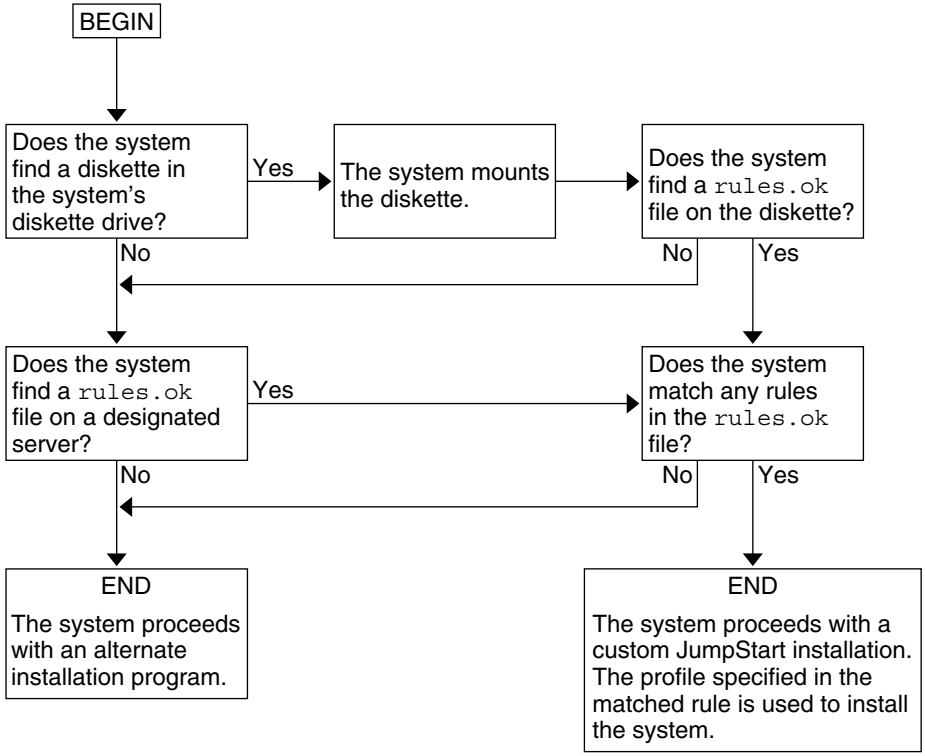

FIGURE 2–3 What Happens During a Custom JumpStart Installation

◆ ◆ ◆ CHAPTER 3

Preparing Custom JumpStart Installations (Tasks)

This chapter provides step-by-step instructions about how to prepare the systems at your site from which and on which you intend to install the Solaris software by using the custom JumpStart installation method.

Note – If you are installing a Solaris ZFS root pool, see Chapter 9, "Installing a ZFS Root Pool With JumpStart," for limitations and profile examples.

Task Map: Preparing Custom JumpStart Installations

TABLE 3–1 Task Map: Preparing Custom JumpStart Installations

Task	Description	For Instructions
Decide how to upgrade the system if a previous version of the Solaris software is installed on the system.	If a previous release of Solaris is installed on the system, you need to determine how to upgrade the system. Ensure that you know what to do before and after you upgrade a system. Planning helps you to create your profiles, begin scripts, and finish scripts.	"Upgrade Planning" in *Solaris 10 5/09 Installation Guide: Planning for Installation and Upgrade*

TABLE 3–1 Task Map: Preparing Custom JumpStart Installations *(Continued)*

Task	Description	For Instructions
Create a JumpStart directory.	**On a server** If you want to perform custom JumpStart installations on systems that are connected to a network, you must create a profile server. The profile server contains a JumpStart directory for the custom JumpStart files. **On a diskette** If you want to perform custom JumpStart installations on systems that are not connected to a network, you must create a profile diskette. A profile diskette contains the custom JumpStart files.	"Creating a Profile Server for Networked Systems" on page 25 "Creating a Profile Diskette for Standalone Systems" on page 29
Add rules to the `rules` file.	After you decide how you want each group of systems or single systems to be installed, create a rule for each group that you want to install. Each rule distinguishes a group, based on one or more system attributes. The rule links each group to a profile.	"Creating the `rules` File" on page 33
Create a profile for every rule.	A profile is a text file that defines how to install the Solaris software, for example, which software group to install on a system. Every rule specifies a profile to define how a system is to be installed with the Solaris software when the rule is matched. You usually create a different profile for every rule. However, the same profile can be used in more than one rule.	"Creating a Profile" on page 36
(Optional) Test the profiles.	After you create a profile, use the `pfinstall(1M)` command to test the profile before you use the profile to install or upgrade a system.	"Testing a Profile" on page 49
Validate the `rules` file.	The `rules.ok` file is a generated version of the `rules` file that the JumpStart program uses to match the system to be installed with a profile. You must use the `check` script to validate the `rules` file.	"Validating the `rules` File" on page 54

Creating a Profile Server for Networked Systems

When setting up custom JumpStart installations for systems on the network, you need to create a directory on a server that is called a JumpStart directory. The JumpStart directory contains all of the essential custom JumpStart files, for example, the `rules` file, `rules.ok` file, and profiles. You must save the JumpStart directory in the root (`/`) directory of the profile server.

The server that contains a JumpStart directory is called a profile server. A profile server can be the same system as an install server or a boot server, or the server can be a completely different server. A profile server can provide custom JumpStart files for different platforms. For example, an x86 server can provide custom JumpStart files for both SPARC based systems and x86 based systems.

Note – After you create a profile server, you must allow systems to access the server. For detailed instructions, see "To Allow All Systems Access to the Profile Server" on page 27.

▼ To Create a JumpStart Directory on a Server

Note – This procedure assumes that the system is running *Volume Manager*. If you are not using Volume Manager to manage discs, refer to *System Administration Guide: Devices and File Systems* for detailed information about managing removable media without Volume Manager.

---

1. **Locate the server on which you want to create the JumpStart directory.**

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

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

3. **Create the JumpStart directory anywhere on the server.**

   ```
   # mkdir -m 755 jumpstart_dir_path
   ```

   In the command, *jumpstart_dir_path* is the absolute path of the JumpStart directory.

   For example, the following command creates a directory that is called `jumpstart` in the root (`/`) directory and sets the permissions to 755:

   ```
   # mkdir -m 755 /jumpstart
   ```

4. **Edit the `/etc/dfs/dfstab` file by adding the following entry.**

   ```
   share -F nfs -o ro,anon=0 jumpstart_dir_path
   ```

For example, the following entry shares the `/jumpstart` directory:

```
share -F nfs -o ro,anon=0 /jumpstart
```

5 **Type `shareall` and press Enter.**

6 **Determine if you want to copy examples of custom JumpStart files to your JumpStart directory.**

- If no, go to Step 9.
- If yes, use the following decision table to determine what to do next.

Example Locations	Instructions
The Solaris Operating System DVD or the Solaris Software - 1 CD for your platform	Insert the Solaris Operating System DVD or the Solaris Software - 1 CD into the server's CD-ROM drive. Volume Manager automatically mounts the CD or DVD.
An image of the Solaris Operating System DVD or the Solaris Software - 1 CD for your platform on a local disk	Change directory to the location of the Solaris Operating System DVD or the Solaris Software - 1 image. For example, type the following command: `cd /export/install`

7 **Copy the example custom JumpStart files into the JumpStart directory on the profile server.**

`#` **`cp -r`** *media_path*`/Solaris_10/Misc/jumpstart_sample/*` *jumpstart_dir_path*

media_path	The path to the CD, DVD, or image on the local disk
jumpstart_dir_path	The path on the profile server where you are placing the example custom JumpStart files

For example, the following command copies the `jumpstart_sample` directory into the `/jumpstart` directory on the profile server:

```
cp -r /cdrom/cdrom0/Solaris_10/Misc/jumpstart_sample/* /jumpstart
```

8 **Update the example JumpStart files so that the files work in your environment.**

9 **Ensure that `root` owns the JumpStart directory and that the permissions are set to 755.**

10 **Allow systems on the network to access the profile server.**

For detailed instructions, see "To Allow All Systems Access to the Profile Server" on page 27.

Allowing All Systems Access to the Profile Server

When you create a profile server, you must ensure that systems can access the JumpStart directory on the profile server during a custom JumpStart installation. Use one of the following ways to ensure access.

Command or File	Providing Access	Instructions
`add_install_client` command	Each time that you add a system for network installation, use the `-c` option with the `add_install_client` command to specify the profile server. **Note –** If you are not using NFS, then you must use another means to provide access. ■ **For SPARC based systems**, use the `boot` command ■ **For x86 based systems**, edit the GRUB menu	■ For DVD media, see "Adding Systems to Be Installed From the Network With a DVD Image" in *Solaris 10 5/09 Installation Guide: Network-Based Installations* ■ For CD media, see "Adding Systems to Be Installed From the Network With a CD Image" in *Solaris 10 5/09 Installation Guide: Network-Based Installations*
Specify the location of the JumpStart directory when you boot the system	■ **For SPARC based systems**, use the `boot` command to boot the system. Specify the location of the JumpStart directory on the profile server when you boot the system. You must compress the custom JumpStart configuration files into one file. Then, save the compressed configuration file on an HTTP or HTTPS server.	■ "Creating a Compressed Configuration File" on page 66 ■ Step 5 in "SPARC: To Perform an Installation or Upgrade With the Custom JumpStart Program" on page 86
	■ **For x86 based systems**, specify the location of the JumpStart directory on the profile server when you boot the system by editing the boot entry on the GRUB menu. You must compress the custom JumpStart configuration files into one file. Then, save the compressed configuration file on an HTTP or HTTPS server. When you edit the GRUB menu entry, specify the location of the compressed file.	■ "Creating a Compressed Configuration File" on page 66 ■ "x86: Performing a Custom JumpStart Installation by Editing the GRUB Boot Command" on page 92
`/etc/bootparams` file	Add a wildcard in the `/etc/bootparams` file.	"To Allow All Systems Access to the Profile Server" on page 27

▼ To Allow All Systems Access to the Profile Server

Use the following procedure only if you store network installation information in the following places:

- In the `/etc/bootparams` file.

- In the naming service `bootparams` database. To update the `bootparams` database, add the entry that is shown in Step 3.

If you use the following procedure, the systems must be of the same type, such as all SPARC systems.

Do not use this procedure under the following conditions:

- If you save the JumpStart directory on a diskette.
- If you specify the location of the profile server when you boot the system. If you have systems of different architectures, you must specify the location of the profile server when you boot the system

If you have the above conditions, use the SPARC `boot` command or use the x86 GRUB menu.

Note – You also can store network installation information on a DHCP server.

- **For SPARC based systems**, you use the `add_install_client` command and the `-d` option to specify that the custom JumpStart program use the DHCP server. Or you use the `boot` command with the `dhcp` option to specify that the custom JumpStart program use the DHCP server. For instructions about using this option, see "SPARC: Command Reference for the `boot` Command" on page 87.
- **For x86 based systems**, you use dhcp in one of the following ways:
  - If you use an install server, use the `add_install_client` command and the `-d` option to specify that the custom JumpStart program use the DHCP server with PXE.
  - You can edit the GRUB entry on the GRUB menu and add the dhcp option. For instructions about editing the GRUB entry, see "x86: Performing a Custom JumpStart Installation by Editing the GRUB Boot Command" on page 92

---

1 On the installation or boot server, log in as superuser.

2 Use a text editor to open `/etc/bootparams`.

3 Add this entry.

```
* install_config=server:jumpstart_dir_path
```

*	A wildcard character that specifies that all systems have access
server	The host name of the profile server where the JumpStart directory is located
jumpstart_dir_path	The absolute path of the JumpStart directory

For example, the following entry enables all systems to access the `/jumpstart` directory on the profile server that is named `sherlock`:

```
* install_config=sherlock:/jumpstart
```

Caution – Use of this procedure might produce the following error message when an installation client is booted:

```
WARNING: getfile: RPC failed: error 5: (RPC Timed out).
```

"Booting From the Network, Error Messages" on page 174 contains details about this error message.

All systems can now access the profile server.

Creating a Profile Diskette for Standalone Systems

A diskette that contains a JumpStart directory is called a profile diskette. A system that is not connected to the network does not have access to a profile server. As a result, you must create a JumpStart directory on a diskette if a system is not connected to a network. The system on which you create a profile diskette must have a diskette drive.

The JumpStart directory contains all of the essential custom JumpStart files, for example, the `rules` file, `rules.ok` file, and profiles. You must save the JumpStart directory in the root (`/`) directory of the profile diskette.

See one of the following procedures:

- "SPARC: To Create a Profile Diskette" on page 29
- "x86: To Create a Profile Diskette With GRUB" on page 31

▼ SPARC: To Create a Profile Diskette

Note – This procedure assumes that the system is running Volume Manager. If you are not using Volume Manager to manage diskettes, CDs, and DVDs, refer to *System Administration Guide: Devices and File Systems* for detailed information about managing removable media without Volume Manager.

1 Locate a SPARC based system to which a diskette drive is attached.

2 Become superuser or assume an equivalent role.

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

3 Insert a blank diskette or a diskette that can be overwritten in the diskette drive.

4 Mount the diskette.

```
# volcheck
```

5 Determine if the diskette contains a UNIX file system (UFS).

Examine the contents of the file `/etc/mnttab` on the system for an entry such as the following:

```
/vol/dev/diskette0/scrap  /floppy/scrap  ufs  suid,rw,largefiles,dev=1740008  927147040
```

- If the entry exists, go to Step 7.
- If the entry does not exist, go to the next step.

6 Create a UFS on the diskette.

```
# newfs /vol/dev/aliases/floppy0
```

7 Determine if you want to copy examples of custom JumpStart files to your JumpStart directory.

- If no, go to Step 10.
- If yes, use the following decision table to determine what to do next.

Example Locations	Instructions
The Solaris Operating System for SPARC Platforms DVD or the Solaris Software for SPARC Platforms - 1 CD	Insert the Solaris Operating System for SPARC Platforms DVD or the Solaris Software for SPARC Platforms - 1 CD into the server's CD-ROM drive. Volume Manager automatically mounts the CD or DVD.
An image of the Solaris Operating System for SPARC Platforms DVD or the Solaris Software for SPARC Platforms - 1 CD on a local disk	Change the directory to the location of the Solaris Operating System for SPARC Platforms DVD or the Solaris Software for SPARC Platforms - 1 CD image. For example, type the following command: `cd /export/install`

8 Copy the example custom JumpStart files into the JumpStart directory on the profile diskette.

```
# cp -r media_path/Solaris_10/Misc/jumpstart_sample/* jumpstart_dir_path
```

media_path The path to the CD, DVD, or image on the local disk

jumpstart_dir_path The path to the profile diskette where you want to place the example custom JumpStart files

Note – You must place all custom JumpStart installation files in the root (`/`) directory on the diskette.

For example, the following command copies the contents of `jumpstart_sample` on the Solaris Software for SPARC Platforms - 1 CD to the root (`/`) directory on a profile diskette that is named `scrap`:

```
cp -r /cdrom/cdrom0/Solaris_10/Misc/jumpstart_sample/* /floppy/scrap
```

9 **Update the example JumpStart files on the profile diskette so that the files work in your environment.**

10 **Ensure that `root` owns the JumpStart directory and that permissions are set to 755.**

11 **Eject the diskette.**

```
# eject floppy
```

You have completed the creation of a profile diskette. You can now update the `rules` file and create profiles on the profile diskette to perform custom JumpStart installations. To continue, go to "Creating the `rules` File" on page 33.

▼ x86: To Create a Profile Diskette With GRUB

Use this procedure to create a profile diskette with GRUB. A GRUB menu is provided during the installation procedure that enables the boot process. The GRUB menu replaces the Solaris Device Configuration Assistant that might have been needed to boot a system in past releases.

Note – This procedure assumes that the system is running Volume Manager. If you are not using Volume Manager to manage diskettes, CDs, and DVDs, refer to *System Administration Guide: Devices and File Systems* for detailed information about managing removable media without Volume Manager.

1 **Locate an x86 based system to which a diskette drive is attached.**

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

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

3 **Insert a blank diskette or a diskette that can be overwritten into the diskette drive.**

4 Mount the diskette.

```
# volcheck
```

5 Determine if you want to copy examples of custom JumpStart files to your JumpStart directory.

- If no, go to Step 8.
- If yes, use the following decision table to determine what to do next.

Example Locations	Instructions
The Solaris Operating System for x86 Platforms DVD or the Solaris Software for x86 Platforms - 1 CD	Insert the Solaris Operating System for x86 Platforms DVD or the Solaris Software for x86 Platforms - 1 CD into the server's CD-ROM drive. Volume Manager automatically mounts the DVD or CD.
An image of the Solaris Operating System for x86 Platforms DVD or the Solaris Software for x86 Platforms - 1 CD on a local disk	Change directory to the location of the Solaris Operating System for x86 Platforms DVD or the Solaris Software for x86 Platforms - 1 CD image. For example, type the following: `cd /export/install`

6 Copy the example custom JumpStart files into the JumpStart directory on the profile diskette.

```
# cp -r media_path/Solaris_10/Misc/jumpstart_sample/* jumpstart_dir_path
```

media_path	The path to the CD, DVD, or image on the local disk
jumpstart_dir_path	The path to the profile diskette where you want to place the example custom JumpStart files

Note – You must place all custom JumpStart installation files in the root (`/`) directory on the profile diskette.

For example, the following command copies the contents of `jumpstart_sample` on the Solaris Software for x86 Platforms - 1 CD to the root (`/`) directory on a profile diskette that is named `scrap`:

```
cp -r /cdrom/cdrom0/Solaris_10/Misc/jumpstart_sample/* /floppy/scrap
```

7 Update the example JumpStart files on the profile diskette so that the files work in your environment.

8 Ensure that `root` owns the JumpStart directory and that permissions are set to 755.

9. **Eject the diskette by clicking Eject Disk in the File Manager window or by typing `eject floppy` on the command line.**
10. **In the Removable Media Manager dialog box, click OK.**
11. **Manually eject the diskette.**

See Also You have completed the creation of a profile diskette. Now you can update the `rules` file and create profiles on the profile diskette to perform custom JumpStart installations. To continue, go to "Creating the `rules` File" on page 33.

Creating the `rules` File

The `rules` file is a text file that contains a rule for each group of systems on which you want to install the Solaris OS. Each rule distinguishes a group of systems that are based on one or more system attributes. Each rule also links each group to a profile. A profile is a text file that defines how the Solaris software is to be installed on each system in the group. For example, the following rule specifies that the JumpStart program use the information in the `basic_prof` profile to install any system with the `sun4u` platform group.

```
karch sun4u - basic_prof -
```

The `rules` file is used to create the `rules.ok` file, which is required for custom JumpStart installations.

Note – If you set up the JumpStart directory by using the procedures in "Creating a Profile Diskette for Standalone Systems" on page 29 or "Creating a Profile Server for Networked Systems" on page 25, an example `rules` file is already located in the JumpStart directory. The sample `rules` file contains documentation and some example rules. If you use the sample `rules` file, ensure that you comment out the example rules you do not intend to use.

Syntax of the `rules` File

The `rules` file must have the following attributes:

- The file must be assigned the name `rules`.
- The file must contain at least one rule.

The `rules` file can contain any of the following:

- Commented text

Any text that is included after the # symbol on a line is treated by JumpStart as commented text. If a line begins with the # symbol, the entire line is treated as a comment.

- One or more blank lines
- One or more multiline rules

  To continue a single rule onto a new line, include a backslash character (\) just before pressing Return.

▼ To Create a rules File

1 **Use a text editor to create a text file that is named rules. Or, open the sample rules file in the JumpStart directory that you created.**

2 **Add a rule in the rules file for each group of systems on which you want to install the Solaris software.**

For a list of rules file keywords and values, see "Rule Keywords and Values" on page 107.

A rule within a rules file must adhere to the following syntax:

!*rule_keyword rule_value* && !*rule_keyword rule_value* ... *begin profile finish*

!	A symbol that is used before a keyword to indicate negation.
rule_keyword	A predefined lexical unit or word that describes a general system attribute, such as host name, hostname, or memory size, memsize. *rule_keyword* is used with the rule value to match a system with the same attribute to a profile. For the list of rule keywords, see "Rule Keywords and Values" on page 107.
rule_value	A value that provides the specific system attribute for the corresponding rule keyword. Rule values are described in "Rule Keywords and Values" on page 107.
&&	A symbol you must use to join rule keyword and rule value pairs in the same rule (a logical AND). During a custom JumpStart installation, a system must match every pair in the rule before the rule matches.
begin	The name of an optional Bourne shell script that can be executed before the installation begins. If no begin script exists, you must type a minus sign (-) in this field. All begin scripts must be located in the JumpStart directory. Information about how to create begin scripts is presented in "Creating Begin Scripts" on page 57.
profile	The name of a text file that defines how the Solaris software is to be installed on the system when a system matches the rule. The information in a profile

consists of profile keywords and their corresponding profile values. All profiles must be located in the JumpStart directory.

Note – Optional ways to use the profile field are described in "Using a Site-Specific Installation Program" on page 73 and "Creating Derived Profiles With a Begin Script" on page 58.

finish The name of an optional Bourne shell script that can be executed after the installation is completed. If no finish script exists, you must type a minus sign (-) in this field. All finish scripts must be located in the JumpStart directory.

Information about how to create finish scripts is presented in "Creating Finish Scripts" on page 59.

At the minimum, each rule must contain the following:

- A keyword, a value, and a corresponding profile
- A minus sign (-) in the *begin* and *finish* fields if no begin or finish scripts are specified

3 Save the `rules` file in the JumpStart directory.

4 Ensure that `root` owns the `rules` file and that the permissions are set to 644.

`rules` File Example

The following example shows several example rules in a `rules` file. Each line has a rule keyword and a valid value for that keyword. The JumpStart program scans the `rules` file from top to bottom.

When the JumpStart program matches a rule keyword and value with a known system, the JumpStart program installs the Solaris software that is specified by the profile that is listed in the profile field.

For a complete list of `rules` file limitations, see "Syntax of the rules File" on page 33.

EXAMPLE 3–1 `rule` File

```
# rule keywords and rule values       begin script       profile       finish script
# -----------------------------       ------------       --------      -------------
  hostname eng-1                      -                  basic_prof    -
  network 192.168.255.255 && !model \
'SUNW,Sun-Blade-100'                  -                  net_prof      -
  model SUNW,SPARCstation-LX          -                  lx_prof       complete
```

EXAMPLE 3–1 `rule` File *(Continued)*

```
network 192.168.2.0 && karch i86pc  setup              x86_prof      done
memsize 64-128 && arch i386          -                 prog_prof     -
any   -                              -                 generic_prof  -
```

The following list describes some of the keywords and values from this example.

`hostname` The rule matches if the system's host name is `eng-1`. The `basic_prof` profile is used to install the Solaris software on the system that matches the rule.

`network` The rule matches if the system is on subnet 192.168.255.255 and if the system is *not* a Sun Blade ™ 100 (`SUNW,Sun-Blade-100`). The `net_prof` profile is used to install the Solaris software on systems that match this rule. This rule also provides an example of continuing a single rule onto a new line by using the backslash character (\).

`model` The rule matches if the system is a SPARCstation LX. The `lx_prof` profile and the `complete` finish script are used to install the Solaris software on systems that match this rule.

`network` The rule matches if the system is on subnet 192.168.2.0 and is an x86 based system. The `setup` begin script, the `x864u_prof` profile, and the `done` finish script are used to install the Solaris software on systems that match the rule.

`memsize` The rule matches if the system has between 64 and 128 Mbytes of memory and is an x86 based system. The `prog_prof` profile is used to install the Solaris software on systems that match the rule.

`any` The rule matches any system that did not match the previous rules. The `generic_prof` profile is used to install the Solaris software on systems that match the rule. If `any` is used, it should always be the last rule in the `rules` file.

Creating a Profile

A profile is a text file that defines how to install the Solaris software on a system. A profile defines elements of the installation, for example, the software group to install. Every rule specifies a profile that defines how a system is to be installed. You can create different profiles for every rule or the same profile can be used in more than one rule.

A profile consists of one or more profile keywords and their values. Each profile keyword is a command that controls one aspect of how the JumpStart program is to install the Solaris software on a system. For example, the following profile keyword and value specify that the JumpStart program install the system as a server:

```
system_type  server
```

Note – Sample profiles are already located in the JumpStart directory if you created the JumpStart directory by using either of these procedures:

- "Creating a Profile Server for Networked Systems" on page 25
- "Creating a Profile Diskette for Standalone Systems" on page 29

Syntax of Profiles

A profile must contain the following:

- The `install_type` profile keyword as the first entry
- One keyword per line
- The `root_device` keyword if the systems that are being upgraded by the profile contain more than one root (`/`) file system that can be upgraded

A profile can contain the following:

- Commented text

  Any text that is included after the # symbol on a line is treated by the JumpStart program as commented text. If a line begins with the # symbol, the entire line is treated as a comment.
- One or more blank lines

▼ To Create a Profile

1 **Use a text editor to create a text file. Name the file descriptively. Or, open a sample profile in the JumpStart directory that you created.**

Note – Ensure that the name of the profile reflects how you intend to use the profile to install the Solaris software on a system. For example, you might name the profiles `basic_install`, `eng_profile`, or `user_profile`.

2 **Add profile keywords and values to the profile.**

For a list of profile keywords and values, see "Profile Keywords and Values" on page 111.

Note – Profile keywords and their values are case sensitive.

3 **Save the profile in the JumpStart directory.**

4 **Ensure that `root` owns the profile and that the permissions are set to 644.**

5 **Test the profile (optional).**

"Testing a Profile" on page 49 contains information about testing profiles.

Profile Examples

The following examples of profiles show how to use different profile keywords and profile values to control how the Solaris software is installed on a system. "Profile Keywords and Values" on page 111 contains a description of profile keywords and values.

Note – If you are installing a Solaris ZFS™ root pool, see Chapter 9, "Installing a ZFS Root Pool With JumpStart," for limitations and profile examples.

EXAMPLE 3–2 Mounting Remote File Systems and Adding and Deleting Packages

```
# profile keywords        profile values
# -----------------       -----------------
  install_type            initial_install
  system_type             standalone
  partitioning            default
  filesys                 any 512 swap   # specify size of /swap
  cluster                 SUNWCprog
  package                 SUNWman delete
  cluster                 SUNWCacc
```

The following list describes some of the keywords and values from this example.

`install_type` The `install_type` keyword is required in every profile.

`system_type` The `system_type` keyword defines that the system is to be installed as a standalone system.

`partitioning` The file system slices are determined by the software to be installed with the value `default`. The size of `swap` is set to 512 Mbytes and is installed on any disk, value `any`.

`cluster` The Developer Solaris Software Group, `SUNWCprog`, is installed on the system.

`package` If the standard man pages are mounted from the file server, `s_ref`, on the network, the man page packages are not to be installed on the system. The packages that contain the System Accounting utilities are selected to be installed on the system.

EXAMPLE 3–3 Mounting Remote File Systems and Adding a Third-Party Package

```
# profile keywords        profile values
# ----------------        ----------------
  install_type            initial_install
  system_type             standalone
  partitioning            default
  filesys                 any 512 swap   # specify size of /swap
  cluster                 SUNWCprog
  cluster                 SUNWCacc
  package                 apache_server  \
                           http://package.central/packages/apache timeout 5
```

The following list describes some of the keywords and values from this example.

`install_type` The `install_type` keyword is required in every profile.

`system_type` The `system_type` keyword defines that the system is to be installed as a standalone system.

`partitioning` The file system slices are determined by the software to be installed with the value `default`. The size of `swap` is set to 512 Mbytes and is installed on any disk, value `any`.

`cluster` The Developer Solaris Software Group, `SUNWCprog`, is installed on the system.

`package` A third-party package is installed on the system located on an HTTP server.

EXAMPLE 3–4 Specifying Where to Install File Systems

```
# profile keywords        profile values
# ----------------        -------------------
  install_type            initial_install
  system_type             standalone
  partitioning            explicit
  filesys                 c0t0d0s0 auto /
  filesys                 c0t3d0s1 auto swap
  filesys                 any auto usr
  cluster                 SUNWCall
```

The following list describes some of the keywords and values from this example.

`partitioning` The file system slices are determined by the `filesys` keywords, value `explicit`. The size of root (`/`) is based on the selected software, value `auto`, and is installed on `c0t0d0s0`. The size of `swap` is set to the necessary size and

EXAMPLE 3–4 Specifying Where to Install File Systems *(Continued)*

	is installed on `c0t3d0s1`. `usr` is based on the selected software and the installation program determines where `usr` is installed, based on the value `any`.
`cluster`	The Entire Solaris Software Group, `SUNWCall`, is installed on the system.

EXAMPLE 3–5 Upgrading and Installing Patches

```
# profile keywords         profile values
# ----------------         -------------------
  install_type             upgrade
  root_device              c0t3d0s2
  backup_media             remote_filesystem timber:/export/scratch
  package                  SUNWbcp delete
  package                  SUNWxwman add
  cluster                  SUNWCacc add
  patch                    patch_list nfs://patch_master/Solaris_10/patches \
                           retry 5
  locale                   de
```

The following list describes some of the keywords and values from this example.

`install_type`	The profile upgrades a system by reallocating disk space. In this example, disk space must be reallocated because some file systems on the system did not have enough space for the upgrade.
`root_device`	The root file system on `c0t3d0s2` is upgraded.
`backup_media`	A remote system that is named `timber` is to be used to back up data during the disk space reallocation. For more backup-media keyword values, see "`backup_media` Profile Keyword" on page 119.
`package`	The binary compatibility package, `SUNWbcp`, is not installed on the system after the upgrade.
`package`	The code ensures that the X Window System man pages and the System Accounting Utilities are to be installed if they are not already installed on the system. All packages already on the system are automatically upgraded.
`patch`	A list of patches that are installed with the upgrade. The patch list is located on an NFS server named `patch_master` under the directories `Solaris_10/patches`. In case of a mount failure, the NFS mount is tried five times.
`locale`	The German localization packages are to be installed on the system.

EXAMPLE 3–6 Reallocating Disk Space for an Upgrade

```
# profile keywords         profile values
# ----------------         -------------------
  install_type             upgrade
  root_device              c0t3d0s2
  backup_media             remote_filesystem timber:/export/scratch
  layout_constraint        c0t3d0s2 changeable 100
  layout_constraint        c0t3d0s4 changeable
  layout_constraint        c0t3d0s5 movable
  package                  SUNWbcp delete
  package                  SUNWxwman add
  cluster                  SUNWCacc add
  locale                   de
```

The following list describes some of the keywords and values from this example.

`install_type` — The profile upgrades a system by reallocating disk space. In this example, disk space must be reallocated because some file systems on the system did not have enough space for the upgrade.

`root_device` — The root file system on `c0t3d0s2` is upgraded.

`backup_media` — A remote system that is named `timber` is to be used to back up data during the disk space reallocation. For more backup-media keyword values, see "`backup_media` Profile Keyword" on page 119.

`layout_constraint` — The `layout_constraint` keywords designate that auto-layout can perform the following when auto-layout attempts to reallocate disk space for the upgrade.

- Change slices 2 and 4. The slices can be moved to another location and the size can be changed.
- Move slice 5. The slice can be moved to another location but its size cannot change.

`package` — The binary compatibility package, `SUNWbcp`, is not installed on the system after the upgrade.

`package` — The code ensures that the X Window System man pages and the System Accounting Utilities are to be installed if they are not already installed on the system. All packages already on the system are automatically upgraded.

`locale` — The German localization packages are to be installed on the system.

EXAMPLE 3–7 Retrieving a Solaris Flash Archive From an HTTP Server

In the following example, the profile indicates that the custom JumpStart program retrieves the Solaris Flash archive from an HTTP server.

```
# profile keywords        profile values
# ----------------        -------------------
install_type              flash_install
archive_location          http://192.168.255.255/flasharchive/solarisarchive
partitioning              explicit
filesys                   c0t1d0s0 4000 /
filesys                   c0t1d0s1 512 swap
filesys                   c0t1d0s7 free /export/home
```

The following list describes some of the keywords and values from this example.

`install_type` The profile installs a Solaris Flash archive on the clone system. All files are overwritten as in an initial installation.

`archive_location` The Solaris Flash archive is retrieved from an HTTP server.

`partitioning` The file system slices are determined by the `filesys` keywords, value `explicit`. The size of root (`/`) is based on the size of the Solaris Flash archive. The root file system is installed on `c0t1d0s0`. The size of `swap` is set to the necessary size and is installed on `c0t1d0s1`. `/export/home` is based on the remaining disk space. `/export/home` is installed on `c0t1d0s7`.

EXAMPLE 3–8 Retrieving a Solaris Flash Archive From a Secure HTTP Server

In the following example, the profile indicates that the custom JumpStart program retrieves the Solaris Flash archive from a secure HTTP server.

```
# profile keywords        profile values
# ----------------        -------------------
install_type              flash_install
archive_location          https://192.168.255.255/solarisupdate.flar
partitioning              explicit
filesys                   c0t1d0s0 4000 /
filesys                   c0t1d0s1 512 swap
filesys                   c0t1d0s7 free /export/home
```

The following list describes some of the keywords and values from this example.

`install_type` The profile installs a Solaris Flash archive on the clone system. All files are overwritten as in an initial installation.

EXAMPLE 3–8 Retrieving a Solaris Flash Archive From a Secure HTTP Server *(Continued)*

`archive_location`	The compressed Solaris Flash archive is retrieved from a secure HTTP server.
`partitioning`	The file system slices are determined by the `filesys` keywords, value `explicit`. The size of root (`/`) is based on the size of the Solaris Flash archive. The size of `swap` is set to the necessary size and is installed on `c0t1d0s1`. `/export/home` is based on the remaining disk space. `/export/home` is installed on `c0t1d0s7`.

EXAMPLE 3–9 Retrieving a Solaris Flash Archive and Installing a Third-Party Package

In the following example, the profile indicates that the custom JumpStart program retrieves the Solaris Flash archive from an HTTP server.

```
# profile keywords         profile values
# ----------------         ------------------
install_type               flash_install
archive_location           http://192.168.255.255/flasharchive/solarisarchive
partitioning               explicit
filesys                    c0t1d0s0 4000 /
filesys                    c0t1d0s1 512 swap
filesys                    c0t1d0s7 free /export/home
package                    SUNWnew http://192.168.254.255/Solaris_10 timeout 5
```

The following list describes some of the keywords and values from this example.

`install_type`	The profile installs a Solaris Flash archive on the clone system. All files are overwritten as in an initial installation.
`archive_location`	The Solaris Flash archive is retrieved from an HTTP server.
`partitioning`	The file system slices are determined by the `filesys` keywords, value `explicit`. The size of root (`/`) is based on the size of the Solaris Flash archive. The root file system is installed on `c0t1d0s0`. The size of `swap` is set to the necessary size and is installed on `c0t1d0s1`. `/export/home` is based on the remaining disk space. `/export/home` is installed on `c0t1d0s7`.
`package`	The SUNWnew package is added from the `Solaris_10` directory from the HTTP server `192.168.254.255`.

EXAMPLE 3–10 Retrieving a Solaris Flash Differential Archive From an NFS Server

In the following example, the profile indicates that the custom JumpStart program retrieves the Solaris Flash archive from an NFS server. The `flash_update` keyword indicates that this is a differential archive. A differential archive installs only the differences between two system images.

```
# profile keywords         profile values
# ----------------         -------------------
install_type               flash_update
archive_location           nfs installserver:/export/solaris/flasharchive \
                           /solarisdiffarchive
no_master_check
```

The following list describes some of the keywords and values from this example.

`install_type` The profile installs a Solaris Flash differential archive on the clone system. Only files that are specified by the archive are installed.

`archive_location` The Solaris Flash archive is retrieved from an NFS server.

`no_master_check` The clone system is not checked for a valid system image. A valid system image would have been built from the original master system.

EXAMPLE 3–11 Creating an Empty Boot Environment

In the following example, the profile indicates that the custom JumpStart program creates an empty boot environment. An empty boot environment contains no file systems and no copy from the current boot environment occurs. The boot environment can be populated later with a Solaris Flash archive and then activated.

```
# profile keywords        profile values
# ----------------        -------------------
  install_type            initial_install
  system_type             standalone
  partitioning            explicit
  filesys                 c0t0d0s0 auto /
  filesys                 c0t3d0s1 auto swap
  filesys                 any auto usr
  cluster                 SUNWCall
  bootenv createbe bename second_BE \
  filesystem /:/dev/dsk/c0t1d0s0:ufs \
  filesystem -:/dev/dsk/c0t1d0s0:swap \
  filesystem /export:shared:ufs
```

The following list describes some of the keywords and values from this example.

