



Solaris 10 Installation Guide: Custom JumpStart and Advanced Installations

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

Part No: 817-5506-10
January 2005

Copyright 2005 Sun Microsystems, Inc. 4150 Network Circle, Santa Clara, CA 95054 U.S.A. All rights reserved.

This product or document is protected by copyright and distributed under licenses restricting its use, copying, distribution, and decompilation. No part of this product or document may be reproduced in any form by any means without prior written authorization of Sun and its licensors, if any. Third-party software, including font technology, is copyrighted and licensed from Sun suppliers.

Parts of the product may be derived from Berkeley BSD systems, licensed from the University of California. UNIX is a registered trademark in the U.S. and other countries, exclusively licensed through X/Open Company, Ltd.

Sun, Sun Microsystems, the Sun logo, docs.sun.com, AnswerBook, AnswerBook2, JumpStart, Solaris Flash, Sun Java, and Solaris are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. in the U.S. and other countries. Products bearing SPARC trademarks are based upon an architecture developed by Sun Microsystems, Inc.

The OPEN LOOK and Sun™ Graphical User Interface was developed by Sun Microsystems, Inc. for its users and licensees. Sun acknowledges the pioneering efforts of Xerox in researching and developing the concept of visual or graphical user interfaces for the computer industry. Sun holds a non-exclusive license from Xerox to the Xerox Graphical User Interface, which license also covers Sun's licensees who implement OPEN LOOK GUIs and otherwise comply with Sun's written license agreements.

U.S. Government Rights – Commercial software. Government users are subject to the Sun Microsystems, Inc. standard license agreement and applicable provisions of the FAR and its supplements.

DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

Copyright 2005 Sun Microsystems, Inc. 4150 Network Circle, Santa Clara, CA 95054 U.S.A. Tous droits réservés.

Ce produit ou document est protégé par un copyright et distribué avec des licences qui en restreignent l'utilisation, la copie, la distribution, et la décompilation. Aucune partie de ce produit ou document ne peut être reproduite sous aucune forme, par quelque moyen que ce soit, sans l'autorisation préalable et écrite de Sun et de ses bailleurs de licence, s'il y en a. Le logiciel détenu par des tiers, et qui comprend la technologie relative aux polices de caractères, est protégé par un copyright et licencié par des fournisseurs de Sun.

Des parties de ce produit pourront être dérivées du système Berkeley BSD licenciés par l'Université de Californie. UNIX est une marque déposée aux Etats-Unis et dans d'autres pays et licenciée exclusivement par X/Open Company, Ltd.

Sun, Sun Microsystems, le logo Sun, docs.sun.com, AnswerBook, AnswerBook2, JumpStart, Solaris Flash, Sun Java, et Solaris sont des marques de fabrique ou des marques déposées, de Sun Microsystems, Inc. aux Etats-Unis et dans d'autres pays. Toutes les marques SPARC sont utilisées sous licence et sont des marques de fabrique ou des marques déposées de SPARC International, Inc. aux Etats-Unis et dans d'autres pays. Les produits portant les marques SPARC sont basés sur une architecture développée par Sun Microsystems, Inc.

L'interface d'utilisation graphique OPEN LOOK et Sun™ a été développée par Sun Microsystems, Inc. pour ses utilisateurs et licenciés. Sun reconnaît les efforts de pionniers de Xerox pour la recherche et le développement du concept des interfaces d'utilisation visuelle ou graphique pour l'industrie de l'informatique. Sun détient une licence non exclusive de Xerox sur l'interface d'utilisation graphique Xerox, cette licence couvrant également les licenciés de Sun qui mettent en place l'interface d'utilisation graphique OPEN LOOK et qui en outre se conforment aux licences écrites de Sun.

CETTE PUBLICATION EST FOURNIE "EN L'ETAT" ET AUCUNE GARANTIE, EXPRESSE OU IMPLICITE, N'EST ACCORDEE, Y COMPRIS DES GARANTIES CONCERNANT LA VALEUR MARCHANDE, L'APTITUDE DE LA PUBLICATION A REpondre A UNE UTILISATION PARTICULIERE, OU LE FAIT QU'ELLE NE SOIT PAS CONTREFAISANTE DE PRODUIT DE TIERS. CE DENI DE GARANTIE NE S'APPLIQUERAIT PAS, DANS LA MESURE OU IL SERAIT TENU JURIDIQUEMENT NUL ET NON AVENU.



041129@10536



Contents

Preface	9
Part I Overall Planning of Any Solaris Installation or Upgrade	13
1 Solaris Installation and Upgrade (Roadmap)	15
Task Map: Installing or Upgrading the Solaris Software	15
Installing From the Network or From DVD or CDs?	18
Initial Installation, or Upgrade?	19
Initial Installation	19
Upgrade	19
Choosing a Solaris Installation Method	20
Sun Java System Application Server Platform Edition 8	21
2 Solaris Installation and Upgrade (Planning)	23
System Requirements and Recommendations	23
Allocating Disk and Swap Space	25
General Disk Space Planning and Recommendations	25
Disk Space Recommendations for Software Groups	27
Upgrade	29
Upgrade Limitations	29
Upgrade Programs	29
Installing a Solaris Flash Archive Instead of Upgrading	30
Upgrading With Disk Space Reallocation	31
Using the Solaris Installation Program to Upgrade From DVD or CD Media	32
Backing Up Systems Before Upgrading	32

How to Find the Version of the Solaris OS That Your System Is Running	32
Locale Values	32
Platform Names and Groups	33
Planning to Install and Configure Zones	33
Disk Space Requirements for Non-Global Zones	34
Restricting Non-Global Zone Size	35
SPARC: 64-bit Packaging Changes	35
x86: Partitioning Recommendations	36
Default Boot-Disk Partition Layout Preserves the Service Partition	36

Part II Using Custom JumpStart 37

3 Custom JumpStart (Overview) 39	
Custom JumpStart Introduction	39
Custom JumpStart Example Scenario	39
How the JumpStart Program Installs Solaris Software	40
4 Preparing Custom JumpStart Installations (Tasks) 45	
Task Map: Preparing Custom JumpStart Installations	45
Creating a Profile Server for Networked Systems	47
▼ To Create a JumpStart Directory on a Server	47
▼ To Allow All Systems Access to the Profile Server	49
Creating a Profile Diskette for Standalone Systems	51
▼ SPARC: To Create a Profile Diskette	51
▼ x86: To Create a Profile Diskette	53
Creating the rules File	56
Syntax of the rules File	56
▼ To Create a rules File	57
rules File Example	58
Creating a Profile	59
Syntax of Profiles	60
▼ To Create a Profile	60
Profile Examples	61
Testing a Profile	70
▼ To Create a Temporary Solaris Environment to Test a Profile	71
▼ To Test a Profile	72
Profile Test Examples	74

Validating the rules File	75
▼ To Validate the rules File	75
5 Using Optional Custom JumpStart Features (Tasks)	77
Creating Begin Scripts	77
Important Information About Begin Scripts	78
Creating Derived Profiles With a Begin Script	78
Creating Finish Scripts	79
Important Information About Finish Scripts	79
▼ To Add Files With a Finish Script	80
Adding Packages or Patches With a Finish Script	80
Customizing the Root Environment With a Finish Script	83
Setting a System's Root Password With a Finish Script	83
Non-Interactive Installations With Finish Scripts	85
Creating a Compressed Configuration File	85
▼ To Create a Compressed Configuration File	85
Compressed Configuration File Example	86
Creating Disk Configuration Files	86
▼ SPARC: To Create a Disk Configuration File	86
SPARC: Disk Configuration File Example	88
▼ x86: To Create a Disk Configuration File	88
x86: Disk Configuration File Example	90
Using a Site-Specific Installation Program	92
6 Creating Custom Rule and Probe Keywords (Tasks)	93
Probe Keywords	93
Creating a custom_probes File	94
Syntax of the custom_probes File	94
Syntax of Function Names in custom_probes	95
▼ To Create a custom_probes File	95
Examples of a custom_probes File and Keyword	95
Validating the custom_probes File	97
▼ To Validate the custom_probes File	97
7 Performing a Custom JumpStart Installation (Tasks)	99
SPARC: Task Map: Setting Up a System for a Custom JumpStart Installation	99
SPARC: Performing a Custom JumpStart Installation	101

- ▼ To Prepare to Install a Solaris Flash Archive With a Custom JumpStart Installation 101
- ▼ SPARC: To Perform an Installation or Upgrade With the Custom JumpStart Program 103
- x86: Task Map: Setting Up a System for a Custom JumpStart Installation 105
- x86: Performing a Custom JumpStart Installation 106
 - ▼ x86: To Perform an Installation or Upgrade With the Custom JumpStart Program 106

8 Installing With Custom JumpStart (Examples) 111

- Sample Site Setup 111
- Create an Install Server 113
- x86: Create a Boot Server for Marketing Systems 114
- Create a JumpStart Directory 115
- Share the JumpStart Directory 115
- SPARC: Create the Engineering Group's Profile 115
- x86: Create the Marketing Group's Profile 116
- Update the rules File 117
- Validate the rules File 117
- SPARC: Set Up Engineering Systems to Install From the Network 118
- x86: Set Up Marketing Systems to Install From the Network 118
- SPARC: Boot the Engineering Systems and Install Solaris 10 Software 119
- x86: Boot the Marketing Systems and Install Solaris 10 Software 120

9 Custom JumpStart (Reference) 121

- Rule Keywords and Values 121
- Profile Keywords and Values 125
 - Profile Keywords Quick Reference 126
 - Profile Keyword Descriptions and Examples 127
- Custom JumpStart Environment Variables 165
- Probe Keywords and Values 167

Part III Using RAID-1 Volumes 169

10 Creating RAID-1 Volumes (Mirrors) During Installation (Overview) 171

- Why Use RAID-1 Volumes? 171
- How Do RAID-1 Volumes Work? 172

	Overview of Solaris Volume Manager Components	174
	State Database and State Database Replicas	174
	RAID-0 Volumes (Concatenations)	175
	RAID-1 Volumes (Mirrors)	176
	Example of RAID-1 Volume Disk Layout	177
11	Creating RAID-1 Volumes (Mirrors) During Installation (Planning)	179
	System Requirement	179
	State Database Replicas Guidelines and Requirements	180
	Selecting Slices for State Database Replicas	180
	Choosing the Number of State Database Replicas	181
	Distributing State Database Replicas Across Controllers	181
	RAID-1 and RAID-0 Volume Requirements and Guidelines	181
	Custom JumpStart and Solaris Live Upgrade Guidelines	181
	Guidelines for Selecting Disks and Controllers	184
	Guidelines for Selecting Slices	184
	How Booting Into Single-User Mode Affects RAID-1 Volumes	185
Part IV	Appendixes	187
A	Troubleshooting (Tasks)	189
	Problems With Setting Up Network Installations	189
	Problems With Booting a System	190
	Booting From Media, Error Messages	190
	Booting From Media, General Problems	191
	Booting From the Network, Error Messages	193
	Booting From the Network, General Problems	196
	Initial Installation of the Solaris OS	196
	▼ x86: To Check IDE Disk for Bad Blocks	197
	Upgrading the Solaris OS	198
	Upgrading, Error Messages	198
	Upgrading, General Problems	199
	▼ To Continue Upgrading After a Failed Upgrade	200
	▼ System Panics When Upgrading With Solaris Live Upgrade Running Veritas VxVm	200
	x86: Service Partition Not Created by Default on Systems With No Existing Service Partition	202

▼ To Install Software From a Network Installation Image or From the Solaris 10 Operating System DVD 203

▼ To Install From the Solaris 10 Software - 1 CD or From a Network Installation Image 203

B Additional SVR4 Packaging Requirements (Reference) 205

Preventing Modification of the Current Solaris OS 205

Using Absolute Paths 205

Using the pkgadd -R Command 206

Differences Between \$PKG_INSTALL_ROOT and \$BASEDIR Overview 206

Guidelines for Writing Scripts 207

Maintaining Diskless Client Compatibility 208

Verifying Packages 208

Preventing User Interaction When Installing or Upgrading 209

For More Information 210

Glossary 213

Index 229

Preface

This book describes how to install and upgrade the Solaris™ 10 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 10 Hardware Compatibility List* at <http://www.sun.com/bigadmin/hcl>. This document cites any implementation differences between the platform types.

In this document the term “x86” refers to 64-bit and 32-bit systems manufactured using processors compatible with the AMD64 or Intel Xeon/Pentium product families. For supported systems, see the *Solaris 10 Hardware Compatibility List*.

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 related information that you need when you install the Solaris software.

TABLE P-1 Related Information

Information	Description
<i>Solaris 10 Installation Guide: Basic Installations</i>	Describes a basic OS installation with a graphical user interface (GUI).
<i>Solaris 10 Installation Guide: Network-Based Installations</i>	Describes how to perform a remote Solaris installation over a local area network or a wide area network.
<i>Solaris 10 Installation Guide: Custom JumpStart and Advanced Installations</i>	Describes how to create the files and directories necessary to perform an unattended custom JumpStart™ installation. This book also describes how to create RAID-1 volumes.
<i>Solaris 10 Installation Guide: Solaris Live Upgrade and Upgrade Planning</i>	Provides planning information when using CD or DVD media to upgrade a system to the Solaris OS. This book also describes how to use Solaris Live Upgrade to create and upgrade new boot environments.
<i>Solaris 10 Installation Guide: Solaris Flash Archives (Creation and Installation)</i>	Provides instructions for creating Solaris Flash archives and using Solaris Flash archives to install the Solaris OS on multiple systems.
<i>System Administration Guide: Devices and File Systems</i>	Describes how to back up system files and other system administration tasks.
<i>Solaris 10 Release Notes</i>	Describes any bugs, known problems, software that is being discontinued, and patches that are related to the Solaris release.
SPARC: <i>Solaris 10 Sun Hardware Platform Guide</i> at http://docs.sun.com	Contains information about supported hardware.
<i>Solaris 10 Package List</i>	Lists and describes the packages in the Solaris 10 OS.
x86: <i>Solaris Hardware Compatibility List</i>	Contains supported hardware information and device configuration.

Accessing Sun Documentation Online

The docs.sun.comSM Web site enables you to access Sun technical documentation online. You can browse the docs.sun.com archive or search for a specific book title or subject. The URL is <http://docs.sun.com>.

Ordering Sun Documentation

Sun Microsystems offers select product documentation in print. For a list of documents and how to order them, see “Buy printed documentation” at <http://docs.sun.com>.

Typographic Conventions

The following table describes the typographic changes used in this book.

TABLE P-2 Typographic Conventions

Typeface or Symbol	Meaning	Example
AaBbCc123	The names of commands, files, and directories; on-screen computer output	Edit your <code>.login</code> file. Use <code>ls -a</code> to list all files. <code>machine_name% you have mail.</code>
AaBbCc123	What you type, contrasted with on-screen computer output	<code>machine_name% su</code> <code>password:</code>
<i>AaBbCc123</i>	Command-line placeholder: replace with a real name or value	To delete a file, type <code>rm <i>filename</i></code> .
<i>AaBbCc123</i>	Book titles, new words, or terms, or words to be emphasized.	Read Chapter 6 in <i>User's Guide</i> . These are called <i>class</i> options. You must be <i>root</i> to do this.

Shell Prompts in Command Examples

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

TABLE P-3 Shell Prompts

Shell	Prompt
C shell prompt	machine_name%
C shell superuser prompt	machine_name#
Bourne shell and Korn shell prompt	\$
Bourne shell and Korn shell superuser prompt	#

PART I

Overall Planning of Any Solaris Installation or Upgrade

This part guides you through planning the installation or upgrade of the Solaris operating system when using any installation program.

Solaris Installation and Upgrade (Roadmap)

This chapter provides you with information about decisions you need to make before you install or upgrade the Solaris Operating System (Solaris OS). This chapter contains the following sections:

- “Task Map: Installing or Upgrading the Solaris Software” on page 15
- “Installing From the Network or From DVD or CDs?” on page 18
- “Initial Installation, or Upgrade?” on page 19
- “Choosing a Solaris Installation Method” on page 20
- “Sun Java System Application Server Platform Edition 8” on page 21

Note – This book uses the term *slice*, but some Solaris documentation and programs might refer to a slice as a partition.

x86: To avoid confusion, this book distinguishes between x86 `fdisk` partitions and the divisions within the Solaris `fdisk` partition. The x86 `fdisk` divisions are called partitions. The divisions within the Solaris `fdisk` partition are called slices.

Task Map: Installing or Upgrading the Solaris Software

The following task map is an overview of the steps necessary to install or upgrade the Solaris OS when using any installation program. Use this task map to identify all of the decisions that you need to make to complete the most efficient installation for your environment.

TABLE 1-1 Task Map: Installing or Upgrading the Solaris Software

Task	Description	For Instructions
Choose initial installation or upgrade.	Decide if you want to perform an initial installation or an upgrade.	“Initial Installation, or Upgrade?” on page 19.
Choose an installation program.	The Solaris OS provides several programs for installation or upgrade. Choose the installation method that is most appropriate for your environment.	“Choosing a Solaris Installation Method” on page 20.
(Solaris installation program) Choose a default or custom installation.	Decide which type installation is suitable for your environment: <ul style="list-style-type: none"> ■ If you are using a graphical user interface (GUI) you can choose a default or a custom installation: <ul style="list-style-type: none"> ■ A default installation formats the hard disk and installs a preselected set of software, including the Sun Java™ Enterprise System. ■ A custom installation enables you to modify the hard disk layout and select the software that you want to install. ■ If you use a text installer (non-graphical interface), you can select the default values or edit the values to select the software you want to install. 	For information about the Sun Java Enterprise System, see <i>Sun Java Enterprise System Technical Overview</i> at http://docs.sun.com
Review system requirements. Also, plan and allocate disk space and swap space.	Determine if your system meets the minimum requirements to install or upgrade. Allocate disk space on your system for the components of the Solaris OS that you want to install. Determine the appropriate swap-space layout for your system.	Chapter 2.
Choose to install a system from local media or from the network.	Decide on the most appropriate installation media for your environment.	“Installing From the Network or From DVD or CDs?” on page 18.