EXAMPLE 3–11 Creating an Empty Boot Environment *(Continued)*

`partitioning`	The file system slices are determined by the `filesys` keywords, value `explicit`. The size of root (`/`) is based on the selected software, value `auto`, and is installed on `c0t0d0s0`. The size of `swap` is set to the necessary size and is installed on `c0t3d0s1`. `usr` is based on the selected software and the installation program determines where `usr` is installed, based on the value `any`.
`cluster`	The Entire Solaris Software Group, `SUNWCall`, is installed on the system.
`bootenv createbe`	An empty, inactive boot environment is set up on disk `c0t1d0`. File systems for root (`/`), swap, and `/export` are created, but left empty. This second boot environment can be installed with a Solaris Flash archive at a later time. The new boot environment can then be activated to become the current boot environment. For keyword values and background about using this keyword, see the following references: ■ For descriptions of keyword values, see "Profile Keywords and Values" on page 111. ■ For background about using Solaris Live Upgrade that creates, upgrades, and activates inactive boot environments, see Chapter 2, "Solaris Live Upgrade (Overview)," in *Solaris 10 5/09 Installation Guide: Solaris Live Upgrade and Upgrade Planning*. ■ For background about using a Solaris Flash archive, see Chapter 1, "Solaris Flash (Overview)," in *Solaris 10 5/09 Installation Guide: Solaris Flash Archives (Creation and Installation)*.

EXAMPLE 3–12 Creating RAID-1 Volumes When Installing a Solaris Flash Archive

In the following example, the profile indicates that the custom JumpStart program uses Solaris Volume Manager technology to create RAID-1 volumes (mirrors) for the root (`/`), `swap`, `/usr` and `/export/home` file systems. A Solaris Flash archive is installed on the boot environment.

```
# profile keywords        profile values
# ----------------        ------------------
  install_type            flash_install
  arhcive_location        nfs server:/export/home/export/flash.s10.SUNWCall
  partitioning            explicit
  filesys                 mirror:d10 c0t0d0s0 c0t1d0s0 4096 /
  filesys                 mirror c0t0d0s1 2048 swap
  filesys                 mirror:d30 c0t0d0s3 c0t1d0s3 4096 /usr
  filesys                 mirror:d40 c0t0d0s4 c0t1d0s4 4096 /usr
```

EXAMPLE 3–12 Creating RAID-1 Volumes When Installing a Solaris Flash Archive *(Continued)*

```
filesys                 mirror:d50 c0t0d0s5 c0t1d0s5 free /export/home
metadb                  c0t1d0s7 size 8192 count 3
```

The following list describes some of the keywords and values from this example.

`install_type` — The profile installs a Solaris Flash archive on the clone system. All files are overwritten as in an initial installation.

`archive_location` — The Solaris Flash archive is retrieved from an NFS server.

`partitioning` — The file system slices are determined by the `filesys` keywords, value `explicit`.

`filesys` — The root (`/`) file system is created and mirrored on the slices `c0t0d0s0` and `c0t1d0s0`. The size of the root (`/`) file system is set to 4096 Mbytes. The RAID-1 volume that mirrors `c0t0d0s0` and `c0t1d0s0` is named `d10`.

`filesys` — The `swap` file system is created and mirrored on the slice `c0t0d0s1`, and is sized at 2048 Mbytes. The custom JumpStart program assigns a name to the mirror.

`filesys` — The `/usr` file system is created and mirrored on the slices `c0t1d0s3` and `c0t0d0s3`. The size of the `/usr` file system is set to 4096 Mbytes. The RAID-1 volume is named `d30`.

`filesys` — The `/usr` file system is created and mirrored on the slices `c0t1d0s4` and `c0t0d0s4`. The size of the `/usr` file system is set to 4096 Mbytes. The RAID-1 volume is named `d40`.

`metadb` — Three state database replicas (metadbs) are installed on slice `c0t1d0s7`, and are sized at 8192 blocks (4 Mbytes).

- For overview information about how to create mirrored file systems during your installation, see Chapter 9, "Creating RAID-1 Volumes (Mirrors) During Installation (Overview)," in *Solaris 10 5/09 Installation Guide: Planning for Installation and Upgrade*.
- For guidelines and requirements of creating mirrored file systems, see Chapter 10, "Creating RAID-1 Volumes (Mirrors) During Installation (Planning)," in *Solaris 10 5/09 Installation Guide: Planning for Installation and Upgrade*.
- For descriptions of keyword values, see "`filesys` Profile Keyword (Creating RAID-1 Volumes)" on page 133 and "`metadb` Profile Keyword (Creating State Database Replicas)" on page 140.

EXAMPLE 3–13 Creating a RAID-1 Volume to Mirror the Root File System

In the following example, the profile indicates that the custom JumpStart program uses Solaris Volume Manager technology to create a RAID-1 volume (mirror) for the root (/) file system.

```
# profile keywords        profile values
# ----------------        ------------------
  install_type            initial_install
  cluster                 SUNWCXall
  filesys                 mirror:d30 c0t1d0s0 c0t0d0s0  /
  filesys                 c0t0d0s3 512 swap
  metadb                  c0t0d0s4 size 8192 count 4
  metadb                  c0t1d0s4 size 8192 count 4
```

The following list describes some of the keywords and values from this example.

`cluster` The Entire Solaris Software Plus OEM Support software group, SUNWCXall, is installed on the system.

`filesys` The root (/) file system is created and mirrored on the slices `c0t1d0s0` and `c0t0d0s0`. The RAID-1 volume that mirrors `c0t1d0s0` and `c0t0d0s0` is named `d30`. The custom JumpStart program assigns names to the two submirrors.

`filesys` The `swap` file system is created and mirrored on the slice `c0t0d0s3`, and is sized at 512 Mbytes.

`metadb` Four state database replicas (metadbs) are installed on slice `c0t0d0s4`, and are sized at 8192 blocks (4 Mbytes).

`metadb` Four state database replicas (metadbs) are installed on slice `c0t1d0s4`, and are sized at 8192 blocks (4 Mbytes).

- For overview information about how to create RAID-1 volumes during your installation, see Chapter 9, "Creating RAID-1 Volumes (Mirrors) During Installation (Overview)," in *Solaris 10 5/09 Installation Guide: Planning for Installation and Upgrade*.
- For guidelines and requirements about creating RAID-1 volumes, see Chapter 10, "Creating RAID-1 Volumes (Mirrors) During Installation (Planning)," in *Solaris 10 5/09 Installation Guide: Planning for Installation and Upgrade*.
- For descriptions of keyword values, see "`filesys` Profile Keyword (Creating RAID-1 Volumes)" on page 133 and "`metadb` Profile Keyword (Creating State Database Replicas)" on page 140.

EXAMPLE 3–14 Creating RAID-1 Volumes to Mirror Multiple File Systems

In the following example, the profile indicates that the custom JumpStart program uses Solaris Volume Manager technology to create RAID-1 volumes (mirrors) for the root (/), swap, and /usr file systems.

```
# profile keywords        profile values
# ----------------        ------------------
  install_type            initial_install
  cluster                 SUNWCXall
  filesys                 mirror:d100 c0t1d0s0 c0t0d0s0 200 /
  filesys                 c0t1d0s5 500 /var
  filesys                 c0t0d0s5 500
  filesys                 mirror c0t0d0s1 512 swap
  metadb                  c0t0d0s3 size 8192 count 5
  filesys                 mirror c0t1d0s4 c0t0d0s4 2000 /usr
  filesys                 c0t1d0s7 free /export/home
  filesys                 c0t0d0s7 free
```

The following list describes some of the keywords and values from this example.

`cluster` The Entire Solaris Software Plus OEM Support software group, SUNWCXall, is installed on the system.

`filesys` The root (/) file system is created and mirrored on the slices `c0t1d0s0` and `c0t0d0s0`. The size of the root (/) file system is set to 200 Mbytes. The RAID-1 volume that mirrors `c0t1d0s0` and `c0t0d0s0` is named `d100`.

`filesys` The /var file system is installed on the slice `c0t1d0s5` and is sized at 500 Mbytes. The root (/) file system is created and mirrored on the slices `c0t1d0s0` and `c0t0d0s0`. The size of the root (/) file system is set to 200 Mbytes. The RAID-1 volume that mirrors `c0t1d0s0` and `c0t0d0s0` is named `d100`.

`filesys` The swap file system is created and mirrored on the slice `c0t0d0s1`, and is sized at 512 Mbytes. The custom JumpStart program assigns a name to the mirror.

`metadb` Five state database replicas (metadbs) are installed on slice `c0t0d0s3`, and are sized at 8192 blocks (4 Mbytes).

`filesys` The /usr file system is created and mirrored on the slices `c0t1d0s4` and `c0t0d0s4`. The size of the /usr file system is set to 2000 Mbytes. The custom JumpStart program assigns a name to the mirror.

- For overview information about how to create mirrored file systems during your installation, see Chapter 9, "Creating RAID-1 Volumes (Mirrors) During Installation (Overview)," in *Solaris 10 5/09 Installation Guide: Planning for Installation and Upgrade*.

EXAMPLE 3–14 Creating RAID-1 Volumes to Mirror Multiple File Systems *(Continued)*

- For guidelines and requirements of creating mirrored file systems, see Chapter 10, "Creating RAID-1 Volumes (Mirrors) During Installation (Planning)," in *Solaris 10 5/09 Installation Guide: Planning for Installation and Upgrade*.
- For descriptions of keyword values, see "`filesys` Profile Keyword (Creating RAID-1 Volumes)" on page 133 and "`metadb` Profile Keyword (Creating State Database Replicas)" on page 140.

EXAMPLE 3–15 x86: Using the `fdisk` Keyword

```
# profile keywords      profile values
# ----------------      ------------------
  install_type          initial_install
  system_type           standalone

  fdisk                 c0t0d0 0x04 delete
  fdisk                 c0t0d0 solaris maxfree
  cluster               SUNWCall
  cluster               SUNWCacc delete
```

The following list describes some of the keywords and values from this example.

`fdisk` All `fdisk` partitions of type DOSOS16 (04 hexadecimal) are deleted from the `c0t0d0` disk.

`fdisk` A Solaris `fdisk` partition is created on the largest contiguous free space on the `c0t0d0` disk.

`cluster` The Entire Distribution software group, `SUNWCall`, is installed on the system.

`cluster` The system accounting utilities, `SUNWCacc`, are not to be installed on the system.

Testing a Profile

After you create a profile, use the `pfinstall(1M)` command to test the profile. Test the profile before you use the profile to install or upgrade a system. Testing a profile is especially useful when you are creating upgrade profiles that reallocate disk space.

By looking at the installation output that is generated by `pfinstall`, you can quickly determine if a profile works as you intended. For example, use the profile to determine if a system has enough disk space to upgrade to a new release of the Solaris software before you perform the upgrade on that system.

`pfinstall` enables you to test a profile against the following:

- The system's disk configuration where `pfinstall` is being run.
- Other disk configurations. You use a disk configuration file that represents a structure of a disk, for example, a disk's bytes/sector, flags, and slices. Creating disk configuration files is described in "Creating Disk Configuration Files" on page 67 and "x86: To Create a Disk Configuration File" on page 69.

Note – You cannot use a disk configuration file to test a profile you intend to use to upgrade a system. Instead, you must test the profile against the system's actual disk configuration and the software that is currently installed on that system.

▼ To Create a Temporary Solaris Environment to Test a Profile

To test a profile for a particular Solaris release successfully and accurately, you must test a profile within the Solaris environment of the same release. For example, if you want to test a Solaris initial installation profile, run the `pfinstall` command on a system that is running the Solaris OS.

You need to create a temporary installation environment if you are testing a profile under one of the following conditions:

- You want to test a Solaris 10 5/09 upgrade profile on a system that is running a previous version of the Solaris software.
- You do not have a Solaris 10 5/09 system installed yet to test Solaris 10 5/09 initial installation profiles.

1 Boot a system from an image of one of the following:

For SPARC based systems:

- Solaris Operating System for SPARC Platforms DVD
- Solaris Software for SPARC Platforms - 1 CD

For x86 based systems:

- Solaris Operating System for x86 Platforms DVD
- Solaris Software for x86 Platforms - 1 CD

Note – If you want to test an upgrade profile, boot the system that you are upgrading.

---

2 Respond to the system identification questions.

3 **To exit from the installation program, type ! at the following prompt.**

```
The Solaris installation program  will assist you in installing software for Solaris.
<Press ENTER to continue> {"!" exits}
```

4 **Execute the `pfinstall` command from the shell. For details about using the `pfinstall` command, see Step 7 in "To Test a Profile" on page 51.**

▼ To Test a Profile

x86 only – If you are using the `locale` keyword, the `pfinstall -D` command fails to test the profile. For a workaround, see the error message "could not select locale," in the section, "Upgrading the Solaris OS" on page 180.

1 **Locate a system on which to test the profile that is the same type of platform, SPARC or x86, for which the profile was created.**

If you are testing an upgrade profile, you must test the profile on the actual system that you intend to upgrade.

2 **Use the following decision table to determine what to do next.**

Test Scenario	Instructions
Test an initial installation profile and have a system that is running the Solaris 10 5/09 software.	Become superuser on the system and go to Step 5.
Test an upgrade profile, or you do not have a system that is running Solaris 10 5/09 to test an initial installation profile.	Create a temporary Solaris 10 5/09 environment to test the profile. For details, see "To Create a Temporary Solaris Environment to Test a Profile" on page 50. Then, go to Step 3.

3 **Create a temporary mount point.**

```
# mkdir /tmp/mnt
```

4 **Mount the directory that contains the profile or profiles that you want to test.**

Mount Scenario	Typing Instructions
Mount a remote NFS file system for systems on the network.	**`mount -F nfs`** *server_name*`:`*path* **`/tmp/mnt`**
SPARC: Mount a UFS-formatted diskette.	**`mount -F ufs /dev/diskette /tmp/mnt`**

Mount Scenario	Typing Instructions
Mount a PCFS-formatted diskette.	`mount -F pcfs /dev/diskette /tmp/mnt`

5 To test the profile with a specific system memory size, set `SYS_MEMSIZE` to the specific memory size in Mbytes.

```
# SYS_MEMSIZE=memory_size
# export SYS_MEMSIZE
```

6 Did you mount a directory in Step 4?

- If yes, change the directory to `/tmp/mnt`.

  ```
  # cd /tmp/mnt
  ```

- If no, change the directory to where the profile is located, which is usually the JumpStart directory.

  ```
  # cd jumpstart_dir_path
  ```

7 Test the profile with the `pfinstall`(1M) command.

```
# /usr/sbin/install.d/pfinstall -D:-d disk_config_file -c path profile
```

Caution – You *must* include the `-d` or `-D` option. If you do not include one of these options, `pfinstall` uses the profile you specify to install the Solaris software. All of the data on the system is overwritten.

`-D`	`pfinstall` uses the current system's disk configuration to test the profile. You must use the `-D` option to test an upgrade profile.
`-d` *disk_config_file*	`pfinstall` uses the disk configuration file, *disk_config_file*, to test the profile. If *disk_config_file* is not located in the directory where `pfinstall` is run, you must specify the path. For instructions about how to create a disk configuration file, see "Creating Disk Configuration Files" on page 67.

Note – You cannot use the `-d` *disk_config_file* option with an upgrade profile, `install_type upgrade`. You must always test an upgrade profile against a system's disk configuration, that is, you must use the `-D` option.

`-c` *path*	The path to the Solaris software image. You use this option, for example, if the system is using Volume Manager to mount the Solaris Software - 1 CD for your platform.
	Note – The `-c` option is not required if you booted from a Solaris Operating System DVD or a Solaris Software - 1 CD image for your platform. The DVD or CD image is mounted on `/cdrom` as part of the booting process.
profile	The name of the profile to test. If *profile* is not in the directory where `pfinstall` is being run, you must specify the path.

Profile Test Examples

The following example shows how to use `pfinstall` to test a profile that is named `basic_prof`. The profile is tested against the disk configuration on a system on which the Solaris 10 5/09 software is installed. The `basic_prof` profile is located in the `/jumpstart` directory, and the path to the Solaris Operating System DVD image is specified because Volume Manager is being used.

EXAMPLE 3–16 Profile Test Using a Solaris 10 5/09 System

```
# cd /jumpstart
# /usr/sbin/install.d/pfinstall -D -c /cdrom/pathname basic_prof
```

The following example shows how to use `pfinstall` to test the profile that is named `basic_prof` on a Solaris 10 5/09 system. The test is performed against the `535_test` disk configuration file. The test checks for 64 Mbytes of system memory. This example uses a Solaris Software for SPARC Platforms - 1 CD or Solaris Software for x86 Platforms - 1 CD image that is located in the `/export/install` directory.

EXAMPLE 3–17 Profile Test Using a Disk Configuration File

```
# SYS_MEMSIZE=64
# export SYS_MEMSIZE
# /usr/sbin/install.d/pfinstall -d 535_test -c /export/install basic_prof
```

Validating the rules File

Before you can use a profile and rules file, you must run the check script to validate that the files are set up correctly. If all rules and profiles are correctly set up, the rules.ok file is created, which is required by the custom JumpStart installation software to match a system to a profile.

Table 3–2 describes what the check script does.

TABLE 3–2 What Happens When You Use the check Script

Stage	Description
1	The rules file is checked for syntax. check verifies that the rule keywords are legitimate and that the *begin*, *class*, and *finish* fields are specified for each rule. The *begin* and *finish* fields can consist of a minus sign (-) instead of a file name.
2	If no errors are found in the rules file, each profile that is specified in the rules is checked for syntax.
3	If no errors are found, check creates the rules.ok file from the rules file, removes all comments and blank lines, retains all rules, and adds the following comment line at the end: `# version=2 checksum=num`

▼ To Validate the rules File

1 Ensure that the check script is located in the JumpStart directory.

Note – The check script is in the Solaris_10/Misc/jumpstart_sample directory on the Solaris Operating System DVD or on the Solaris Software - 1 CD.

---

2 Change the directory to the JumpStart directory.

3 Run the check script to validate the rules file:

```
$ ./check -p path -r file_name
```

-p *path* Validates the rules by using the check script from the Solaris software image instead of the check script from the system you are using. *path* is the image on a local disk or a mounted Solaris Operating System DVD or a Solaris Software - 1 CD.

Use this option to run the most recent version of check if your system is running a previous version of Solaris.

`-r` *file_name* Specifies a rules file other than the one that is named `rules`. Using this option, you can test the validity of a rule before you integrate the rule into the `rules` file.

As the `check` script runs, the script reports the checking of the validity of the `rules` file and each profile. If no errors are encountered, the script reports the following information.

```
The custom JumpStart configuration is ok
```

4 Ensure that `root` owns the `rules.ok` file and that the permissions are set to 644.

See Also After you validate the `rules` file, you can learn more about optional custom JumpStart features in Chapter 4, "Using Optional Custom JumpStart Features (Tasks)." You can learn about performing custom JumpStart installations in Chapter 6, "Performing a Custom JumpStart Installation (Tasks)."

◆ ◆ ◆ CHAPTER 4

Using Optional Custom JumpStart Features (Tasks)

This chapter describes the optional features that are available to create additional custom JumpStart installation tools.

Note – If you are installing a Solaris ZFS root pool, see Chapter 9, "Installing a ZFS Root Pool With JumpStart," for limitations and profile examples.

Note – Instructions in this chapter are valid for either a SPARC server or an x86 server that is being used to provide custom JumpStart files, called a profile server. A profile server can provide custom JumpStart files for different platform types. For example, a SPARC server can provide custom JumpStart files for both SPARC based systems and x86 based systems.

Creating Begin Scripts

A begin script is a user-defined Bourne shell script that you specify in the `rules` file. A begin script performs tasks before the Solaris software is installed on a system. You can use begin scripts only when using custom JumpStart to install the Solaris software.

Use a begin script to perform one of the following tasks:

- Creating derived profiles
- Backing up files before upgrading

Important Information About Begin Scripts

- Do not specify something in the script that would prevent the mounting of file systems onto `/a` during an initial or upgrade installation. If the JumpStart program cannot mount the file systems onto `/a`, an error occurs and installation fails.
- During the installation, output from the begin script is deposited in `/tmp/begin.log`. After the installation is completed, the log file is redirected to `/var/sadm/system/logs/begin.log`.
- Ensure that `root` owns the begin script and that the permissions are set to 644.
- You can use custom JumpStart environment variables in your begin scripts. For a list of environment variables, see "Custom JumpStart Environment Variables" on page 155.
- Save begin scripts in the JumpStart directory.

Note – For the Solaris 10 release, a sample JumpStart script, `set_nfs4_domain`, was provided on media to prevent being prompted during a JumpStart installation. This script suppressed the NFSv4 prompt during installation. This script is no longer required. **Starting with the Solaris 10 5/09release**, use the `sysidcfg` keyword, `nfs4_domain` that suppresses being prompted. The `set_nfs4_domain` script no longer works to suppress a prompt.

If you have non-global zones installed and the new `nfs4_domain` keyword exists in the `sysidcfg` file, the first boot of a non-global zone sets the domain. Otherwise, the Solaris interactive installation program comes up and you are prompted to provide a domain name before the boot process completes.

See "nfs4_domain Keyword" in *Solaris 10 5/09 Installation Guide: Network-Based Installations*

Creating Derived Profiles With a Begin Script

A derived profile is a profile that is dynamically created by a begin script during a custom JumpStart installation. Derived profiles are needed when you cannot set up the `rules` file to match specific systems to a profile. For example, you might need to use derived profiles for identical system models that have different hardware components, such as systems that contain different frame buffers.

To set up a rule to use a derived profile, you must perform the following tasks:

- Set the profile field to an equal sign (=) instead of a profile.
- Set the begin field to a begin script that creates a derived profile that depends on the system on which you intend to install Solaris.

When a system matches a rule with the profile field equal to an equal sign (=), the begin script creates the derived profile that is used to install the Solaris software on the system.

The following is an example of a begin script that creates the same derived profile every time. You can write a begin script to create different derived profiles that depend on the evaluation of rules.

EXAMPLE 4–1 Begin Script That Creates a Derived Profile

```
#!/bin/sh
echo "install_type        initial_install"    > ${SI_PROFILE}
echo "system_type         standalone"        >> ${SI_PROFILE}
echo "partitioning        default"           >> ${SI_PROFILE}
echo "cluster             SUNWCprog"         >> ${SI_PROFILE}
echo "package       SUNWman     delete"      >> ${SI_PROFILE}
echo "package       SUNWolman   delete"      >> ${SI_PROFILE}
echo "package       SUNWxwman   delete"      >> ${SI_PROFILE}
```

In the example, the begin script must use the `SI_PROFILE` environment variable for the name of the derived profile, which is set to `/tmp/install.input` by default.

Note – If a begin script is used to create a derived profile, ensure the script does not have any errors. A derived profile is not verified by the `check` script because derived profiles are not created until the execution of the begin script.

Creating Finish Scripts

A finish script is a user-defined Bourne shell script that you specify in the `rules` file. A finish script performs tasks after the Solaris software is installed on a system, but before the system reboots. You can use finish scripts only when using custom JumpStart to install Solaris.

Tasks that you can perform with a finish script include the following:

- Adding files
- Adding individual packages or patches in addition to the ones that are installed in a particular software group
- Customizing the root environment
- Setting the system's root password
- Installing additional software

Important Information About Finish Scripts

- The Solaris installation program mounts the system's file systems on `/a`. The file systems remain mounted on `/a` until the system reboots. You can use the finish script to add, change, or remove files from the newly installed file system hierarchy by modifying the file systems that are respective to `/a`.
- During the installation, output from the finish script is deposited in `/tmp/finish.log`. After the installation is completed, the log file is redirected to `/var/sadm/system/logs/finish.log`.
- Ensure that `root` owns the finish script and that the permissions are set to 644.
- You can use custom JumpStart environment variables in your finish scripts. For a list of environment variables, see "Custom JumpStart Environment Variables" on page 155.
- Save finish scripts in the JumpStart directory.

▼ To Add Files With a Finish Script

Through a finish script, you can add files from the JumpStart directory to an already installed system. You can add the files because the JumpStart directory is mounted on the directory that is specified by the `SI_CONFIG_DIR` variable. The directory is set to `/tmp/install_config` by default.

Note – You can also replace files by copying files from the JumpStart directory to already existing files on the installed system.

---

1 Copy all of the files that you are adding to the installed system to the JumpStart directory.

2 Insert the following line in the finish script for each file that you want to be copied to the newly installed file system hierarchy:

```
cp ${SI_CONFIG_DIR}/file_name  /a/path_name
```

Example 4–2 Adding a File With a Finish Script

For example, assume you have a special application, `site_prog`, developed for all users at your site. If you place a copy of `site_prog` into the JumpStart directory, the following line in a finish script copies `site_prog` from the JumpStart directory into a system's `/usr/bin` directory:

```
cp ${SI_CONFIG_DIR}/site_prog  /a/usr/bin
```

Adding Packages or Patches With a Finish Script

You can create a finish script to automatically add packages or patches after the Solaris software is installed on a system. By adding packages with a finish script, you reduce time and ensure consistency in which packages and patches are installed on different systems at your site.

When you use the `pkgadd(1M)` or `patchadd(1M)` commands in finish scripts, use the `-R` option to specify `/a` as the root path.

- Example 4–3 shows an example of a finish script that adds packages.
- Example 4–4 shows an example of a finish script that adds patches.

EXAMPLE 4–3 Adding Packages With a Finish Script

```
#!/bin/sh

BASE=/a
MNT=/a/mnt
ADMIN_FILE=/a/tmp/admin

mkdir ${MNT}
mount -f nfs sherlock:/export/package ${MNT}
cat >${ADMIN_FILE} <<DONT_ASK
mail=root
instance=overwrite
partial=nocheck
runlevel=nocheck
idepend=nocheck
rdepend=nocheck
space=ask
setuid=nocheck
conflict=nocheck
action=nocheck
basedir=default
DONT_ASK

/usr/sbin/pkgadd -a ${ADMIN_FILE} -d ${MNT} -R ${BASE} SUNWxyz
umount ${MNT}
rmdir ${MNT}
```

The following describes some commands for this example.

- The following command mounts a directory on a server that contains the package to install.

```
mount -f nfs sherlock:/export/package ${MNT}
```

EXAMPLE 4–3 Adding Packages With a Finish Script *(Continued)*

- The following command creates a temporary package administration file, `admin`, to force the `pkgadd(1M)` command not to perform checks or prompt for questions when installing a package. Use the temporary package administration file to maintain a hands-off installation when you are adding packages.

  ```
  cat >${ADMIN_FILE} <<DONT_ASK
  ```

- The following `pkgadd` command adds the package by using the `-a` option, specifying the package administration file, and the `-R` option, specifying the root path.

  ```
  /usr/sbin/pkgadd -a ${ADMIN_FILE} -d ${MNT} -R ${BASE} SUNWxyz
  ```

EXAMPLE 4–4 Adding Patches With a Finish Script

```
 #!/bin/sh

########
#
# USER-CONFIGURABLE OPTIONS
#
########

# The location of the patches to add to the system after it's installed.
# The OS rev (5.x) and the architecture (`mach`) will be added to the
# root.  For example, /foo on a 8 SPARC would turn into /foo/5.8/sparc
LUPATCHHOST=ins3525-svr
LUPATCHPATHROOT=/export/solaris/patchdb
#########
#
# NO USER-SERVICEABLE PARTS PAST THIS POINT
#
#########

BASEDIR=/a

# Figure out the source and target OS versions
echo Determining OS revisions...
SRCREV=`uname -r`
echo Source $SRCREV

LUPATCHPATH=$LUPATCHPATHROOT/$SRCREV/`mach`

#
# Add the patches needed
#
```

EXAMPLE 4–4 Adding Patches With a Finish Script *(Continued)*

```
echo Adding OS patches
mount $LUPATCHHOST:$LUPATCHPATH /mnt >/dev/null 2>&1
if [ $? = 0 ] ; then
    for patch in ‘cat /mnt/*Recommended/patch_order‘ ; do
        (cd /mnt/*Recommended/$patch ; echo yes | patchadd -u -d -R $BASEDIR .)
    done
    cd /tmp
    umount /mnt
else
    echo "No patches found"
if
```

Note – In the past, the `chroot(1M)` command was used with the `pkgadd` and `patchadd` commands in the finish script environment. In rare instances, some packages or patches do not work with the `-R` option. You must create a dummy `/etc/mnttab` file in the `/a` root path before issuing the `chroot` command.

To create a dummy `/etc/mnttab` file, add the following line to your finish script:

```
cp /etc/mnttab /a/etc/mnttab
```

Customizing the Root Environment With a Finish Script

You can also use finish scripts to customize files that are already installed on a system. For example, the finish script in Example 4–5 customizes the root environment by appending information to the `.cshrc` file in the root (`/`) directory.

EXAMPLE 4–5 Customizing the Root Environment With a Finish Script

```
#!/bin/sh
#
# Customize root’s environment
#
echo "***adding customizations in /.cshrc"
test -f a/.cshrc || {
cat >> a/.cshrc <<EOF
set history=100 savehist=200 filec ignoreeof prompt="\$user@‘uname -n‘> "
alias cp cp -i
alias mv mv -i
alias rm rm -i
```

EXAMPLE 4–5 Customizing the Root Environment With a Finish Script *(Continued)*

```
alias ls ls -FC
alias h history
alias c clear
unset autologout
EOF
}
```

Setting a System's Root Password With a Finish Script

After the Solaris software is installed on a system, the system reboots. Before the boot process is completed, the system prompts for the root password. Until someone types a password, the system cannot finish booting.

A finish script that is named `set_root_pw` is saved in the `auto_install_sample` directory. The finish script shows how to set the root password automatically, without prompting. `set_root_pw` is shown in Example 4–6.

Note – If you set the system's root password with a finish script, users might attempt to discover the root password from the encrypted password in your finish script. Ensure that you safeguard against users who might try to determine the root password.

EXAMPLE 4–6 Setting the System's Root Password With a Finish Script

```
    #!/bin/sh
    #
    #       @(#)set_root_pw 1.4 93/12/23 SMI
    #
    # This is an example Bourne shell script to be run after installation.
    # It sets the system’s root password to the entry defined in PASSWD.
    # The encrypted password is obtained from an existing root password entry
    # in /etc/shadow from an installed machine.

    echo "setting password for root"

    # set the root password
PASSWD=dKO5IBkSF42lw
    #create a temporary input file
cp /a/etc/shadow /a/etc/shadow.orig

    mv /a/etc/shadow /a/etc/shadow.orig
    nawk -F: ’{
        if ( $1 == "root" )
```

EXAMPLE 4–6 Setting the System's Root Password With a Finish Script *(Continued)*

```
          printf"%s:%s:%s:%s:%s:%s:%s:%s:%s\n",$1,passwd,$3,$4,$5,$6,$7,$8,$9
     else
               printf"%s:%s:%s:%s:%s:%s:%s:%s:%s\n",$1,$2,$3,$4,$5,$6,$7,$8,$9
     }' passwd="$PASSWD" /a/etc/shadow.orig > /a/etc/shadow
#remove the temporary file
rm -f /a/etc/shadow.orig
# set the flag so sysidroot won't prompt for the root password
sed -e 's/0 # root/1 # root/' ${SI_SYS_STATE} > /tmp/state.$$
 mv /tmp/state.$$ ${SI_SYS_STATE}
```

The following describes some of the commands in this example.

- The following command sets the variable `PASSWD` to an encrypted root password that is obtained from an existing entry in a system's `/etc/shadow` file.

  ```
  #create a temporary input file
  ```

- The following command creates a temporary input file of `/a/etc/shadow`.

  ```
  cp /a/etc/shadow /a/etc/shadow.orig
  ```

- The following command changes the root entry in the `/etc/shadow` file for the newly installed system by using `$PASSWD` as the password field.

  ```
  if ( $1 == "root" )
  ```

- The following command removes the temporary `/a/etc/shadow` file.

  ```
  rm -f /a/etc/shadow.orig
  ```

- The following command changes the entry from `0` to a `1` in the state file so that the user is not prompted for the root password. The state file is accessed by using the variable `SI_SYS_STATE`, which has a value currently of `/a/etc/.sysIDtool.state`. To avoid problems with your scripts if this value changes, always reference this file by using `$SI_SYS_STATE`. The `sed` command that is shown here contains a tab character after the `0` and after the `1`.

  ```
  sed -e 's/0 # root/1 # root/' ${SI_SYS_STATE} > /tmp/state.$$
  ```

Non-Interactive Installations With Finish Scripts

You can use finish scripts to install additional software after the Solaris OS is installed. The Solaris installation program prompts you to enter information during the installation. To maintain a hands-off installation, you can run the Solaris installation program with the `-nodisplay` or `-noconsole` options.

TABLE 4–1 Solaris Installation Options

Option	Description
`-nodisplay`	Runs the installer without a graphic user interface. Use the default product installation unless the installation was modified by the `-locales` option.
`-noconsole`	Runs the installation without any interactive text console device. Useful when paired with `-nodisplay` for UNIX script use.

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

Creating a Compressed Configuration File

Instead of using the `add_install_client` command to specify the location of the custom JumpStart configuration files, you can specify the location of the files when you boot the system. However, you can only specify the name of one file. As a result, you must compress all of the custom JumpStart configuration files into one file.

- **For SPARC based systems**, you specify the location of the file in the `boot` command
- **For x86 based systems**, you specify the location of the files by editing the GRUB entry in the GRUB menu

The compressed configuration file can be one of the following types:

- `tar`
- Compressed `tar`
- `zip`
- `bzip tar`

▼ To Create a Compressed Configuration File

1 Change the directory to the JumpStart directory on the profile server.

```
# cd jumpstart_dir_path
```

2 Use a compression tool to compress the custom JumpStart configuration files into one file.

Note – The compressed configuration file cannot contain relative paths. The custom JumpStart configuration files must be in the same directory as the compressed file.

The compressed configuration file must contain the following files:

- Profile

- `rules`
- `rules.ok`

You can also include the `sysidcfg` file in the compressed configuration file.

3 **Save the compressed configuration file on an NFS server, an HTTP server, or on a local hard disk.**

Compressed Configuration File Example

The following example shows how to use the `tar` command to create a compressed configuration file that is named `config.tar`. The custom JumpStart configuration files are located in the `/jumpstart` directory.

EXAMPLE 4–7 Creating a Compressed Configuration File

```
# cd /jumpstart
# tar -cvf config.tar *
a profile 1K
a rules 1K
a rules.ok 1K
a sysidcfg 1K
```

Creating Disk Configuration Files

This section describes how to create single-disk and multiple-disk configuration files. Disk configuration files enable you to use `pfinstall(1M)` from a single system to test profiles against different disk configurations.

▼ SPARC: To Create a Disk Configuration File

1 **Locate a SPARC based system with a disk you want to test.**

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

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

3 **Create a single–disk configuration file by redirecting the output of the `prtvtoc(1M)` command to a file.**

```
# prtvtoc /dev/rdsk/device_name >disk_config_file
```

`/dev/rdsk/`*device_name* The device name of the system's disk. *device_name* must be in the form c*w*t*x*d*y*s2 or c*x*d*y*s2.

disk_config_file The name of the disk configuration file.

4 Determine if you are testing the installation of Solaris software on multiple disks.

- If no, stop. You are finished.
- If yes, concatenate the single–disk configuration files and save the output in a new file.

  ```
  # cat disk_file1 disk_file2 >multi_disk_config
  ```

  The new file becomes the multiple-disk configuration file, as in the following example.

  ```
  # cat 104_disk2 104_disk3 104_disk5 >multi_disk_test
  ```

5 Determine if the target numbers in the disk device names are unique in the multiple-disk configuration file that you created in the previous step.

- If yes, stop. You are finished.
- If no, open the file with a text editor and make the target numbers unique in the disk device names.

  For example, assume that the file contains the same target number, `t0`, for different disk device names, as shown here.

  ```
  * /dev/rdsk/c0t0d0s2 partition map
  ...
  * /dev/rdsk/c0t0d0s2 partition map
  ```

  Change the second target number to `t2`, as shown here:

  ```
  * /dev/rdsk/c0t0d0s2 partition map
  ...
  * /dev/rdsk/c0t2d0s2 partition map
  ```

SPARC: Disk Configuration File Example

The following example shows how to create a single–disk configuration file, `104_test`, on a SPARC based system with a 104-Mbyte disk.

EXAMPLE 4–8 SPARC: Creating a Disk Configuration File

You redirect the output of the `prtvtoc` command to a single–disk configuration file that is named `104_test`:

```
# prtvtoc /dev/rdsk/c0t3d0s2 >104_test
```

EXAMPLE 4–8 SPARC: Creating a Disk Configuration File *(Continued)*

The contents of the `104_test` file resemble the following:

```
* /dev/rdsk/c0t3d0s2 partition map
*
* Dimensions:
*     512 bytes/sector
*      72 sectors/track
*      14 tracks/cylinder
*    1008 sectors/cylinder
*    2038 cylinders*    2036 accessible cylinders
* Flags:
*   1: unmountable
*  10: read-only
*
*                          First     Sector    Last
* Partition  Tag  Flags    Sector     Count    Sector  Mount Directory
       1      2    00          0     164304   164303   /
       2      5    00          0    2052288  2052287
       3      0    00     164304     823536   987839   /disk2/b298
       5      0    00     987840     614880  1602719   /install/298/sparc/work
       7      0    00    1602720     449568  2052287   /space
```

You have created disk configuration files for a SPARC based system. "Testing a Profile" on page 49 contains information about using disk configuration files to test profiles.