TABLE 1-1 Task Map: Installing or Upgrading the Solaris Software (Continued)

Task	Description	For Instructions
Gather information about your system.	<ul style="list-style-type: none"> ■ For the Solaris installation program, complete the worksheet to collect all of the information that you need to install or upgrade. ■ For the custom JumpStart installation method, decide which profile keywords to use in your profile. Then review the keyword descriptions to find the information about your system that you need. 	<ul style="list-style-type: none"> ■ For the Solaris installation program, see either of the following documents: <ul style="list-style-type: none"> ■ Chapter 3, “Gathering Information Before Installation or Upgrade (Planning),” in <i>Solaris 10 Installation Guide: Network-Based Installations</i> ■ Chapter 3, “Gathering Information Before Upgrade (Planning),” in <i>Solaris 10 Installation Guide: Solaris Live Upgrade and Upgrade Planning</i> ■ For the custom JumpStart installation method, see Chapter 9.
(Optional) Set system parameters.	You can preconfigure system information to avoid being prompted for the information during the installation or upgrade.	Chapter 4, “Preconfiguring System Configuration Information (Tasks),” in <i>Solaris 10 Installation Guide: Network-Based Installations</i> .
(Optional) Prepare to install the Solaris software from the network.	If you chose to install the Solaris software from the network, create an installation server, create a boot server (if necessary), and set up the systems to be installed from the network.	<p>To install over a local area network, see Chapter 7, “Preparing to Install From the Network With CD Media (Tasks),” in <i>Solaris 10 Installation Guide: Network-Based Installations</i>.</p> <p>To install over a wide area network, see Chapter 11, “Preparing to Install With WAN Boot (Tasks),” in <i>Solaris 10 Installation Guide: Network-Based Installations</i>.</p>
(Upgrade only) Perform tasks prior to upgrade.	Back up your system and determine if you can upgrade with disk space reallocation.	“Upgrade” on page 29 .
Perform an installation or upgrade.	Use the Solaris installation method that you chose to install or upgrade the Solaris software.	The chapter or chapters that provide detailed instructions for the installation programs.

TABLE 1-1 Task Map: Installing or Upgrading the Solaris Software (Continued)

Task	Description	For Instructions
Troubleshoot installation problems	Review the troubleshooting information when you encounter problems with your installation.	Appendix A.

Installing From the Network or From DVD or CDs?

The Solaris software is distributed on DVD or CD media so that you can install or upgrade systems that have access to a DVD-ROM or CD-ROM drive.

You can set up the systems to install from the network with remote DVD or CD images. You might want to set up systems this way for the following reasons:

- If you have systems that do not have local DVD-ROM or CD-ROM drives
- If you are installing several systems and do not want to insert the discs into every local drive to install the Solaris software

You can use all of the Solaris installation methods to install a system from the network. However, by installing systems from the network with the Solaris Flash installation feature or with a custom JumpStart installation, you can centralize and automate the installation process in a large enterprise. For more details about the different installation methods, refer to [“Choosing a Solaris Installation Method”](#) on page 20.

Installing the Solaris software from the network requires initial setup. For information about preparing to install from the network, choose one of the following options.

For detailed instructions about preparing to install from a local area network	Chapter 7, “Preparing to Install From the Network With CD Media (Tasks),” in <i>Solaris 10 Installation Guide: Network-Based Installations</i>
For instructions about preparing to install over a wide area network	Chapter 11, “Preparing to Install With WAN Boot (Tasks),” in <i>Solaris 10 Installation Guide: Network-Based Installations</i>
For instructions about how to install x86 based clients over the network by using PXE	“Booting and Installing Over the Network With PXE” in <i>Solaris 10 Installation Guide: Network-Based Installations</i>

Initial Installation, or Upgrade?

You can choose to perform an initial installation or, if your system is already running the Solaris OS, you can upgrade your system.

Initial Installation

An initial installation overwrites the system's disk with the new version of the Solaris OS. If your system is not running the Solaris OS, you must perform an initial installation.

If the system is already running the Solaris OS, you can choose to perform an initial installation. If you want to preserve any local modifications, before you install, you must back up the local modifications. After you complete the installation, you can restore the local modifications.

You can use any of the Solaris installation methods to perform an initial installation. For detailed information about the different Solaris installation methods, refer to [“Choosing a Solaris Installation Method” on page 20](#).

Upgrade

You can upgrade the Solaris OS by using two upgrade methods: standard and Solaris Live Upgrade. A standard upgrade maintains as many existing configuration parameters as possible of the current Solaris OS. Solaris Live Upgrade creates a copy of the current system. This copy can be upgraded with a standard upgrade. The upgraded Solaris OS can then be switched to become the current system by a simple reboot. If a failure occurs, you can switch back to the original Solaris OS with a reboot. Solaris Live Upgrade enables you to keep your system running while you upgrade and enables you to switch back and forth between Solaris OS releases.

You can upgrade any system that is running the Solaris 7, Solaris 8, or Solaris 9 software. For more information about upgrading and the list of upgrade methods, see [“Upgrade” on page 29](#).

Choosing a Solaris Installation Method

The Solaris OS provides several programs for installation or upgrade. Each installation technology offers different features that are designed for specific installation requirements and environments. Use the following table to help you decide which installation method to use.

TABLE 1-2 Choosing Your Installation Method

Task	Installation Method	Instructions
Install one system from CD-ROM or DVD-ROM media with an interactive program.	Solaris installation program	<i>Solaris 10 Installation Guide: Basic Installations</i>
Install one system over a local area network.	Solaris installation program over the network	Part II, "Installing Over a Local Area Network," in <i>Solaris 10 Installation Guide: Network-Based Installations</i>
Automate the installation or upgrade of multiple systems based on profiles you create.	Custom JumpStart	Chapter 4
Replicate the same software and configuration on multiple systems.	Solaris Flash archives	Chapter 1, "Solaris Flash (Overview)," in <i>Solaris 10 Installation Guide: Solaris Flash Archives (Creation and Installation)</i>
Install systems over a wide area network (WAN) or the Internet.	WAN boot	Chapter 9, "WAN Boot (Overview)," in <i>Solaris 10 Installation Guide: Network-Based Installations</i>
Upgrade a system while it is running.	Solaris Live Upgrade	Chapter 4, "Solaris Live Upgrade (Overview)," in <i>Solaris 10 Installation Guide: Solaris Live Upgrade and Upgrade Planning</i>
After installing the Solaris OS, create an isolated application environment.	Solaris Zones	Chapter 16, "Introduction to Solaris Zones," in <i>System Administration Guide: Solaris Containers—Resource Management and Solaris Zones</i>

Sun Java System Application Server Platform Edition 8

The Sun Java System Application Server Platform Edition 8 provides for broad deployment of application services and web services. This software is automatically installed with the Solaris OS. You can find documentation for the server in the following areas:

For documentation about starting the server	See <i>Sun Java System Application Server Platform Edition 8 QuickStart Guide</i> in the installation directory at <code>/docs/QuickStart.html</code>
For the full Application Server documentation set	http://docs.sun.com/db/coll/ApplicationServer8_04q2
For a tutorial	http://java.sun.com/j2ee/1.4/docs/tutorial/doc/index.html

Solaris Installation and Upgrade (Planning)

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. This chapter contains the following sections:

- “System Requirements and Recommendations” on page 23
- “Allocating Disk and Swap Space” on page 25
- “Upgrade” on page 29
- “How to Find the Version of the Solaris OS That Your System Is Running” on page 32
- “Locale Values” on page 32
- “Platform Names and Groups” on page 33
- “Planning to Install and Configure Zones” on page 33
- “SPARC: 64-bit Packaging Changes” on page 35
- “x86: Partitioning Recommendations” on page 36

System Requirements and Recommendations

TABLE 2-1 Memory, Swap, and Processor Recommendations

System	Size
Memory to install or upgrade	256 MB is the recommended size. 64 MB is the minimum size. Note – Some optional installation features are enabled only when sufficient memory is present. For example, if you install from a DVD with insufficient memory, you install through the Solaris installation program’s text installer, not through the graphical user interface (GUI). For more information about these memory requirements, see Table 2-2 .

TABLE 2-1 Memory, Swap, and Processor Recommendations (Continued)

System	Size
Swap area	512 Mbytes is the default size. Note – You might need to customize the swap space. Swap space is based on the size of the system’s hard disk.
x86: Processor requirements	SPARC: 200-MHz or faster processor is required. x86: 120-MHz or faster processor is recommended. Hardware floating-point support is required.

You can choose to install the software with a GUI or with or without a windowing environment. If there is sufficient memory, the GUI is displayed by default. Other environments are displayed by default if memory is insufficient for the GUI. You can override defaults with the `nowin` or `text` boot options. But, you are limited by the amount of memory in your system or by installing remotely. Also if the Solaris installation program does not detect a video adapter, it automatically displays in a console-based environment. Table 2-2 describes these environments and lists minimal memory requirements for displaying them.

TABLE 2-2 Memory Requirements for Display Options

Minimal Memory	Type of Installation	Description	Windowing Environment
64–127 MB	Console-based	Contains no graphics and no windowing environment. If no video adapter is detected, the installer displays a console-based environment. If you are installing remotely through a <code>tip</code> line or using the <code>nowin</code> boot option, you are limited to the console-based installation.	
128–383 MB	Console-based	Contains no graphics, but provides a window and the ability to open other windows. Requires a local or remote DVD-ROM or CD-ROM drive or network connection, video adapter, keyboard, monitor. If you install by using the <code>text</code> boot option and have enough memory, you are installing in a windowing environment.	X
384MB or greater	GUI-based	Provides windows, pull-down menus, buttons, scrollbars, and iconic images. A GUI requires a local or remote DVD-ROM or CD-ROM drive or network connection, video adapter, keyboard, monitor.	X

Allocating Disk and Swap Space

Before you install the Solaris software, you can determine if your system has enough disk space by doing some high-level planning.

General Disk Space Planning and Recommendations

Planning disk space is different for everyone. Consider allocating space for the following conditions, depending on your needs.

TABLE 2-3 General Disk Space and Swap Space Planning

Conditions for Space Allocations	Description
File systems	<p>For each file system that you create, allocate an additional 30 percent more disk space than you need to enable you to upgrade to future Solaris versions.</p> <p>By default, the Solaris installation methods create only root (/) and /swap. When space is allocated for OS services, the /export directory is also created. If you are upgrading to a major Solaris release, you might need to reslice your system or allocate double the space that you need at installation time. If you are upgrading to an update, you could prevent having to reslice your system by allocating extra disk space for future upgrades. A Solaris update release needs approximately 10 percent more disk space than the previous release. You can allocate an additional 30 percent of disk space for each file system to allow space for several Solaris updates.</p>
The /var file system	<p>If you intend to use the crash dump feature <code>savecore(1M)</code>, allocate double the amount of your physical memory in the /var file system.</p>

TABLE 2-3 General Disk Space and Swap Space Planning (Continued)

Conditions for Space Allocations	Description
Swap	<p>The Solaris installation program allocates a default swap area of 512 Mbytes under the following conditions:</p> <ul style="list-style-type: none"> ■ If you use the installation program’s automatic layout of disk slices ■ If you avoid manually changing the size of the swap slice <p>By default, the Solaris installation programs allocate swap space by placing swap so that it starts at the first available disk cylinder (typically cylinder 0 on SPARC based systems). This placement provides maximum space for the root (/) file system during the default disk layout and enables the growth of the root (/) file system during an upgrade.</p> <p>If you think you might need to expand the swap area in the future, you can place the swap slice so that it starts at another disk cylinder by using one of the following methods.</p> <ul style="list-style-type: none"> ■ For the Solaris installation program, you can customize the disk layout in cylinder mode and manually assign the swap slice to the desired location. ■ For the custom JumpStart installation program, you can assign the swap slice in the profile file. For more information about the JumpStart profile file, see “Creating a Profile” on page 59. <p>For an overview of the swap space, see Chapter 20, “Configuring Additional Swap Space (Tasks),” in <i>System Administration Guide: Devices and File Systems</i>.</p>
A server that is providing home directory file systems	By default, home directories are usually located in the /export file system.
The Solaris software group you are installing	A software group is a grouping of software packages. When you are planning disk space, remember that you can add or remove individual software packages from the software group that you select. For information about software groups, see “Disk Space Recommendations for Software Groups” on page 27 .
Upgrade	<ul style="list-style-type: none"> ■ If you are using Solaris Live Upgrade to upgrade an inactive boot environment and want information about disk space planning, see “Solaris Live Upgrade Disk Space Requirements” in Solaris 10 Installation Guide: Solaris Live Upgrade and Upgrade Planning. ■ If you are using other Solaris installation methods to plan disk space, see “Upgrading With Disk Space Reallocation” on page 31.
Language support	For example, Chinese, Japanese, or Korean.
Printing or mail support	Allocate additional space.
Additional software or third-party software	Allocate additional space.

Disk Space Recommendations for Software Groups

The Solaris software groups are collections of Solaris packages. Each software group includes support for different functions and hardware drivers.

- For an initial installation, you select the software group to install, based on the functions that you want to perform on the system.
- For an upgrade, you must upgrade to a software group that is installed on the system. For example, if you previously installed the End User Solaris Software Group on your system, you cannot use the upgrade option to upgrade to the Developer Solaris Software Group. However, during the upgrade you can add software to the system that is not part of the currently installed software group.

When you are installing the Solaris software, you can choose to add or remove packages from the Solaris software group that you selected. When you are selecting which packages to add or remove, you need to know about software dependencies and how the Solaris software is packaged.

The following figure shows the grouping of software packages. Reduced Network Support contains the minimal number of packages and Entire Solaris Software Group Plus OEM Support contains all the packages.

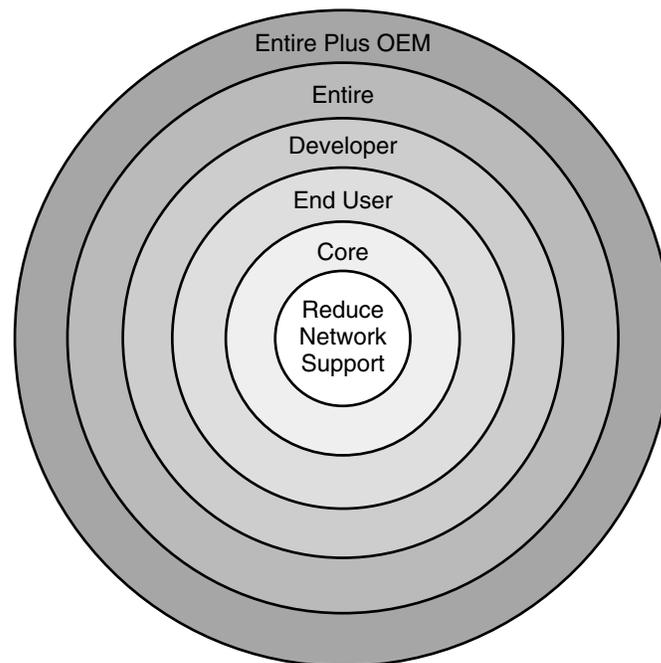


FIGURE 2-1 Solaris Software Groups

Table 2–4 lists the Solaris software groups and the recommended amount of disk space that you need to install each group.

Note – The disk space recommendations in Table 2–4 include space for the following items.

- Swap space
- Patches
- Additional software packages

You might find that the software groups require less disk space than the amount that is listed in this table.

TABLE 2–4 Disk Space Recommendations for Software Groups

Software Group	Description	Recommended Disk Space
Entire Solaris Software Group Plus OEM Support	Contains the packages for the Entire Solaris Software Group plus additional hardware drivers, including drivers for hardware that is not on the system at the time of installation.	6.7 Gbytes
Entire Solaris Software Group	Contains the packages for the Developer Solaris Software Group and additional software that is needed for servers.	6.5 Gbytes
Developer Solaris Software Group	Contains the packages for the End User Solaris Software Group plus additional support for software development. The additional software development support includes libraries, include files, man pages, and programming tools. Compilers are not included.	6.0 Gbytes
End User Solaris Software Group	Contains the packages that provide the minimum code that is required to boot and run a networked Solaris system and the Common Desktop Environment.	5.0 Gbytes
Core System Support Software Group	Contains the packages that provide the minimum code that is required to boot and run a networked Solaris system.	2.0 Gbytes
Reduced Network Support Software Group	Contains the packages that provide the minimum code that is required to boot and run a Solaris system with limited network service support. The Reduced Network Support 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.	2.0 Gbytes

Upgrade

You can upgrade a system by using one of three different upgrade methods: Solaris Live Upgrade, the Solaris installation program, and custom JumpStart.

TABLE 2-5 Solaris Upgrade Methods

Current Solaris OS	Solaris Upgrade Methods
Solaris 7, Solaris 8, Solaris 9	<ul style="list-style-type: none">■ Solaris Live Upgrade – Upgrades a system by creating and upgrading a copy of the running system■ The Solaris installation program – Provides an interactive upgrade with a graphical user interface or command-line interface■ Custom JumpStart method – Provides an automated upgrade

Upgrade Limitations

Issue	Description
Upgrading to a different software group	You cannot upgrade your system to a software group that is not installed on the system. For example, if you previously installed the End User Solaris Software Group on your system, you cannot use the upgrade option to upgrade to the Developer Solaris Software Group. However, during the upgrade you can add software to the system that is not part of the currently installed software group.
Using the Solaris installation program to upgrade from a CD or DVD	You must have a free slice on the disk that does not store files and can be overwritten by the installation software. The <code>swap</code> slice is preferred, but you can use any slice that is not located in any of the “upgradable” root slices that are listed in <code>/etc/vfstab</code> . The size of this slice must be at least 512 Mbytes.

Upgrade Programs

You can perform a standard interactive upgrade with the Solaris installation program or an unattended upgrade with the custom JumpStart installation method. Solaris Live Upgrade enables you to upgrade a running system.