▼ x86: To Create a Disk Configuration File

1 Locate an x86 based system that contains a disk that you are testing.

2 Become superuser or assume an equivalent role.

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

3 Create part of the single-disk configuration file by saving the output of the `fdisk`(1M) command in a file.

```
# fdisk -R -W disk_config_file -h /dev/rdsk/device_name
```

disk_config_file	The name of a disk configuration file.
`/dev/rdsk/`*device_name*	The device name of the `fdisk` layout of the entire disk. *device_name* must be in the form `c`*w*`t`*x*`d`*y*`s0` or `c`*x*`d`*y*`s0`.

4 **Append the output of the `prtvtoc`(1M) command to the disk configuration file:**

```
# prtvtoc /dev/rdsk/device_name >>disk_config
```

`/dev/rdsk/`*device_name*	The device name of the system's disk. *device_name* must be in the form `c`*w*`t`*x*`d`*y*`s2` or `c`*x*`d`*y*`s2`.
disk_config	The name of the disk configuration file.

5 **Determine if you are testing the installation of Solaris software on multiple disks.**

- If no, stop. You are finished.
- If yes, concatenate the single-disk configuration files and save the output in a new file.

  ```
  # cat disk_file1 disk_file2 >multi_disk_config
  ```

  The new file becomes the multiple-disk configuration file, as in the following example.

  ```
  # cat 104_disk2 104_disk3 104_disk5 >multi_disk_test
  ```

6 **Determine if the target numbers in the disk device names are unique in the multiple-disk configuration file that you created in the previous step.**

- If yes, stop. You are finished.
- If no, open the file with a text editor and make the target numbers unique.

  For example, the file might contain the same target number, `t0`, for different disk device names as shown here:

  ```
  * /dev/rdsk/c0t0d0s2 partition map
  ...
  * /dev/rdsk/c0t0d0s2 partition map
  ```

  Change the second target number to `t2`, as shown here:

  ```
  * /dev/rdsk/c0t0d0s2 partition map
  ...
  * /dev/rdsk/c0t2d0s2 partition map
  ```

x86: Disk Configuration File Example

The following example shows how to create a single-disk configuration file, `500_test`, on an x86 based system that contains a 500-Mbyte disk.

EXAMPLE 4–9 x86: Creating a Disk Configuration File

First, you save the output of the `fdisk` command to a file that is named `500_test`:

```
# fdisk -R -W 500_test -h /dev/rdsk/c0t0d0p0
```

The `500_test` file looks like the following:

```
 * /dev/rdsk/c0t0d0p0 default fdisk table
* Dimensions:
*     512 bytes/sector
*      94 sectors/track
*      15 tracks/cylinder
*    1455 cylinders
*
*  HBA Dimensions:
*     512 bytes/sector
*      94 sectors/track
*      15 tracks/cylinder
*    1455 cylinders
*
* systid:
*  1:    DOSOS12
*  2:    PCIXOS
*  4:    DOSOS16
*  5:    EXTDOS
*  6:    DOSBIG
*  86:   DOSDATA
*  98:   OTHEROS
*  99:   UNIXOS
* 130:   SUNIXOS
*
* Id  Act Bhead Bsect   Bcyl  Ehead  Esect  Ecyl Rsect  Numsect
 130  128 44    3       0     46    30     1001 1410   2050140
```

Second, you append the output of the `prtvtoc` command to the `500_test` file:

```
# prtvtoc /dev/rdsk/c0t0d0s2 >>500_test
```

The `500_test` file is now a complete disk configuration file:

```
* /dev/rdsk/c0t0d0p0 default fdisk table
* Dimensions:
*     512 bytes/sector
*      94 sectors/track
*      15 tracks/cylinder
*    1455 cylinders
*
```

EXAMPLE 4–9 x86: Creating a Disk Configuration File *(Continued)*

```
*  HBA Dimensions:
*     512 bytes/sector
*      94 sectors/track
*      15 tracks/cylinder
*    1455 cylinders
*
* systid:
*  1:    DOSOS12
*  2:    PCIXOS
*  4:    DOSOS16
*  5:    EXTDOS
*  6:    DOSBIG
*  86:   DOSDATA
*  98:   OTHEROS
*  99:   UNIXOS
*  130:  SUNIXOS
*
* Id  Act Bhead Bsect Bcyl  Ehead  Esec  Ecyl Rsect  Numsect
 130  128 44    3     0     46    30    1001 1410   2050140
* /dev/rdsk/c0t0d0s2 partition map
*
* Dimensions:
*      512 bytes/sector
*       94 sectors/track
*       15 tracks/cylinder
*     1110 sectors/cylinder
*     1454 cylinders
*     1452 accessible cylinders
*
* Flags:
*   1: unmountable
*  10: read-only
*                          First    Sector    Last
* Partition  Tag  Flags    Sector     Count    Sector  Mount Directory
       2      5    01       1410   2045910   2047319
       7      6    00       4230   2043090   2047319  /space
       8      1    01          0      1410     1409
       9      9    01       1410      2820     422987
```

You have created disk configuration files for an x86 based system. "Testing a Profile" on page 49 contains information about using disk configuration files to test profiles.

Using a Site-Specific Installation Program

You can also use begin and finish scripts to create your own installation program to install Solaris software.

When you specify a minus sign (-) in the profile field, begin and finish scripts control how Solaris software is installed on a system instead of the profile and the Solaris installation program.

For example, if the following rule matches a system, the `x_install.beg` begin script and the `x_install.fin` finish script install Solaris software on the system that is named `clover`:

```
hostname clover x_install.beg - x_install.fin
```

◆ ◆ ◆ CHAPTER 5

Creating Custom Rule and Probe Keywords (Tasks)

This chapter provides information and procedures for creating your own custom rule and probe keywords.

Note – If you are installing a Solaris ZFS root pool, see Chapter 9, "Installing a ZFS Root Pool With JumpStart," for limitations and profile examples.

Probe Keywords

To understand what a probe keyword is, you first need to recall what a rule keyword is. A rule keyword is a predefined lexical unit or word that describes a general system attribute, such as host name, `hostname`, or memory size, `memsize`. Rule keywords and the values that are associated with them enable you to match a system that has the same attribute to a profile. This match of a system's attributes defines how the Solaris software is to be installed on each system in the group.

Custom JumpStart environment variables, which you use in begin and finish scripts, are set on demand. For example, information about which operating system is already installed on a system is only available in `SI_INSTALLED` after the `installed` rule keyword is used.

In some situations, you might need to extract the same information in a begin or finish script for a purpose other than to match a system and run a profile. Probe keywords provide the solution. Probe keywords extract attribute information and remove the need for you to set up a matching condition and run a profile.

For a list of probe keywords and values, see "Probe Keywords and Values" on page 157.

Creating a custom_probes File

The rule and probe keywords that are described in "Rule Keywords and Values" on page 107 and "Probe Keywords and Values" on page 157 might not be precise enough for your needs. You can define your own custom rule or probe keywords by creating a custom_probes file.

The custom_probes file is a Bourne shell script that contains two types of functions. You must save the custom_probes file in the same JumpStart directory where you saved the rules file. The two types of functions that you can define in a custom_probes file are as follows:

- Probe – Gathers the information you want or does the actual work and sets a corresponding SI_ environment variable that you define. Probe functions become probe keywords.
- Comparison – Calls a corresponding probe function, compares the output of the probe function, and returns 0 if the keyword matches or 1 if the keyword does not match. Comparison functions become rule keywords.

Syntax of the custom_probes File

The custom_probes file can contain any valid Bourne shell command, variable, or algorithm.

Note – You can define probe and comparison functions that require a single argument in the custom_probes file. When you use the corresponding custom probe keyword in the rules file, the argument after the keyword is interpreted (as $1).

When you use the corresponding custom rule keyword in the rules file, the arguments are interpreted in sequence. The sequence starts after the keyword and ends before the next && or begin script, whichever comes first.

The custom_probes file must meet the following requirements:

- Have the name custom_probes
- Have root as its owner
- Be executable and have permissions set to 755
- Contain at least one probe function and one corresponding comparison function

To improve clarity and organization, define all probe functions first, at the top of the file, followed by all comparison functions.

Syntax of Function Names in custom_probes

The name of a probe function must begin with probe_. The name of a comparison function must begin with cmp_.

Functions that begin with probe_ define new probe keywords. For example, the function probe_tcx defines the new probe keyword tcx. Functions that begin with cmp_ define new rule keywords. For example, cmp_tcx defines the new rule keyword tcx.

▼ To Create a custom_probes File

1. **Use a text editor to create a Bourne shell script text file. Name the file** custom_probes**.**
2. **In the** custom_probes **text file, define your probe and comparison functions.**

Note – You can define probe and comparison functions that require arguments in the custom_probes file. When you use the corresponding custom probe keyword in the rules file, the arguments after the keyword are interpreted in sequence (as $1, $2, and so on).

When you use the corresponding custom rule keyword in the rules file, the arguments are interpreted in sequence. The sequence starts after the keyword and ends before the next && or begin script, whichever comes first.

---

3. **Save the** custom_probes **file in the JumpStart directory next to the** rules **file.**
4. **Ensure that** root **owns the** rules **file and that the permissions are set to 644.**

Examples of a custom_probes File and Keyword

You can find additional examples of probe and comparison functions in the following directories:

- /usr/sbin/install.d/chkprobe on a system that has the Solaris software installed
- /Solaris_10/Tools/Boot/usr/sbin/install.d/chkprobe on the Solaris Operating System DVD or on the Solaris Software - 1 CD

The following custom_probes file contains a probe and comparison function that tests for the presence of a TCX graphics card.

EXAMPLE 5–1 custom_probes File

```
#!/bin/sh
#
# custom_probe script to test for the presence of a TCX graphics card.
#

#
```

EXAMPLE 5–1 custom_probes File *(Continued)*

```
# PROBE FUNCTIONS
#
probe_tcx() {
  SI_TCX=‘modinfo | grep tcx | nawk ’{print $6}’‘
  export SI_TCX
}

#
# COMPARISON FUNCTIONS
#
cmp_tcx() {
  probe_tcx

  if [ "X${SI_TCX}" = "X${1}" ]; then
     return 0
  else
     return 1
  if
}
```

The following example `rules` file shows the use of the probe keyword that is defined in the preceding example, `tcx`. If a TCX graphics card is installed and found in a system, `profile_tcx` is run. Otherwise, `profile` is run.

Note – Always place probe keywords at or near the beginning of the `rules` file. This placement ensures that the keywords are read and run before other rule keywords that might rely on the probe keywords.

EXAMPLE 5–2 Custom Probe Keyword Used in a `rules` File

```
probe tcx
tcx     tcx     -     profile_tcx     -
any     any     -     profile         -
```

Validating the `custom_probes` File

Before you can use a profile, `rules`, and `custom_probes` file, you must run the `check` script to validate that the files are set up correctly. If all profiles, rules, and probe and comparison functions are correctly set up, the `rules.ok` and `custom_probes.ok` files are created. Table 5–1 describes what the `check` script does.

TABLE 5–1 What Happens When You Use the check Script

Stage	Description
1	check searches for a custom_probes file.
2	If the file exists, check creates the custom_probes.ok file from the custom_probes file, removes all comments and blank lines, and retains all Bourne shell commands, variables, and algorithms. Then, check adds the following comment line at the end: `# version=2 checksum=num`

▼ To Validate the custom_probes File

1 **Verify that the check script is located in the JumpStart directory.**

Note – The check script is in the Solaris_10/Misc/jumpstart_sample directory on the Solaris Operating System DVD or on the Solaris Software - 1 CD.

---

2 **Change to the JumpStart directory.**

3 **Run the check script to validate the rules and custom_probes files.**

```
$ ./check -p path -r file_name
```

`-p` *path* — Validates the custom_probes file by using the check script from the Solaris software image for your platform instead of the check script from the system you are using. *path* is the image on a local disk or a mounted Solaris Operating System DVD or Solaris Software - 1 CD.

Use this option to run the most recent version of check if your system is running a previous version of Solaris.

`-r` *file_name* — Specifies a file name other than the one that is named custom_probes. By using the -r option, you can test the validity of a set of functions before integrating the functions into the custom_probes file.

As the check script runs, the script reports the validity of the rules and custom_probes files and each profile. If no errors are encountered, the script reports: "The custom JumpStart configuration is ok" and creates the rules.ok and custom_probes.ok files in the JumpStart directory.

4 **Determine if the custom_probes.ok file is executable.**

- If yes, go to Step 5.
- If no, type the following command.

```
# chmod +x custom_probes
```

5 **Ensure that** `root` **owns the** `custom_probes.ok` **file and that the permissions are set to 755.**

◆ ◆ ◆ CHAPTER 6

Performing a Custom JumpStart Installation (Tasks)

This chapter describes how to perform a custom JumpStart installation on a SPARC based or an x86 based system. You need to follow these procedures on the system on which you intend to install the Solaris software.

Note – If you are installing a Solaris ZFS root pool, see Chapter 9, "Installing a ZFS Root Pool With JumpStart," for limitations and profile examples.

Limitations for a JumpStart Installation

A number of issues might cause problems during a JumpStart installation. Review the table below for specific information.

TABLE 6–1 JumpStart Installation Limitations

Issue	Description	For More Information
The sample JumpStart script is no longer required to suppress the NFSv4 prompt	For the Solaris 10 release, a sample JumpStart script, `set_nfs4_domain`, was provided on media to prevent being prompted during a JumpStart installation. This script suppressed the NFSv4 prompt during installation. This script is no longer required. **Starting with the Solaris 10 8/07 release**, use the `sysidcfg` keyword, `nfs4_domain` that suppresses being prompted. The `set_nfs4_domain` script no longer works to suppress a prompt. If you have non-global zones installed and the new `nfs4_domain` keyword exists in the `sysidcfg` file, the first boot of a non-global zone sets the domain. Otherwise, the Solaris interactive installation program is displayed and you are prompted to provide a domain name before the boot process completes.	"nfs4_domain Keyword" in *Solaris 10 5/09 Installation Guide: Network-Based Installations*
Selecting a keyboard language in the `sysidcfg` file prevents a prompt	If your keyboard is not self-identifying and you want to prevent being prompted during your JumpStart installation, select the keyboard language in your `sysidcfg` file. For JumpStart installations, the default is for the U.S. English language. To select another language and its corresponding keyboard layout, set the keyboard keyword in your `sysidcfg` file.	■ "sysidcfg File Keywords" in *Solaris 10 5/09 Installation Guide: Network-Based Installations* ■ For man pages, see: ■ `sysidtool(1M)` ■ `sysidcfg(4)`
If you have non-global zones, use Solaris Live Upgrade to upgrade	You can upgrade a system that has non-global zones installed with JumpStart, but Solaris Live Upgrade is the recommended program to upgrade. JumpStart might require extensive upgrade time, because the time required to complete the upgrade increases linearly with the number of installed non-global zones.	*Solaris 10 5/09 Installation Guide: Solaris Live Upgrade and Upgrade Planning*
A Solaris Flash archive cannot contain non-global zones	If you use a Solaris Flash archive to install, an archive that contains non-global zones is not properly installed on your system.	For general information about creating non-global zones, see *System Administration Guide: Solaris Containers-Resource Management and Solaris Zones*.
A Solaris Flash archive can only be created and installed from a UFS file system.	You cannot create a Solaris Flash archive from a ZFS root pool. Also, you cannot install a Solaris Flash archive on a ZFS root pool.	
SPARC: Additional hardware requirements	Refer to your hardware documentation for any additional requirements for your platform that might be required to complete a JumpStart installation.	

SPARC: Task Map: Setting Up a System for a Custom JumpStart Installation

TABLE 6–2 Task Map: Setting Up a System for a Custom JumpStart Installation

Task	Description	For Instructions
Check if the system is supported.	Check the hardware documentation for system support in the Solaris environment.	*Solaris Sun Hardware Platform Guide* at `http://docs.sun.com`
Check if the system has enough disk space for the Solaris software.	Verify that you have planned enough space to install the Solaris software on your system.	Chapter 4, "System Requirements, Guidelines, and Upgrade (Planning)," in *Solaris 10 5/09 Installation Guide: Planning for Installation and Upgrade*
(Optional) Set system parameters.	You can preconfigure system information to avoid being prompted for the information during the installation or upgrade.	Chapter 2, "Preconfiguring System Configuration Information (Tasks)," in *Solaris 10 5/09 Installation Guide: Network-Based Installations*
Prepare the system for custom JumpStart installation.	Create and validate a `rules` file and profile files.	Chapter 3, "Preparing Custom JumpStart Installations (Tasks)"
(Optional) Prepare optional custom JumpStart features.	If you are using begin scripts, finish scripts, or other optional features, prepare the scripts or files.	Chapter 4, "Using Optional Custom JumpStart Features (Tasks)," and Chapter 5, "Creating Custom Rule and Probe Keywords (Tasks)"
(Optional) Prepare to install the Solaris software from the network.	To install a system from a remote Solaris Operating System DVD or Solaris Software for SPARC Platforms CD image, you need to set up the system to boot and install from an install server or a boot server.	Chapter 5, "Installing From the Network With DVD Media (Tasks)," in *Solaris 10 5/09 Installation Guide: Network-Based Installations* Chapter 6, "Installing From the Network With CD Media (Tasks)," in *Solaris 10 5/09 Installation Guide: Network-Based Installations*
(Optional) Prepare for a Solaris Flash archive installation.	Set up specifics for a Solaris Flash archive installation.	"To Prepare to Install a Solaris Flash Archive With a Custom JumpStart Installation" on page 84

TABLE 6–2 Task Map: Setting Up a System for a Custom JumpStart Installation *(Continued)*

Task	Description	For Instructions
Perform an installation or upgrade.	Boot the system to initiate the installation or upgrade.	"SPARC: To Perform an Installation or Upgrade With the Custom JumpStart Program" on page 86

SPARC: Performing a Custom JumpStart Installation

During a custom JumpStart installation, the JumpStart program attempts to match the system that is being installed to the rules in the `rules.ok` file. The JumpStart program reads the rules from the first rule through the last. A match occurs when the system that is being installed matches all the system attributes that are defined in the rule. When a system matches a rule, the JumpStart program stops reading the `rules.ok` file and begins to install the system, based on the matched rule's profile.

▼ To Prepare to Install a Solaris Flash Archive With a Custom JumpStart Installation

You can install a full archive for an initial installation or if you have already installed an archive, a differential archive for an update. You can use the custom JumpStart installation method or use Solaris Live Upgrade to install an archive on an inactive boot environment. This procedure provides the instructions to install an archive with custom JumpStart.

- For an overview of a full or differential archive, see Chapter 1, "Solaris Flash (Overview)," in *Solaris 10 5/09 Installation Guide: Solaris Flash Archives (Creation and Installation)*.
- For procedures about installing an archive on an inactive boot environment by using Solaris Live Upgrade, see "To Install a Solaris Flash Archive With a Profile" in *Solaris 10 5/09 Installation Guide: Solaris Live Upgrade and Upgrade Planning*.

1 Review the following limitations.

Description	Example
Caution: When using the `archive_location` keyword to install a Solaris Flash archive, the archive and the installation media must contain identical operating system versions.	For example, if the archive is a Solaris 10 5/09 operating system and you are using DVD media, then you must use Solaris 10 5/09 DVD media to install the archive. If the operating systems versions do not match, the installation on the clone system fails.

Description	Example
Caution – A Solaris Flash archive cannot be properly created when a non-global zone is installed. The Solaris Flash feature is not compatible with the Solaris Zones partitioning technology. If you create a Solaris Flash archive, the resulting archive is not installed properly when the archive is deployed under these conditions: ■ The archive is created in a non-global zone ■ The archive is created in a global zone that has non-global zones installed	

2 On the install server, create the custom JumpStart `rules` file.

For detailed instructions about creating custom JumpStart files, refer to Chapter 3, "Preparing Custom JumpStart Installations (Tasks)."

3 On the install server, create the custom JumpStart profile file.

For examples of Solaris Flash archive profiles, see "Profile Examples" on page 38.

From the existing list of custom JumpStart keywords in Table 8–2, the only keywords valid when you install a Solaris Flash archive are the following:

Keyword	Initial Installation	Differential Archive
(required)`archive_location`	X	X
`fdisk` (x86 only)	X	X
`filesys`	X	
Note – You cannot set the `filesys` keyword to the value `auto`.		
`forced_deployment`		X
(required) `install_type`	X	X
`local_customization`	X	X
`no_content_check`		X
`no_master_check`		X
`package`	X	
`root_device`	X	X

a. Set the value of the keyword `install_type` to one of the following types.

- For a full archive installation, set the value to `flash_install`.

- For a differential archive installation, set the value to `flash_update`.

b. Add the path to the Solaris Flash archive by using the `archive_location` keyword.

For details about the `archive_location` keyword, refer to "`archive_location` Keyword" on page 113.

c. Specify the file system configuration.

The Solaris Flash archive extraction process does not support auto-layout of partitions.

d. (Optional) If you want to install additional packages at the same time you install an archive, use the `package` keyword. For more information, see "`package` Profile Keyword (UFS and ZFS)" on page 142.

e. (Optional) If you want to install an additional Solaris Flash archive on the clone system, add one `archive_location` line for each archive that you want to install.

4 On the install server, add the clients that you are installing with the Solaris Flash archive.

For detailed instructions, refer to the following:

- "Adding Systems to Be Installed From the Network With a DVD Image" in *Solaris 10 5/09 Installation Guide: Network-Based Installations*
- "Adding Systems to Be Installed From the Network With a CD Image" in *Solaris 10 5/09 Installation Guide: Network-Based Installations*

5 Perform the custom JumpStart installation on the clone systems.

For detailed instructions, refer to "SPARC: To Perform an Installation or Upgrade With the Custom JumpStart Program" on page 86.

▼ SPARC: To Perform an Installation or Upgrade With the Custom JumpStart Program

1 If the system is part of a network, ensure that an Ethernet connector or similar network adapter is attached to your system.

2 If you are installing a system that is connected through a `tip(1)` line, ensure that your window display is at least 80 columns wide and 24 rows long.

To determine the current dimensions of your `tip` window, use the `stty(1)` command.

3 If you are using the system's DVD-ROM or CD-ROM drive to install the Solaris software, insert the Solaris Operating System for SPARC Platforms DVD or the Solaris Software for SPARC Platforms - 1 CD in the drive.

4 **If you are using a profile diskette, insert the profile diskette in the system's diskette drive.**

5 **Boot the system.**

- If the system is new, out–of–the–box, turn on the system.
- If you want to install or upgrade an existing system, shut down the system. At the `ok` prompt, type the appropriate options for the boot command. The syntax of the `boot` command is the following.

  ```
  ok boot [cd–dvd|net] - install [url|ask] options
  ```

  For example, if you type the following command, the OS is installed over the network by using a JumpStart profile.

  ```
  ok boot net - install http://131.141.2.32/jumpstart/config.tar
  ```

  For a description of the boot command options, see the following table.

SPARC only – The system checks hardware and system components and your SPARC based system boots. Booting lasts several minutes.

---

6 **If you did not preconfigure system information in the `sysidcfg` file, when prompted, answer the questions about system configuration.**

7 **Follow the instructions on the screen to install the software.**

When the JumpStart program finishes installing the Solaris software, the system reboots automatically.

After the installation is finished, installation logs are saved in a file. You can find the installation logs in the following directories:

- `/var/sadm/system/logs`
- `/var/sadm/install/logs`

SPARC: Command Reference for the `boot` Command

The syntax of the `boot` command is the following.

```
ok boot [cd–dvd|net] - install [url|ask] options
```

The following table describes the command-line options for the `boot` command that are appropriate for a JumpStart installation.

Option	Description
[*cd–dvd*|net]	Specifies to boot from a CD or a DVD or to boot from an install server on the network. ■ *cd-dvd* - Use `cdrom` to boot from a CD or a DVD. ■ `net` - Specifies to boot from an install server on the network.

[*url*| ask]	Specifies the location of the custom JumpStart files or prompts you for the location.
	■ *url* – Specifies the path to the files. You can specify a URL for files that are located in an HTTP or HTTPS server:
	HTTP server
	`http://`*server_name:IP_address/jumpstart_dir_path/compressed_config_file&proxy_info*
	■ If you placed a `sysidcfg` file in the compressed configuration file, you must specify the IP address of the server that contains the file, as in the following example:
	`http://131.141.2.32/jumpstart/config.tar`
	■ If you saved the compressed configuration file on an HTTP server that is behind a firewall, you must use a proxy specifier during boot. You do not need to specify an IP address for the server that contains the file. You must specify an IP address for the proxy server, as in the following example:
	`http://www.shadow.com/jumpstart/config.tar&proxy=131.141.6.151`
	■ `ask` – Specifies that the installation program prompt you to type the location of the compressed configuration file. The prompt happens after the system boots and connects to the network. If you use this option, you are not able to do a completely hands off JumpStart installation.
	If you bypass the prompt by pressing Return, the Solaris installation program interactively configures the network parameters. The installation program then prompts you for the location of the compressed configuration file.

options	■ `dhcp` – Specifies to use a DHCP server to obtain network installation information that is needed to boot the system. This option is not needed for a JumpStart installation. If you do not specify to use a DHCP server by typing `dhcp`, the system uses the `/etc/bootparams` file or the naming service `bootparams` database. For example, you would not specify `dhcp` if you wanted keep a static IP address. ■ The options `nowin` and `text` do not apply to a JumpStart installation. These options are useful with an interactive installation. For more information, see "To Install or Upgrade With the Solaris Installation Program" in *Solaris 10 5/09 Installation Guide: Basic Installations*.

x86: Task Map: Setting Up a System for a Custom JumpStart Installation

TABLE 6–3 x86: Task Map: Setting Up a System for a Custom JumpStart Installation

Task	Description	For Instructions
Determine if you need to preserve an existing operating system and user data.	If the existing operating system on the system uses the entire disk, you must preserve the existing operating system so it can co-exist with the Solaris 10 5/09 software. This decision determines how to specify the `fdisk`(1M) keyword in the system's profile.	"x86: `fdisk` Profile Keyword (UFS and ZFS)" on page 127
Check if the system is supported.	Check the hardware documentation for system support in the Solaris environment.	Hardware manufacturer's documentation
Check if the system has enough disk space for the Solaris software.	Verify that you have planned enough space to install the Solaris software on your system.	Chapter 4, "System Requirements, Guidelines, and Upgrade (Planning)," in *Solaris 10 5/09 Installation Guide: Planning for Installation and Upgrade*
(Optional) Set system parameters.	You can preconfigure system information to avoid being prompted for the information during the installation or upgrade.	Chapter 2, "Preconfiguring System Configuration Information (Tasks)," in *Solaris 10 5/09 Installation Guide: Network-Based Installations*
Prepare the system for custom JumpStart installation.	Create and validate a `rules` file and profile files.	Chapter 3, "Preparing Custom JumpStart Installations (Tasks)"
(Optional) Prepare optional custom JumpStart features.	If you are using begin scripts, finish scripts, or other optional features, prepare the scripts or files.	Chapter 4, "Using Optional Custom JumpStart Features (Tasks)," and Chapter 5, "Creating Custom Rule and Probe Keywords (Tasks)"
(Optional) Prepare to install the Solaris software from the network.	To install a system from a remote Solaris Operating System for x86 Platforms DVD or Solaris Software For x86 Platforms CD image, you need to set up the system to boot and install from an install server or a boot server.	Chapter 6, "Installing From the Network With CD Media (Tasks)," in *Solaris 10 5/09 Installation Guide: Network-Based Installations*

TABLE 6–3 x86: Task Map: Setting Up a System for a Custom JumpStart Installation *(Continued)*

Task	Description	For Instructions
(Optional) Prepare for a Solaris Flash archive installation.	Set up specifics for a Solaris Flash archive installation.	"To Prepare to Install a Solaris Flash Archive With a Custom JumpStart Installation" on page 84
Perform an installation or upgrade.	Boot the system to initiate the installation or upgrade.	"x86: To Perform an Installation or Upgrade With the Custom JumpStart Program and With GRUB" on page 90

x86: Performing a Custom JumpStart Installation

During a custom JumpStart installation, the JumpStart program attempts to match the system that is being installed to the rules in the `rules.ok` file. The JumpStart program reads the rules from the first rule through the last rule. A match occurs when the system that is being installed matches all of the system attributes that are defined in the rule. As soon as a system matches a rule, the JumpStart program stops reading the `rules.ok` file and begins to install the system, based on the matched rule's profile.

You can install a Solaris Flash archive with custom JumpStart. For instructions, see "To Prepare to Install a Solaris Flash Archive With a Custom JumpStart Installation" on page 84.

Choose one of the following procedures:

- For a standard custom JumpStart procedure, see "x86: To Perform an Installation or Upgrade With the Custom JumpStart Program and With GRUB" on page 90.
- To perform a custom JumpStart by editing the GRUB command, see "x86: Performing a Custom JumpStart Installation by Editing the GRUB Boot Command" on page 92.

▼ x86: To Perform an Installation or Upgrade With the Custom JumpStart Program and With GRUB

Use this procedure to install the Solaris OS for an x86 based system with the GRUB menu.

1 If the system is part of a network, ensure that an Ethernet connector or similar network adapter is attached to your system.

2 If you want to install a system that is connected through a `tip(1)` line, ensure that your window display is at least 80 columns wide and 24 rows long.

To determine the current dimensions of your `tip` window, use the `stty(1)` command.

3 Decide if you want to use a profile diskette.

A profile diskette is no longer used to boot the system but, a diskette can be prepared that includes only the JumpStart directory. The diskette can then be used situations such as performing a JumpStart installation and booting off the CD-ROM.

- If you are using a profile diskette, insert the profile diskette into the system's diskette drive.
- If you are not using a profile diskette, continue with step Step 4.

4 Decide how to boot the system.

- If you boot from the Solaris Operating System DVD or the Solaris Software - 1 CD, insert the disc. Your system's BIOS must support booting from a DVD or CD.
- If you boot from the network, use Preboot Execution Environment (PXE) network boot. The system must support PXE. Enable the system to use PXE by using the system's BIOS setup tool or the network adapter's configuration setup tool.

5 (Optional) If you are booting from a DVD or CD, change the boot setting in your system's BIOS and set to boot from DVD or CD media. See your hardware documentation for instructions.

6 If the system is off, turn the system on. If the system is on, reboot the system.

The GRUB menu is displayed. This menu provides a list of boot entries.

```
GNU GRUB version 0.95 (631K lower / 2095488K upper memory)
+-------------------------------------------------------------------+
|Solaris 10 5/09 image_directory                                    |
|Solaris Serial Console ttya                                          |
|Solaris Serial Console ttyb (for lx50, v60x and v65x               |
+-------------------------------------------------------------------+
Use the ^ and v keys to select which entry is highlighted. Press
enter to boot the selected OS, 'e' to edit the commands before
booting, or 'c' for a command-line.
```

The *image_directory* is the name of the directory where the installation image is located. The path to the JumpStart files was defined with the `add_install_client` command and the `-c` option.

Note – Instead of booting from the GRUB entry now, you can edit the boot entry. After editing the GRUB entry, you then perform the JumpStart installation. For instructions about how to edit the GRUB entry and a list of installation options, see "x86: Performing a Custom JumpStart Installation by Editing the GRUB Boot Command" on page 92.

---

7 At the prompt, perform one of the following instructions:

```
Select the type of installation you want to perform:

         1 Solaris Interactive
```

```
        2 Custom JumpStart
        3 Solaris Interactive Text (Desktop session)
        4 Solaris Interactive Text (Console session)
        5. Apply driver updates
        6. Single User Shell
Enter the number of your choice.
Please make a selection (1-6).
```

To select the custom JumpStart method, type **2** and press Enter.

The JumpStart installation begins.

Note –

- If you do not make a selection within 30 seconds, the Solaris interactive installation program begins. You can stop the timer by typing any key at the command line.
- If you select items 1, 3, or 4, you install with an interactive installation. For information about interactive installations, see *Solaris 10 5/09 Installation Guide: Basic Installations*.
- If you select item 5, you install driver updates.
- If you select item 6, you can perform maintenance tasks.

---

8 If you did not preconfigure system information in the `sysidcfg` file, when prompted, answer the questions about system configuration.

9 Follow the instructions on the screen to install the software.

When the JumpStart program finishes installing the Solaris software, the system reboots automatically. Also, the GRUB `menu.lst` file is automatically updated. Then the instance of Solaris that you have installed appears in the next use of the GRUB menu.

After the installation is finished, installation logs are saved in a file. You can find the installation logs in the following directories:

- `/var/sadm/system/logs`
- `/var/sadm/install/logs`

x86: Performing a Custom JumpStart Installation by Editing the GRUB Boot Command

In some circumstances such as for debugging purposes, you might want to modify the GRUB boot command. The following procedure describes the steps to edit the GRUB boot command before performing the custom JumpStart installation.

▼ x86: To Modify the GRUB Boot Command

1 **To begin the installation, proceed with Step 1 through Step 5 in the preceding procedure, "x86: To Perform an Installation or Upgrade With the Custom JumpStart Program and With GRUB" on page 90.**

2 **If the system is off, turn the system on. If the system is on, reboot the system.**

The GRUB menu is displayed. This menu provides a list of boot entries. The entry that is provided is the Solaris instance to be installed.

```
GNU GRUB version 0.95 (631K lower / 2095488K upper memory)
+-------------------------------------------------------------------+
|Solaris 10 5/09 image_directory                                    |
|Solaris Serial Console ttya                                              |
|Solaris Serial Console ttyb (lx50, v60x and v68)                 |
+-------------------------------------------------------------------+
Use the ^ and v keys to select which entry is highlighted. Press
enter to boot the selected OS, 'e' to edit the commands before
booting, or 'c' for a command-line.
```

The *image_directory* is the name of the directory where the installation image is located.

Note –

- If you used the NFS to set the path to the JumpStart directory with the `add_install_client` command and the `-c` option, then you do not need to include the path in the boot entry.
- If you are not using NFS, then you must note the path to the compressed configuration file that contains the JumpStart directory.

---

3 **To stop the booting process and use the menu entry editor, type e.**

The GRUB edit menu is displayed.

```
kernel /I86PC.Solaris_11-8/multiboot kernel/unix -B console=ttyb,\
install_media=131.141.2.32:/export/mary/v11 \
module /I86PC.Solaris_11-8/x86.new
```

4 **Use the arrow keys to select the boot entry.**

5 **To edit the selected command, type e.**

A command that is similar to the following example displays.

```
grub edit>kernel /I86PC.Solaris_11-8/multiboot kernel/unix -B \
console=ttyb,install_media=131.141.2.32:/export/mary/_\
module /I86PC.Solaris_11-8/x86.new
```

6 Edit the command by typing the options that you need.

The syntax for a JumpStart installation is the following.

```
grub edit>kernel /I86PC.Solaris_11-image_directory/multiboot kernel/unix/ \
- install [url|ask] options -B install_media=media_type
```

For a description of JumpStart options, see "x86: Command Reference for Booting the System" on page 94.

In the following example, the OS is installed over the network with a custom JumpStart profile.

```
kernel /I86PC.Solaris_11-8/multiboot kernel/unix/ - install \
-B install_media=131.141.2.32:/export/mary/v11 \
module /I86PC.Solaris_11-8/x86.new
```

7 To accept the edits, press Enter.

Your changes are saved and the GRUB main menu is displayed.

Note – Pressing the Escape key returns you to the GRUB main menu without saving your changes.

---

8 To begin the installation, type b.

x86: Command Reference for Booting the System

The following table describes the command-line options for the GRUB menu boot command. The options listed are appropriate for a JumpStart installation.

The syntax of the boot command is the following.

```
kernel /I86PC.Solaris_11-image_directory/multiboot kernel/unix/ - install \
[url|ask] options -B install_media=media_type
```

TABLE 6–4 GRUB Menu Boot Command Reference

Option	Description

`- install`	Performs a custom JumpStart installation.
	