Upgrade Program	Description	For More Information
Solaris Live Upgrade	Enables you to create a copy of the currently running system. The copy can be upgraded and then a reboot switches the upgraded copy to become the currently running system. Using Solaris Live Upgrade reduces the downtime that is required to upgrade the Solaris OS. Also, Solaris Live Upgrade can prevent problems with upgrading. An example is the inability to recover from an upgrade if the power fails, because the copy being upgraded is not the currently running system.	To plan for disk space allocation when using Solaris Live Upgrade, see “Solaris Live Upgrade Requirements” in <i>Solaris 10 Installation Guide: Solaris Live Upgrade and Upgrade Planning</i> .
The Solaris installation program	Guides you through an upgrade with a GUI.	Chapter 2, “Installing With the Solaris Installation Program (Tasks),” in <i>Solaris 10 Installation Guide: Basic Installations</i> .
Custom JumpStart program	Provides an automated upgrade. A profile file and optional preinstallation and postinstallation scripts provide the information required. When creating a custom JumpStart profile for an upgrade, specify <code>install_type upgrade</code> . You must test the custom JumpStart profile against the system’s disk configuration and currently installed software before you upgrade. Use the <code>pfinstall -D</code> command on the system that you are upgrading to test the profile. You cannot test an upgrade profile by using a disk configuration file.	<ul style="list-style-type: none"> ■ For more information about testing the upgrade option, refer to “Testing a Profile” on page 70. ■ For more information about creating a upgrade profile, see “Profile Examples” on page 61 ■ For more information about performing an upgrade, see “SPARC: Performing a Custom JumpStart Installation” on page 101

Installing a Solaris Flash Archive Instead of Upgrading

The Solaris Flash installation feature provides a method of creating a copy of the whole installation from a master system that can be replicated on many clone systems. This copy is called a Solaris Flash archive. You can install an archive by using any installation program. For information about installing an archive, see the following table.

Solaris Live Upgrade	“Installing Solaris Flash Archives on a Boot Environment” in <i>Solaris 10 Installation Guide: Solaris Live Upgrade and Upgrade Planning</i>
Custom JumpStart	Chapter 7

Solaris installation program	Chapter 4, “Installing and Administering Solaris Flash Archives (Tasks),” in <i>Solaris 10 Installation Guide: Solaris Flash Archives (Creation and Installation)</i>
WAN boot installation method	Chapter 12, “Installing With WAN Boot (Tasks),” in <i>Solaris 10 Installation Guide: Network-Based Installations</i>

Upgrading With Disk Space Reallocation

The upgrade option in the Solaris installation program and the `upgrade` keyword in the custom JumpStart program provide the ability to reallocate disk space. This reallocation automatically changes the sizes of the disk slices. You can reallocate disk space if the current file systems do not have enough space for the upgrade. For example, file systems might need more space for the upgrade for the following reasons:

- The Solaris software group that is currently installed on the system contains new software in the new release. Any new software that is included in a software group is automatically selected to be installed during the upgrade.
- The size of the existing software on the system has increased in the new release.

The auto-layout feature attempts to reallocate the disk space to accommodate the new size requirements of the file system. Initially, auto-layout attempts to reallocate space, based on a set of default constraints. If auto-layout cannot reallocate space, you must change the constraints on the file systems.

Note – Auto-layout does not have the ability to “grow” file systems. Auto-layout reallocates space by the following process:

1. Backing up required files on the file systems that need to change.
2. Repartitioning the disks on the basis of the file system changes.
3. Restoring the backup files before the upgrade happens.

-
- If you are using the Solaris installation program, and auto-layout cannot determine how to reallocate the disk space, you must use the custom JumpStart program to upgrade.
 - If you are using the custom JumpStart method to upgrade and you create an upgrade profile, disk space might be a concern. If the current file systems do not contain enough disk space for the upgrade, you can use the `backup_media` and `layout_constraint` keywords to reallocate disk space. For an example of how to use the `backup_media` and `layout_constraint` keywords in a profile, refer to “Profile Examples” on page 61.

Using the Solaris Installation Program to Upgrade From DVD or CD Media

When you use the Solaris installation program from a DVD or CD to upgrade, you must have a slice on the disk that does not store files. The swap slice is preferred, but you can use any slice that is not located in any of the upgradable root slices that are listed in the `/etc/vfstab`. The size of this slice must be at least 512 Mbytes.

Backing Up Systems Before Upgrading

Backing up your existing file systems before you upgrade to the Solaris OS is highly recommended. If you copy file systems to removable media, such as tape, you can safeguard against data loss, damage, or corruption. For detailed instructions to back up your system, refer to Chapter 23, “Backing Up and Restoring File Systems (Overview),” in *System Administration Guide: Devices and File Systems*.

How to Find the Version of the Solaris OS That Your System Is Running

To see the version of Solaris software that is running on your system, type either of the following commands.

```
$ uname -a
```

The `cat` command provides more detailed information.

```
$ cat /etc/release
```

Locale Values

As a part of your installation, you can preconfigure the locale that you want the system to use. A *locale* determines how online information is displayed in a specific language and specific region. A language might also include more than one locale to accommodate regional differences, such as differences in the format of date and time, numeric and monetary conventions, and spelling.

You can preconfigure the system locale in a custom JumpStart profile or in the `sysidcfg` file.

Setting the locale in a profile	"Creating a Profile" on page 59.
Setting the locale in the <code>sysidcfg</code> file	"Preconfiguring With the <code>sysidcfg</code> File" in <i>Solaris 10 Installation Guide: Network-Based Installations</i>
List of locale values	<i>International Language Environments Guide</i>

Platform Names and Groups

When you are adding clients for a network installation, you must know your system architecture (platform group). If you are writing a custom JumpStart installation rules file, you need to know the platform name.

Some examples of platform names and groups follow. For a full list of SPARC based systems, see *Solaris 10 Sun Hardware Platform Guide* at <http://docs.sun.com/>.

TABLE 2-6 Example of Platform Names and Groups

System	Platform Name	Platform Group
Sun Blade™	SUNW,Sun-Blade-100	sun4u
x86 based	i86pc	i86pc

Note – On a running system, you can also use the `uname -i` command to determine a system's *platform name* or the `uname -m` command to determine a system's *platform group*.

Planning to Install and Configure Zones

The following introduction provides high-level planning information for global and non-global zones. For more specific planning information and specific procedures, see Chapter 16, "Introduction to Solaris Zones," in *System Administration Guide: Solaris Containers—Resource Management and Solaris Zones*.

After the Solaris OS is installed, you can install and configure zones. In a zones environment, the global zone is the single instance of the operating system that is running and is contained on every Solaris system. The global zone is both the default zone for the system and the zone that is used for system-wide administrative control. A non-global zone is a virtualized operating system environment.

Solaris Zones are a software partitioning technology used to virtualize operating system services and provide an isolated and secure environment for running applications. When you create a zone, you produce an application execution environment in which processes are isolated from all other zones. This isolation prevents processes that are running in one zone from monitoring or affecting processes that are running in any other zones. Even a process running in a non-global zone with superuser credentials cannot view or affect activity in any other zones. A process running in the global zone with superuser credentials can affect any process in any zone.

The global zone is the only zone from which a non-global zone can be configured, installed, managed, or uninstalled. Only the global zone is bootable from the system hardware. 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 any or all other zones.

When installing the OS, the software group installed in the global zone is the set of packages that is shared by all the non-global zones. For example, if you install the Entire software group, all zones contain these packages. By default any additional packages installed in the global zone also populate the non-global zones. You can segregate into non-global zones applications, namespaces, servers, and network connections such as NFS and DHCP as well as other software. Each non-global zone is not aware of nor does it contain the segregated software in other non-global zones and each can operate independently. For example, you might have installed the Entire software group on the global zone and have running on separate non-global zones the Java Enterprise System Messaging Server, a database, DHCP, and a web server. When installing non-global zones keep in mind performance requirements of the applications running in each non-global zone.

Disk Space Requirements for Non-Global Zones

When installing the global zone, be sure to reserve enough disk space to house all of the zones you might create. Each non-global zone might have unique disk space requirements. The following description is a brief overview of planning information. For complete planning requirements and recommendations, see Chapter 18, "Planning and Configuring Non-Global Zones (Tasks)," in *System Administration Guide: Solaris Containers—Resource Management and Solaris Zones*.

No limits are placed on how much disk space can be consumed by a zone. The global zone administrator is responsible for space restriction. Even a small uniprocessor system can support a number of zones running simultaneously.

The nature of the packages installed in the global zone affects the space requirements of the non-global zones that are created. The number of packages and space requirements are factors. The following are general disk space guidelines.

- Approximately 100 Mbytes of free disk space is suggested when the global zone has been installed with all of the standard Solaris packages. Increase this amount if additional packages are installed in the global zone. By default, any additional packages installed in the global zone also populate the non-global zones. The directory location in the non-global zone for these additional packages is specified through the `inherit-pkg-dir` resource.
- Add 40 Mbytes of RAM per zone if the system has sufficient swap space. This addition is recommended to make each zone operational. When planning your system size, consider this addition of RAM.

Restricting Non-Global Zone Size

The following options can be used to restrict zone size.

- You can place the zone on a `lofi`-mounted partition. This action limits the amount of space consumed by the zone to that of the file used by `lofi`. For more information, see the `lofiadm(1M)` and `lofi(7D)` man pages.
- You can use soft partitions to divide disk slices or logical volumes into partitions. You can use these partitions as zone roots, and thus limit per-zone disk consumption. The soft partition limit is 8192 partitions. For more information, see Chapter 12, “Soft Partitions (Overview),” in *Solaris Volume Manager Administration Guide*.
- You can use the standard partitions of a disk for zone roots, and thus limit per-zone disk consumption.

SPARC: 64-bit Packaging Changes

In previous Solaris releases, the Solaris OS was delivered in separate packages for 32-bit and 64-bit components. In the Solaris 10 OS, packaging has been simplified with the delivery of most 32-bit and 64-bit components in a single package. The combined packages retain the names of the original 32-bit packages, and the 64-bit packages are no longer delivered. This change reduces the number of packages and simplifies installation.

The 64-bit packages are renamed with the following conventions:

- If a 64-bit package has a 32-bit counterpart, the 64-bit package is named with the 32-bit package name. For example, a 64-bit library such as `/usr/lib/sparcv9/libc.so.1` previously would have been delivered in `SUNWcslx`, but now is delivered in `SUNWcsl`. The 64-bit `SUNWcslx` package is no longer delivered.
- If a package does not have a 32-bit counterpart, the “x” suffix is removed from the name. For example, `SUNW1394x` becomes `SUNW1394`. This change means that you might need to modify your custom JumpStart script or other package installation

scripts to remove references to the 64-bit packages.

x86: Partitioning Recommendations

When using the Solaris OS on x86 based systems, follow these guidelines for partitioning your system.

The Solaris installation program uses a default boot-disk partition layout. These partitions are called `fdisk` partitions. An `fdisk` partition is 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.

TABLE 2-7 x86: Default Partitions

Partitions	Partition Name	Partition Size
First partition (on some systems)	Diagnostic or Service partition	Existing size on system
Second partition	x86 boot partition	Greater than 10 Mbytes, depending on disk size
Third partition	Solaris OS partition	Remaining space on the boot disk

Default Boot-Disk Partition Layout Preserves the Service Partition

The Solaris installation program uses a default boot-disk partition layout to accommodate the diagnostic or Service partition. If your system currently includes a diagnostic or Service partition, the default boot-disk partition layout enables you to preserve this partition.

Note – If you install the Solaris OS on an x86 based system that does not currently include a diagnostic or Service partition, the installation program does not create a new diagnostic or Service partition by default. If you want to create a diagnostic or Service partition on your system, see your hardware documentation.

PART II Using Custom JumpStart

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

Custom JumpStart (Overview)

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

- [“Custom JumpStart Introduction” on page 39](#)
- [“How the JumpStart Program Installs Solaris Software” on page 40](#)

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 3-1 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.

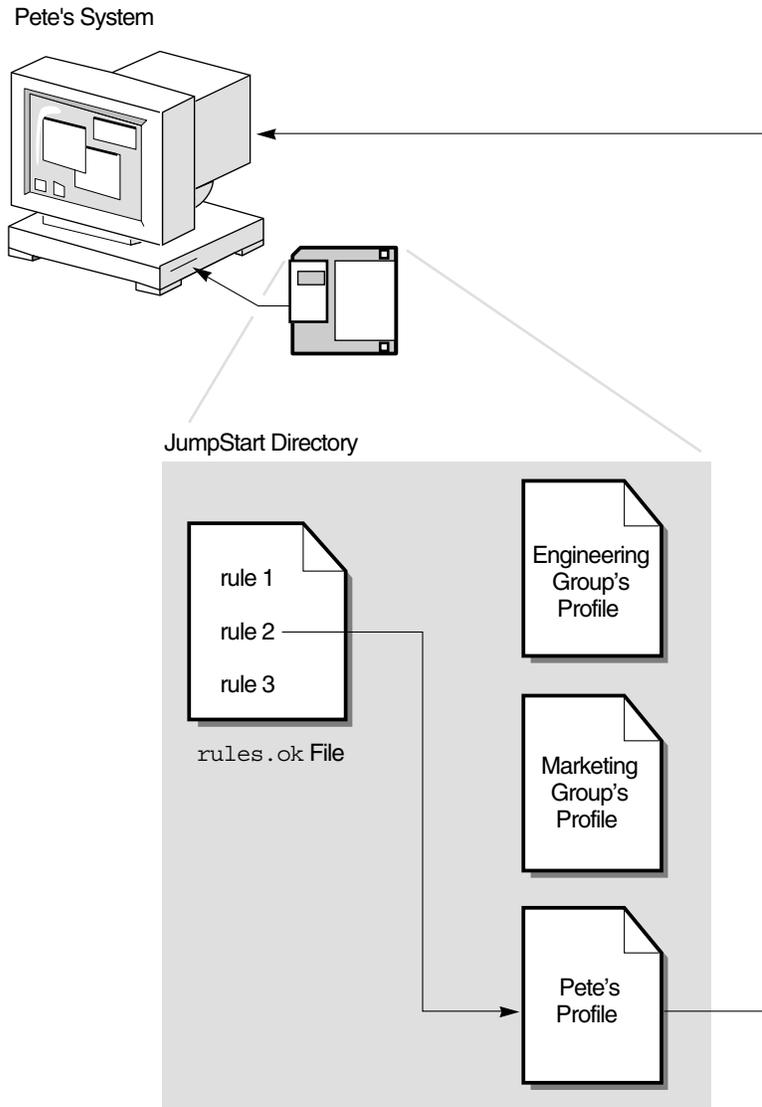


FIGURE 3-1 How a Custom JumpStart Installation Works: nonnetworked Example

Figure 3-2 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.