	In the following example, the system boots from DVD media and the following options were used:
	■ `- install` performs a custom JumpStart
	■ `file://jumpstart/config.tar` finds the JumpStart profile on the local disk
	
	`kernel /I86pc.Solaris_11.8/multiboot - install file://jumpstart/config.tar \`
	`-B install_media=dvdrom module /I86Solaris_11.8/x86.new`

TABLE 6–4 GRUB Menu Boot Command Reference *(Continued)*

Option	Description

[*url*| ask]	Specifies the location of the custom JumpStart files or prompts you for the location.
	■ *url* – Specifies the path to the files. You can specify a URL for files that are located on an HTTP or HTTPS server:
	The syntax for an HTTP server is the following:
	`http://`*server_name:IP_address/jumpstart_dir_path/ compressed_config_file&proxy_info*
	■ If you placed a `sysidcfg` file in the compressed configuration file, you must specify the IP address of the server that contains the file, as in the following example:
	`kernel /I86pc.Solaris_11.8/multiboot` **`install`** `\`
	`http://192.168.2.1/jumpstart/config.tar` `\`
	`-B install_media=192.168.2.1/export/Solaris_11.8/boot \`
	`module /I86PC.Solaris_11.8/x86.new`
	■ If you saved the compressed configuration file on an HTTP server that is behind a firewall, you must use a proxy specifier during boot. You do not need to specify an IP address for the server that contains the file. You must specify an IP address for the proxy server, as in the following example:
	`kernel /I86pc.Solaris_11.8/multiboot` **`install`** `\`
	`http://www.shadow.com/jumpstart/config.tar&proxy=131.141.6.151` `\`
	`-B install_media=192.168.2.1/export/Solaris_11.8/boot \`
	`module /I86PC.Solaris_11.8/x86.new`
	■ `ask` – Specifies that the installation program prompt you to type the location of the compressed configuration file. You are prompted after the system boots and connects to the network. If you use this option, you are not able to do a completely hands off JumpStart installation.
	If you bypass the prompt by pressing Return, the Solaris installation program interactively configures the network parameters. The installation program then prompts you for the location of the compressed configuration file.
	The following example performs a custom JumpStart and boots from DVD media. You are prompted to type the location of the configuration file after the system connects to the network.
	`kernal /boot/multiboot kernel/unix` **`install ask`** `-B \`
	`install_media=192.168.2.1:export/sol_11_x86/boot module \`
	`/I86PC.Solaris_11.8_`

TABLE 6–4 GRUB Menu Boot Command Reference *(Continued)*

Option	Description

options	■ `dhcp` – Specifies to use a DHCP server to obtain network installation information that is needed to boot the system. This option is not needed for a JumpStart installation. If you do not specify to use a DHCP server by typing `dhcp`, the system uses the `/etc/bootparams` file or the naming service `bootparams` database. For example, you would not specify `dhcp` if you wanted keep a static IP address. For example:
	
	`kernel /I86pc.Solaris_11.8/multiboot` **`install`** `\`
	`dhcp` `-B install_media=192.168.2.1:/export/Solaris_11.8/ \`
	`boot module /I86PC.Solaris_11.8/x86.new`
	
	■ The options `nowin` and `text` do not apply to a JumpStart installation. These options are useful with an interactive installation. For more information, see "To Install or Upgrade With the Solaris Installation Program With GRUB" in *Solaris 10 5/09 Installation Guide: Basic Installations*.

◆ ◆ ◆ CHAPTER 7

Installing With Custom JumpStart (Examples)

This chapter provides an example of setting up and installing Solaris software on both SPARC based and x86 based systems by using a custom JumpStart installation.

Note – If you are installing a Solaris ZFS root pool, see Chapter 9, "Installing a ZFS Root Pool With JumpStart," for limitations and profile examples.

Sample Site Setup

Figure 7–1 shows the site setup for this example.

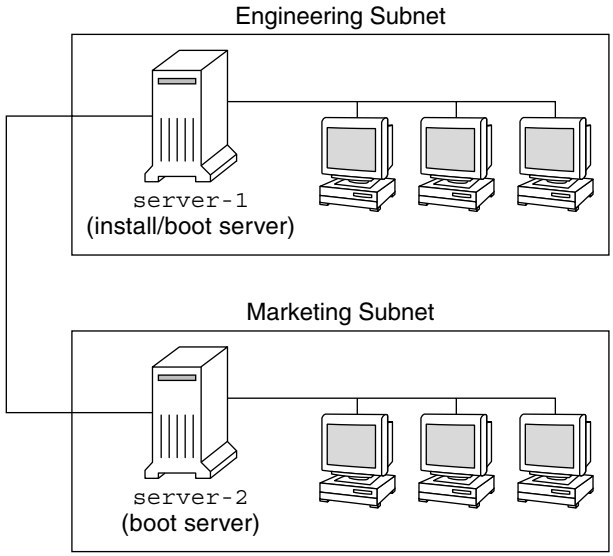

FIGURE 7–1 Sample Site Setup

At this sample site, the conditions are as follows:

- SPARC: The engineering group is located on its own subnet. This group uses SPARCstation™ systems for software development.
- x86: The marketing group is located on its own subnet. This group uses x86 based systems for running word processors, spreadsheets, and other office productivity tools.
- The site uses NIS. The Ethernet addresses, IP addresses, and host names of the systems are preconfigured in the NIS maps. The subnet mask, date and time, and geographic region for the site are also preconfigured in the NIS maps.

  Note – The peripheral devices for the marketing systems are preconfigured in the `sysidcfg` file.

- Both the engineering and marketing systems are to be installed with Solaris 10 5/09 software from the network.

Create an Install Server

Because the groups need to install Solaris 10 5/09 software from the network, you make `server-1` an install server for both groups. You use the `setup_install_server(1M)` command to copy the images to the `server-1` local disk (in the `/export/install` directory). Copy the images from the either of the following media.

- Solaris Software CDs and the Solaris Languages CDs
- Solaris Operating System DVD

You must copy the image from the disc to an empty directory, in these examples the `sparc_10` directory and the `x86_10` directory.

EXAMPLE 7–1 SPARC: Copying the Solaris 10 5/09 CDs

Insert the Solaris Software for SPARC Platforms - 1 CD in the CD-ROM drive that is attached to `server-1` and type the following commands:

```
server-1# mkdir -p /export/install/sparc_10
server-1# cd /CD_mount_point/Solaris_10/Tools
server-1# ./setup_install_server /export/install/sparc_10
```

Insert the Solaris Software for SPARC Platforms - 2 CD in the CD-ROM drive that is attached to `server-1` and type the following commands:

```
server-1# cd /CD_mount_point/Solaris_10/Tools
server-1# ./add_to_install_server /export/install/sparc_10
```

Repeat the previous command for each Solaris Software you want to install.

Insert the first SPARC: Solaris Languages for SPARC Platforms CD in the CD-ROM drive that is attached to `server-1` and type the following commands:

```
server-1# cd /CD_mount_point/Solaris_10/Tools
server-1# ./add_to_install_server /export/install/sparc_10
```

Repeat the previous command for each SPARC: Solaris Languages for SPARC Platforms CD.

EXAMPLE 7–2 x86: Copying the Solaris 10 5/09 CDs

Insert the Solaris Software for x86 Platforms - 1 CD in the CD-ROM drive that is attached to `server-1` and type the following commands:

```
server-1# mkdir -p /export/install/x86_10
server-1# cd /CD_mount_point/Solaris_10/Tools
server-1# ./setup_install_server /export/install/x86_10
```

EXAMPLE 7–2 x86: Copying the Solaris 10 5/09 CDs *(Continued)*

Insert the Solaris Software for x86 Platforms - 2 CD in the CD-ROM drive that is attached to `server-1` and type the following commands:

```
server-1# cd /CD_mount_point/Solaris_10/Tools
server-1# ./add_to_install_server /export/install/x86_10
```

Repeat the previous command for each Solaris Software you want to install.

Insert the first Solaris Languages for x86 Platforms CD in the CD-ROM drive that is attached to `server-1` and type the following commands:

```
server-1# cd /CD_mount_point/Solaris_10/Tools
server-1# ./add_to_install_server /export/install/x86_10
```

Repeat the previous command for each Solaris Languages for x86 Platforms CD.

EXAMPLE 7–3 SPARC: Copying the Solaris 10 5/09 DVD

Insert the Solaris Operating System for SPARC Platforms DVD in the DVD-ROM drive that is attached to `server-1` and type the following commands:

```
server-1# mkdir -p /export/install/sparc_10
server-1# cd /DVD_mount_point/Solaris_10/Tools
server-1# ./setup_install_server /export/install/sparc_10
```

EXAMPLE 7–4 x86: Copying the Solaris Operating System for x86 Platforms DVD

Insert the Solaris Operating System for x86 Platforms DVD in the DVD-ROM drive that is attached to `server-1` and type the following commands:

```
server-1# mkdir -p /export/install/x86_10
server-1# cd /DVD_mount_point/Solaris_10/Tools
server-1# ./setup_install_server /export/install/x86_10
```

x86: Create a Boot Server for Marketing Systems

Systems cannot boot from an install server on a different subnet, so you make `server-2` a boot server on the marketing group's subnet. You use the `setup_install_server(1M)` command to copy the boot software from the Solaris Operating System for x86 Platforms DVD or the Solaris Software for x86 Platforms - 1 CD. The boot software is copied to the `server-2` local disk in the `/export/boot` directory.

Choose the media and install the boot software to local disk.

- If you insert the Solaris Software for x86 Platforms - 1 CD in the CD-ROM drive that is attached to `server-2`, type the following command:

```
server-2# cd /CD_mount_point/Solaris_10/Tools
server-2# ./setup_install_server -b /export/boot
```

- If you insert the Solaris Operating System for x86 Platforms DVD in the DVD-ROM drive that is attached to `server-2`, type the following command:

```
server-2# cd /DVD_mount_point/Solaris_10/Tools
server-2# ./setup_install_server -b /export/boot
```

In the `setup_install_server` command, `-b` specifies that `setup_install_server` is to copy the boot information to the directory that is named `/export/boot`.

Create a JumpStart Directory

Now that you have the install server and boot server set up, you create a JumpStart directory on `server-1`. You can use any system on the network. This directory holds files that are required for a custom JumpStart installation of Solaris software. You set up this directory by copying the sample directory from the Solaris Operating System DVD image or from the Solaris Software - 1 CD image that has been copied to `/export/install`:

```
server-1# mkdir /jumpstart
server-1# cp -r /export/install/sparc_10/Solaris_10/Misc/jumpstart_sample /jumpstart
```

Share the JumpStart Directory

To make the `rules` file and profiles accessible to systems on the network, you share the `/jumpstart` directory. To enable the sharing of a directory, you add the following line to the `/etc/dfs/dfstab` file:

```
share -F nfs -o ro,anon=0 /jumpstart
```

Then, at the command line, you type the `shareall` command:

```
server-1# shareall
```

SPARC: Create the Engineering Group's Profile

For the engineering systems, you create a file that is named `eng_prof` in the `/jumpstart` directory. The `eng_prof` file contains the following entries, which define the Solaris 10 5/09 software to be installed on systems in the engineering group:

```
install_type  initial_install
system_type   standalone
partitioning  default
cluster       SUNWCprog
filesys       any 512 swap
```

The previous example profile specifies the following installation information.

`install_type`	The installation is to be treated as an initial installation, as opposed to an upgrade.
`system_type`	The engineering systems are standalone systems.
`partitioning`	The JumpStart software uses default disk partitioning for installing Solaris software on the engineering systems.
`cluster`	The Developer System Support software group is to be installed.
`filesys`	Each system in the engineering group is to have 512 Mbytes of swap space.

x86: Create the Marketing Group's Profile

For the marketing systems, you create a file that is named `marketing_prof` in the `/jumpstart` directory. The `marketing_prof` file contains the following entries, which define the Solaris 10 5/09 software to be installed on systems in the marketing group:

```
install_type  initial_install
system_type   standalone
partitioning  default
cluster       SUNWCuser
package       SUNWaudio
```

The previous example profile specifies the following installation information.

`install_type`	The installation is to be treated as an initial installation, as opposed to an upgrade.
`system_type`	The marketing systems are standalone systems.
`partitioning`	The JumpStart software is to use default disk partitioning for installing Solaris on the marketing systems.

`cluster`	The End User Solaris Software Group is to be installed.
`package`	The audio demo software package is to be added to each system.

Update the `rules` File

Now you must add rules to the `rules` file. The Solaris installation program uses the rules to select the correct installation (profile) for each system during a custom JumpStart installation.

At this site, each department is located on its own *subnet* and has its own network address. The engineering department is located on subnet 255.222.43.0. The marketing department is located on 255.222.44.0. You can use this information to control how the engineering and marketing systems are installed with the Solaris 10 5/09 software. In the `/jumpstart` directory, you edit the `rules` file, delete all of the example rules, and add the following lines to the file:

```
network 255.222.43.0 - eng_prof -
network 255.222.44.0 - marketing_prof -
```

Basically, these rules state that systems on the 255.222.43.0 network are to be installed with the Solaris 10 5/09 software by using the `eng_prof` profile. The systems on the 255.222.44.0 network are to be installed with the Solaris 10 5/09 software by using the `marketing_prof` profile.

Note – You can use the sample rules to use a network address to identify the systems to be installed with the Solaris 10 5/09 software by using `eng_prof` and `marketing_prof`, respectively. You can also use host names, memory size, or model type as the rule keyword. Table 8–1 contains a complete list of keywords you can use in a `rules` file.

Validate the `rules` File

After the `rules` and profiles are set up, you run the `check` script to verify that the files are correct:

```
server-1# cd /jumpstart
server-1# ./check
```

If the `check` script does not find any errors, the script creates the `rules.ok` file.

SPARC: Set Up Engineering Systems to Install From the Network

After setting up the `/jumpstart` directory and files, you use the `add_install_client` command on the install server, `server-1`, to set up the engineering systems to install the Solaris software from the install server. `server-1` is also the boot server for the engineering group's subnet.

```
server-1# cd /export/install/sparc_10/Solaris_10/Tools
server-1# ./add_install_client -c server-1:/jumpstart host-eng1 sun4u
server-1# ./add_install_client -c server-1:/jumpstart host-eng2 sun4u
```

In the `add_install_client` command, the options that are used have the following meanings:

`-c` Specifies the server (`server-1`) and path (`/jumpstart`) to the JumpStart directory. Use this option if you are using NFS.

Note – If you are not using NFS, you specify the path to the JumpStart directory by using the following commands:

- **For SPARC based systems**, specify the path in the `boot` command
- **For x86 based systems**, specify the path by editing the GRUB menu entry

---

host-eng1 The name of a system in the engineering group.

host-eng2 The name of another system in the engineering group.

sun4u Specifies the platform group of the systems that use `server-1` as an install server. The platform group is for Ultra 5 systems.

x86: Set Up Marketing Systems to Install From the Network

Next, you use the `add_install_client` command on the boot server (`server-2`). This command sets up the marketing systems to boot from the boot server and install the Solaris software from the install server (`server-1`):

```
server-2# cd /marketing/boot-dir/Solaris_10/Tools
server-2# ./add_install_client -s server-1:/export/install/x86_10 \
-c server-1:/jumpstart host-mkt1 i86pc
server-2# ./add_install_client -s server-1:/export/install/x86_10 \
-c server-1:/jumpstart host-mkt2 i86pc
server-2# ./add_install_client -d -s server-1:/export/install/x86_10 \
-c server-1:/jumpstart SUNW.i86pc i86pc
```

```
server-2# ./add_install_client -c server-1:/jumpstart host-mkt1 sun4u
server-2# ./add_install_client -c server-1:/jumpstart host-mkt2 sun4u
```

In the `add_install_client` command, the options that are used have the following meanings:

-d	Specifies that the client is to use DHCP to obtain the network install parameters. This option is required for clients to use PXE network boot to boot from the network. `-d` is optional for network boot clients that do not use PXE network boot.
-s	Specifies the install server (`server-1`) and the path to the Solaris software (`/export/install/x86_10`).
-c	Specifies the server (`server-1`) and path (`/jumpstart`) to the JumpStart directory. Use this option if you are using NFS.

Note – If you are not using NFS, you specify the path to the JumpStart directory by using the following commands:

- **For SPARC based systems**, specify the path in the `boot` command
- **For x86 based systems**, specify the path by editing the GRUB menu entry

host-mkt1	The name of a system in the marketing group.
host-mkt2	The name of another system in the marketing group.
sun4u	Specifies the platform group of the systems that use `server-1` as an install server. The platform group is for Ultra 5 systems.
SUNW.i86pc	The DHCP class name for all Solaris x86 clients. If you want to configure all Solaris x86 DHCP clients with a single command, use this class name.
`i86pc`	Specifies the platform group of the systems that use this boot server. The platform name represents x86 based systems.

SPARC: Boot the Engineering Systems and Install Solaris Software

After setting up the servers and files, you can boot the engineering systems by using the following `boot` command at the `ok` (PROM) prompt of each system:

```
ok boot net - install
```

The Solaris OS is automatically installed on the engineering group's systems.

x86: Boot the Marketing Systems and Install Solaris Software

You can boot the system from one of the following:

- Solaris Software for x86 Platforms - 1 CD
- Solaris Operating System for x86 Platforms DVD
- The network by using PXE network boot

Solaris software is automatically installed on the marketing group's systems.

◆ ◆ ◆ CHAPTER 8

Custom JumpStart (Reference)

This chapter lists keywords and values that you can use in the `rules` file, profiles, and begin and finish scripts.

Note – If you are installing a Solaris ZFS root pool, see Chapter 9, "Installing a ZFS Root Pool With JumpStart," for limitations and profile examples. Also, for a list of ZFS-specific keywords and keywords that can be used within a profile, see Table 8–2.

Rule Keywords and Values

Table 8–1 describes the keywords and values that you can use in the `rules` file. For detailed instructions to create a `rules` file, see "Creating the `rules` File" on page 33.

TABLE 8–1 Descriptions of Rule Keywords and Values

Keyword	Value	Matches
`any`	minus sign (`-`)	Anything. The `any` keyword always succeeds.
`arch`	*processor_type* Valid values for *processor_type* are the following: ■ SPARC: `sparc` ■ x86: `i386`	A system's processor type. The `uname -p` command reports the system's processor type.

TABLE 8–1 Descriptions of Rule Keywords and Values *(Continued)*

Keyword	Value	Matches

`disksize`	*actual_disk_name size_range*
	
	actual_disk_name – A disk name in the form `c`*x*`t`*y*`d`*z*, such as `c0t3d0` or `c0d0`, or the special word `rootdisk`. If `rootdisk` is used, the disk to be matched is determined in the following order:
	■ SPARC: The disk that contains the preinstalled boot image, which is a new SPARC based system with factory JumpStart installed
	■ The `c0t3d0s0` disk, if the disk exists
	■ The first available disk that is searched in kernel probe order
	
	size_range – The size of the disk, which must be specified as a range of Mbytes (*x*-*x*).
	
	Note – When calculating *size_range*, remember that a Mbyte equals 1,048,576 bytes. A disk might be advertised as a "535–Mbyte" disk, but the disk might contain only 510 million bytes of disk space. The JumpStart program views the "535–Mbyte" disk as a 510–Mbyte disk because 535,000,000 / 1,048,576 = 510. A "535–Mbyte" disk does not match a *size_range* equal to 530–550.
		The name and size of a system's disk in Mbytes.
		
		Example:
		
		`disksize c0t3d0 250-300`
		
		In the example, the JumpStart program attempts to match a system disk that is named `c0t3d0`. The disk can hold between 250 and 300 Mbytes of information.
		
		Example:
		
		`disksize rootdisk 750-1000`
		
		In the example, the JumpStart program attempts to match a disk in the following order:
		1. A system disk that contains a preinstalled boot image
		2. The `c0t3d0s0` disk, if the disk exists
		3. The first available disk that can hold between 750 Mbytes and 1 Gbyte of information

`domainname`	*actual_domain_name*	A system's domain name, which controls how a naming service determines information. If you have a system already installed, the `domainname` command reports the system's domain name.
`hostaddress`	*actual_IP_address*	A system's IP address.
`hostname`	*actual_host_name*	A system's host name. If you have a system that is already installed, the `uname -n` command reports the system's host name.

TABLE 8–1 Descriptions of Rule Keywords and Values *(Continued)*

Keyword	Value	Matches
`installed`	*slice version* *slice* – A disk slice name in the form `c`*w*`t`*x*`d`*y*`s`*z*, such as `c0t3d0s5`, or the special words `any` or `rootdisk`. If `any` is used, the JumpStart program attempts to match all of the system's disks in kernel probe order. If `rootdisk` is used, the disk to be matched is determined in the following order: ■ SPARC: The disk that contains the preinstalled boot image, which is a new SPARC based system with factory JumpStart installed ■ The `c0t3d0s0` disk, if the disk exists ■ The first available disk that is searched in kernel probe order *version* – A version name or the special words `any` or `upgrade`. If `any` is used, any Solaris or SunOS release is matched. If `upgrade` is used, any Solaris release that is supported and can be upgraded is matched. If the JumpStart program finds a Solaris release but is unable to determine the version, the version that is returned is `SystemV`.	A disk that has a root (`/`) file system that corresponds to a particular version of Solaris software. Example: `installed c0t3d0s1 Solaris 10` In the example, the JumpStart program attempts to match a system that has a Solaris root (`/`) file system on `c0t3d0s1`.
`karch`	*actual_platform_group* Valid values are `sun4u`, `i86pc`, and `prep`. A list of systems and their corresponding platform group is presented in the *Solaris Sun Hardware Platform Guide* at `http://docs.sun.com`.	A system's platform group. If you have a system that is already installed, the `arch -k` command or the `uname -m` command reports the system's platform group.
`memsize`	*physical_mem* The value must be a range of Mbytes, *x*-*x*, or a single Mbyte value.	A system's physical memory size in Mbytes. Example: `memsize 64-128` The example tries to match a system with a physical memory size between 64 and 128 Mbytes. If you have a system that is already installed, the output of the `prtconf` command, line 2, reports the system's physical memory size.

TABLE 8–1 Descriptions of Rule Keywords and Values *(Continued)*

Keyword	Value	Matches
`model`	*actual_platform_name*	A system's platform name. See the *Solaris Sun Hardware Platform Guide* at `http://docs.sun.com` for a list of valid platform names. To find the platform name of an installed system, use the `uname -i` command or the output of the `prtconf` command, line 5. **Note –** If the *actual_platform_name* contains spaces, you must replace spaces with underscores (_). Example: SUNW,Sun_4_50
`network`	*network_num*	A system's network number, which the JumpStart program determines by performing a logical AND between the system's IP address and the subnet mask. Example: `network 192.168.2.0` The example tries to match a system with a 192.168.2.8 IP address, if the subnet mask is 255.255.255.0.
`osname`	`Solaris_`*x*	A version of Solaris software that is already installed on a system. Example: `osname Solaris 10` In the example, the JumpStart program attempts to match a system with the Solaris 10 5/09 OS already installed.

TABLE 8–1 Descriptions of Rule Keywords and Values *(Continued)*

Keyword	Value	Matches
`probe`	*probe_keyword*	A valid probe keyword or a valid custom probe keyword. Example: `probe disks` The example returns the size of a system's disks in Mbytes and in kernel probe order, for example, `c0t3d0s1`, `c0t4d0s0`, on a SPARC based system. The JumpStart program sets the `SI_DISKLIST`, `SI_DISKSIZES`, `SI_NUMDISKS`, and `SI_TOTALDISK` environment variables. **Note –** The `probe` keyword is unique in that the keyword does not attempt to match an attribute and run a profile. The `probe` keyword returns a value. Consequently, you cannot specify begin scripts, profiles, and finish scripts with the `probe` rule keyword. Probe keywords are described in Chapter 5, "Creating Custom Rule and Probe Keywords (Tasks)."
`totaldisk`	*size_range* The value must be specified as a range of Mbytes (*x*-*x*). **Note –** When calculating *size_range*, remember that one Mbyte equals 1,048,576 bytes. A disk might be advertised as a "535–Mbyte" disk, but the disk might have only 510 million bytes of disk space. The JumpStart program views the "535–Mbyte" disk as a 510–Mbyte disk because 535,000,000 / 1,048,576 = 510. A "535–Mbyte" disk does not match a *size_range* equal to 530–550.	The total disk space on a system in Mbytes. The total disk space includes all the operational disks that are attached to a system. Example: `totaldisk 300-500` In the example, the JumpStart program tries to match a system with a total disk space between 300 and 500 Mbytes.

Profile Keywords and Values

This section describes the profile keywords and values that you can use in a profile. For detailed instructions to create a profile, see "Creating a Profile" on page 36. These keywords are for installing UFS and ZFS file systems. If the keyword can be used in a ZFS profile, the term "ZFS" is noted.

Profile Keywords Quick Reference

Table 8–2 provides a quick way to determine which keywords you can use, based on your installation scenario. Unless otherwise noted in the keyword descriptions, the keyword can only be used with the initial installation option. Also, these keywords are for a UFS file system unless noted that the keyword can be used in a ZFS root pool profile.

TABLE 8–2 Profile Keywords Overview

	Installation Scenarios					
Profile Keyword	**Standalone System (Nonnetworked)**	**Standalone System (Networked) or Server**	**OS Server**	**Upgrade**	**Upgrade With Disk Space Reallocation**	**Can be Used for a ZFS Root Pool**
`archive_location` (installing Solaris Flash archives)	X	X				
`backup_media`					X	
`boot_device` (UFS and ZFS)	X	X	X			X
`bootenv` (UFS and ZFS)	X	X	X			X
`client_arch`			X			
`client_root`			X			
`client_swap`			X			
`cluster` (adding software groups) (UFS and ZFS)	X	X	X			X
`cluster` (adding or deleting clusters) (UFS and ZFS)	X	X	X	X	X	X
`dontuse` (UFS and ZFS)	X	X	X			X
`fdisk` (x86 only) (UFS and ZFS)	X	X	X			X
`filesys` (mounting remote file systems) (UFS and ZFS)		X	X			X
`filesys` (creating local file systems)	X	X	X			
`filesys` (creating mirrored file systems)	X	X	X			
`forced_deployment` (installing Solaris Flash differential archives)	X	X				

TABLE 8–2 Profile Keywords Overview *(Continued)*

	Installation Scenarios					
Profile Keyword	**Standalone System (Nonnetworked)**	**Standalone System (Networked) or Server**	**OS Server**	**Upgrade**	**Upgrade With Disk Space Reallocation**	**Can be Used for a ZFS Root Pool**
`geo` (UFS and ZFS)	X	X	X	X	X	X
`install_type` (UFS and ZFS)	X	X	X	X	X	X
`layout_constraint`					X	
`local_customization` (installing Solaris Flash archives)	X	X				
`locale` (UFS and ZFS)	X	X	X	X	X	X
`metadb` (creating state database replicas)	X	X	X			
`no_master_check` (installing Solaris Flash differential archives)	X	X				
`no_content_check` (installing Solaris Flash differential archives)	X	X				
`num_clients`			X			
`package` (UFS and ZFS)	X	X	X	X	X	X
`partitioning`	X	X	X			
`patch`	X	X	X	X	X	
`pool` (ZFS root pools only)	X	X	X			X
`root_device` (UFS and ZFS)	X	X	X	X	X	X
`system_type`	X	X	X			
`usedisk` (UFS and ZFS)	X	X	X			X

Profile Keyword Descriptions and Examples

`archive_location` Keyword

`archive_location` *retrieval_type location*

retrieval_type The values of *retrieval_type* and *location* depend on where the Solaris Flash archive is stored. The following sections contain the values you can use for *retrieval_type* and *location* and examples of how to use the `archive_location` keyword.

- "Archive Stored on an NFS Server" on page 114
- "Archive Stored on an HTTP or HTTPS Server" on page 115
- "Archive Stored on an FTP Server" on page 116
- "Archive Stored on a Local Tape" on page 117
- "Archive Stored on a Local Device" on page 118
- "Archive Stored on a Local File" on page 119

location Specifics for locations are noted in the following sections.

Caution – Solaris Flash archive cannot be properly created when a non-global zone is installed. The Solaris Flash feature is not compatible with the Solaris Zones partitioning technology. If you create a Solaris Flash archive, the resulting archive is not installed properly when the archive is deployed under these conditions:

- The archive is created in a non-global zone
- The archive is created in a global zone that has non-global zones installed

Archive Stored on an NFS Server

If the archive is stored on an NFS server, use the following syntax for the `archive_location` keyword.

`archive_location nfs` *server_name:/path/filename retry n*

server_name The name of the server where you stored the archive.

path The location of the archive to be retrieved from the specified server. If the path contains $HOST, the Solaris Flash installation utilities replace $HOST with the name of the clone system that you are installing.

filename The name of the Solaris Flash archive file.

retry n An optional keyword. *n* is the maximum number of times the Solaris Flash utilities attempt to mount the archive.

EXAMPLE 8–1 Archive Stored on an NFS Server

```
archive_location nfs golden:/archives/usrarchive

archive_location nfs://golden/archives/usrarchive
```

Archive Stored on an HTTP or HTTPS Server

If the archive is stored on an HTTP server, use the following syntax for the `archive_location` keyword.

```
archive_location http://server_name:port/path/filename  optional_keywords
```

If the archive is stored on an HTTPS server, use the following syntax for the `archive_location` keyword.

```
archive_location https://server_name:port/path/filename  optional_keywords
```

server_name The name of the server where you stored the archive.

port An optional port. *port* can be a port number or the name of a TCP service that has a port number that is determined at runtime.

If you do not specify a port, the Solaris Flash installation utilities use the default HTTP port number, 80.

path The location of the archive to be retrieved from the specified server. If the path contains $HOST, the Solaris Flash installation utilities replace $HOST with the name of the clone system that you are installing.

filename The name of the Solaris Flash archive file.

optional_keywords The optional keywords that you can specify when you retrieve a Solaris Flash archive from an HTTP server.

TABLE 8–3 Optional Keywords to Use With `archive_location` HTTP

Keyword	Value Definition
`auth basic` *user_name password*	If the archive is located on an HTTP server that is password protected, you must include the user name and password that you need to access the HTTP server in the profile file. **Note –** The use of this authentication method in a profile that is intended for use with custom JumpStart is risky. Unauthorized users might have access to the profile file that contains the password.

TABLE 8–3 Optional Keywords to Use With `archive_location` HTTP *(Continued)*

Keyword	Value Definition
`timeout` *min*	The `timeout` keyword enables you to specify, in minutes, the maximum length of time that is allowed to pass without receipt of data from the HTTP server. If a timeout occurs, the connection is closed, reopened, and resumed. If you specify a `timeout` value of 0 (zero), the connection is not reopened. ■ If a timeout reconnection occurs, the Solaris Flash installation utilities attempt to resume the installation at the last known position in the archive. If the Solaris Flash installation utilities cannot resume the installation at the last known position, the retrieval restarts from the beginning of the archive and the data that was retrieved prior to the timeout is discarded. ■ If a timeout reconnection occurs while a package is being installed, the package is retried from the beginning of the package and the data that was retrieved prior to the timeout is discarded.
`proxy` *host:port*	The `proxy` keyword enables you to specify a proxy host and proxy port. You can use a proxy host to retrieve a Solaris Flash archive from the other side of a firewall. You must supply a proxy port when you specify the `proxy` keyword.

EXAMPLE 8–2 Archive Stored on a HTTP or HTTPS Server

```
archive_location http://silver/archives/usrarchive.flar timeout 5
```

Example of the `auth basic` *user_name password* keyword:

```
archive_location http://silver/archives/usrarchive.flar timeout 5 user1 secret
```

Archive Stored on an FTP Server

If the archive is stored on an FTP server, use the following syntax for the `archive_location` keyword.

```
archive_location ftp://user_name:password@server_name:port/path/filename  optional_keywords
```

user_name:password — The user name and password that you need to access the FTP server in the profile file.

server_name — The name of the server where you stored the archive.

port — A is an optional port. *port* can be a port number or the name of a TCP service that has a port number that is determined at runtime.

If you do not specify a port, the Solaris Flash installation utilities use the default FTP port number, 21.

path The location of the archive to be retrieved from the specified server. If the path contains $HOST, the Solaris Flash installation utilities replace $HOST with the name of the clone system that you are installing.

filename The name of the Solaris Flash archive file.

optional_keywords The optional keywords that you can specify when you retrieve a Solaris Flash archive from an FTP server.

TABLE 8–4 Optional Keywords to Use With `archive_location` FTP

Keyword	Value Definition
`timeout` *min*	The `timeout` keyword enables you to specify, in minutes, the maximum length of time that is allowed to pass without receipt of data from the HTTP server. If a timeout occurs, the connection is closed, reopened, and resumed. If you specify a `timeout` value of 0 (zero), the connection is not reopened. ■ If a timeout reconnection occurs, the Solaris Flash installation utilities attempt to resume the installation at the last known position in the archive. If the Solaris Flash installation utilities cannot resume the installation at the last known position, the retrieval restarts from the beginning of the archive and the data that was retrieved prior to the timeout is discarded. ■ If a timeout reconnection occurs while a package is being installed, the package is retried from the beginning of the package and the data that was retrieved prior to the timeout is discarded.
`proxy` *host:port*	The `proxy` keyword enables you to specify a proxy host and proxy port. You can use a proxy host to retrieve a Solaris Flash archive from the other side of a firewall. You must supply a proxy port when you specify the `proxy` keyword.

EXAMPLE 8–3 Archive Stored on an FTP Server

```
archive_location ftp://user1:secret@silver/archives/usrarchive.flar timeout 5
```

Archive Stored on a Local Tape

If the archive is stored on a tape, use the following syntax for the `archive_location` keyword.

```
archive_location local_tape device position
```

device The name of the tape drive where you stored the Solaris Flash archive. If the device name is a canonical path, the Solaris Flash installation utilities retrieve the archive from the path to the device node. If you supply a device name that is not a canonical path, the Solaris Flash installation utilities add `/dev/rmt/` to the path.

position Designates the place on the tape drive where you saved the archive. If you do not supply a position, the Solaris Flash installation utilities retrieve the archive from the current position on the tape drive. By specifying a *position*, you can place a begin script or a `sysidcfg` file on the tape drive before the archive.

EXAMPLE 8–4 Archive Stored on a Local Tape

```
archive_location local_tape /dev/rmt/0n 5

archive_location local_tape 0n 5
```

Archive Stored on a Local Device

You can retrieve a Solaris Flash archive from a local device if you stored the Solaris Flash archive on a file system-oriented, random-access device, such as a diskette or a DVD. Use the following syntax for the `archive_location` keyword.

Note – You can retrieve an archive from stream-oriented devices, such as tape, by using the syntax for local tape.

`archive_location local_device` *device path/filename file_system_type*

device The name of the drive where you stored the Solaris Flash archive. If the device name is a canonical path, the device is mounted directly. If you supply a device name that is not a canonical path, the Solaris Flash installation utilities add `/dev/dsk/` to the path.

path The path to the Solaris Flash archive, relative to the root of the file system on the device you specified. If the path contains $HOST, the Solaris Flash installation utilities replace $HOST with the name of the clone system that you are installing.

filename The name of the Solaris Flash archive file.

file_system_type Specifies the type of file system on the device. If you do not supply a file system type, the Solaris Flash installation utilities attempt to mount a UFS file system. If the UFS mount fails, the Solaris Flash installation utilities attempt to mount an HSFS file system.

EXAMPLE 8–5 Archive Stored on a Local Device

To retrieve an archive from a local hard drive that is formatted as a UFS file system, use the following command:

```
archive_location local_device c0t0d0s0 /archives/$HOST
```

EXAMPLE 8–5 Archive Stored on a Local Device *(Continued)*

To retrieve an archive from a local CD-ROM that has an HSFS file system, use the following command:

```
archive_location local_device c0t0d0s0 /archives/usrarchive
```

Archive Stored on a Local File

You can retrieve an archive that you stored in the miniroot from which you booted the clone system as a local file. When you perform a custom JumpStart installation, you boot the system from a DVD, CD, or an NFS-based miniroot. The installation software is loaded and run from this miniroot. Therefore, a Solaris Flash archive that you stored in the DVD, CD, or NFS-based miniroot is accessible as a local file. Use the following syntax for the `archive_location` keyword.

`archive_location local_file` *path/filename*

path The location of the archive. The path must be accessible to the system as a local file while the system is booted from the Solaris Software - 1 CD or from the Solaris Operating System DVD. The system cannot access `/net` or any other automounted directory when it is booted from the Solaris Software - 1 CD or from the Solaris Operating System DVD.

filename The name of the Solaris Flash archive file.

EXAMPLE 8–6 Archive Stored on a Local File

```
archive_location local_file /archives/usrarchive
```

`backup_media` Profile Keyword

`backup_media` *type path*

You can use `backup_media` only with the upgrade option when disk space reallocation is required.

`backup_media` defines the media that is to be used to back up file systems if space needs to be reallocated during an upgrade because of insufficient space. If multiple tapes or diskettes are required for the backup, you are prompted to insert tapes or diskettes during the upgrade.

Valid *type* Value	Valid *path* Value	Specification
`local_tape`	`/dev/rmt/`*n*	A local tape drive on the system that is being upgraded. *path* must be the character (raw) device path for the tape drive. *n* is the number of the tape drive.
`local_diskette`	`/dev/rdiskette`*n*	A local diskette drive on the system that is being upgraded. *path* must be the character (raw) device path for the diskette drive. *n* is the number of the diskette drive. Diskettes that you use for the backup must be formatted.
`local_filesystem`	`/dev/dsk/c`*w*`t`*x*`d`*y*`s`*z* */file_system*	A local file system on the system that is being upgraded. You cannot specify a local file system that is being changed by the upgrade. *path* can be a block device path for a disk slice. For example, the `t`*x* in `/dev/dsk/c`*w*`t`*x*`d`*y*`s`*z* might not be needed. Or, *path* can be the absolute path to a file system that is mounted by the `/etc/vfstab` file.
`remote_filesystem`	*host*`:`*/file_system*	An NFS file system on a remote system. *path* must include the name or IP address of the remote system, *host*, and the absolute path to the NFS file system, *file_system*. The NFS file system must have read/write access.
`remote_system`	*user*`@`*host*`:`*/directory*	A directory on a remote system that can be reached by a remote shell, `rsh`. The system that is being upgraded must have access to the remote system through the remote system's `.rhosts` file. *path* must include the name of the remote system *host* and the absolute path to the directory *directory*. If a user login ID *user* is not specified, `root` is used by default.