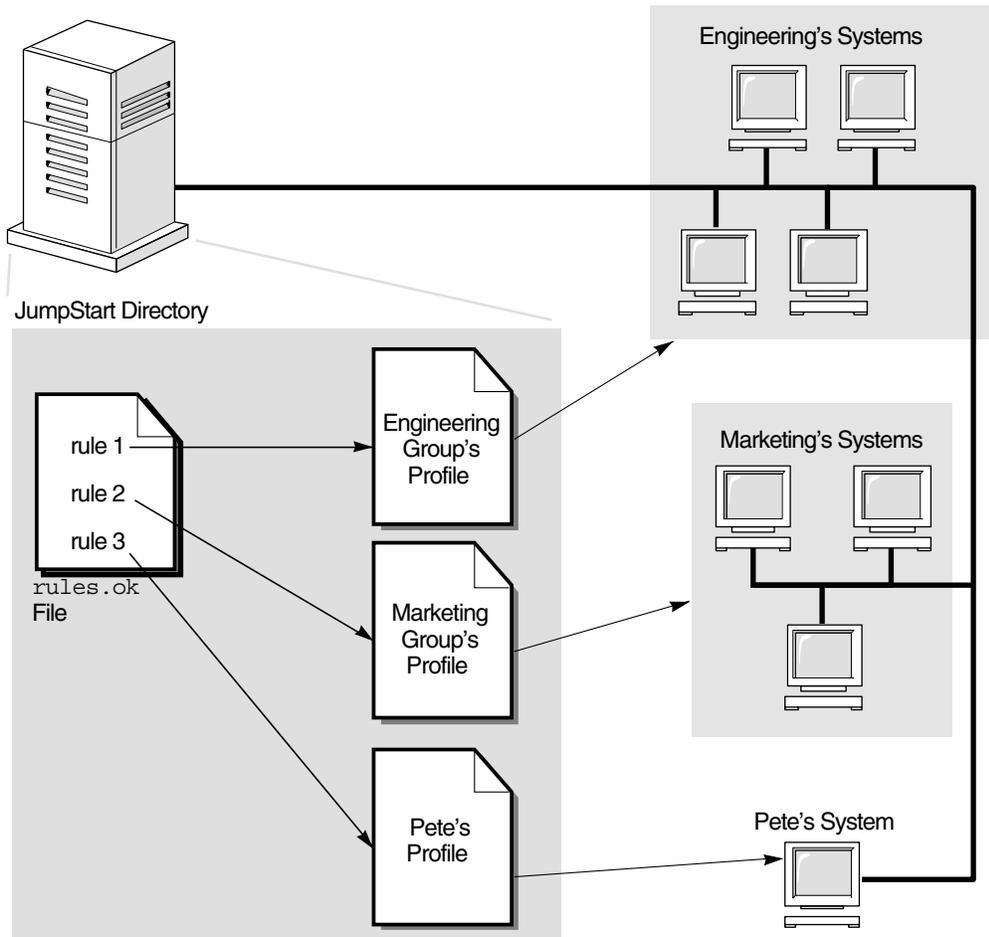


FIGURE 3-2 How a Custom JumpStart Installation Works: Networked Example

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

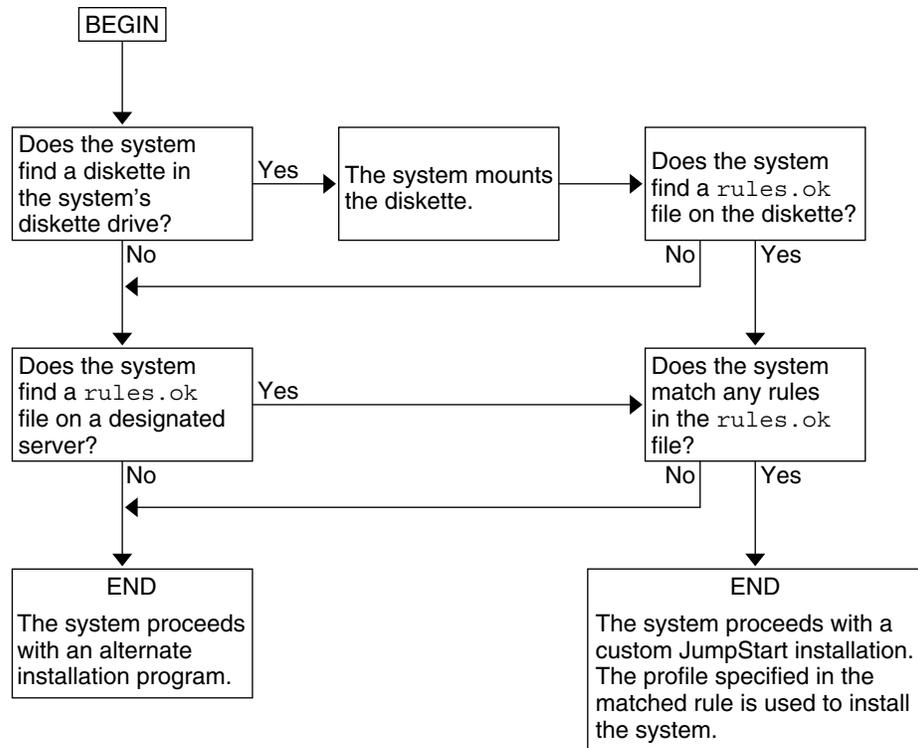


FIGURE 3-3 What Happens During a Custom JumpStart Installation

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.

- “Task Map: Preparing Custom JumpStart Installations” on page 45
- “Creating a Profile Server for Networked Systems” on page 47
- “Creating a Profile Diskette for Standalone Systems” on page 51
- “Creating the rules File” on page 56
- “Creating a Profile” on page 59
- “Testing a Profile” on page 70
- “Validating the rules File” on page 75

Task Map: Preparing Custom JumpStart Installations

TABLE 4-1 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” on page 29

TABLE 4-1 Task Map: Preparing Custom JumpStart Installations (Continued)

Task	Description	For Instructions
Create a JumpStart directory.	<p>On a server</p> <p>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.</p> <p>On a diskette</p> <p>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.</p>	<p>“Creating a Profile Server for Networked Systems” on page 47</p> <p>“Creating a Profile Diskette for Standalone Systems” on page 51</p>
Add rules to the <code>rules</code> file.	<p>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.</p>	<p>“Creating the <code>rules</code> File” on page 56</p>
Create a profile for every rule.	<p>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.</p>	<p>“Creating a Profile” on page 59</p>
(Optional) Test the profiles.	<p>After you create a profile, use the <code>pfinstall(1M)</code> command to test the profile before you use the profile to install or upgrade a system.</p>	<p>“Testing a Profile” on page 70</p>

TABLE 4-1 Task Map: Preparing Custom JumpStart Installations (Continued)

Task	Description	For Instructions
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 75

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 49](#).

▼ 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.

- Steps**
1. **Log in as superuser on the server on which you want to create the JumpStart directory.**

2. 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
```

3. 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
```

4. Type `shareall` and press Enter.

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 10 Operating System DVD or the Solaris 10 Software - 1 CD for your platform	Insert the Solaris 10 Operating System DVD or the Solaris 10 Software - 1 CD into the server's CD-ROM drive. Volume Manager automatically mounts the CD.
An image of the Solaris 10 Operating System DVD or the Solaris 10 Software - 1 CD for your platform on a local disk	Change directory to the location of the Solaris 10 Operating System DVD or the Solaris 10 Software - 1 image. For example, type the following command: <pre>cd /export/install</pre>

6. 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:

- For SPARC based systems:

```
cp -r /cdrom/cdrom0/s0/Solaris_10/Misc/jumpstart_sample/* /jumpstart
```

- For x86 based systems:

```
cp -r /cdrom/cdrom0/s2/Solaris_10/Misc/jumpstart_sample/* /jumpstart
```

```
cp -r /cdrom/cdrom0/s0/Solaris_10/Misc/jumpstart_sample/* /jumpstart
```

7. Update the example JumpStart files so that the files work in your environment.
8. Ensure that root owns the JumpStart directory and that the permissions are set to 755.
9. 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 49.

▼ To Allow 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:

- `add_install_client` command – Each time that you add a system for network installation, use the `-c` option with the `add_install_client` command. For detailed instructions, refer to [“Adding Systems to Be Installed From the Network With a CD Image”](#) in *Solaris 10 Installation Guide: Network-Based Installations*.
- `boot` command – 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 NFS server, an HTTP server, or on media that the system can access locally. For detailed instructions, refer to [“Creating a Compressed Configuration File”](#) on page 85.

When you boot the system to initiate the custom JumpStart installation, specify the location of the compressed file.

- For detailed instructions for SPARC based systems, refer to [Step 5](#) in [“SPARC: To Perform an Installation or Upgrade With the Custom JumpStart Program”](#) on page 103.
- For x86 based systems, refer to [Step 8](#) in [“x86: To Perform an Installation or Upgrade With the Custom JumpStart Program”](#) on page 106.
- `/etc/bootparams` file – Use a wildcard in the `/etc/bootparams` file. Use the following steps to add a wildcard in the `etc/bootparams` file.

Note – The following procedure is not necessary if you save the JumpStart directory on a diskette or if you specify the location of the profile server when you boot the system.

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.

▼ 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.

- Steps**
1. Log in as superuser on a SPARC based system to which a diskette drive is attached.
 2. Insert a blank diskette or a diskette that can be overwritten in the diskette drive.
 3. Mount the diskette.

```
# volcheck
```

4. 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.
5. Format the diskette.



Caution – Formatting erases all data on the diskette.

```
# fdformat -U
```

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 10 Operating System for SPARC Platforms DVD or the Solaris 10 Software for SPARC Platforms - 1 CD	Insert the Solaris 10 Operating System for SPARC Platforms DVD or the Solaris 10 Software for SPARC Platforms - 1 CD into the server's CD-ROM drive. Volume Manager automatically mounts the CD.
An image of the Solaris 10 Operating System for SPARC Platforms DVD or the Solaris 10 Software for SPARC Platforms - 1 CD on a local disk	Change the directory to the location of the Solaris 10 Operating System for SPARC Platforms DVD or the Solaris 10 Software for SPARC Platforms - 1 CD image. For example, type the following command: <code>cd /export/install</code>

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 10 Software for SPARC Platforms - 1 CD to the root (`/`) directory on a profile diskette that is named `scrap`:

```
cp -r /cdrom/sol_10_sparc/s0/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 56](#).

▼ x86: 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.

- Steps**
1. Log in as superuser on an x86 based system to which a diskette drive is attached.
 2. Insert the Solaris 10 Device Configuration Assistant diskette into the diskette drive (usually drive A). You use this diskette as the profile diskette.

x86 only – You can copy the Device Configuration Assistant software to a diskette from the Solaris 10 Operating System for x86 Platforms DVD or Solaris 10 Software for x86 Platforms - 2 CD by using the procedure in Appendix C, “Preparing to Boot From the Solaris 10 Device Configuration Assistant or the Network (Tasks),” in *Solaris 10 Installation Guide: Network-Based Installations*.

3. Mount the diskette.

```
# volcheck
```

4. Copy the image of the Solaris 10 Device Configuration Assistant to the system's hard disk.

```
# dd if=/vol/dev/aliases/floppy0 of=boot_image
```

In the command, *boot_image* is the name of the file into which you want to copy the image of the Solaris 10 Device Configuration Assistant. You can specify an absolute path name.

For example, the following command copies the boot diskette to a file that is named *boot_save*:

```
dd if=/vol/dev/aliases/floppy0 of=boot_save
```

5. Eject the diskette by clicking Eject Disk in the File Manager window or by typing `eject floppy` on the command line.
6. In the Removable Media Manager dialog box, click OK.
7. Manually eject the Solaris 10 Device Configuration Assistant diskette.
8. Insert a blank diskette or a diskette that can be overwritten into the diskette drive.
9. Mount the diskette.

```
# volcheck
```

10. Format the diskette.



Caution – Formatting erases all data on the diskette.

```
# fdformat -d -U
```

11. Copy the Solaris 10 Device Configuration Assistant image from the system's hard disk to the formatted diskette.