EXAMPLE 8–7 `backup_media` Profile Keyword

```
backup_media local_tape /dev/rmt/0

backup_media local_diskette /dev/rdiskette1

backup_media local_filesystem /dev/dsk/c0t3d0s4

backup_media local_filesystem /export

backup_media remote_filesystem system1:/export/temp

backup_media remote_system user1@system1:/export/temp
```

`boot_device` Profile Keyword (UFS and ZFS)

Note – The `boot_device` keyword can be used when you install either a UFS file system or a ZFS root pool. The usage for this keyword is the same in both UFS and ZFS installations.

- For a complete list of keywords that can be used in a UFS or ZFS installation, see Table 8–2
- For information on performing a ZFS installation, see Chapter 9, "Installing a ZFS Root Pool With JumpStart"

`boot_device` *device eeprom*

`boot_device` designates the device where the JumpStart program is to install the root (`/`) file system and the system's boot device. `boot_device` must match any `filesys` keywords that specify the root (`/`) file system and the `root_device` keyword.

If you do not specify the `boot_device` keyword in a profile, the following `boot_device` keyword is specified by default during the installation:

```
boot_device any update
```

device Use one of the following values.

SPARC: c*w*t*x*d*y*s*z* or c*x*d*y*s*z*	The disk slice where the JumpStart program places the root (`/`) file system, for example, `c0t0d0s0`.
x86: c*w*t*x*d*y* or c*x*d*y*	The disk where the JumpStart program places the root (`/`) file system, for example, `c0d0`.
`existing`	The JumpStart program places the root (`/`) file system on the system's existing boot device.
`any`	The JumpStart program chooses where to place the root (`/`) file system. The JumpStart program attempts to use the system's existing boot device. The JumpStart program might choose a different boot device if necessary.

eeprom Choose to update or preserve the system's EEPROM.

The *eeprom* value enables you to update the system's EEPROM if you change the system's current boot device. By updating the system's EEPROM, the system can automatically boot from the new boot device.

Note – x86: You must specify the `preserve` value.

`update`	The JumpStart program updates the system's EEPROM to the specified boot device so that the installed system automatically boots from it.
`preserve`	The boot device value in the system's EEPROM is not changed. If you specify a new boot device without changing the system's EEPROM, you need to change the system's EEPROM manually so it can automatically boot from the new boot device.

EXAMPLE 8–8 `boot_device` Profile Keyword

```
boot_device c0t0d0s2 update
```

`bootenv` Profile Keyword (UFS and ZFS)

Note – The `bootenv` keyword can be used for either a UFS file system or a ZFS root pool installation. The usage is different for a ZFS installation.

- For a complete list of keywords that can be used in a UFS or ZFS installation, see Table 8–2
- For a description of how the `bootenv` keyword can be used when installing a ZFS root pool, see "JumpStart Keywords for a ZFS Root (/) File System (Reference)" on page 164

```
bootenv createbe bename new_BE_name filesystem mountpoint:device:fs_options
[filesystem...]
```

`bootenv createbe` keyword enables you to quickly create an empty-and-inactive boot environment at the same time you are installing the Solaris OS. At the least, you must create the root (/) file system. The slices are reserved for the file systems specified, but no file systems are copied. The boot environment is named, but not actually created until installed with a Solaris Flash archive. When the empty boot environment is installed with an archive, file systems are installed on the reserved slices. The following lists the values for *bename* and *filesystem*.

`bename` *new_BE_name*
: `bename` specifies the name of the new boot environment to be created. *new_BE_name* can be no longer than 30 characters, can contain only alphanumeric characters, and can contain no multibyte characters. The name must be unique on the system.

`filesystem` *mountpoint*:*device*:*fs_options*
: `filesystem` determines the type and number of file systems that are to be created in the new boot environment. At least one slice that contains the root (/) file system must be defined. File systems can be on the same disk or spread across multiple disks.

  - *mountpoint* can be any valid mount point or – (hyphen), indicating a swap slice.

- *device* must be available when the operating system that is being installed is first booted. The device has no relation to JumpStart special storage devices such as *free*. The device cannot be a Solaris Volume Manager volume or Veritas Volume Manager volume. *device* is the name of a disk device, of the form `/dev/dsk/c`*w*`t`*x*`d`*y*`s`*z*.
- *fs_options* can be one of the following:
  - `ufs`, which indicates a UFS file system.
  - `swap`, which indicates a swap file system. The swap mount point must be a – (hyphen).

For a profile example and background about using this keyword, see the following references:

For an example of a profile	Example 3–11
For background about using Solaris Live Upgrade that creates, upgrades, and activates inactive boot environments	Chapter 2, "Solaris Live Upgrade (Overview)," in *Solaris 10 5/09 Installation Guide: Solaris Live Upgrade and Upgrade Planning*
For background about using a Solaris Flash archive	Chapter 1, "Solaris Flash (Overview)," in *Solaris 10 5/09 Installation Guide: Solaris Flash Archives (Creation and Installation)*

`client_arch` Profile Keyword

`client_arch` *karch_value* `...`

`client_arch` specifies that the operating system server is to support a different platform group than the server uses. If you do not specify `client_arch` in the profile, any diskless client that uses the operating system server must contain the same platform group as the server. You must specify each platform group that you want the operating system server to support.

Valid values for *karch_value* are `sun4u` and `i86pc`. For a detailed list of platform names and various systems, see *Solaris Sun Hardware Platform Guide* at `http://docs.sun.com`.

Note – You can use `client_arch` only when `system_type` is specified as `server`.

`client_root` Profile Keyword

`client_root` *root_size*

`client_root` defines the amount of root space, *root_size* in Mbytes, to allocate for each client. If you do not specify `client_root` in a server's profile, the installation software allocates 15 Mbytes of root space per client. The size of the client root area is used in combination with the `num_clients` keyword to determine how much space to reserve for the `/export/root` file system.

Note – You can use `client_root` only when `system_type` is specified as `server`.

`client_swap` Profile Keyword

`client_swap` *swap_size*

`client_swap` defines the amount of swap space, *swap_size* in Mbytes, to allocate for each diskless client. If you do not specify `client_swap` in the profile, 32 Mbytes of swap space is allocated by default.

Note – You can use `client_swap` only when `system_type` is specified as `server`.

EXAMPLE 8–9 `client_swap` Profile Keyword

The following example specifies that each diskless client is to have a swap space of 64 Mbytes.

```
client_swap 64
```

How the Size of `swap` Is Determined

If a profile does not specify the size of `swap`, the JumpStart program determines the size of the swap space, based on the system's physical memory. Table 8–5 shows how the size of `swap` is determined during a custom JumpStart installation.

TABLE 8–5 Determining `swap` Size

Physical Memory (in Mbytes)	Swap Space (in Mbytes)
16–64	32
64–128	64
128–512	128
Greater than 512	256

The JumpStart program makes the size of `swap` no more than 20 percent of the disk where `swap` is located. The allocation is different if the disk contains free space after laying out the other file systems. If free space exists, the JumpStart program allocates the free space to `swap`, and if possible, allocates the amount that is shown in Table 8–5.

Note – Physical memory plus swap space must total a minimum of 32 Mbytes.

`cluster` Profile Keyword (Adding Software Groups) (UFS and ZFS)

Note – The `cluster` keyword can be used when you install either a UFS file system or a ZFS root pool. The usage for this keyword is the same in both UFS and ZFS installations.

- For a complete list of keywords that can be used in a UFS or ZFS installation, see Table 8–2
- For a description of how the `cluster` keyword can be used in a profile for an installation of a ZFS root pool, see "JumpStart Profile Examples for a ZFS Root Pool" on page 161

`cluster` *group_name*

`cluster` designates the software group to add to the system.

A software group is a metacluster that contains a collection of clusters and packages. The software group is installed by using the `cluster` keyword and *group_name* variable. This cluster keyword can only be installed in an initial installation. This `cluster` keyword refers to metaclusters found in the `clustertoc(4)` file.

A cluster is a collection of packages that is named SUNW*name*. A cluster is installed by using the `cluster` keyword and *cluster_name* variable. A cluster can be added or removed from a software group (metacluster) in an initial install or an upgrade.

The *group_name* for each software group is listed in the following table.

Software Group	*group_name*
Reduced Network Support Software Group	`SUNWCrnet`
Core System Support Software Group	`SUNWCreq`
End User Solaris Software Group	`SUNWCuser`
Developer Solaris Software Group	`SUNWCprog`
Entire Solaris Software Group	`SUNWCall`
Entire Solaris Software Group Plus OEM Support	`SUNWCXall`

The following limitations apply:

- You can specify only one software group in a profile.
- The software group must be specified before other `cluster` and `package` entries.
- If you do not specify a software group with `cluster` in the profile, the end–user software group, `SUNWCuser`, is installed on the system.

For more information about software groups, see "Disk Space Recommendations for Software Groups" in *Solaris 10 5/09 Installation Guide: Planning for Installation and Upgrade*.

`cluster` Profile Keyword (Adding or Deleting Clusters) (UFS and ZFS)

`cluster` *cluster_name add_delete_switch*

Note – The `cluster` keyword can be used when you install either a UFS file system or a ZFS root pool. The usage for this keyword is the same in both UFS and ZFS installations.

- For a complete list of keywords that can be used in a UFS or ZFS installation, see Table 8–2
- For a description of how the `cluster` keyword can be used in a profile for an installation of a ZFS root pool, see "JumpStart Profile Examples for a ZFS Root Pool" on page 161

`cluster` designates whether a cluster is to be added or deleted from the software group that is to be installed on the system.

cluster_name	The name of the cluster that must be in the form `SUNWC`*name*.
add_delete_switch	An optional keyword that indicates whether to add or delete the cluster that is specified. Use the value `add` or `delete`. If you do not specify `add` or `delete`, `add` is used by default.

When you use `cluster` during an upgrade, the following conditions apply:

- All clusters that are already on the system are automatically upgraded.
- If you specify *cluster_name* `add`, and *cluster_name* is not installed on the system, the cluster is installed.
- If you specify *cluster_name* `delete`, and *cluster_name* is installed on the system, the package is deleted *before* the upgrade begins.

Note – A software group is a metacluster that contains a collection of clusters and packages. The software group is installed by using the `cluster` keyword and *group_name* variable. This cluster keyword can only be installed in an initial installation. This `cluster` keyword refers to metaclusters found in the `clustertoc(4)` file.

A cluster is collection of packages. Clusters can be grouped together to form a software group (metacluster). A cluster name is always in the form of SUNW<name>. A cluster is installed by using the `cluster` keyword and *cluster_name* variable. A cluster can be added or removed from a software group (metacluster) in an initial install or an upgrade.

`dontuse` Profile Keyword (UFS and ZFS)

Note – The `dontuse` keyword can be used when you install either a UFS file system or a ZFS root pool. The usage for this keyword is the same in both UFS and ZFS installations.

- For a complete list of keywords that can be used in a UFS or ZFS installation, see Table 8–2
- For information on performing a ZFS installation, see Chapter 9, "Installing a ZFS Root Pool With JumpStart"

`dontuse` *disk_name* `...`

By default, the JumpStart program uses all of the operational disks on the system when `partitioning default` is specified. `dontuse` designates one or more disks that you do not want the JumpStart program to use. *disk_name* must be specified in the form c*x*t*y*d*z*or c*y*d*z*, for example, `c0t0d0`.

Note – You cannot specify the `dontuse` keyword and the `usedisk` keyword in the same profile.

x86: `fdisk` Profile Keyword (UFS and ZFS)

Note – The `fdisk` keyword can be used when you install either a UFS file system or a ZFS root pool. The usage for this keyword is the same in both UFS and ZFS installations.

- For a complete list of keywords that can be used in a UFS or ZFS installation, see Table 8–2
- For information on performing a ZFS installation, see Chapter 9, "Installing a ZFS Root Pool With JumpStart"

`fdisk` *disk_name type size*

`fdisk` defines how the `fdisk` partitions are set up on an x86 based system. You can specify `fdisk` more than once. When `fdisk` partitions an x86 based system, the following occurs:

- All `fdisk` partitions on the disk are preserved unless you delete the partitions with the `fdisk` keyword by assigning *size* the value of `delete` or `0`. Also, all existing `fdisk` partitions are deleted when *size* is set to `all`.
- A Solaris `fdisk` partition that contains a root (`/`) file system is always designated as the active partition on the disk.

  Note – The system boots from the active partition by default.

- If the `fdisk` keyword is not specified in a profile, the following `fdisk` keyword is used by default during the installation.

  ```
  fdisk all solaris maxfree
  ```

- `fdisk` entries are processed in the order in which the entries are listed in the profile.

disk_name Use the following values to specify where the `fdisk` partition is to be created or deleted:

- c*x*t*y*d*z* or c*y*d*z* – A specific disk, for example, `c0t3d0`.
- `rootdisk` – The variable that contains the value of the system's root disk, which is where the installation takes place. The root disk is determined by the JumpStart program as described in "How the System's Root Disk Is Determined" on page 154.
- `all` – All the selected disks.

type Use the following values to specify the type of `fdisk` partition that is to be created or deleted on the specified disk:

- `solaris` – A Solaris `fdisk` partition (`SUNIXOS fdisk` type).
- `dosprimary` – An alias for primary DOS `fdisk` partitions, not for `fdisk` partitions that are extended or reserved for data DOS. When you delete `fdisk` partitions by assigning *size* the value `delete`, `dosprimary` is an alias for the DOSHUGE, DOSOS12, and DOSOS16 `fdisk` types. When you create an `fdisk` partition, `dosprimary` is an alias for the DOSHUGE `fdisk` partition.
- *DDD* – An integer `fdisk` partition. *DDD* is an integer between 1 and 255 inclusive.

  Note – You can specify this value only if *size* is `delete`.

- 0x*HH* – A hexadecimal `fdisk` partition. *HH* is a hexadecimal number between 01 and FF.

  Note – You can specify this value only if *size* is `delete`.

The following table shows the integer and hexadecimal numbers for some of the `fdisk` types.

`fdisk` **Type**	*DDD*	*HH*
DOSOS12	`1`	`01`
PCIXOS	`2`	`02`
DOSOS16	`4`	`04`
EXTDOS	`5`	`05`
DOSHUGE	`6`	`06`
DOSDATA	`86`	`56`
OTHEROS	`98`	`62`
UNIXOS	`99`	`63`

size Use one of the following values:

- *DDD* – An `fdisk` partition of size *DDD* in Mbytes is created on the specified disk. *DDD* must be an integer, and the JumpStart program automatically rounds the number up to the nearest cylinder boundary. Specifying a value of `0` is the same as specifying `delete`.
- `all` – An `fdisk` partition is created on the entire disk. All existing `fdisk` partitions are deleted.

  x86 only – The `all` value can be specified only if *type* is `solaris`.

- `maxfree` – An `fdisk` partition is created in the largest contiguous free space on the specified disk. If an `fdisk` partition of the specified *type* already exists on the disk, the existing `fdisk` partition is used. A new `fdisk` partition is *not* created on the disk.

  x86 only – The disk must contain at least one unused `fdisk` partition. Also, the disk must have free space or the installation fails. The `maxfree` value can be specified only if *type* is `solaris` or `dosprimary`.

- `delete` – All `fdisk` partitions of the specified *type* are deleted on the specified disk.

`filesys` Profile Keyword (Mounting Remote File Systems) (UFS and ZFS)

Note – The `filesys` keyword can be used when you install either a UFS file system or a ZFS root pool. The usage for this keyword is the same in both UFS and ZFS installations.

- For a complete list of keywords that can be used in a UFS or ZFS installation, see Table 8–2
- For information on performing a ZFS installation, see Chapter 9, "Installing a ZFS Root Pool With JumpStart"

`filesys` *server*:*path server_address mount_pt_name mount_options*

By using `filesys` with the listed values, the JumpStart program sets up the installed system to automatically mount remote file systems when the system boots. You can specify `filesys` more than once.

server — The name of the server where the remote file system is located, followed by a colon.

path — The remote file system's mount-point name. For example, `/usr` or `/export/home`

server_address — The IP address of the server that is specified in *server*:*path*. If a naming service is not running on the network, the *server_address* value can be used to populate the `/etc/hosts` file with the server's host name and IP address. If you are not specifying the server's IP address, you must specify a minus sign (`-`). For example, if you have a naming service that is running on the network, you do not need to specify the server's IP address.

mount_pt_name — The name of the mount point on which the remote file system is to be mounted.

mount_options — One or more mount options, which is the same as the `-o` option of the `mount(1M)` command. The mount options are added to the `/etc/vfstab` entry for the specified *mount_pt_name*.

Note – If you need to specify more than one mount option, the mount options must be separated by commas and no spaces (`ro,quota` for example).

EXAMPLE 8–10 `filsys` Profile Keyword

```
filesys sherlock:/export/home/user2 - /home
```

`filesys` Profile Keyword (Creating Local File Systems)

`filesys` *slice size file_system optional_parameters*

By using `filesys` with the values that are listed, the JumpStart program creates local file systems during the installation. You can specify `filesys` more than once.

slice — Use one of the following values:

- `any` — The JumpStart program places the file system on any disk.

  Note – You cannot specify `any` when `size` is `existing`, `all`, `free`, *start*:*size*, or `ignore`.

- c*w*t*x*d*y*s*z* or c*x*d*y*s*z* — The disk slice where the JumpStart program places the file system, for example, `c0t0d0s0` or `c0d0s0`.
- `rootdisk.s`*n* — The variable that contains the value for the system's root disk, which is determined by the JumpStart program as described in "How the System's Root Disk Is Determined" on page 154. The `s`*n* suffix indicates a specific slice on the disk.

  Note – The root disk is determined by the JumpStart program and determines where the OS is to be installed. The rules file uses a probe keyword "`rootdisk`," but this keyword is used differently than the "rootdisk" keyword used in the JumpStart profile. You cannot set the place of installation by using the probe keyword "`rootdisk`" in the rules file. The probe keyword, `rootdisk`, determines where to boot from during the installation. See Table 8–10.

size — Use one of the following values:

- *num* — The size of the file system is set to *num* in Mbytes.
- `existing` — The current size of the existing file system is used.

Note – When you use the `existing` value, you can change the name of an existing slice by specifying *file_system* as a different *mount_pt_name*.

`auto` The size of the file system is automatically determined, depending on the software that is selected.

`all` The specified *slice* uses the entire disk for the file system. When you specify the `all` value, no other file systems can be placed on the specified disk.

`free` The remaining unused space on the disk is used for the file system.

Note – If `free` is used as the value to `filesys`, the `filesys` entry must be the last entry in a profile.

start:*size* The file system is explicitly partitioned. *start* is the cylinder where the slice begins. *size* is the number of cylinders for the slice.

file_system The *file_system* value is optional and used when *slice* is specified as `any` or c*w*t*x*d*y*s*z*. If *file_system* is not specified, `unnamed` is set by default. If `unnamed` is set, you cannot specify the *optional_parameters* value. Use one of the following values:

mount_pt_name The file system's mount-point name, for example, `/var`.

`swap` The specified *slice* is used as `swap`.

`overlap` The specified *slice* is defined as a representation of a disk region. The VTOC value is V_BACKUP. By default, slice 2 is an overlap slice that is a representation of the whole disk.

Note – You can specify `overlap` only when *size* is `existing`, `all`, or *start*:*size*.

`unnamed` The specified *slice* is defined as a raw slice, so *slice* does not have a mount-point name. If you do not specify *file_system*, `unnamed` is used by default.

`ignore` — The specified *slice* is not used or recognized by the JumpStart program. You can use this option to specify that you want a file system to be ignored on a disk during installation. The JumpStart program creates a new file system on the same disk with the same name. You can use `ignore` only when `partitioning existing` is specified.

optional_parameters — Use one of the following values:

`preserve` — The file system on the specified *slice* is preserved.

Note – `preserve` can be specified only when *size* is `existing` and *slice* is c*w*t*x*d*y*s*z*.

mount_options — One or more mount options, which is the same as the `-o` option of the `mount(1M)` command. The mount options are added to the `/etc/vfstab` entry for the specified *mount_pt_name*.

Note – If you need to specify more than one mount option, the mount options must be separated by commas and no space (`ro,quota`, for example).

`filesys` Profile Keyword (Creating RAID-1 Volumes)

`filesys mirror[:`*name*`]`*slice* `[`*slice*`]` *size file_system optional_parameters*

By using the `filesys mirror` keywords with the values that are listed, the JumpStart program creates the RAID-1 and RAID-0 volumes that are necessary to create a mirrored file system. You can specify `filesys mirror` more than once to create RAID-1 volumes (mirrors) for different file systems.

Note – The `filesys mirror` keyword is only supported for initial installations.

name — This optional keyword enables you to name the RAID-1 volume (mirror). Mirror names must start with the letter “d”, followed by a number between 0 and 127, for example, `d100`. If you do not specify a mirror name, the custom JumpStart program assigns a mirror name for you. For guidelines about how to name mirrors, see “RAID

Volume Name Requirements and Guidelines for Custom JumpStart and Solaris Live Upgrade" in *Solaris 10 5/09 Installation Guide: Planning for Installation and Upgrade*.

slice

This value specifies the disk slice where the custom JumpStart program places the file system you want to duplicate. The slice value must follow the format *cwtxdysz*, for example `c0t0d0s0` or `c0t0d0s5`. The custom JumpStart program creates a RAID-0 volume (single-slice concatenation) on the slice, and creates a RAID-1 volume to mirror the concatenation. You can specify up to two slices for two RAID-0 volumes.

size

This value specifies the size, in Mbytes, of the file system.

file_system

This value specifies the file system that you are duplicating. The custom JumpStart program creates the RAID-1 volume from the slices that are specified and mounts the RAID-1 volume on the specified file system. In addition to critical file systems, such as root (`/`), `/usr`, and `/var`, you can also specify `swap` as the file system.

- If *file_system* is not specified, `unnamed` is set by default.
- If `unnamed` is set, you cannot specify the *optional_parameters* value. Use one of the following values:

  mount_pt_name — Specifies the file system's mount-point name, for example, `/var`.

  `swap` — Defines the slice to be used as swap.

  `overlap` — Defines the slice as a representation of a disk region. The VTOC value is `V_BACKUP`. By default, slice 2 is an overlap slice that is a representation of the whole disk. You can specify `overlap` only when `size` is one of the following values:

  - `existing`
  - `all`
  - `start:size.`

  unnamed — Defines the slices as a raw slice. Therefore, the slice does not have a mount-point name. If you do not specify *file_system*, `unnamed` is used by default.

  ignore — Specifies that the slice is not to be used or recognized by the JumpStart program. You can use this option to specify that you want a

file system to be ignored on a disk during installation. The JumpStart program creates a new file system on the same disk with the same name. You can use `ignore` only when the `partitioning` Keyword and the `existing` value is specified.

optional_parameters One or more mount options, which is the same as the `-o` option of the `mount(1M)` command. The mount options are added to the `/etc/vfstab` entry for the specified *file_system*. If you need to specify more than one mount option, the mount options must be separated by commas and no spaces, for example, `ro,quota`.

Note – If `unnamed` is set for the *file_system* value, you cannot specify the *optional_parameters* value. See *file_system* for the values that can be set.

For more information about creating mirrored file systems during your installation, see Chapter 9, "Creating RAID-1 Volumes (Mirrors) During Installation (Overview)," in *Solaris 10 5/09 Installation Guide: Planning for Installation and Upgrade*.

`forced_deployment` Profile Keyword (Installing Solaris Flash Differential Archives)

`forced_deployment`

`forced_deployment` forces the installation of a Solaris Flash differential archive onto a clone system that is different than the software expects.

Caution – If you use `forced_deployment`, all new files are deleted to bring the clone system to the expected state. If you are not certain that you want files deleted, use the default, which protects new files by stopping the installation.

`geo` Profile Keyword (UFS and ZFS)

Note – The `geo` keyword can be used when you install either a UFS file system or a ZFS root pool. The usage for this keyword is the same in both UFS and ZFS installations.

- For a complete list of keywords that can be used in a UFS or ZFS installation, see Table 8–2
- For information on performing a ZFS installation, see Chapter 9, "Installing a ZFS Root Pool With JumpStart"

`geo` *region*

`geo` designates the regional locale or locales that you want to install on a system or to add when upgrading a system. *region* designates a geographical area that contains the locales that you want to install. Values you can specify for *region* are listed in the following table.

Value	Description
`N_Africa`	Northern Africa, including Egypt
`C_America`	Central America, including Costa Rica, El Salvador, Guatemala, Mexico, Nicaragua, Panama
`N_America`	North America, including Canada, United States
`S_America`	South America, including Argentina, Bolivia, Brazil, Chile, Colombia, Ecuador, Paraguay, Peru, Uruguay, Venezuela
`Asia`	Asia, including Japan, Republic of Korea, People's Republic of China, Taiwan, Thailand
`Ausi`	Australasia, including Australia, New Zealand
`C_Europe`	Central Europe, including Austria, Czech Republic, Germany, Hungary, Poland, Slovakia, Switzerland
`E_Europe`	Eastern Europe, including Albania, Bosnia, Bulgaria, Croatia, Estonia, Latvia, Lithuania, Macedonia, Romania, Russia, Serbia, Slovenia, Turkey
`N_Europe`	Northern Europe, including Denmark, Finland, Iceland, Norway, Sweden
`S_Europe`	Southern Europe, including Greece, Italy, Portugal, Spain
`W_Europe`	Western Europe, including Belgium, France, Great Britain, Ireland, Netherlands
`M_East`	Middle East, including Israel

A complete list of the component locale values that compose each regional locale that is listed previously is presented in *International Language Environments Guide*.

Note – You can specify a `geo` keyword for each locale you need to add to a system.

`install_type` Profile Keyword (UFS and ZFS)

Note – The `install_type` keyword can be used for either a UFS file system or a ZFS root pool installation. The usage is limited for a ZFS installation. You can only use the `intial_install` option for a ZFS installation.

- If you want to migrate your UFS file system to a ZFS root pool or upgrade a ZFS root pool, you must use Solaris Live Upgrade. See Chapter 11, "Solaris Live Upgrade and ZFS (Overview)," in *Solaris 10 5/09 Installation Guide: Solaris Live Upgrade and Upgrade Planning*.
- For a complete list of keywords that can be used in a UFS or ZFS installation, see Table 8–2
- For a description of how the `install_type` keyword can be used when installing a ZFS root pool, see "JumpStart Keywords for a ZFS Root (/) File System (Reference)" on page 164.

`install_type` *initial_upgrade_flash_switch*

`install_type` defines whether to erase and install a new Solaris OS on a system, upgrade the existing Solaris OS on a system, or install a Solaris Flash archive on the system.

Note – You must specify `install_type` in a profile, and `install_type` must be the first profile keyword in every profile.

You must use one of the following options for the *initial_upgrade_flash_switch*:

`initial_install`	Specifies to perform an initial installation of the Solaris OS
`upgrade`	Specifies to perform an upgrade of the Solaris OS
`flash_install`	Specifies to install a Solaris Flash archive that overwrites all files
`flash_update`	Specifies to install a Solaris Flash differential archive that overwrites only the files that are specified

Note – Some profile keywords can only be used with the `initial_install` option. Some profile keywords can only be used with the `upgrade` option. Some profile keywords can only be used with the `flash_install` option.

`layout_constraint` Profile Keyword

`layout_constraint` *slice constraint minimum_size*

`layout_constraint` designates the constraint auto-layout has on a file system if auto-layout needs to reallocate space during an upgrade because of space problems.

Limitation	Description
This keyword is used only with upgrade option.	You can use `layout_constraint` only for the upgrade option when you need to reallocate disk space.
If you do not specify the `layout_constraint` keyword	The JumpStart program lays out the disk as follows: ■ File systems that require more space for the upgrade are marked changeable. ■ File systems that are on the same disk as the file system that requires more space and that are mounted by the `/etc/vfstab` file are marked changeable. ■ Remaining file systems are marked fixed because auto-layout cannot change the file systems.
If you specify one or more `layout_constraint` keywords	The JumpStart program lays out the disk as follows: ■ File systems that require more space for the upgrade are marked changeable. ■ File systems for which you specified a `layout_constraint` keyword are marked with the specified constraint. ■ The remaining file systems are marked fixed.
If the file system is not marked changeable	You cannot change the constraint on file systems that require more space for the upgrade because the file systems must be marked changeable. You can use the `layout_constraint` keyword to change the *minimum_size* values on file systems that require more space for the upgrade.
If file systems require more space for upgrade	To help auto-layout reallocate space, select more file systems to be changeable or movable, especially those file systems that are located on the same disks as the file systems that require more space for the upgrade.

slice Specifies the file system's disk slice on which to specify the constraint. You must specify the system's disk slice in the form c*w*t*x*d*y*s*z* or c*x*d*y*s*z*.

constraint Use one of the following constraints for the specified file system:

`changeable` Auto-layout can move the file system to another location and it can change the file system size. The `changeable` constraint can only be specified on file systems that are mounted by the `/etc/vfstab` file. You can change the file system's size by specifying the *minimum_size* value.

When you mark a file system as changeable and *minimum_size* is not specified, the file system's minimum size is set to 10 percent more than the minimum size that is

required. For example, if the minimum size for a file system is 100 Mbytes, the changed size is 110 Mbytes. If *minimum_size* is specified, any free space that remains, original size minus minimum size, is used for other file systems.

`movable` Auto-layout can move the file system to another slice on the same disk or different disk. The file system size remains the same.

`available` Auto-layout can use all of the space on the file system to reallocate space. All of the data in the file system is lost. The `available` constraint can only be specified on file systems that are not mounted by the `/etc/vfstab` file.

`collapse` Auto-layout moves and collapses the specified file system into the parent file system. You can use the `collapse` option to reduce the number of file systems on a system as part of the upgrade. For example, if a system has the `/usr` and `/usr/share` file systems, collapsing the `/usr/share` file system moves the file system into `/usr`, the parent file system. You can specify the `collapse` constraint only on file systems that are mounted by the `/etc/vfstab` file.

minimum_size Specifies the size of the file system after auto-layout reallocates space. The *minimum_size* option enables you to change the size of a file system. The size of the file system might be larger if unallocated space is added to the file system. But, the size is never less than the value you specify. The *minimum_size* value is optional. Use this value only if you have marked a file system as changeable and the minimum size cannot be less than what the file system needs for the existing file system contents.

EXAMPLE 8–11 `layout_constraint` Profile Keyword

```
layout_constraint c0t3d0s1 changeable 200

layout_constraint c0t3d0s4 movable

layout_constraint c0t3d1s3 available

layout_constraint c0t2d0s1 collapse
```

`local_customization` Profile Keyword (Installing Solaris Flash Archives)

`local_customization` *local_directory*

Before you install a Solaris Flash archive on a clone system, you can create custom scripts to preserve local configurations on the clone system. The `local_customization` keyword designates the directory where you have stored these scripts. *local_directory* is the path to the script on the clone system.

For information about predeployment and postdeployment scripts, see "Creating Customization Scripts" in *Solaris 10 5/09 Installation Guide: Solaris Flash Archives (Creation and Installation)*.

`locale` Profile Keyword (UFS and ZFS)

Note – The `locale` keyword can be used when you install either a UFS file system or a ZFS root pool. The usage for this keyword is the same in both UFS and ZFS installations.

- For a complete list of keywords that can be used in a UFS or ZFS installation, see Table 8–2
- For information on performing a ZFS installation, see Chapter 9, "Installing a ZFS Root Pool With JumpStart"

`locale` *locale_name*

Note – You can use `locale` with both the initial installation and upgrade options.

`locale` designates the locale packages you want to install or add when upgrading for the specified *locale_name*. The *locale_name* values are the same as those values that are used for the `$LANG` environment variable. *International Language Environments Guide* contains a list of valid locale values.

When you use the `locale` keyword, consider the following:

- If you have preconfigured a default locale, the locale is automatically installed. The English language packages are installed by default.
- You can specify a `locale` keyword for each locale you need to add to a system.

`metadb` Profile Keyword (Creating State Database Replicas)

`metadb` *slice* `[size` *size-in-blocks*`] [count` *number-of-replicas*`]`

The `metadb` keyword enables you to create Solaris Volume Manager state database replicas (mediates) during your custom JumpStart installation. You can use the `metadb` keyword multiple times in your profile file to create state database replicas on different disk slices.

slice	You must specify the disk slice on which you want the custom JumpStart program to place the state database replica. The *slice* value must follow the format *cwtxdysz*.
`size` *size-in-blocks*	The `size` optional keyword enables you to specify the size, in blocks, of the state database replica to be created. If you do not specify a `size` value, the custom JumpStart program uses a default size of 8192 blocks for the state database replica.
`count` *number-of-replicas*	You can specify the number of state database replicas you are creating by setting the optional `count` keyword value in your profile. If you do not specify a `count` value, the custom JumpStart program creates three state database replicas by default.

For more information about creating Solaris Volume Manager state database replicas during your installation, see "State Database Replicas Guidelines and Requirements" in *Solaris 10 5/09 Installation Guide: Planning for Installation and Upgrade*.

`no_content_check` Profile Keyword (Installing Solaris Flash Archives)

`no_content_check`

When installing a clone system with a Solaris Flash differential archive, you can use the `no_content_check` keyword to ignore file-by-file validation. File-by-file validation ensures that the clone system is a duplicate of the master system. Avoid using this keyword unless you are sure the clone system is a duplicate of the original master system.

Caution – If you use `no_content_check`, all new files are deleted to bring the clone system to the expected state. If you are not certain that you want files deleted, use the default, which protects new files by stopping the installation.

For information about installing Solaris Flash differential archives, see "To Prepare to Install a Solaris Flash Archive With a Custom JumpStart Installation" on page 84.

`no_master_check` Profile Keyword (Installing Solaris Flash Archives)

`no_master_check`

When installing a clone system with a Solaris Flash differential archive, you can use the `no_master_check` keyword to ignore checking the clone system to make sure it was built from the original master system. Avoid using this keyword unless you are sure the clone system is a duplicate of the original master system.

For information about installing Solaris Flash differential archives, see "To Prepare to Install a Solaris Flash Archive With a Custom JumpStart Installation" on page 84.

num_clients Profile Keyword

num_clients *client_num*

When a server is installed, space is allocated for each diskless client's root (/) and swap file systems. num_clients defines the number of diskless clients, *client_num*, that a server supports. If you do not specify num_clients in the profile, five diskless clients are allocated by default.

Note – You can use num_clients only when system_type is specified as server.

---

package Profile Keyword (UFS and ZFS)

Note – The package keyword can be used when you install either a UFS file system or a ZFS root pool. The usage for this keyword is the same in both UFS and ZFS installations.

- For a complete list of keywords that can be used in a UFS or ZFS installation, see Table 8–2
- For information on performing a ZFS installation, see Chapter 9, "Installing a ZFS Root Pool With JumpStart"

---

package *package_name* [add [*retrieval_type location*]| delete]

You can use package with both the initial installation and upgrade options. The package keyword enables you to do the following:

- Add a package to the software group from the Solaris distribution that is to be installed.
- Add a package to the software group from outside the distribution that is being installed.
- Exclude or remove a package from the software group that is to be installed or upgraded.
- Add a package from outside the distribution that is being installed when installing a Solaris Flash archive.

package_name — Specifies the package name in the form SUNW*name*. To view detailed information about packages and their names, on an installed system, use the pkginfo -l command.

add | delete — Specifies to add or remove the specified package. If you do not specify add or delete, add is used by default.

Note – You can add more than one package by adding another package entry to the profile and omitting the location. The location of the previous package is used for all subsequent packages if the location is left blank.

[*retrieval_type location*] Specifies the addition of a package or packages that are located outside the Solaris distribution that is being installed. The values of *retrieval_type* and *location* depend on where the package is stored. The following sections contain the values you can use for *retrieval_type* and *location* and examples of how to use the `package_name` keyword.

Packages Stored on an NFS Server

If the package is stored on an NFS server, use one of the following syntaxes for the `package` keyword.

```
package package_name add nfs server_name:/path [retry n]
package package_name add nfs://server_name:/path [retry n]
```

package_name Specifies the package name in the form `SUNW`*name*. To view detailed information about packages and their names, on an installed system, use the `pkginfo -l` command.

server_name Specifies the name of the server where you stored the package.

path Specifies the location of the package directory on the specified server. If the path contains $HOST, $HOST is replaced with the name of the host system that you are installing.

retry *n* Is an optional keyword. *n* is the maximum number of times the installation process attempts to mount the directory.

EXAMPLE 8–12 Adding a Package by Using NFS

In this example, the `package` profile keyword adds the SUNWnew package from the NFS location `nfs://golden/packages/Solaris_10/`. If a mount fails, the NFS mount is tried five times.

```
package SUNWnew add nfs golden:/packages/Solaris_10 retry 5
```

Packages Stored on an HTTP Server

If the package is stored on an HTTP server, use one of the following syntaxes for the `package` keyword.

```
package package_name add http://server_name[:port] path  optional_keywords
package package_name add http server_name[:port] path  optional_keywords
```

package_name Specifies the package name in the form `SUNW`*name*. To view detailed information about packages and their names, on an installed system, use the `pkginfo -l` command.

server_name	Specifies the name of the server where you stored the package.
port	Specifies an optional port. *port* can be a port number or the name of a TCP service that has a port number that is determined at runtime. If you do not specify a port, the default HTTP port number 80 is used.
path	Specifies the location of the package to be retrieved from the specified server. When using an HTTP server, the package must be in package datastream format.
optional_keywords	Specifies the optional keywords to use when you retrieve a package from an HTTP server.

TABLE 8–6 Optional `package` Keywords to Use With HTTP

Keyword	Value Definition
`timeout` *min*	The `timeout` keyword enables you to specify, in minutes, the maximum length of time that is allowed to pass without receipt of data from the HTTP server. If a timeout occurs, the connection is closed, reopened, and resumed. If you specify a `timeout` value of 0 (zero), the connection is not reopened. If a timeout reconnection occurs, the package is retried from the beginning of the package and the data that was retrieved prior to the timeout is discarded.
`proxy` *host*:*port*	The `proxy` keyword enables you to specify a proxy host and proxy port. You can use a proxy host to retrieve a Solaris package from the other side of a firewall. You must supply a proxy port when you specify the `proxy` keyword.

EXAMPLE 8–13 Adding a Package by Using HTTP

In this example, the `package` profile keyword adds all the packages listed in the `Solaris 10` directory from the HTTP location `http://package.central/Solaris_10`. If five minutes pass and no data is received, the package data is retrieved again. Previous package data is discarded. Either of the following formats can be used.

```
package SUNWnew add http package.central/Solaris_10 timeout 5

package SUNWnew add http://package.central/Solaris_10 timeout 5
```

EXAMPLE 8–14 Adding a Package by Using HTTP with a Proxy Port

In this example, the `package` profile keyword adds all the packages listed in the `Solaris_10` directory from the HTTP location `http://package.central/Solaris_10`. The package is retrieved across a firewall by using the `proxy` keyword.

```
package SUNWnew add http://package.central/Solaris_10 proxy webcache.east:8080
```

Packages Stored on a Local Device

You can retrieve a Solaris package from a local device if you stored the package on a file system-oriented, random-access device, such as a diskette or a DVD-ROM. Use the following syntax for the `package` keyword.

`package` *package_name* `add local_device` *device path file_system_type*

package_name	Specifies the package name in the form `SUNW`*name*. To view detailed information about packages and their names, on an installed system, use the `pkginfo -l` command.
device	Specifies the name of the drive where the Solaris package resides. If the device name is a canonical path, the device is mounted directly. If you supply a device name that is not a canonical path, the installation utility adds `/dev/dsk/` to the path.
path	Specifies the path to the Solaris package, relative to the root (`/`) file system on the device you specified.
file_system_type	Specifies the type of file system on the device. If you do not supply a file system type, the installation utility attempts to mount a UFS file system. If the UFS mount fails, the installation utility attempts to mount an HSFS file system.

EXAMPLE 8–15 Adding a Package by Using a Local Device With a UFS File System

In this example, the `package` profile keyword adds the `SUNWnew` package from the directory `/Solaris_10/Product` from the local device c0t6d0s0. This is a UFS file system.

```
package SUNWnew add local_device c0t6d0s0 /Solaris_10/Product ufs
```

EXAMPLE 8–16 Adding a Package by Using a Local Device From an HSFS File System

In this example, the `package` profile keyword adds the `SUNWnew` package from the directory `/Solaris_10/Product` from the local device c0t6d0s0. This is an HSFS file system.

```
package SUNWnew add local_device c0t6d0s0 /Solaris_10/Product  hsfs
```

Packages Stored on a Local File

A package can be installed from the miniroot from which you booted the system. When you perform a custom JumpStart installation, you boot the system from a DVD, CD, or an NFS-based miniroot. The installation software is loaded and run from this miniroot. Therefore, a package that you stored in the DVD, CD, or NFS-based miniroot is accessible as a local file. Use the following syntax for the `package` keyword.

`package` *package_name* `add local_file` *path*

package_name Specifies the package name in the form `SUNW`*name*. To view detailed information about packages and their names, on an installed system, use the `pkginfo -l` command.

path Specifies the location of the package. The path must be accessible to the system as a local file while the system is booted from the Solaris Software - 1 CD or from the Solaris Operating System DVD. The system cannot access `/net` when it is booted from the Solaris Software - 1 CD or from the Solaris Operating System DVD.

EXAMPLE 8–17 Adding a Package by Using a Local File

In this example, the `package` profile keyword adds the `SUNWnew` package from the `/Solaris_10/Product` directory.

```
package SUNWnew add local_file /Solaris_10/Product
```

Limitations When Using the `package` Keyword

Note these limitations when using the `package` keyword:

- Some packages are required and cannot be deleted.
- You cannot individually add or delete localization packages by using the `package` profile keyword. To add localization packages, use the `locale` profile keyword.
- Packages cannot be retrieved from an FTP server location or local backup, such as tape.
- Packages within the Solaris distribution being installed cannot be added from alternate locations. If a package from the Solaris distribution is specified, the package cannot be followed by an alternative location in order to maintain consistency with the resulting installed system.
- In order to install without manual intervention, the package must be installable by using the `pkgadd` command. The same `admin` file must be used to install the software group packages and the package that resides in another location.
  - If the retrieval_type is HTTP, then the package must be in stream format.
  - If the retrieval_type is NFS server, local device, or local file, then the package should follow standard packaging format with the directory name being the same as the package being installed.
  - If a package is being added from a separate location and a package depends on another package that is not currently installed, the package is not installed. An error message is logged into the install or upgrade log file.
- If the package is being installed with a Solaris Flash archive, follow these guidelines.
  - Any package installed must be compatible with the archive.

- If a package is present in the archive, the JumpStart overwrites the existing package.

Upgrade Behavior When Using the `package` Keyword

When you use `package` for an upgrade, the JumpStart program performs the following actions:

- All packages already on the system are automatically upgraded.
- If you specify *package_name* `add` and *package_name* is not installed on the system, the package is installed.
- If you specify *package_name* `delete` and *package_name* is installed on the system, the package is deleted *before* the upgrade begins.
- If you specify *package_name* `delete` and *package_name* is not installed on the system, the package is not installed if the package is part of a cluster that is designated to be installed.

`partitioning` Profile Keyword

`partitioning` *type*

`partitioning` defines how the disks are divided into slices for file systems during the installation.

If you do not specify `partitioning` in the profile, the `default` type of partitioning is used by default.

type Use one of the following values:

`default` The JumpStart program selects the disks and creates the file systems on which to install the specified software, except for any file systems that are specified by the `filesys` keywords. `rootdisk` is selected first. The JumpStart program uses additional disks if the specified software does not fit on `rootdisk`.

`existing` The JumpStart program uses the existing file systems on the system's disks. All file systems except `/`, `/usr`, `/usr/openwin`, `/opt`, and `/var` are preserved. The JumpStart program uses the last mount-point field from the file system superblock to determine which file-system mount point the slice represents.

Note – When you use both the `filesys` and `partitioning existing` profile keywords, you must set size *size* to `existing`.

`explicit` The JumpStart program uses the disks and creates the file systems that are specified by the `filesys` keywords. If you specify only the root (`/`) file system with the `filesys` keyword, all of the Solaris software is installed in the root (`/`) file system.

Note – If you use the `explicit` profile value, you must use the `filesys` keyword to specify the disks to use and file systems to create.

`patch` Profile Keyword

`patch` *patch_id_list* | *patch_file patch_location optional_keywords*]

patch_id_list Specifies the patch ID numbers that are to be installed. The list should be a list of comma-separated Solaris patch IDs. The patches are installed in the order specified in the list. Do not add a space after the comma, for example: 112467-01,112765-02.

patch_file A file with a list of patches that is found in the *patch_location*. The patches are installed in the order specified in the file.

patch_location Specifies the location where the patches reside. The locations allowed are the following:

- NFS server
- HTTP server
- Local device
- Local file

optional_keywords Optional keywords depend on where patches are stored. The following sections describe the possible locations and optional keywords.

Patches Stored on an NFS Server

If the patch is stored on an NFS server, use one of the following syntaxes for the `patch` keyword.

```
patch patch_id_list | patch_file nfs server_name:/patch_directory [retry n]
patch patch_id_list | patch_file nfs://server_name/patch_director  [retry n]
```

patch_id_list Specifies the patch ID numbers that are to be installed. The list should be a list of comma-separated Solaris patch IDs. The patches are installed in the order specified in the list.

patch_file A file with a list of patches that is found in the *patch_location*. The patches are installed in the order specified in the file.

server_name Specifies the name of the server where you stored the patches.

patch_directory Specifies the location of the patch directory on the specified server. The patches must be in standard patch format.

retry *n* Is an optional keyword. *n* is the maximum number of times the install utility attempts to mount the directory.

EXAMPLE 8–18 Adding a Patch With an Ordered List by Using NFS

In this example, the `patch` profile keyword adds all the patches listed in the `patch` file from the NFS patch directory `nfs://patch_master/Solaris/v10/patches`. Patches are installed in the order listed in the `patch`. If a mount fails, the NFS mount is tried five times.

```
patch patch_file nfs://patch_master/Solaris/v10/patches retry 5
```

EXAMPLE 8–19 Adding a Patch by Using NFS

In this example, the `patch` profile keyword adds the patches 112467–01 and 112765–02 from the patch directory `/Solaris/v10/patches` on the server `patch_master`.

```
patch 112467-01,112765-02 nfs patch_master:/Solaris/v10/patches
```

Patches Stored on an HTTP Server

If the patch is stored on an HTTP server, use one of the following syntaxes for the `patch` keyword.

`patch` *patch_id_list* | *patch_file* `http://`*server_name [:port] patch_directory optional_http_keywords*

`patch` *patch_id_list* | *patch_file* `http` *server_name [:port] patch_directory optional_http_keywords*

patch_id_list Specifies the patch ID numbers that are to be installed. The list should be a list of comma-separated Solaris patch IDs. The patches are installed in the order specified in the list. Do not add a space after the comma, for example: 112467-01,112765-02.

patch_file A file with a list of patches that is found in the *patch_location*. The patches are installed in the order specified in the file.

server_name Specifies the name of the server where you stored the patch.

port Specifies an optional port. *port* can be a port number or the name of a TCP service that has a port number that is determined at runtime.

If you do not specify a port, the default HTTP port number 80 is used.

patch_directory	Specifies the location of the patch directory to be retrieved from the specified server. When using an HTTP server, the patch must be in JAR format.
optional_keywords	Specifies the optional keywords to use when you retrieve a patch from an HTTP server.

TABLE 8–7 Optional `patch` Keywords to Use With HTTP

Keyword	Value Definition
`timeout` *min*	The `timeout` keyword enables you to specify, in minutes, the maximum length of time that is allowed to pass without receipt of data from the HTTP server. If a timeout occurs, the connection is closed, reopened, and resumed. If you specify a `timeout` value of 0 (zero), the connection is not reopened. If a timeout reconnection occurs, the package is retried from the beginning of the package and the data that was retrieved prior to the timeout is discarded.
`proxy` *host*:*port*	The `proxy` keyword enables you to specify a proxy host and proxy port. You can use a proxy host to retrieve a Solaris package from the other side of a firewall. You must supply a proxy port when you specify the `proxy` keyword.

EXAMPLE 8–20 Adding a Patch With an Ordered List by Using HTTP

In this example, the `patch` profile keyword adds all the patches listed in the `patch_file` file from the HTTP location `http://patch.central/Solaris/v10/patches`. The patches are installed in the order specified in the file the `patch` file. If five minutes pass and no data is received, the patch data is retrieved again. Previous patch data is discarded.

```
patch patch_file http://patch.central/Solaris/v10/patches timeout 5
```

EXAMPLE 8–21 Adding a Patch by Using HTTP

In this example, the `patch` profile keyword entry adds the patches 112467–01 and 112765–02 from the patch location `http://patch_master/Solaris/v10/patches`.

```
patch 112467-01,112765-02 http://patch.central/Solaris/v10/patches
```

Patches Stored on a Local Device

You can retrieve a Solaris package from a local device if you stored the package on a file system-oriented, random-access device, such as a diskette or a DVD-ROM. Use the following syntax for the `patch` keyword.

```
patch patch_id_list | patch_file local_device \
device path file_system_type
```

patch_id_list Specifies the patch ID numbers that are to be installed. The list should be a list of comma-separated Solaris patch IDs. The patches are installed in the order specified in the list. Do not add a space after the comma, for example: 112467-01,112765-02.

patch_file A file with a list of patches that is found in the *patch_location*. The patches are installed in the order specified in the file.

device Specifies the name of the drive where the Solaris package resides. If the device name is a canonical path, the device is mounted directly. If you supply a device name that is not a canonical path, the installation utility adds `/dev/dsk/` to the path.

path Specifies the path to the Solaris patch, relative to the root (`/`) file system on the device you specified.

file_system_type Specifies the type of file system on the device. If you do not supply a file system type, the installation utility attempts to mount a UFS file system. If the UFS mount fails, the installation utility attempts to mount an HSFS file system.

EXAMPLE 8–22 Adding a Patch With an Ordered List by Using a Local Device

In this example, the `patch` profile keyword adds all the patches listed in the `patch_file` file from the directory `/Solaris_10/patches` from the local device c0t6d0s0. The `patch` file determines the order of patches to be installed.

```
patch patch_file c0t6d0s0 /Solaris_10/patches
```

EXAMPLE 8–23 Adding a Patch by Using a Local Device

In this example, the `patch` profile keyword adds the patches 112467–01 and 112765–02 from the patch directory `/Solaris_10/patches` from local device c0t6d0s0.

```
patch 112467-01,112765-02 local_device c0t6d0s0 /Solaris_10/patches
```

Patches Stored on a Local File

A patch can be installed from the miniroot from which you booted the system. When you perform a custom JumpStart installation, you boot the system from a DVD, CD, or an NFS-based miniroot. The installation software is loaded and run from this miniroot. Therefore, a patch that you stored in the DVD, CD, or NFS-based miniroot is accessible as a local file. Use the following syntax for the `patch` keyword.

`patch` *patch_id_list* | *patch_file* `local_file` *patch _directory*

patch_id_list Specifies the patch ID numbers that are to be installed. The list should be a list of comma-separated Solaris patch IDs. The patches are installed in the order specified in the list. Do not add a space after the comma, for example: 112467-01,112765-02.

patch_file A file with a list of patches that is found in the *patch_location*. The patches are installed in the order specified in the file.

patch_directory Specifies the location of the patch directory. The patch directory must be accessible to the system as a local file while the system is booted from the Solaris Software - 1 CD or from the Solaris Operating System DVD. The system cannot access `/net` when it is booted from the Solaris Software - 1 CD or from the Solaris Operating System DVD.

EXAMPLE 8–24 Adding a Patch With an Ordered List by Using a Local File

In this example, the `patch` profile keyword adds all the patches that are listed in the `patch_file` file from the `/Solaris_10/patches` directory. The `patch` file determines the order of patches to be installed.

```
patch patch_cal_file /Solaris_10/patches
```

EXAMPLE 8–25 Adding a Patch by Using a Local File

In this example, the `patch` profile keyword adds the patches 112467–01 and 112765–02 from the patch directory `/Solaris_10/patches`.

```
patch 112467-01,112765-02 local_file /Solaris_10/patches
```

Limitations When Using the `patch` Keyword

Note the following limitations when using the `patch` keyword:

- Patches cannot be retrieved from FTP locations or local backup, such as tape.
- Signed patches cannot be added.
- Patches must be installable with the `patchadd` command.
- If a patch depends on a patch that is not currently installed, the patch is not installed. An error message is logged into the installation or upgrade log file.
- You must determine the correct order of the patches for a correct installation of the patches.

`pool` Profile Keyword (ZFS Only)

The `pool` keyword defines the installation of a ZFS root pool. The pool is installed with a software group specified with the `cluster` keyword. The *poolsize*, *swapsize*, *dumpsize*, and *vdevlist* options are needed for creating a new root pool.

For a complete description of the `pool` keyword and other keywords that can be used for a ZFS root pool, see "`pool` Profile Keyword (ZFS Only)" on page 166.

`root_device` Profile Keyword (UFS and ZFS)

Note – The `root_device` keyword can be used when you install either a UFS file system or a ZFS root pool. The usage for this keyword is the limited to a single system for ZFS installations.

- For a complete list of keywords that can be used in a UFS or ZFS installation, see Table 8–2
- For a description of how the `root_device` keyword can be used when installing a ZFS root pool, see "JumpStart Keywords for a ZFS Root (/) File System (Reference)" on page 164

`root_device` *slice*

`root_device` designates the system's root disk. "How the System's Root Disk Is Determined" on page 154 contains additional information.

Note – The root disk is determined by the JumpStart program and determines where the OS is to be installed. The rules file uses a probe keyword "`rootdisk`," but this keyword is used differently than the "rootdisk" keyword used in the JumpStart profile. You cannot set the place of installation by using the probe keyword "`rootdisk`" in the rules file. The probe keyword, `rootdisk`, determines where to boot from during the installation. See Table 8–10.

When you are upgrading a system, `root_device` designates the root (`/`) file system and the file systems that are mounted by its `/etc/vfstab` file to be upgraded. You must specify `root_device` if more than one root (`/`) file system can be upgraded on a system. You must specify *slice* in the form c*w*t*x*d*y*s*z* or c*x*d*y*s*z*.

When you use the `root_device` keyword, consider the following:

- If you specify `root_device` on a system with only one disk, the `root_device` and the disk must match. Also, any `filesys` keywords that specify the root (`/`) file system must match `root_device`.
- If you are upgrading a RAID-1 volume (mirror), the value that is specified for `root_device` should be one side of the mirror. The other side of the mirror is automatically upgraded.

EXAMPLE 8–26 `root_device` Profile Keyword

```
root_device c0t0d0s2
```

How the System's Root Disk Is Determined

A system's root disk is the disk on the system that contains the root (`/`) file system. In a profile, you can use the `rootdisk` variable in place of a disk name, which the JumpStart program sets to the system's root disk. Table 8–8 describes how the JumpStart program determines the system's root disk for the installation.

Note – The JumpStart program only determines a system's root disk size during an initial installation. You cannot change a system's root disk during an upgrade.

TABLE 8–8 How JumpStart Determines a System's Root Disk (Initial Installation)

Stage	Action
1	If the `root_device` keyword is specified in the profile, the JumpStart program sets `rootdisk` to the root device.
2	If `rootdisk` is not set and the `boot_device` keyword is specified in the profile, the JumpStart program sets `rootdisk` to the boot device.
3	If `rootdisk` is not set and a `filesys` *cwtxdysz size* `/` entry is specified in the profile, the JumpStart program sets `rootdisk` to the disk that is specified in the entry.
4	If `rootdisk` is not set and a `rootdisk.s`*n* entry is specified in the profile, the JumpStart program searches the system's disks in kernel probe order for an existing root file system on the specified slice. If a disk is found, the JumpStart program sets `rootdisk` to the found disk.
5	If `rootdisk` is not set and `partitioning existing` is specified in the profile, the JumpStart program searches the system's disks in kernel probe order for an existing root file system. If a root file system is not found or more than one is found, an error occurs. If a root file system is found, the JumpStart program sets `rootdisk` to the found disk.
6	If `rootdisk` is not set, the JumpStart program sets `rootdisk` to the disk where the root (`/`) file system is installed.

`system_type` Profile Keyword

`system_type` *type_switch*

`system_type` defines the type of system on which the Solaris OS is to be installed.

type_switch represents the option `standalone` or `server`, which you use to indicate the type of system on which the Solaris software is to be installed. If you do not specify `system_type` in a profile, `standalone` is used by default.

`usedisk` Profile Keyword (UFS and ZFS)

Note – The `usedisk` keyword can be used when you install either a UFS file system or a ZFS root pool. The usage for this keyword is the same in both UFS and ZFS installations.

- For a complete list of keywords that can be used in a UFS or ZFS installation, see Table 8–2
- For information on performing a ZFS installation, see Chapter 9, "Installing a ZFS Root Pool With JumpStart"

`usedisk` *disk_name* `...`

By default, the JumpStart program uses all of the operational disks on the system when you specify `partitioning default`. The `usedisk` profile keyword designates one or more disks that you want the JumpStart program to use. You must specify *disk_name* in the form c*x*t*y*d*z* or c*y*d*z*, for example, `c0t0d0` or `c0d0s0`.

If you specify `usedisk` in a profile, the JumpStart program uses only the disks that you specify after the `usedisk` keyword.

Note – You cannot specify the `usedisk` keyword and the `dontuse` keyword in the same profile.

Custom JumpStart Environment Variables

You can use environment variables in your begin and finish scripts. For example, a begin script might extract the disk size, `SI_DISKSIZES`, and install or not install particular packages on a system, based on the actual disk size the script extracts.

Information that is gathered about a system is stored in these environment variables, which are generally set or not, depending on the rule keywords and values you use in the `rules` file.

For example, information about which operating system is already installed on a system is only available in `SI_INSTALLED` after the `installed` keyword is used.

Table 8–9 describes these variables and their values.

TABLE 8–9 Installation Environment Variables

Environment Variable	Value
`SI_ARCH`	The hardware architecture of the install client. The `SI_ARCH` variable is set when the `arch` keyword is used in the `rules` file.

TABLE 8–9 Installation Environment Variables *(Continued)*

Environment Variable	Value
SI_BEGIN	The name of the begin script, if one is used.
SI_CLASS	The name of the profile that is used to install the install client.
SI_DISKLIST	A comma-separated list of disk names on the install client. The SI_DISKLIST variable is set when the disksize keyword is used and matched in the rules file. The SI_DISKLIST and SI_NUMDISKS variables are used to determine the physical disk to use for the rootdisk. rootdisk is described in "How the System's Root Disk Is Determined" on page 154.
SI_DISKSIZES	A comma-separated list of disk sizes on the install client. The SI_DISKSIZES variable is set when the disksize keyword is used and matched in the rules file.
SI_DOMAINNAME	The domain name. The SI_DOMAINNAME variable is set when the dommainname keyword is used and matched in the rules file.
SI_FINISH	The name of the finish script, if one is used.
SI_HOSTADDRESS	The install client's IP address.
SI_HOSTNAME	The install client's host name. The SI_HOSTNAME variable is set when the hostname keyword is used and matched in the rules file.
SI_INSTALLED	The device name of a disk with a specific operating system on the disk, for example, Solaris, SunOS, or System V. The SI_INSTALLED variable is set when the installed keyword is used and matched in the rules file. SI_INST_OS and SI_INST_VER are used to determine the value of SI_INSTALLED.
SI_INST_OS	The name of the operating system. SI_INST_OS and SI_INST_VER are used to determine the value of SI_INSTALLED.
SI_INST_VER	The version of the operating system. SI_INST_OS and SI_INST_VER are used to determine the value of SI_INSTALLED.
SI_KARCH	The install client's kernel architecture. The SI_KARCH variable is set when the karch keyword is used and matched in the rules file.
SI_MEMSIZE	The amount of physical memory on the install client. The SI_MEMSIZE variable is set when the memsize keyword is used and matched in the rules file.
SI_MODEL	The install client's model name. The SI_MODEL variable is set when the model keyword is used and matched in the rules file.
SI_NETWORK	The install client's network number. The SI_NETWORK variable is set when the network keyword is used and matched in the rules file.
SI_NUMDISKS	The number of disks on an install client. The SI_NUMDISKS variable is set when the disksize keyword is used and matched in the rules file. The SI_NUMDISKS and SI_DISKLIST variables are used to determine the physical disk to use for the rootdisk. rootdisk is described in "How the System's Root Disk Is Determined" on page 154.

TABLE 8–9 Installation Environment Variables *(Continued)*

Environment Variable	Value
SI_OSNAME	The operating system release on the Solaris software image. For example, you can use the SI_OSNAME variable in a script if you are installing the Solaris software on systems that are based on the version of the operating system on the Solaris Operating System DVD or the Solaris Software - 1 CD image.
SI_ROOTDISK	The device name of the disk that is represented by the logical name rootdisk. The SI_ROOTDISK variable is set when the disksize or the installed keyword is set to rootdisk in the rules file. The SI_ROOTDISK variable sets the device to boot from during the installation. **Note –** You cannot set the place of installation by using the probe keyword "rootdisk" in the rules file. For information on the "rootdisk" variable that is set in a JumpStart profile, see "How the System's Root Disk Is Determined" on page 154.
SI_ROOTDISKSIZE	The size of the disk that is represented by the logical name rootdisk. The SI_ROOTDISKSIZE variable is set when the disksize or the installed keyword is set to rootdisk in the rules file.
SI_TOTALDISK	The total amount of disk space on the install client. The SI_TOTALDISK variable is set when the totaldisk keyword is used and matched in the rules file.

Probe Keywords and Values

Table 8–10 describes each rule keyword and its equivalent probe keyword.

Note – Always place probe keywords at or near the beginning of the rules file.

TABLE 8–10 Descriptions of Probe Keywords

Rule Keyword	Equivalent Probe Keyword	Description of Probe Keyword
any	None	
arch	arch	Determines the kernel architecture, i386 or SPARC, and sets SI_ARCH.
disksize	disks	Returns the size of a system's disks in Mbytes in kernel probe order, c0t3d0s0, c0t3d0s1, c0t4d0s0. disksize sets SI_DISKLIST, SI_DISKSIZES, SI_NUMDISKS, and SI_TOTALDISK.
domainname	domainname	Returns a system's NIS or NIS+ domain name or blank and sets SI_DOMAINNAME. The domainname keyword returns the output of domainname(1M).
hostaddress	hostaddress	Returns a system's IP address, the first address that is listed in the output of ifconfig(1M) -a that is not lo0, and sets SI_HOSTADDRESS.
hostname	hostname	Returns a system's host name that is the output from uname(1) -n and sets SI_HOSTNAME.

TABLE 8–10 Descriptions of Probe Keywords *(Continued)*

Rule Keyword	Equivalent Probe Keyword	Description of Probe Keyword
`installed`	`installed`	Returns the version name of the Solaris OS that is installed on a system and sets `SI_ROOTDISK` and `SI_INSTALLED`. If the JumpStart program finds a Solaris release but is unable to determine the version, the version that is returned is `SystemV`.
`karch`	`karch`	Returns a system's platform group, for example i86pc or sun4u, and sets `SI_KARCH`. For a list of platform names, see *Solaris Sun Hardware Platform Guide* at `http://docs.sun.com`.
`memsize`	`memsize`	Returns the size of physical memory on a system in Mbytes and sets `SI_MEMSIZE`.
`model`	`model`	Returns a system's platform name and sets `SI_MODEL`. For a list of platform names, see the *Solaris Sun Hardware Platform Guide* at `http://docs.sun.com`.
`network`	`network`	Returns a system's network number, which the JumpStart program determines by performing a logical AND between the system's IP address and the subnet mask. The system's IP address and the subnet mask are extracted from the first address that is listed in the output of `ifconfig(1M) -a` that is not lo0. The `network` keyword sets `SI_NETWORK`.
`osname`	`osname`	Returns the version and operating system name of the Solaris OS that is found on a CD and sets `SI_OSNAME`. If the JumpStart program finds a Solaris release but is unable to determine the version, the version that is returned is `SystemV`.
	`rootdisk`	Returns the name and size in Mbytes of a system's root disk and sets `SI_ROOTDISK`.
`totaldisk`	`totaldisk`	Returns the total disk space on a system (in Mbytes) and sets `SI_TOTALDISK`. The total disk space includes all of the operational disks that are attached to a system.

◆ ◆ ◆ CHAPTER 9

Installing a ZFS Root Pool With JumpStart

This chapter provides the information necessary for performing a JumpStart installation for ZFS root pool. The following sections provides planning information, profile examples, and profile keyword descriptions.

JumpStart Installation for a ZFS Root (/) File System (Overview and Planning)

This chapter provides the information for you to create a JumpStart profile to install a ZFS root pool.

Note – If you want to install a UFS root (/) file system, all existing profile keywords work as in previous Solaris releases. For a list of UFS profile keywords, see Chapter 8, "Custom JumpStart (Reference)."

A ZFS specific profile must contain the `pool` keyword. The `pool` keyword installs a new root pool and a new boot environment is created by default. You can provide the name of the boot environment and you can create a separate `/var` dataset with existing `bootenv installbe` keywords and the new `bename` and `dataset` options. Some keywords that are allowed in a UFS-specific profile are not allowed in a ZFS specific profile, such as those specifying the creation of UFS mount points.

For overall ZFS planning information, see Chapter 6, "ZFS Root File System Installation (Planning)," in *Solaris 10 5/09 Installation Guide: Planning for Installation and Upgrade*.

Limitations for a JumpStart installation for a ZFS Root Pool

Keep the following issues in mind before considering a JumpStart installation of a bootable ZFS root pool.

TABLE 9–1 JumpStart Limitations for ZFS Root Pools

Limitation	Description	For More Information

For a JumpStart installation, you cannot use an existing ZFS storage pool to create a bootable ZFS root pool.	You must create a new ZFS storage pool with syntax similar to the following:
	`pool rpool 20G 4G 4G c0t0d0s0`
	The complete `pool` keyword line is required because you cannot use an existing pool. The `bootenv` keyword line is optional. If you do not use `bootenv`, a default boot environment is created for you. For example:
	`install_type initial_install`
	`cluster SUNWCall`
	`pool rpool 20G 4g 4g any`
	`bootenv installbe bename newBE`
		"`pool` Profile Keyword (ZFS Only)" on page 166

You cannot create a pool with whole disks.	You must create your pool with disk slices rather than whole disks. If in the profile you create a pool with whole disks, such as `c0t0d0`, the installation fails. You will receive an error message similar to the following. `Invalid disk name (c0t0d0)`	
Some keywords that are allowed in a UFS specific profile are not allowed in a ZFS specific profile, such as those specifying the creation of UFS mount points.		"Profile Keywords Quick Reference" on page 112
You cannot upgrade with JumpStart. You must use Solaris Live Upgrade	With Solaris Live Upgrade, you can create a copy of the currently running system. This copy can be upgraded and then activated to become the currently running system.	Chapter 11, "Solaris Live Upgrade and ZFS (Overview)," in *Solaris 10 5/09 Installation Guide: Solaris Live Upgrade and Upgrade Planning*

JumpStart Profile Examples for a ZFS Root Pool

This section provides examples of ZFS specific JumpStart profiles.

Note – For the ZFS root pool to be upgradeable and bootable, you must create your pool with disk slices rather than whole disks. If in the profile you create a pool with whole disks, such as `c0t0d0`, you will receive an error message similar to the following.

```
Invalid disk name (c0t0d0)
```

EXAMPLE 9–1 Installing a Mirrored ZFS Root Pool

```
install_type initial_install
cluster SUNWCall
pool newpool auto auto auto mirror c0t0d0s0 c0t1d0s0
bootenv installbe bename solaris10_6
```

The following list describes some of the keywords and values from this example.

`install_type initial_install`
: The `install_type` keyword is required in every profile. The `initial_install` keyword performs an initial installation that installs a new Solaris OS in a new ZFS root pool.

`cluster`
: The Entire Distribution software group, `SUNWCall`, is installed on the system. For more information about software groups, see "Disk Space Recommendations for Software Groups" in *Solaris 10 5/09 Installation Guide: Planning for Installation and Upgrade*.

`pool`
: The `pool` keyword defines the characteristics of the new ZFS root pool.

  newpool
  : Defines the name of the root pool.

  `auto`
  : Specifies the size of the disks automatically. The size is determined by the size of the specified disks.

  `auto`
  : The swap area is automatically sized with the `auto` keyword. The default size is 1/2 the size of physical memory, but no less than 512 Mbytes and no greater than 2 Gbytes. You can set the size outside this range by using the `size` option.

  `auto`
  : The dump device is automatically sized.

EXAMPLE 9–1 Installing a Mirrored ZFS Root Pool *(Continued)*

`mirror`	The mirrored configuration of disks has the `mirror` keyword and disk slices specified as *c0t0d0s0* and *c0t1d0s0*.
`bootenv`	`installbe` changes the characteristics of the default boot environment that is created during the installation.
`bename`	Names the new boot environment *solaris10_6*.

EXAMPLE 9–2 Customizing the Disk Size For a ZFS Root Pool

```
install_type initial_install
cluster SUNWCall
pool newpool 80g 2g 2g mirror any any
bootenv installbe bename solaris10_6
```

The following list describes some of the keywords and values from this example.

`install_type initial_install`	The `install_type` keyword is required in every profile. The `initial_install` keyword performs an initial installation that installs a new Solaris OS in a new ZFS root pool.
`cluster`	The Entire Distribution software group, `SUNWCall`, is installed on the system. For more information about software groups, see "Disk Space Recommendations for Software Groups" in *Solaris 10 5/09 Installation Guide: Planning for Installation and Upgrade*.
`pool`	The `pool` keyword defines the characteristics of the new ZFS root pool.
newpool	Specifies the name of the root pool.
`80g`	Specifies the size of the disk slice.
`2g`	The swap area and dump volumes are 2-Gbytes.
`mirror`	The mirrored configuration of disks has the `mirror` keyword and disk slices specified as *c0t0d0s0* and *c0t1d0s0*.
	The `any` options in the mirrored configuration finds any two available

EXAMPLE 9–2 Customizing the Disk Size For a ZFS Root Pool *(Continued)*

devices that are large enough to create a 80-Gbyte pool. If two such devices are not available, the install fails.

`bootenv` `installbe` changes the characteristics of the default boot environment that is created during the installation.

- `bename` Names the new boot environment *solaris10_6*.

EXAMPLE 9–3 Specifying Where to Install the OS

```
install_type initial_install
cluster SUNWCall
root_device c0t0d0s0
pool nrpool auto auto auto rootdisk.s0
bootenv installbe bename bnv dataset /var
```

The following list describes some of the keywords and values from this example.

`install_type initial_install` The `install_type` keyword is required in every profile. The `initial_install` keyword performs an initial installation that installs a new Solaris OS in a new ZFS root pool.

`cluster` The Entire Distribution software group, `SUNWCall`, is installed on the system. For more information about software groups, see "Disk Space Recommendations for Software Groups" in *Solaris 10 5/09 Installation Guide: Planning for Installation and Upgrade*.

`root_device` Specifies the disk slice where the OS is to be installed. The *c0t0d0s0* defines the specific disk and slice for the OS.

`pool` The `pool` keyword defines the characteristics of the new ZFS root pool.

- *nrpool* Defines the name of the root pool.
- `auto` Specifies the size of the disks automatically. The size is determined by the size of the specified disks.
- `auto` The swap area is automatically sized with the `auto` keyword. The default size is 1/2 the size of physical memory,

EXAMPLE 9–3 Specifying Where to Install the OS *(Continued)*

		but no less than 512 Mbytes and no greater than 2 Gbytes. You can set the size outside this range by using the `size` option.
	`auto`	The dump device is automatically sized.
	`rootdisk.s0`	The device used to create the root pool is specified as slice 0.
`bootenv`	`installbe` changes the characteristics of the default boot environment that is created during the installation.	
	`bename`	Names the new boot environment *bnv*.
	`dataset`	Creates a `/var` dataset that is separate from the ROOT dataset. `/var` is the only value for `dataset`.

JumpStart Keywords for a ZFS Root (/) File System (Reference)

This section provides descriptions of some of the ZFS specific keywords that you can use in a JumpStart profile. The usage of the keywords in this section is either different from their usage in a UFS profile or used only in a ZFS profile.

- For a quick reference of both UFS and ZFS profile keywords, see "Profile Keywords Quick Reference" on page 112.
- The following list of keywords can be used in a ZFS profile. The usage is the same for both UFS and ZFS profiles. For descriptions of these keywords, see "Profile Keyword Descriptions and Examples" on page 113.
  - boot_device
  - cluster
  - dontuse
  - fdisk
  - filesys (mounting remote file systems)
  - geo
  - locale
  - package
  - usedisk

`bootenv` Profile Keyword (ZFS and UFS)

The `bootenv` keyword identifies boot environment characteristics. A boot environment is created by default during installation with the `pool` keyword. If you use the `bootenv` keyword with the `installbe` option, you can name the new boot environment and create a `/var` dataset within the boot environment.

This keyword can be used in a profile for installing a UFS file system or a ZFS root pool.

- In a UFS file system, this keyword is used for creating an empty boot environment for the future installation of a Solaris Flash archive. For the complete description of the bootenv keyword for UFS, see "`bootenv` Profile Keyword (UFS and ZFS)" on page 122.
- For a ZFS root pool, the `bootenv` keyword changes the characteristics of the default boot environment that is created at install time. This boot environment is a copy of the root file system that you are installing.

  The `bootenv` keyword can be used with the `installbe`, `bename` and `dataset` options. These options name the boot environment and create a separate `/var` dataset.

  ```
  bootenv installbe bename new-BE-name [dataset mount-point]
  ```

`installbe` — Changes the characteristics of the default boot environment that is created during the installation.

`bename` — Specifies the name of the new boot environment to be created, *new_BE_name*. The name can be no longer than 30 characters, can contain only alphanumeric characters, and can contain no multibyte characters. The name must be unique on the system.

`dataset` *mount-point* — Use the optional `dataset` keyword to identify a `/var` dataset that is separate from the ROOT dataset. The *mount-point* value is limited to `/var`. For example, a `bootenv` syntax line for separate `/var` dataset would be similar to the following:

```
bootenv installbe bename zfsroot dataset /var
```

For more information about upgrading and activating a boot environment, see Chapter 11, "Solaris Live Upgrade and ZFS (Overview)," in *Solaris 10 5/09 Installation Guide: Solaris Live Upgrade and Upgrade Planning*.

`install_type` Keyword (ZFS and UFS)

The `install_type` keyword is required in every profile. For a UFS installation, Several options are available. The only option available for a ZFS installation is the `initial_install` keyword. This option installs a new Solaris OS on a system. The profile syntax is the following:

```
install_type initial_install
```

Note – The following UFS options are not available for a ZFS installation.

- `upgrade` - You must use Solaris Live Upgrade to upgrade ZFS root pool. See Chapter 11, "Solaris Live Upgrade and ZFS (Overview)," in *Solaris 10 5/09 Installation Guide: Solaris Live Upgrade and Upgrade Planning*.
- `flash_install` - A Solaris Flash archive cannot be installed.
- `flash_update` - A Solaris Flash archive cannot be installed.

`pool` Profile Keyword (ZFS Only)

The pool keyword defines the new root pool to be created. The pool is then installed with a software group specified with the `cluster` keyword. The *poolsize*, *swapsize*, *dumpsize*, and *vdevlist* options are needed for creating a new root pool.

`pool` *poolname poolsize swapsize dumpsize vdevlist*

poolname Specifies the name of the new pool to be created. A new pool is created with the specified *size* and with the specified devices, *vdevlist*.

poolsize Size of the new pool to be created. If you denote the amount of space, the size is assumed to be in Mbytes, unless specified by `g` (Gbytes). You can also use the `auto` option.

`auto` Allocates the largest possible pool size given the constraints, such as size of the disks and preserved slices.

Note – The meaning of `auto` for the *poolsize* keyword is different from the `filesys` keyword use of `auto` in a UFS file system. In ZFS, the size of the disk is checked to verify that the minimum size can be accommodated. If the minimize size is available, the largest possible pool size is allocated given the constraints, such as size of the disks and preserved slices.

swapsize Size of the swap volume (`zvol`) to be created within a new root pool. The options are either `auto` or *size*.

`auto` The swap area is automatically sized. The default size is 1/2 the size of physical memory, but no less than 512 Mbytes and no greater than 2 Gbytes. You can set the size outside this range by using the `size` option.

size Can be used to specify an amount. Size is assumed to be in Mbytes, unless specified by `g` (Gbytes).

dumpsize Size of the dump volume to be created within a new pool.

`auto` Uses the default swap size.

size Can be used to specify an amount. Size is assumed to be in Mbytes, unless specified by `g` (Gbytes).

vdevlist One or more devices used to create the pool.

Devices in the *vdevlist* must be slices for the root pool. *vdevlist* can be either a *single-device-name* in the form c*w*t*x*d*y*s*z* or `mirror` or `any` option.

Note – The format of the *vdevlist* is the same as the format of the `zpool create` command.

---

single-device-name A disk slice in the form or c*w*t*x*d*y*s*z*, such as `c0t0d0s0`.

`mirror` [*device-names*|`any`] Specifies the mirroring of the disk.

At this time, only mirrored configurations are supported when multiple devices are specified. You can mirror as many as disks you like, but the size of the pool created is determined by the smallest of the specified disks. For more information about creating mirrored storage pools, see "Mirrored Storage Pool Configuration" in *Solaris ZFS Administration Guide*.

- *device-names* lists the devices to be mirrored. The names are in the form of c*w*t*x*d*y*s*z*, for example `c0t0d0s0` and `c0t0d1s5`.
- The `any` option enables the installer to choose the devices.

`any` Enables the installer to select a suitable device.

`root_device` Profile Keyword (ZFS and UFS)

`root_device` c*w*t*x*d*y*s*z*

`root_device` specifies the device to be used for the root pool. The `root_device` keyword determines where the operating system is installed. This keyword is used the same in both ZFS and a UFS file system with some limitations. For the ZFS root pool, the root device is limited to a single system. This keyword is not useful for mirrored pools.

*c*w*t*x*d*y*s*z Identifies the root disk where the operating system is installed.

Additional Resources

For additional information about the topics included in this chapter, see the resources listed in Table 9–2.

TABLE 9–2 Additional Resources

Resource	Location
For ZFS information, including overview, planning, and step-by-step instructions	*Solaris ZFS Administration Guide*
For a list of all JumpStart keywords	Chapter 8, "Custom JumpStart (Reference)"
For information about using Solaris Live Upgrade to migrate from UFS to ZFS or create a new boot environment in a ZFS root pool	Chapter 11, "Solaris Live Upgrade and ZFS (Overview)," in *Solaris 10 5/09 Installation Guide: Solaris Live Upgrade and Upgrade Planning*

PART II

Appendixes

This part contains troubleshooting and reference information.

◆ ◆ ◆ APPENDIX A

Troubleshooting (Tasks)

This chapter contains a list of specific error messages and general problems you might encounter when installing Solaris 10 5/09 software. The chapter also explains how to fix the problems. Start by using this list of sections to determine where in the installation process the problem occurred.

Note – When you see the phrase "bootable media," this means the Solaris installation program and JumpStart installation method.

Problems With Setting Up Network Installations

`Unknown client "`*host_name*`"`

Cause: The *host_name* argument in the `add_install_client` command is not a host in the naming service.

Solution: Add the host *host_name* to the naming service and execute the `add_install_client` command again.

`Error: <system name> does not exist in the NIS ethers map`

`Add it, and rerun the add_install_client command`

Description: When you run the `add_install_client` command, the command fails with the above error.

Cause: The client you are adding to the install server does not exist in the server's `/etc/ethers` file.

Solution: Add the needed information to the `/etc/ethers` file on the install server and run the `add_install_client` command again.

1. Become superuser or assume an equivalent role.
2. On the client, find the ethers address.

```
# ifconfig -a grep ethers
ether 8:0:20:b3:39:1d
```

3. On the install server, open the `/etc/ethers` file in an editor. Add the address to the list.
4. On the client, run `add_install_client` again as in this example.

```
# ./add_install_client bluegill sun4u
```

Problems With Booting a System

Booting From Media, Error Messages

`le0: No carrier - transceiver cable problem`

Cause: The system is not connected to the network.

Solution: If this is a nonnetworked system, ignore this message. If this is a networked system, ensure that the Ethernet cabling is attached securely.

`The file just loaded does not appear to be executable`

Cause: The system cannot find the proper media for booting.

Solution: Verify that the system has been set up properly to install the Solaris 10 5/09 software from the network from an install server. The following are examples of checks you can make.

- If you copied the images of the Solaris Operating System DVD or the Solaris Software CDs to the install server, ensure that you specified the correct platform group for the system when you set it up.
- If you are using DVD or CD media, ensure that the Solaris Operating System DVD or Solaris Software - 1 CD is mounted and accessible on the install server.

`boot: cannot open <filename> (SPARC based systems only)`

Cause: This error occurs when you override the location of the `boot -file` by explicitly setting it.

Note – *filename* is a variable for the name of the file affected.

Solution: Follow these instructions:

- Reset the `boot -file` in the PROM to " " (blank).
- Ensure that the diag-switch is set to off and to true.

`Can't boot from file/device`
Cause: The installation media cannot find the bootable media.

Solution: Ensure that the following conditions are met:

- The DVD-ROM or CD-ROM drive is installed properly and turned on.
- Solaris Operating System DVD or the Solaris Software - 1 CD is inserted into the drive.
- The disc is free of damage or dirt.

`WARNING: clock gained` *xxx* `days -- CHECK AND RESET DATE! (`**`SPARC based systems only`**`)`
Description: This is an informational message.

Solution: Ignore the message and continue with the installation.

`Not a UFS file system (`**`x86 based systems only`**`)`
Cause: When Solaris 10 5/09 software was installed (either through the Solaris installation program or custom JumpStart), no boot disk was selected. You now must edit the BIOS to boot the system.

Solution: Select the BIOS to boot. See your BIOS documentation for instructions.

Booting From Media, General Problems

`The system does not boot.`
Description: When initially setting up a custom JumpStart server, you might encounter boot problems that do not return an error message. To verify information about the system and how the system is booting, run the boot command with the `-v` option. When you use the `-v` option, the boot command displays verbose debugging information about the screen.

Note – If this flag is not given, the messages are still printed, but the output is directed to the system log file. For more information, see `syslogd(1M)`.

Solution: For SPARC based systems, at the `ok` prompt, type the following command.

```
ok boot net -v - install
```

`Boot from DVD media fails on systems with Toshiba SD-M 1401 DVD-ROM`

Description: If your system has a Toshiba SD-M1401 DVD-ROM with firmware revision 1007, the system cannot boot from the Solaris Operating System DVD.

Solution: Apply patch 111649–03, or later version, to update the Toshiba SD-M1401 DVD-ROM drive's firmware. The patch 111649–03 is available at sunsolve.sun.com.

`The system hangs or panics when nonmemory PC cards are inserted. (x86 based systems only)`

Cause: Nonmemory PC cards cannot use the same memory resources that are used by other devices.

Solution: To correct this problem, see the instructions for your PC card and check for the address range.

`The system hangs before displaying the system prompt. (x86 based systems only)`

Solution: You have hardware that is not supported. Check your hardware manufacturer's documentation.

Booting From the Network, Error Messages

`WARNING: getfile: RPC failed: error 5 (RPC Timed out).`

Description: This error occurs when you have two or more servers on a network responding to an install client's boot request. The install client connects to the wrong boot server, and the installation hangs. The following specific reasons might cause this error to occur:

Cause: *Reason 1:* `/etc/bootparams` files might exist on different servers with an entry for this install client.

Solution: *Reason 1:* Ensure that servers on the network do not have multiple `/etc/bootparams` entries for the install client. If they do have multiple entries, remove duplicate client entries in the `/etc/bootparams` file on all install servers and boot servers except the one you want the install client to use.

Cause: *Reason 2:* Multiple `/tftpboot` or `/rplboot` directory entries might exist for this install client.

Solution: *Reason 2:* Ensure that servers on the network do not have multiple `/tftpboot` or `/rplboot` directory entries for the install client. If they do have multiple entries, remove duplicate client entries from the `/tftpboot` or `/rplboot` directories on all install servers and boot servers except the one you want the install client to use.

Cause: *Reason 3:* An install client entry might exist in the `/etc/bootparams` file on a server and an entry in another `/etc/bootparams` file that enables all systems to access the profile server. Such an entry resembles the following:

```
* install_config=profile_server:path
```

A line that resembles the previous entry in the NIS or NIS+ `bootparams` table can also cause this error.

Solution: *Reason 3:* If a wildcard entry is in the naming service `bootparams` map or table (for example, `* install_config=`), delete it and add it to the `/etc/bootparams` file on the boot server.

`No network boot server. Unable to install the system. See installation instructions. (`**`SPARC based systems only`**`)`

Cause: This error occurs on a system that you are attempting to install from the network. The system is not set up correctly.

Solution: Ensure that you correctly set up the system to install from the network. See "Adding Systems to Be Installed From the Network With a CD Image" in *Solaris 10 5/09 Installation Guide: Network-Based Installations*.

`prom_panic: Could not mount file system (`**`SPARC based systems only`**`)`

Cause: This error occurs when you are installing Solaris from a network, but the boot software cannot locate the following:

- Solaris Operating System DVD, either the DVD or a copy of the DVD image on the install server
- Solaris Software - 1 CD image, either the Solaris Software - 1 CD or a copy of the CD image on the install server

Solution: Ensure that the installation software is mounted and shared.

- If you are installing Solaris from the install server's DVD-ROM or CD-ROM drive, ensure that the Solaris Operating System DVD or Solaris Software - 1 CD is inserted in the CD-ROM drive, is mounted, and is shared in the `/etc/dfs/dfstab` file.
- If installing from a copy of the Solaris Operating System DVD image or Solaris Software - 1 CD image on the install server's disk, ensure that the directory path to the copy is shared in the `/etc/dfs/dfstab` file.

`Timeout waiting for ARP/RARP packet...(`**`SPARC based systems only`**`)`

Cause: *Reason 1:* The client is trying to boot from the network, but it cannot find a system that knows about the client.

Solution: *Reason 1:* Verify the system's host name is in the NIS or NIS+ naming service. Also, verify the `bootparams` search order in the boot server's `/etc/nsswitch.conf` file.

For example, the following line in the `/etc/nsswitch.conf` file indicates that JumpStart or the Solaris installation program first looks in the NIS maps for `bootparams` information. If the program does not find any information, the installer looks in the boot server's `/etc/bootparams` file.

```
bootparams: nis files
```

Cause: *Reason 2:* The client's Ethernet address is not correct.

Solution: *Reason 2:* Verify that the client's Ethernet address in the install server's `/etc/ethers` file is correct.

Cause: *Reason 3:* In a custom JumpStart installation, the `add_install_client` command specifies the platform group that uses a specified server as an install server. If the wrong architecture value is used when using the `add_install_client`, this problem occurs. For example, the machine you want to install is a sun4u, but you used i86pc instead.

Solution: *Reason 3:* Rerun `add_install_client` with the correct architecture value.

`ip: joining multicasts failed on tr0 - will use link layer broadcasts for multicast (`**`x86 based systems only`**`)`

Cause: This error message is displayed when you boot a system with a token ring card. Ethernet multicast and token ring multicast do not work the same way. The driver returns this error message because an invalid multicast address was provided to it.

Solution: Ignore this error message. If multicast does not work, IP uses layer broadcasts instead and does not cause the installation to fail.

`Requesting Internet address for` *Ethernet_Address* `(`**`x86 based systems only`**`)`

Cause: The client is trying to boot from the network, but it cannot find a system that knows about the client.

Solution: Verify the system's host name is listed in the naming service. If the system's host name is listed in the NIS or NIS+ naming service, and the system continues to print this error message, try rebooting.

`RPC: Timed out No bootparams (whoami) server responding; still trying... (`**`x86 based systems only`**`)`

Cause: The client is trying to boot from the network, but it cannot find a system with an entry in the `/etc/bootparams` file on the install server.

Solution: Use `add_install_client` on the install server. Using this command adds the proper entry in the `/etc/bootparams` file, enabling the client to boot from the network.

`Still trying to find a RPL server... (`**`x86 based systems only`**`)`

Cause: The system is trying to boot from the network, but the server is not set up to boot this system.

Solution: On the install server, execute `add_install_client` for the system to be installed. The `add_install_client` command sets up an `/rplboot` directory, which contains the necessary network boot program.

`CLIENT MAC ADDR: FF FF FF FF FF FF (network installations with DHCP only)`
Cause: The DHCP server is not configured correctly. This error might occur if the options or macros are not correctly defined in the DHCP Manager software.

Solution: In the DHCP Manager software, verify that the options and macros are correctly defined. Confirm that the Router option is defined, and that the value of the Router option is correct for the subnet you are using for the network installation.

Booting From the Network, General Problems

`The system boots from the network, but from a system other than the specified install server.`
Cause: An `/etc/bootparams` and perhaps an `/etc/ethers` entry exist on another system for the client.

Solution: On the name server, update the `/etc/bootparams` entry for the system that is being installed. The entry should conform to the following syntax:

install_system `root=`*boot_server*`:`*path* `install=`*install_server*`:`*path*

Also, ensure that only one `bootparams` entry is on the subnet for the install client.

`The system does not boot from the network (network installations with DHCP only).`
Cause: The DHCP server is not configured correctly. This error might occur if the system is not configured as an installation client on the DHCP server.

Solution: In the DHCP manager software, verify that installation options and macros are defined for the client system. For more information, see "Preconfiguring System Configuration Information With the DHCP Service (Tasks)" in *Solaris 10 5/09 Installation Guide: Network-Based Installations*.

Initial Installation of the Solaris OS

`Initial installation fails`
Solution: If the Solaris installation fails, you must restart the installation. To restart the installation, boot the system from the Solaris Operating System DVD, the Solaris Software - 1 CD, or from the network.

You cannot uninstall the Solaris software after the software has been partially installed. You must restore your system from a backup or begin the Solaris installation process again.

`/cdrom/cdrom0/SUNW`*xxxx*`/reloc.cpio: Broken pipe`
Description: This error message is informational and does not affect the installation. The condition occurs when a write on a pipe does not have a reading process.

Solution: Ignore the message and continue with the installation.

`WARNING: CHANGE DEFAULT BOOT DEVICE (x86 based systems only)`
Cause: This is an informational message. The default boot device set in the system's BIOS might be set to a device that requires you to use the Solaris Device Configuration Assistant to boot the system.

Solution: Continue with the installation and, if necessary, change the system's default boot device specified in the BIOS after you install the Solaris software to a device that does not require the Solaris Device Configuration Assistant.

---

x86 only – If you are using the `locale` keyword to test a custom JumpStart profile for an initial installation, the `pfinstall -D` command fails to test the profile. For a workaround, see the error message "could not select locale," in the section, "Upgrading the Solaris OS" on page 180.

▼ x86: To Check IDE Disk for Bad Blocks

IDE disk drives do not automatically map out bad blocks like other drives supported by Solaris software. Before installing Solaris on an IDE disk, you might want to perform a surface analysis on the disk. To perform surface analysis on an IDE disk, follow this procedure.

1 Become superuser or assume an equivalent role.

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

2 Boot to the installation media.

3 When you are prompted to select an installation type, select option 6, Single user shell.

4 Start the `format`(1M) program.

```
# format
```

5 Specify the IDE disk drive on which you want to perform a surface analysis.

```
# cxdy
```

cx Is the controller number

d*y* Is the device number

6 **Determine if you have an `fdisk` partition.**

- If a Solaris `fdisk` partition already exists, proceed to Step 7.
- If a Solaris `fdisk` partition does not exist, use the `fdisk` command to create a Solaris partition on the disk.

```
format> fdisk
```

7 **To begin the surface analysis, type:**

```
format> analyze
```

8 **Determine the current settings, type:**

```
analyze> config
```

9 **(Optional) To change settings, type:**

```
analyze> setup
```

10 **To find bad blocks, type:**

`analyze>` *type_of_surface_analysis*

type_of_surface_analysis Is read, write, or compare

If `format` finds bad blocks, it remaps them.

11 **To exit the analysis, type:**

```
analyze> quit
```

12 **Determine if you want to specify blocks to remap.**

- If no, go to Step 13.
- If yes, type:

```
format> repair
```

13 **To exit the format program, type:**

```
quit
```

14 **Restart the media in multiuser mode by typing the following command.**

```
# exit
```

Upgrading the Solaris OS

Upgrading, Error Messages

`No upgradable disks`
Cause: A swap entry in the `/etc/vfstab` file is causing the upgrade to fail.

Solution: Comment out the following lines in the `/etc/vfstab` file:

- All swap files and slices on disks not being upgraded
- Swap files that are no longer present
- Any unused swap slices

`usr/bin/bzcat not found`
Cause: Solaris Live Upgrade fails because of needing a patch cluster.

Solution: A patch is needed to install Solaris Live Upgrade. Ensure that you have the most recently updated patch list by consulting http://sunsolve.sun.com. Search for the info doc 72099 on the SunSolve web site.

`Upgradeable Solaris root devices were found, however, no suitable partitions to hold the Solaris install software were found. Upgrading using the Solaris Installer is not possible. It might be possible to upgrade using the Solaris Software 1 CDROM. (x86 based systems only)`
Cause: You cannot upgrade with the Solaris Software - 1 CD because you do not have enough space.

Solution: To upgrade, you can either create a swap slice that is larger than or equal to 512 Mbytes or use another method of upgrading such as the Solaris installation program from Solaris Operating System DVD, a net installation image, or JumpStart.

`ERROR: Could not select locale (x86 based systems only)`
Cause: When you test your JumpStart profile by using the `pfinstall -D` command, the dry run test fails under the following conditions:

- The profile contains the locale keyword.
- You're testing a release that contains GRUB software. **Starting with the Solaris 10 1/06 release**, the GRUB boot loader facilitates booting different operating systems installed on your system with the GRUB menu.

With the introduction of GRUB software, the miniroot is compressed. The software can no longer find the list of locales from the compressed miniroot. The miniroot is the smallest possible Solaris root (`/`) file system and is found on the Solaris installation media.

Solution: Perform the following steps. Use the following values.

- `MEDIA_DIR` is `/cdrom/cdrom0/`
- `MINIROOT_DIR` is `$MEDIA_DIR/Solaris_10/Tools/Boot`
- `MINIROOT_ARCHIVE` is `$MEDIA_DIR/boot/x86.miniroot`
- `TEMP_FILE_NAME` is `/tmp/test`

1. Become superuser or assume an equivalent role.

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

2. Uncompress the miniroot archive.

   ```
   # /usr/bin/gzcat $MINIROOT_ARCHIVE > $TEMP_FILE_NAME
   ```

3. Create the miniroot device by using the `lofiadm` command.

   ```
   # LOFI_DEVICE=/usr/sbin/lofiadm -a $TEMP_FILE_NAME
   # echo $LOFI_DEVICE
   /dev/lofi/1
   ```

4. Mount the miniroot with the `lofi` command under the Miniroot directory.

   ```
   # /usr/sbin/mount -F ufs  $LOFI_DEVICE  $MINIROOT_DIR
   ```

5. Test the profile.

   ```
   # /usr/sbin/install.d/pfinstall -D -c $MEDIA_DIR $path-to-jumpstart_profile
   ```

6. After the testing is completed, unmount the lofi device.

   ```
   # umount  $LOFI_DEVICE
   ```

7. Delete the `lofi` device.

   ```
   # lofiadm -d $TEMP_FILE_NAME
   ```

Upgrading, General Problems

`The upgrade option is not presented even though there is a version of Solaris software that's upgradable on the system.`

Cause: *Reason 1:* The `/var/sadm` directory is a symlink or it is mounted from another file system.

Solution: *Reason 1:* Move the `/var/sadm` directory into the root (`/`) or `/var` file system.

Cause: *Reason 2:* The `/var/sadm/softinfo/INST_RELEASE` file is missing.

Solution: *Reason 2:* Create a new `INST_RELEASE` file by using the following template:

```
OS=Solaris
VERSION=x
REV=0
```

x Is the version of Solaris software on the system

Cause: *Reason 3:* SUNWusr is missing from `/var/sadm/softinfo`.

Solution: *Solution 3:* You need to do an initial installation. The Solaris software is not upgradable.

`Couldn't shut down or initialize the md driver`

Solution: Follow these instructions:

- If the file system is not a RAID-1 volume, comment out in the `vsftab` file.
- If the file system is a RAID-1 volume, break the mirror and reinstall. For information about unmirroring, see "Removing RAID-1 Volumes (Unmirroring)" in *Solaris Volume Manager Administration Guide*.

`The upgrade fails because the Solaris installation program cannot mount a file system.`

Cause: During an upgrade, the script attempts to mount all the file systems that are listed in the system's `/etc/vfstab` file on the root (`/`) file system that is being upgraded. If the installation script cannot mount a file system, it fails and exits.

Solution: Ensure that all file systems in the system's `/etc/vfstab` file can be mounted. Comment out any file systems in the `/etc/vfstab` file that cannot be mounted or that might cause the problem so that the Solaris installation program does not try to mount them during the upgrade. Any system-based file systems that contain software to be upgraded (for example, `/usr`) cannot be commented out.

`The upgrade fails`

Description: The system does not have enough space for the upgrade.

Cause: Check "Upgrading With Disk Space Reallocation" in *Solaris 10 5/09 Installation Guide: Planning for Installation and Upgrade* for the space problem and see if you can fix it without using auto-layout to reallocate space.

`Problems upgrading RAID–1 volume root (/) file systems`

Solution: If you have problems upgrading when using Solaris Volume Manager RAID-1 volumes that are the root (`/`) file system, see Chapter 25, "Troubleshooting Solaris Volume Manager (Tasks)," in *Solaris Volume Manager Administration Guide*.

▼ To Continue Upgrading After a Failed Upgrade

The upgrade fails and the system cannot be soft-booted. The failure is for reasons beyond your control, such as a power failure or a network connection failure.

1 **Reboot the system from the Solaris Operating System DVD, the Solaris Software - 1 CD, or from the network.**

2 **Choose the upgrade option for installation.**

The Solaris installation program determines if the system has been partially upgraded and continues the upgrade.

x86: Problems With Solaris Live Upgrade When You Use GRUB

The following errors can occur when you use Solaris Live Upgrade and the GRUB boot loader on an x86 based system.

`ERROR: The media product tools installation directory` *path-to-installation-directory* `does not exist.`

`ERROR: The media` *dirctory* `does not contain an operating system upgrade image.`

Description: The error messages are seen when using the `luupgrade` command to upgrade a new boot environment.

Cause: An older version of Solaris Live Upgrade is being used. The Solaris Live Upgrade packages you have installed on your system are incompatible with the media and the release on that media.

Solution: Always use the Solaris Live Upgrade packages from the release you are upgrading to.

Example: In the following example, the error message indicates that the Solaris Live Upgrade packages on the system are not the same version as on the media.

```
# luupgrade -u -n s10u1 -s /mnt
    Validating the contents of the media </mnt>.
    The media is a standard Solaris media.
    ERROR: The media product tools installation directory
</mnt/Solaris_10/Tools/Boot/usr/sbin/install.d/install_config> does
not exist.
    ERROR: The media </mnt> does not contain an operating system upgrade
image.
```

`ERROR: Cannot find or is not executable: </sbin/biosdev>.`

`ERROR: One or more patches required by Solaris Live Upgrade has not been installed.`

Cause: One or more patches required by Solaris Live Upgrade are not installed on your system. Beware that this error message does not catch all missing patches.

Solution: Before using Solaris Live Upgrade, always install all the required patches. Ensure that you have the most recently updated patch list by consulting http://sunsolve.sun.com. Search for the info doc 72099 on the SunSolve web site.

`ERROR: Device mapping command </sbin/biosdev> failed. Please reboot and try again.`

Cause: *Reason 1*: Solaris Live Upgrade is unable to map devices because of previous administrative tasks.

Solution: *Reason 1*: Reboot the system and try Solaris Live Upgrade again

Cause: *Reason 2*: If you reboot your system and get the same error message, you have two or more identical disks. The device mapping command is unable to distinguish between them.

Solution: *Reason 2*: Create a new dummy `fdisk` partition on one of the disks. See the `fdisk(1M)` man page. Then reboot the system.

`Cannot delete the boot environment that contains the GRUB menu`

Cause: Solaris Live Upgrade imposes the restriction that a boot environment cannot be deleted if the boot environment contains the GRUB menu.

Solution: Use `lumake(1M)` or `luupgrade(1M)` commands to reuse that boot environment.

`The file system containing the GRUB menu was accidentally remade. However, the disk has the same slices as before. For example, the disk was not re-sliced.`

Cause: The file system that contains the GRUB menu is critical to keeping the system bootable. Solaris Live Upgrade commands do not destroy the GRUB menu. But, if you accidentally remake or otherwise destroy the file system containing the GRUB menu with a command other than a Solaris Live Upgrade command, the recovery software attempts to reinstall the GRUB menu. The recovery software puts the GRUB menu back in the same file system at the next reboot. For example, you might have used the `newfs` or `mkfs` commands on the file system and accidentally destroyed the GRUB menu. To restore the GRUB menu correctly, the slice must adhere to the following conditions:

- Contain a mountable file system
- Remain a part of the same Solaris Live Upgrade boot environment where the slice resided previously

Before rebooting the system, make any necessary corrective actions on the slice.

Solution: Reboot the system. A backup copy of the GRUB menu is automatically installed.

`The GRUB menu's menu.lst file was accidentally deleted.`

Solution: Reboot the system. A backup copy of the GRUB menu is automatically installed.

System Panics When Upgrading With Solaris Live Upgrade Running Veritas VxVm

When you use Solaris Live Upgrade while upgrading and running Veritas VxVM, the system panics on reboot unless you upgrade by using the following procedure. The problem occurs if packages do not conform to Solaris advanced packaging guidelines.

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

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

2. **Create an inactive boot environment. See "Creating a New Boot Environment" in *Solaris 10 5/09 Installation Guide: Solaris Live Upgrade and Upgrade Planning*.**

3. **Before upgrading the inactive boot environment, you must disable the existing Veritas software on the inactive boot environment.**

   a. **Mount the inactive boot environment.**

      ```
      # lumount inactive_boot_environment_name mount_point
      ```

      For example:

      ```
      # lumount solaris8 /mnt
      ```

   b. **Change to the directory that contains the `vfstab`, for example:**

      ```
      # cd /mnt/etc
      ```

   c. **Make a copy of the inactive boot environment's `vfstab` file, for example:**

      ```
      # cp vfstab vfstab.501
      ```

   d. **In the copied `vfstab`, comment out all Veritas file system entries, for example:**

      ```
      #  sed  '/vx\/dsk/s/^/#/g'  < vfstab >  vfstab.novxfs
      ```

      The first character of each line is changed to #, which makes the line a comment line. Note that this comment line is different than the system file-comment lines.

e. **Copy the changed `vfstab` file, for example:**

```
# cp vfstab.novxfs vfstab
```

f. **Change directories to the inactive boot environment's system file, for example:**

```
# cd /mnt/etc
```

g. **Make a copy of the inactive boot environment's system file, for example:**

```
# cp system system.501
```

h. **Comment out all "forceload:" entries that include `drv/vx`.**

```
# sed '/forceload:   drv\/vx/s/^/*/' <system> system.novxfs
```

The first character of each line is changed to *, which makes the line a command line. Note that this comment line is different than the `vfstab` file comment lines.

i. **Create the Veritas `install-db` file, for example:**

```
# touch vx/reconfig.d/state.d/install-db
```

j. **Unmount the inactive boot environment.**

```
# luumount inactive_boot_environment_name
```

4 **Upgrade the inactive boot environment. See Chapter 5, "Upgrading With Solaris Live Upgrade (Tasks)," in *Solaris 10 5/09 Installation Guide: Solaris Live Upgrade and Upgrade Planning*.**

5 **Activate the inactive boot environment. See "Activating a Boot Environment" in *Solaris 10 5/09 Installation Guide: Solaris Live Upgrade and Upgrade Planning*.**

6 **Shut down the system.**

```
# init 0
```

7 **Boot the inactive boot environment in single-user mode:**

```
OK boot -s
```

Several messages and error messages that contain "vxvm" or "VXVM" are displayed that can be ignored. The inactive boot environment becomes active.

8 **Upgrade Veritas.**

a. **Remove the Veritas VRTSvmsa package from the system, for example:**

```
# pkgrm VRTSvmsa
```

b. **Change directories to the Veritas packages.**

```
# cd /location_of_Veritas_software
```

c. **Add the latest Veritas packages to the system:**

```
# pkgadd -d 'pwd' VRTSvxvm VRTSvmsa VRTSvmdoc VRTSvmman VRTSvmdev
```

9 **Restore the original `vfstab` and system files:**

```
# cp /etc/vfstab.original /etc/vfstab
# cp /etc/system.original /etc/system
```

10 **Reboot the system.**

```
# init 6
```

x86: Service Partition Not Created by Default on Systems With No Existing Service Partition

If you install the Solaris 10 5/09 OS on a system that does not currently include a service or diagnostic partition, the installation program might not create a service partition by default. If you want to include a service partition on the same disk as the Solaris partition, you must re-create the service partition before you install the Solaris 10 5/09 OS.

If you installed the Solaris 8 2/02 OS on a system with a service partition, the installation program might not have preserved the service partition. If you did not manually edit the `fdisk` boot partition layout to preserve the service partition, the installation program deleted the service partition during the installation.

Note – If you did not specifically preserve the service partition when you installed the Solaris 8 2/02 OS, you might not be able to re-create the service partition and upgrade to the Solaris 10 5/09 OS.

If you want to include a service partition on the disk that contains the Solaris partition, choose one of the following workarounds.

▼ To Install Software From a Network Installation Image or From the Solaris Operating System DVD

To install the software from a net installation image or from the Solaris Operating System DVD over the network, follow these steps.

1 **Delete the contents of the disk.**

2. **Before you install, create the service partition by using the diagnostics CD for your system.**
   For information about how to create the service partition, see your hardware documentation.
3. **Boot the system from the network.**
   The Customize `fdisk` Partitions screen is displayed.
4. **To load the default boot disk partition layout, click Default.**
   The installation program preserves the service partition and creates the Solaris partition.

▼ To Install From the Solaris Software - 1 CD or From a Network Installation Image

To use the Solaris installation program to install from the Solaris Software - 1 CD or from a network installation image on a boot server, follow these steps.

1. **Delete the contents of the disk.**
2. **Before you install, create the service partition by using the diagnostics CD for your system.**
   For information about how to create the service partition, see your hardware documentation.
3. **The installation program prompts you to choose a method for creating the Solaris partition.**
4. **Boot the system.**
5. **Select the `Use rest of disk for Solaris partition` option.**
   The installation program preserves the service partition and creates the Solaris partition.
6. **Complete the installation.**

◆ ◆ ◆ APPENDIX B

Additional SVR4 Packaging Requirements (Reference)

This appendix is for system administrators who install or remove packages, especially third-party packages. Following these packaging requirements enables the following:

- Avoids modifying the currently running system so you can upgrade with Solaris Live Upgrade and create and maintain non-global zones and diskless clients
- Prevents a package from being interactive to automate installations when using installation programs such as custom JumpStart

This chapter contains the following sections:

Preventing Modification of the Current OS

Following the requirements in this section keeps the currently running OS unaltered.

Using Absolute Paths

For an installation of an operating system to be successful, packages must recognize and correctly respect alternate root (/) file systems, such as a Solaris Live Upgrade inactive boot environment.

Packages can include absolute paths in their `pkgmap` file (package map). If these files exist, they are written relative to the `-R` option of the `pkgadd` command. Packages that contain both absolute and relative (relocatable) paths can be installed to an alternative root (`/`) file system as well. `$PKG_INSTALL_ROOT` is prepended to both absolute and relocatable files so all paths are resolved properly when being installed by `pkgadd`.

Using the pkgadd -R Command

Packages being installed by using the `pkgadd -R` option or being removed using the `pkgrm -R` option must not alter the currently running system. This feature is used by custom JumpStart, Solaris Live Upgrade, non-global zones and diskless client.

Any procedure scripts that are included in the packages being installed with the `pkgadd` command `-R` option or being removed by using the `pkgrm` command `-R` option must not alter the currently running system. Any installation scripts that you provide must reference any directory or file that is prefixed with the `$PKG_INSTALL_ROOT` variable. The package must write all directories and files with the `$PKG_INSTALL_ROOT` prefix. The package must not remove directories without a `$PKG_INSTALL_ROOT` prefix.

Table B–1 provides examples of script syntax.

TABLE B–1 Examples of Installation Script Syntax

Script Type	Correct Syntax	Incorrect Syntax
Bourne shell "if" statement fragments	`if [ -f ${PKG_INSTALL_ROOT}\ /etc/myproduct.conf ] ; then`	`if [ -f /etc/myproduct.conf ] ; \ then`
Removing a file	`/bin/rm -f ${PKG_INSTALL_ROOT}\ /etc/myproduct.conf`	`/bin/rm -f /etc/myproduct.conf`
Changing a file	`echo "test=no" > ${PKG_INSTALL_ROOT}\ /etc/myproduct.conf`	`echo "test=no" > \ /etc/myproduct.conf`

Differences Between $PKG_INSTALL_ROOT and $BASEDIR Overview

`$PKG_INSTALL_ROOT` is the location of the root (`/`) file system of the machine to which you are adding the package. The location is set to the `-R` argument of the `pkgadd` command. For example, if the following command is invoked, then `$PKG_INSTALL_ROOT` becomes `/a` during the installation of the package.

```
# pkgadd -R /a SUNWvxvm
```

`$BASEDIR` points to the *relocatable* base directory into which relocatable package objects are installed. Only relocatable objects are installed here. Nonrelocatable objects (those that have *absolute* paths in the `pkgmap` file) are always installed relative to the inactive boot environment, but not relative to the `$BASEDIR` in effect. If a package has no relocatable objects, then the package is said to be an absolute package (or nonrelocatable), and `$BASEDIR` is undefined and not available to package procedure scripts.

For example, suppose a package's `pkgmap` file has two entries:

```
1 f none sbin/ls 0555 root sys 3541 12322 1002918510
1 f none /sbin/ls2 0555 root sys 3541 12322 2342423332
```

The `pkginfo` file has a specification for `$BASEDIR`:

```
BASEDIR=/opt
```

If this package is installed with the following command, then `ls` is installed in `/a/opt/sbin/ls`, but `ls2` is installed as `/a/sbin/ls2`.

```
# pkgadd -R /a SUNWtest
```

Guidelines for Writing Scripts

Your package procedure scripts must be independent of the currently running OS to prevent modifying the OS. Procedure scripts define actions that occur at particular points during package installation and removal. Four procedure scripts can be created with these predefined names: `preinstall`, `postinstall`, `preremove`, and `postremove`.

TABLE B–2 Guidelines For Creating Scripts

Guidelines	Affects Solaris Live Upgrade	Affects non-global zones
Scripts must be written in Bourne shell (`/bin/sh`). Bourne shell is the interpreter that is used by the `pkgadd` command to execute the procedure scripts.	X	X
Scripts must not start or stop any processes or depend on the output of commands such as `ps` or `truss`, which are operating system dependent and report information about the currently running system.	X	X
Scripts are free to use other standard UNIX commands such as `expr`, `cp`, and `ls` and other commands that facilitate shell scripting.	X	X
Any commands that a script invokes must be available in all supported releases, since a package must run on all of those releases. Therefore, you cannot use commands that were added or removed after the Solaris 8 release.	X	
To verify that a specific command or option is supported in a Solaris 8, 9, or 10 release, see the specific version of *Solaris Reference Manual AnswerBook* on http://docs.sun.com.		

Maintaining Diskless Client Compatibility

Packages must not execute commands delivered by the package itself. This is to maintain diskless client compatibility and avoids running commands that might require shared libraries that are not installed yet.

Verifying Packages

All packages must pass `pkgchk` validation. After a package is created and before it is installed, it must be checked with the following command.

 # pkgchk -d *dir_name pkg_name*

dir_name Specifies the name of the directory where the package resides

pkg_name Specifies the name of the package

EXAMPLE B–1 Testing a Package

After a package is created, it must be tested by installing it in an alternate root (`/`) file system location by using the `-R` *dir_name* option to `pkgadd`. After the package is installed, it must be checked for correctness by using `pkgchk`, as in this example.

```
# pkgadd -d . -R /a SUNWvxvm
# pkgchk -R /a SUNWvxvm
```

No errors should be displayed.

EXAMPLE B–2 Testing a Package on `/export/SUNWvxvm`

If a package exists at `/export/SUNWvxvm`, then you would issue the following command.

```
# pkgchk -d /export SUNWvxvm
```

No errors should be displayed.

Other commands can check the package when you are creating, modifying, and deleting files. The following commands are some examples.

- For example, the `dircmp` or `fssnap` commands can be used to verify that packages behave properly.
- Also, the `ps` command can be used for testing daemon compliance by making sure daemons are not stopped or started by the package.
- The `truss`, `pkgadd -v`, and `pkgrm` commands can test runtime package installation compliance, but might not work in all situations. In the following example, the `truss` command strips out all read-only, non-`$TEMPDIR` access and shows only non-read-only access to paths that do not lie within the specified inactive boot environment.

```
# TEMPDIR=/a; export TEMPDIR
# truss -t open /usr/sbin/pkgadd -R ${TEMPDIR} SUNWvxvm \
```

```
2>&1 > /dev/null | grep -v O_RDONLY | grep -v \
'open("'${TEMPDIR}
```

Preventing User Interaction When Installing or Upgrading

Packages must be added or removed without the user being prompted for information when using the following standard Solaris utilities.

- The custom JumpStart program
- Solaris Live Upgrade
- Solaris installation program program
- Solaris Zones

To test a package to ensure that it will install with no user interaction, a new administration file can be set up with the `pkgadd` command `-a` option. The `-a` option defines an installation administration file to be used in place of the default administration file. Using the default file might result in the user being prompted for more information. You can create an administration file that indicates to `pkgadd` that it should bypass these checks and install the package without user confirmation. For details, see the man page `admin(4)` or `pkgadd(1M)`.

The following examples show how the `pkgadd` command uses the administration file.

- If no administration file is provided, `pkgadd` uses `/var/sadm/install/admin/default`. Using this file might result in user interaction.

  ```
  # pkgadd
  ```

- If a relative administration file is provided on the command line, `pkgadd` looks in `/var/sadm/install/admin` for the file name and uses it. In this example, the relative administration file is named `nocheck` and `pkgadd` looks for `/var/sadm/install/admin/nocheck`.

  ```
  # pkgadd -a nocheck
  ```

- If an absolute file is provided `pkgadd` uses it. In this example, `pkgadd` looks in `/tmp` for the `nocheck` administration file.

  ```
  # pkgadd -a /tmp/nocheck
  ```

EXAMPLE B–3 Installation Administration File

The following is an example of an installation administration file that requires very little user interaction with the `pkgadd` utility. Unless the package requires more space than is available on the system, the `pkgadd` utility uses this file and installs the package without prompting the user for more information.

EXAMPLE B–3 Installation Administration File *(Continued)*

```
mail=
instance=overwrite
partial=nocheck
runlevel=nocheck
idepend=nocheck
space=ask
setuid=nocheck
confiict=nocheck
action=nocheck
basedir=default
```

Setting Package Parameters For Zones

Packages have parameters that control how their content is distributed and made visible on a system with non-global zones installed. The `SUNW_PKG_ALLZONES`, `SUNW_PKG_HOLLOW`, and `SUNW_PKG_THISZONE` package parameters define the characteristics of packages on a system with zones installed. These parameters must be set so that packages can be administered in a system with non-global zones.

The following table lists the four valid combinations for setting package parameters. If you choose setting combinations that are not listed in the following table, those settings are invalid and result in the package failing to install.

Note – Ensure that you have set all three package parameters. You can leave all three package parameters blank. The package tools interpret a missing zone package parameter as if the setting were "false," but not setting the parameters is strongly discouraged. By setting all three package parameters, you specify the exact behavior the package tools should exhibit when installing or removing the package.

TABLE B–3 Valid Package Parameter Settings For Zones

SUNW_PKG_ALLZONES Setting	SUNW_PKG_HOLLOW Setting	SUNW_PKG_THISZONE Setting	Package Description
false	false	false	This is the default setting for packages that do not specify values for all the zone package parameters. A package with these settings can be installed in either the global zone or a non-global zone. ■ If the `pkgadd` command is run in the global zone, the package is installed in the global zone and in all non-global zones. ■ If the `pkgadd` command is run in a non-global zone, the package is installed in the non-global zone only. In both cases, the entire contents of the package is visible in all zones where the package is installed.
false	false	true	A package with these settings can be installed in either the global zone or a non-global zone. If new non-global zones are created after the installation, the package is not propagated to these new non-global zones. ■ If the `pkgadd` command is run in the global zone, the package is installed in the global zone only. ■ If the `pkgadd` command is run in a non-global zone, the package is installed in the non-global zone only. In both cases, the entire contents of the package is visible in the zone where the package is installed.
true	false	false	A package with these settings can be installed in the global zone only. When the `pkgadd` command is run, the package is installed in the global zone and in all non-global zones. The entire contents of the package is visible in all zones. **Note –** Any attempt to install the package in a non-global zone fails.

TABLE B–3 Valid Package Parameter Settings For Zones *(Continued)*

SUNW_PKG_ALLZONES Setting	SUNW_PKG_HOLLOW Setting	SUNW_PKG_THISZONE Setting	Package Description
true	true	false	A package with these settings can only be installed in the global zone, by the global administrator. When the `pkgadd` command is run, the contents of the package is fully installed in the global zone. If a package has the package parameters set to these values, the package content itself is not delivered on any non-global zone. Only the package installation information necessary to make the package appear to be installed is installed on all non-global zones. This enables the installation of other packages to be installed that depend on this package. For more information on "hollow" packages, see Chapter 24, "About Packages and Patches on a Solaris System With Zones Installed (Overview)," in *System Administration Guide: Solaris Containers-Resource Management and Solaris Zones*. For package dependency checking purposes, the package appears to be installed in all zones. ■ In the global zone, the entire contents of the package is visible. ■ In whole root non-global zones, the entire contents of the package is not visible. ■ When a non-global zone inherits a file system from the global zone, a package installed in this file system is visible in a non-global zone. All other files delivered by the package are not visible within the non-global zone. For example, a sparse root non-global zone shares certain directories with the global zone. These directories are read-only. Sparse root non-global zones share the `/platform` file system among others. Another example is packages that deliver files relevant only to booting hardware. **Note –** Any attempt to install the package in a non-global zone fails.

Description	For More Information
For more details on packages and zones	Chapter 24, "About Packages and Patches on a Solaris System With Zones Installed (Overview)," in *System Administration Guide: Solaris Containers-Resource Management and Solaris Zones*

Description	For More Information
For an overview of sparse and whole root zones	Chapter 16, "Introduction to Solaris Zones," in *System Administration Guide: Solaris Containers-Resource Management and Solaris Zones*
For information about package characteristics and parameters	`pkginfo`(4)
For information about displaying package parameter values	`pkgparam`(1)

For Background Information

The following references provide background information about packaging requirements and specific command syntax.

For more specific information about packaging requirements and definitions of terminology	Chapter 6, "Advanced Techniques for Creating Packages," in *Application Packaging Developer's Guide*
For basic information about adding and removing packages and the installation administration file	Chapter 19, "Managing Software (Overview)," in *System Administration Guide: Basic Administration*
For detailed information about specific commands that are referenced in this appendix, see these man pages	`dircmp`(1), `fssnap`(1M), `ps`(1), or `truss`(1) `pkgadd`(1M), `pkgchk`(1M), or `pkgrm`(1M)
For an overview of Solaris Live Upgrade	Chapter 2, "Solaris Live Upgrade (Overview)," in *Solaris 10 5/09 Installation Guide: Solaris Live Upgrade and Upgrade Planning*
For an overview of custom JumpStart	Chapter 2, "Custom JumpStart (Overview)"
For an overview of Solaris Zones	Chapter 16, "Introduction to Solaris Zones," in *System Administration Guide: Solaris Containers-Resource Management and Solaris Zones*

Glossary

3DES ([Triple DES] Triple-Data Encryption Standard). A symmetric-key encryption method that provides a key length of 168 bits.

AES (Advanced Encryption Standard) A symmetric 128-bit block data encryption technique. The U.S. government adopted the Rijndael variant of the algorithm as its encryption standard in October 2000. AES replaces DES encryption as the government standard.

archive A file that contains a collection of files that were copied from a master system. The file also contains identification information about the archive, such as a name and the date that you created the archive. After you install an archive on a system, the system contains the exact configuration of the master system.

An archive could be a differential archive, which is a Solaris Flash archive that contains only the differences between two system images, an unchanged master image and an updated master image. The differential archive contains files to be retained, modified, or deleted from the clone system. A differential update changes only the files specified and is restricted to systems that contain software consistent with the unchanged master image.

arrow keys One of the four directional keys on the numeric keypad.

begin script A user-defined Bourne shell script, specified within the `rules` file, that performs tasks before the Solaris software is installed on the system. You can use begin scripts only with custom JumpStart installations.

boot To load the system software into memory and start it.

boot archive **x86 only**: A boot archive is a collection of critical files that is used to boot the Solaris OS. These files are needed during system startup before the root (`/`) file system is mounted. Two boot archives are maintained on a system:

- The boot archive that is used to boot the Solaris OS on a system. This boot archive is sometimes called the primary boot archive.
- The boot archive that is used for recovery when the primary boot archive is damaged. This boot archive starts the system without mounting the root (`/`) file system. On the GRUB menu, this boot archive is called failsafe. The archive's essential purpose is to regenerate the primary boot archive, which is usually used to boot the system.

boot environment A collection of mandatory file systems (disk slices and mount points) that are critical to the operation of the Solaris OS. These disk slices might be on the same disk or distributed across multiple disks.

The active boot environment is the one that is currently booted. Exactly one active boot environment can be booted. An inactive boot environment is not currently booted, but can be in a state of waiting for activation on the next reboot.

boot loader — **x86 only**: The boot loader is the first software program that runs after you turn on a system. This program begins the booting process.

boot server — A server system that provides client systems on the same network subnet with the programs and information that they need to start. A boot server is required to install over the network if the install server is on a different subnet than the systems on which Solaris software is to be installed.

`bootlog-cgi` **program** — The CGI program that enables a web server to collect and store remote client-booting and installation console messages during a WAN boot installation.

certificate authority — (CA) A trusted third-party organization or company that issues digital certificates that are used to create digital signatures and public-private key pairs. The CA guarantees that the individual who is granted the unique certificate is who she or he claims to be.

`certstore` **file** — A file that contains a digital certificate for a specific client system. During an SSL negotiation, the client might be asked to provide the certificate file to the server. The server uses this file to verify the identity of the client.

CGI — (Common Gateway Interface) An interface by which external programs communicate with the HTTP server. Programs that are written to use CGI are called CGI programs or CGI scripts. CGI programs handle forms or parse output the server does not normally handle or parse.

checksum — The result of adding a group of data items that are used for checking the group. The data items can be either numerals or other character strings that are treated as numerals during the checksum calculation. The checksum value verifies that communication between two devices is successful.

client — In the client-server model for communications, the client is a process that remotely accesses resources of a compute server, such as compute power and large memory capacity.

clone system — A system that you install by using a Solaris Flash archive. The clone system has the same installation configuration as the master system.

cluster — A logical collection of packages (software modules). The Solaris software is divided into *software groups*, which are each composed of clusters and *packages*.

command line — A string of characters that begins with a command, often followed by arguments, including options, file names, and other expressions, and terminated by the end-of-line character.

concatenation — A RAID-0 volume. If slices are concatenated, the data is written to the first available slice until that slice is full. When that slice is full, the data is written to the next slice, serially. A concatenation provides no data redundancy unless it is contained in a mirror. See also RAID-0 volume.

Core Software Group — A software group that contains the minimum software that is required to boot and run the Solaris OS on a system. Core includes some networking software and the drivers that are required to run the Common Desktop Environment (CDE) desktop. Core does not include the CDE software.

critical file systems File systems that are required by the Solaris OS. When you use Solaris Live Upgrade, these file systems are separate mount points in the `vfstab` file of the active and inactive boot environments. Example file systems are `root` (/), `/usr`, `/var`, and `/opt`. These file systems are always copied from the source to the inactive boot environment.

custom JumpStart A type of installation in which the Solaris software is automatically installed on a system that is based on a user-defined profile. You can create customized profiles for different types of users and systems. A custom JumpStart installation is a JumpStart installation you create.

custom probes file A file, which must be located in the same JumpStart directory as the `rules` file, that is a Bourne shell script that contains two types of functions: probe and comparison. Probe functions gather the information you want or do the actual work and set a corresponding `SI_` environment variable you define. Probe functions become probe keywords. Comparison functions call a corresponding probe function, compare the output of the probe function, and return 0 if the keyword matches or 1 if the keyword doesn't match. Comparison functions become rule keywords. See also *rules file*.

dataset A generic name for the following ZFS entities: clones, file systems, snapshots, or volumes.

decryption The process of converting coded data to plain text. See also encryption.

derived profile A profile that is dynamically created by a begin script during a custom JumpStart installation.

DES (Data Encryption Standard) A symmetric-key encryption method that was developed in 1975 and standardized by ANSI in 1981 as ANSI X.3.92. DES uses a 56-bit key.

Developer Solaris Software Group A software group that contains the End User Solaris Software Group plus the libraries, include files, man pages, and programming tools for developing software.

DHCP (Dynamic Host Configuration Protocol) An application-layer protocol. Enables individual computers, or clients, on a TCP/IP network to extract an IP address and other network configuration information from a designated and centrally maintained DHCP server or servers. This facility reduces the overhead of maintaining and administering a large IP network.

differential archive A Solaris Flash archive that contains only the differences between two system images, an unchanged master image and an updated master image. The differential archive contains files to be retained, modified, or deleted from the clone system. A differential update changes only the files that are specified and is restricted to systems that contain software consistent with the unchanged master image.

digital certificate A nontransferable, nonforgeable, digital file issued from a third party that both communicating parties already trust.

disc An optical disc, as opposed to a magnetic disk, which recognizes the common spelling that is used in the compact disc (CD) market. For example, a CD-ROM or DVD-ROM is an optical disc.

disk A round platter, or set of platters, of a magnetized medium that is organized into concentric tracks and sectors for storing data such as files. See also disc.

disk configuration file A file that represents a structure of a disk (for example, bytes/sector, flags, slices). Disk configuration files enable you to use the `pfinstall` command from a single system to test profiles on different–size disks.

diskless client A client on a network that relies on a server for all of its disk storage.

document root directory The root of a hierarchy on a web server machine that contains the files, images, and data you want to present to users who are accessing the web server.

domain A part of the Internet naming hierarchy. A domain represents a group of systems on a local network that share administrative files.

domain name The name that is assigned to a group of systems on a local network that share administrative files. The domain name is required for the Network Information Service (NIS) database to work properly. A domain name consists of a sequence of component names that are separated by periods (for example: `tundra.mpk.ca.us`). As you read a domain name from left to right, the component names identify more general (and usually remote) areas of administrative authority.

encryption The process of protecting information from unauthorized use by making the information unintelligible. Encryption is based on a code, called a key, which is used to decrypt the information. See also decryption.

End User Solaris Software Group A software group that contains the Core Software Group plus the recommended software for an end user, including the Common Desktop Environment (CDE) and DeskSet software.

Entire Solaris Software Group A software group that contains the entire Solaris release.

Entire Solaris Software Group Plus OEM Support A software group that contains the entire Solaris release plus additional hardware support for OEMs. This software group is recommended when installing Solaris software on SPARC based servers.

`/etc` **directory** A directory that contains critical system configuration files and maintenance commands.

`/etc/netboot` **directory** The directory on a WAN boot server that contains the client configuration information and security data that are required for a WAN boot installation.

`/export` **file system** A file system on an OS server that is shared with other systems on a network. For example, the `/export` file system can contain the root (`/`) file system and swap space for diskless clients and the home directories for users on the network. Diskless clients rely on the `/export` file system on an OS server to boot and run.

failsafe boot archive **x86 only**: A boot archive that is used for recovery when the primary boot archive is damaged. This boot archive starts the system without mounting the root (`/`) file system. This boot archive is called failsafe on the GRUB menu. The archive's essential purpose is to regenerate the primary boot archive, which is usually used to boot the system. See *boot archive*.

fallback A reversion to the environment that ran previously. Use fallback when you are activating an environment and the boot environment that is designated for booting fails or shows some undesirable behavior.

`fdisk` **partition** A logical partition of a disk drive that is dedicated to a particular operating system on x86 based systems. To install the Solaris software, you must set up at least one Solaris `fdisk` partition on an x86 based system. x86 based systems allow up to four different `fdisk` partitions on a disk. These partitions can be used to hold individual operating systems. Each operating system must be located on a unique `fdisk` partition. A system can only have one Solaris `fdisk` partition per disk.

file server A server that provides the software and file storage for systems on a network.

file system In the SunOS™ operating system, a tree-structured network of files and directories that you can access.

finish script A user-defined Bourne shell script, specified within the `rules` file, that performs tasks after the Solaris software is installed on the system but before the system reboots. You use finish scripts with custom JumpStart installations.

format To put data into a structure or divide a disk into sectors for receiving data.

function key One of the 10 or more keyboard keys that are labeled F1, F2, F3, and so on that are mapped to particular tasks.

global zone In Solaris Zones, the global zone is both the default zone for the system and the zone used for system-wide administrative control. The global zone is the only zone from which a non-global zone can be configured, installed, managed, or uninstalled. Administration of the system infrastructure, such as physical devices, routing, or dynamic reconfiguration (DR), is only possible in the global zone. Appropriately privileged processes running in the global zone can access objects associated with other zones. See also *Solaris Zones* and *non-global zone*.

GRUB **x86 only**: GNU GRand Unified Bootloader (GRUB) is an open source boot loader with a simple menu interface. The menu displays a list of operating systems that are installed on a system. GRUB enables you to easily boot these various operating systems, such as the Solaris OS, Linux, or Microsoft Windows.

GRUB edit menu **x86 only**: A boot menu that is a submenu of the GRUB main menu. GRUB commands are displayed on this menu. These commands can be edited to change boot behavior.

GRUB main menu **x86 only**: A boot menu that lists the operating systems that are installed on a system. From this menu, you can easily boot an operating system without modifying the BIOS or `fdisk` partition settings.

hard link A directory entry that references a file on disk. More than one such directory entry can reference the same physical file.

hash A number that is produced by taking some input and generating a number that is significantly shorter than the input. The same output value is always generated for identical inputs. Hash functions can be used in table search algorithms, in error detection, and in tamper detection. When used for tamper detection, hash functions are chosen such that it is difficult to find two inputs that yield the same hash result. MD5 and SHA-1 are examples of one-way hash functions. For example, a message digest takes a variable-length input such as a disk file and reduces it to a small value.

hashing The process of changing a string of characters into a value or key that represents the original string.

HMAC Keyed hashing method for message authentication. HMAC is used with an iterative cryptographic hash function, such as MD5 or SHA-1, in combination with a secret shared key. The cryptographic strength of HMAC depends on the properties of the underlying hash function.

host name The name by which a system is known to other systems on a network. This name must be unique among all the systems within a particular domain (usually, this means within any single organization). A host name can be any combination of letters, numbers, and minus signs (-), but it cannot begin or end with a minus sign.

HTTP (Hypertext Transfer Protocol) (n.) The Internet protocol that fetches hypertext objects from remote hosts. This protocol is based on TCP/IP.

HTTPS A secure version of HTTP, implemented by using the Secure Sockets Layer (SSL).

initial installation An installation that overwrites the currently running software or initializes a blank disk.

An initial installation of the Solaris OS overwrites the system's disk or disks with the new version of the Solaris OS. If your system is not running the Solaris OS, you must perform an initial installation. If your system is running an upgradable version of the Solaris OS, an initial installation overwrites the disk and does not preserve the OS or local modifications.

install server A server that provides the Solaris DVD or CD images from which other systems on a network can install Solaris (also called a *media server*). You can create an install server by copying the Solaris DVD or CD images to the server's hard disk.

IPv6 IPv6 is a version (version 6) of Internet Protocol (IP) that is designed to be an evolutionary step from the current version, IPv4 (version 4). Deploying IPv6, by using defined transition mechanisms, does not disrupt current operations. In addition, IPv6 provides a platform for new Internet functionality.

job A user-defined task to be completed by a computer system.

JumpStart directory When you use a profile diskette for custom JumpStart installations, the JumpStart directory is the root directory on the diskette that contains all the essential custom JumpStart files. When you use a profile server for custom JumpStart installations, the JumpStart directory is a directory on the server that contains all the essential custom JumpStart files.

JumpStart installation A type of installation in which the Solaris software is automatically installed on a system by using the factory-installed JumpStart software.

Kerberos A network authentication protocol that uses strong, secret-key cryptography to enable a client and server to identify themselves to each other over an insecure network connection.

key The code for encrypting or decrypting data. See also encryption.

`keystore` **file** A file that contains keys shared by a client and server. During a WAN boot installation, the client system uses the keys to verify the integrity of, or decrypt the data and files transmitted from, the server.

LAN (local area network) A group of computer systems in close proximity that can communicate by way of some connecting hardware and software.

LDAP (Lightweight Directory Access Protocol) A standard, extensible directory access protocol that is used by LDAP naming service clients and servers to communicate with each other.

locale A geographic or political region or community that shares the same language, customs, or cultural conventions (English for the U.S. is `en_US`, and English for the U.K. is `en_UK`).

logical device A group of physical slices on one or more disks that appear to the system as a single device. A logical device is called a volume in Solaris Volume Manager. A volume is functionally identical to a physical disk for the purposes of an application or file system.

manifest section A section of a Solaris Flash archive that is used to validate a clone system. The manifest section lists the files on a system to be retained, added to, or deleted from the clone system. This section is informational only. The section lists the files in an internal format and cannot be used for scripting.

master system A system that you use to create a Solaris Flash archive. The system configuration is saved in the archive.

MD5 (Message Digest 5) An iterative cryptographic hash function that is used for message authentication, including digital signatures. The function was developed in 1991 by Rivest.

media server See *install server*.

`menu.lst` **file** **x86 only**: A file that lists all the operating systems that are installed on a system. The contents of this file dictate the list of operating systems that is displayed on the GRUB menu. From the GRUB menu, you can easily boot an operating system without modifying the BIOS or `fdisk` partition settings.

metadevice See *volume*.

miniroot A minimal, bootable root (`/`) file system that is included in Solaris installation media. A miniroot consists of the Solaris software that is required to install and upgrade systems. On x86 based systems, the miniroot is copied to the system to be used as the failsafe boot archive. See *failsafe boot archive*.

mirror See *RAID-1 volume*.

mount The process of accessing a directory from a disk that is attached to a machine that is making the mount request or a remote disk on a network. To mount a file system, you need a mount point on the local system and the name of the file system to be mounted (for example, `/usr`).

mount point A workstation directory to which you mount a file system that exists on a remote machine.

name server A server that provides a naming service to systems on a network.

naming service A distributed network database that contains key system information about all the systems on a network so that the systems can communicate with each other. With a naming service, the system information can be maintained, managed, and accessed on a network-wide basis. Without a naming service, each system has to maintain its own copy of the system information in the local `/etc` files. Sun supports the following naming services: LDAP, NIS, and NIS+.

network installation A way to install software over the network from a system with a CD-ROM or DVD-ROM drive to a system without a CD-ROM or DVD-ROM drive. Network installations require a *name server* and an *install server*.

networked systems A group of systems (called hosts) that are connected through hardware and software so that they can communicate and share information. Referred to as a local area network (LAN). One or more servers are usually needed when systems are networked.

NIS The SunOS 4.0 (minimum) Network Information Service. A distributed network database that contains key information about the systems and the users on the network. The NIS database is stored on the master server and all the slave servers.

NIS+ The SunOS 5.0 (minimum) Network Information Service. NIS+ replaces NIS, the SunOS 4.0 (minimum) Network Information Service.

non-global zone A virtualized operating system environment created within a single instance of the Solaris Operating System. One or more applications can run in a non-global zone without interacting with the rest of the system. Non-global zones are also called zones. See also *Solaris Zones* and *global zone*.

nonnetworked systems Systems that are not connected to a network or do not rely on other systems.

/opt file system A file system that contains the mount points for third-party and unbundled software.

OS server A system that provides services to systems on a network. To serve diskless clients, an OS server must have disk space set aside for each diskless client's root (/) file system and swap space (`/export/root`, `/export/swap`).

package A collection of software that is grouped into a single entity for modular installation. The Solaris software is divided into *software groups*, which are each composed of *clusters* and packages.

panel A container for organizing the contents of a window, a dialog box, or applet. The panel might collect and confirm user input. Panels might be used by wizards and follow an ordered sequence to fulfill a designated task.

patch analyzer A script that you can run manually or as part of the Solaris installation program. The patch analyzer performs an analysis on your system to determine which (if any) patches will be removed by upgrading to a Solaris update.

platform group A vendor-defined grouping of hardware platforms for the purpose of distributing specific software. Examples of valid platform groups are i86pc and sun4u.

platform name The output of the `uname -i` command. For example, the platform name for the Ultra 60 is SUNW,Ultra-60.

pool A logical group of devices describing the layout and physical characteristics of the available ZFS storage. Space for datasets is allocated from a pool.

Power Management Software that automatically saves the state of a system and turns it off after it is idle for 30 minutes. When you install the Solaris software on a system that complies with Version 2 of the U.S. Environmental Protection Agency's Energy Star guidelines, the Power Management software is installed by default. A `sun4u` SPARC based system is an example of a system that has Power Management installed by default. After a subsequent reboot, you are prompted to enable or disable the Power Management software.

Energy Star guidelines require that systems or monitors automatically enter a “sleep state” (consume 30 watts or less) after the system or monitor becomes inactive.

primary boot archive A boot archive that is used to boot the Solaris OS on a system. This boot archive is sometimes called the primary boot archive. See *boot archive*.

private key The decryption key used in public-key encryption.

probe keyword A syntactical element that extracts attribute information about a system when using the custom JumpStart method to install. A probe keyword does not require you to set up a matching condition and run a profile as required for a rule. See also *rule*.

profile A text file that defines how to install the Solaris software when using the custom JumpStart method. For example, a profile defines which software group to install. Every rule specifies a profile that defines how a system is to be installed when the rule is matched. You usually create a different profile for every rule. However, the same profile can be used in more than one rule. See also *rules file*.

profile diskette A diskette that contains all the essential custom JumpStart files in its root directory (JumpStart directory).

profile server A server that contains all the essential custom JumpStart files in a JumpStart directory.

public key The encryption key used in public-key encryption.

public-key cryptography A cryptographic system that uses two keys: a public key known to everyone, and a private key known only to the recipient of the message.

RAID-0 volume A class of volume that can be a stripe or a concatenation. These components are also called submirrors. A stripe or concatenation is the basic building block for mirrors.

RAID-1 volume A class of volume that replicates data by maintaining multiple copies. A RAID-1 volume is composed of one or more RAID-0 volumes called *submirrors*. A RAID-1 volume is sometimes called a *mirror*.

RAID-Z storage pool A virtual device that stores data and parity on multiple disks that can be used as a ZFS storage pool. RAID-Z is similar to RAID-5.

Reduced Network Support Software Group A software group that contains the minimum code that is required to boot and run a Solaris system with limited network service support. The Reduced Networking Software Group provides a multiuser text-based console and system administration utilities. This software group also enables the system to recognize network interfaces, but does not activate network services.

root The top level of a hierarchy of items. Root is the one item from which all other items are descended. See *root directory* or *root* (`/`) *file system*.

root (`/`) file system The top-level file system from which all other file systems stem. The root (`/`) file system is the base on which all other file systems are mounted, and is never unmounted. The root (`/`) file system contains the directories and files critical for system operation, such as the kernel, device drivers, and the programs that are used to start (boot) a system.

root directory The top-level directory from which all other directories stem.

rule A series of values that assigns one or more system attributes to a profile. A rule is used in a custom JumpStart installation.

`rules` **file** A text file that contains a rule for each group of systems or single systems that you want to install automatically. Each rule distinguishes a group of systems, based on one or more system attributes. The `rules` file links each group to a profile, which is a text file that defines how the Solaris software is to be installed on each system in the group. A `rules` file is used in a custom JumpStart installation. See also *profile*.

`rules.ok` **file** A generated version of the `rules` file. The `rules.ok` file is required by the custom JumpStart installation software to match a system to a profile. You *must* use the `check` script to create the `rules.ok` file.

Secure Sockets Layer (SSL) A software library establishing a secure connection between two parties (client and server) used to implement HTTPS, the secure version of HTTP.

server A network device that manages resources and supplies services to a client.

SHA1 (Secure Hashing Algorithm) The algorithm that operates on any input length less than 2^{64} to produce a message digest.

shareable file systems File systems that are user-defined files such as `/export/home` and `/swap`. These file systems are shared between the active and inactive boot environment when you use Solaris Live Upgrade. Shareable file systems contain the same mount point in the `vfstab` file in both the active and inactive boot environments. Updating shared files in the active boot environment also updates data in the inactive boot environment. Shareable file systems are shared by default, but you can specify a destination slice, and then the file systems are copied.

slice The unit into which the disk space is divided by the software.

snapshot A read-only image of a ZFS file system or volume at a given point in time.

software group A logical grouping of the Solaris software (clusters and packages). During a Solaris installation, you can install one of the following software groups: Core, End User Solaris Software, Developer Solaris Software, or Entire Solaris Software, and for SPARC systems only, Entire Solaris Software Group Plus OEM Support.

Solaris DVD or CD images The Solaris software that is installed on a system, which you can access on the Solaris DVDs or CDs or an install server's hard disk to which you have copied the Solaris DVD or CD images.

Solaris Flash A Solaris installation feature that enables you to create an archive of the files on a system, called the *master system*. You can then use the archive to install other systems, making the other systems identical in their configuration to the master system. See also *archive*.

Solaris installation program A graphical user interface (GUI) or command-line interface (CLI) installation program that uses wizard panels to guide you step-by-step through installing the Solaris software and third-party software.

Solaris Live Upgrade An upgrade method that enables a duplicate boot environment to be upgraded while the active boot environment is still running, thus eliminating downtime of the production environment.

Solaris Zones A software partitioning technology used to virtualize operating system services and provide an isolated and secure environment for running applications. When you create a non-global zone, you produce an application execution environment in which processes are isolated from all other zones. This isolation prevents processes that are running in a zone from monitoring or affecting processes that are running in any other zones. See also *global zone* and *non-global zone*.

standalone A computer that does not require support from any other machine.

state database A database that stores information about the state of your Solaris Volume Manager configuration. The state database is a collection of multiple, replicated database copies. Each copy is referred to as a *state database replica*. The state database tracks the location and status of all known state database replicas.

state database replica A copy of a state database. The replica ensures that the data in the database is valid.

submirror See *RAID-0 volume*.

subnet A working scheme that divides a single logical network into smaller physical networks to simplify routing.

subnet mask A bit mask that is used to select bits from an Internet address for subnet addressing. The mask is 32 bits long and selects the network portion of the Internet address and 1 or more bits of the local portion.

superuser A special user who has privileges to perform all administrative tasks on the system. The superuser has the ability to read and write to any file, run all programs, and send kill signals to any process.

swap space A slice or file that temporarily holds the contents of a memory area till it can be reloaded in memory. Also called the `/swap` or `swap` volume.

`sysidcfg` **file** A file in which you specify a set of special system configuration keywords that preconfigure a system.

system configuration file (`system.conf`) A text file in which you specify the locations of the `sysidcfg` file and the custom JumpStart files you want to use in a WAN boot installation.

time zone Any of the 24 longitudinal divisions of the earth's surface for which a standard time is kept.

`truststore` **file** A file that contains one or more digital certificates. During a WAN boot installation, the client system verifies the identity of the server that is trying to perform the installation by consulting the data in the `truststore` file.

unmount The process of removing access to a directory on a disk that is attached to a machine or to a remote disk on a network.

update An installation, or to perform an installation, on a system that changes software that is of the same type. Unlike an upgrade, an update might downgrade the system. Unlike an initial installation, software of the same type that is being installed must be present before an update can occur.

upgrade An installation that merges files with existing files and preserves modifications where possible.

An upgrade of the Solaris OS merges the new version of the Solaris OS with the existing files on the system's disk or disks. An upgrade saves as many modifications as possible that you have made to the previous version of the Solaris OS.

upgrade option An option that is presented by the Solaris installation program . The upgrade procedure merges the new version of Solaris with existing files on your disk or disks. An upgrade also saves as many local modifications as possible since the last time Solaris was installed.

URL (Uniform Resource Locator) The addressing system used by the server and the client to request documents. A URL is often called a location. The format of a URL is *protocol://machine:port/document*.

A sample URL is `http://www.example.com/index.html`.

`/usr` **file system** A file system on a standalone system or server that contains many of the standard UNIX programs. Sharing the large `/usr` file system with a server rather than maintaining a local copy minimizes the overall disk space that is required to install and run the Solaris software on a system.

utility A standard program, usually furnished at no charge with the purchase of a computer, that does the computer's housekeeping.

`/var` **file system** A file system or directory (on standalone systems) that contains system files that are likely to change or grow over the life of the system. These files include system logs, `vi` files, mail files, and UUCP files.

virtual device A logical device in a ZFS pool, which can be a physical device, a file, or a collection of devices.

volume A group of physical slices or other volumes that appear to the system as a single logical device. A volume is functionally identical to a physical disk for the purposes of an application or file system.

In some command-line utilities, a volume is called a metadevice. Volume is also called *pseudo device* or *virtual device* in standard UNIX terms.

Volume Manager A program that provides a mechanism to administer and obtain access to the data on DVD-ROMs, CD-ROMs, and diskettes.

WAN (wide area network) A network that connects multiple local area networks (LANs) or systems at different geographical sites by using telephone, fiber-optic, or satellite links.

WAN boot installation A type of installation that enables you to boot and install software over a wide area network (WAN) by using HTTP or HTTPS. The WAN boot installation method enables you to transmit an encrypted Solaris Flash archive over a public network and perform a custom JumpStart installation on a remote client.

WAN boot miniroot A miniroot that has been modified to perform a WAN boot installation. The WAN boot miniroot contains a subset of the software in the Solaris miniroot. See also miniroot.

WAN boot server A web server that provides the configuration and security files that are used during a WAN boot installation.

`wanboot-cgi` **program** The CGI program that retrieves and transmits the data and files that are used in a WAN boot installation.

`wanboot.conf` **file** A text file in which you specify the configuration information and security settings that are required to perform a WAN boot installation.

`wanboot` **program** The second-level boot program that loads the WAN boot miniroot, client configuration files, and installation files that are required to perform a WAN boot installation. For WAN boot installations, the `wanboot` binary performs tasks similar to the `ufsboot` or `inetboot` second-level boot programs.

ZFS A file system using storage pools to manage physical storage.

zone See *non-global zone*

Index

C

M

N

O

P

R

S

T

U

V

W

Z