```
# dd if=boot_image of=/vol/dev/aliases/floppy0
```

In the command, *boot_image* is the name of the file where you want to copy the image of the Solaris 10 Device Configuration Assistant. You can specify an absolute path name.

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

- If no, go to [Step 15](#).
- If yes, use the following decision table to determine what to do next.

Example Locations	Instructions
The Solaris 10 Operating System for x86 Platforms DVD or the Solaris 10 Software for x86 Platforms - 1 CD	Insert the Solaris 10 Operating System for x86 Platforms DVD or the Solaris 10 Software for x86 Platforms - 1 CD into the server's CD-ROM drive. Volume Manager automatically mounts the CD.
An image of the Solaris 10 Operating System for x86 Platforms DVD or the Solaris 10 Software for x86 Platforms - 1 CD on a local disk	Change directory to the location of the Solaris 10 Operating System for x86 Platforms DVD or the Solaris 10 Software for x86 Platforms - 1 CD image. For example, type the following: <code>cd /export/install</code>

13. 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 10 Software for x86 Platforms - 1 CD to the root (/) directory on a profile diskette that is named `scrap`:

```
cp -r /cdrom/sol_10_x86/s2/Solaris_10/Misc/jumpstart_sample/* /floppy/scrap
```

14. Update the example JumpStart files on the profile diskette so that the files work in your environment.
15. Ensure that root owns the JumpStart directory and that permissions are set to 755.
16. Eject the diskette by clicking Eject Disk in the File Manager window or by typing `eject floppy` on the command line.
17. In the Removable Media Manager dialog box, click OK.
18. 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 56](#).

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 51](#) or [“Creating a Profile Server for Networked Systems” on page 47](#), 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

- Steps**
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 121](#).

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

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

<code>!</code>	A symbol that is used before a keyword to indicate negation.
<code>rule_keyword</code>	A predefined lexical unit or word that describes a general system attribute, such as host name, <code>hostname</code> , or memory size, <code>memsize</code> . <code>rule_keyword</code> 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 121 .
<code>rule_value</code>	A value that provides the specific system attribute for the corresponding rule keyword. Rule values are described in “Rule Keywords and Values” on page 121 .
<code>&&</code>	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.
<code>begin</code>	The name of an optional Bourne shell script that can be executed before the installation begins. If no <code>begin</code> script exists, you must type a minus sign (-) in this field. All <code>begin</code> scripts must be located in the JumpStart directory. Information about how to create <code>begin</code> scripts is presented in “Creating Begin Scripts” on page 77 .
<code>profile</code>	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 92 and “Creating Derived Profiles With a Begin Script” on page 78.

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 79.

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.

EXAMPLE 4-1 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
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.

EXAMPLE 4-1 rule File (Continued)

hostname	The rule matches if the system's host name is eng-1. The <code>basic_prof</code> 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 <i>not</i> a SPARCstation™ 20 (<code>SUNW, SPARCstation-20</code>). The <code>net_prof</code> profile is used to install the Solaris software on systems that match this rule. The rule also provides an example of rule wrap, which is defined in "Syntax of the rules File" on page 56.
model	The rule matches if the system is a SPARCstation LX. The <code>lx_prof</code> profile and the <code>complete</code> 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 193.168.2.0 and is an x86 based sun4u system. The <code>setup begin</code> script, the <code>x864u_prof</code> profile, and the <code>done</code> 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 <code>prog_prof</code> 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 <code>generic_prof</code> profile is used to install the Solaris software on systems that match the rule. If <code>any</code> 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 47](#)
 - [“Creating a Profile Diskette for Standalone Systems” on page 51](#)
-

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

- Steps**
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 125](#).

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 70 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 125 contains a description of profile keywords and values.

EXAMPLE 4-2 Mounting Remote File Systems and Adding and Deleting Packages

```
# profile keywords      profile values
# -----
install_type           initial_install
system_type            standalone
partitioning           default
filesystems            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 4-3 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.

<code>install_type</code>	The <code>install_type</code> keyword is required in every profile.
<code>system_type</code>	The <code>system_type</code> keyword defines that the system is to be installed as a standalone system.
<code>partitioning</code>	The file system slices are determined by the software to be installed with the value <code>default</code> . The size of swap is set to 512 Mbytes and is installed on any disk, value <code>any</code> .
<code>cluster</code>	The Developer Solaris Software Group, <code>SUNWCprog</code> , is installed on the system.
<code>package</code>	A third-party package is installed on the system located on an HTTP server.

EXAMPLE 4-4 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.

<code>partitioning</code>	The file system slices are determined by the <code>filesys</code> keywords, value <code>explicit</code> . The size of root (/) is based on the selected software, value <code>auto</code> , and is installed on <code>c0t0d0s0</code> . The size of swap is set to the necessary size and is installed on <code>c0t3d0s1</code> . <code>usr</code> is based on the selected software and the installation program determines where <code>usr</code> is installed, based on the value <code>any</code> .
<code>cluster</code>	The Entire Solaris Software Group, <code>SUNWCall</code> , is installed on the system.

EXAMPLE 4-5 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.

<code>install_type</code>	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.
<code>root_device</code>	The root file system on <code>c0t3d0s2</code> is upgraded.
<code>backup_media</code>	A remote system that is named <code>timber</code> is to be used to back up data during the disk space reallocation. For more <code>backup-media</code> keyword values, see “backup_media Profile Keyword” on page 132.
<code>package</code>	The binary compatibility package, <code>SUNWbcp</code> , is not installed on the system after the upgrade.
<code>package</code>	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.
<code>patch</code>	A list of patches that are installed with the upgrade. The patch list is located on an NFS server named <code>patch_master</code> under the directories <code>Solaris_10/patches</code> . In case of a mount failure, the NFS mount is tried five times.
<code>locale</code>	The German localization packages are to be installed on the system.

EXAMPLE 4-6 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
```

EXAMPLE 4-6 Reallocating Disk Space for an Upgrade (Continued)

```
package      SUNWxwman add
cluster     SUNWCacc add
locale      de
```

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

<code>install_type</code>	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.
<code>root_device</code>	The root file system on <code>c0t3d0s2</code> is upgraded.
<code>backup_media</code>	A remote system that is named <code>timber</code> is to be used to back up data during the disk space reallocation. For more <code>backup-media</code> keyword values, see “ backup_media Profile Keyword ” on page 132.
<code>layout_constraint</code>	The <code>layout_constraint</code> keywords designate that auto-layout can perform the following when auto-layout attempts to reallocate disk space for the upgrade. <ul style="list-style-type: none">■ 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.
<code>package</code>	The binary compatibility package, <code>SUNWbcp</code> , is not installed on the system after the upgrade.
<code>package</code>	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.
<code>locale</code>	The German localization packages are to be installed on the system.

EXAMPLE 4-7 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
```

EXAMPLE 4-7 Retrieving a Solaris Flash Archive From an HTTP Server (Continued)

```
filesys                c0t1d0s7 free /export/home
```

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

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

EXAMPLE 4-8 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.

<code>install_type</code>	The profile installs a Solaris Flash archive on the clone system. All files are overwritten as in an initial installation.
<code>archive_location</code>	The compressed Solaris Flash archive is retrieved from a secure HTTP server.
<code>partitioning</code>	The file system slices are determined by the <code>filesys</code> keywords, value <code>explicit</code> . 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 <code>c0t1d0s1</code> . <code>/export/home</code> is based on the remaining disk space. <code>/export/home</code> is installed on <code>c0t1d0s7</code> .

EXAMPLE 4-9 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
fileys                 c0t1d0s0 4000 /
fileys                 c0t1d0s1 512 swap
fileys                 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.

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

EXAMPLE 4-10 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.

<code>install_type</code>	The profile installs a Solaris Flash differential archive on the clone system. Only files that are specified by the archive are installed.
---------------------------	--

EXAMPLE 4-10 Retrieving a Solaris Flash Differential Archive From an NFS Server
(Continued)

<code>archive_location</code>	The Solaris Flash archive is retrieved from an NFS server.
<code>no_master_check</code>	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 4-11 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.

<code>partitioning</code>	The file system slices are determined by the <code>filesys</code> keywords, value <code>explicit</code> . The size of root (/) is based on the selected software, value <code>auto</code> , and is installed on <code>c0t0d0s0</code> . The size of swap is set to the necessary size and is installed on <code>c0t3d0s1</code> . <code>usr</code> is based on the selected software and the installation program determines where <code>usr</code> is installed, based on the value <code>any</code> .
<code>cluster</code>	The Entire Solaris Software Group, <code>SUNWCall</code> , is installed on the system.
<code>bootenv createbe</code>	An empty, inactive boot environment is set up on disk <code>c0t1d0</code> . File systems for root (/), swap, and <code>/export</code> 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.

EXAMPLE 4-11 Creating an Empty Boot Environment (Continued)

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 125.
- For background about using Solaris Live Upgrade that creates, upgrades, and activates inactive boot environments, see Chapter 4, “Solaris Live Upgrade (Overview),” in *Solaris 10 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 Installation Guide: Solaris Flash Archives (Creation and Installation)*.

EXAMPLE 4-12 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
filesystems          mirror:d30 c0t1d0s0 c0t0d0s0 /
filesystems          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. |
| filesystems | 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. |
| filesystems | 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). |

EXAMPLE 4-12 Creating a RAID-1 Volume to Mirror the Root File System (Continued)

- For overview information about how to create RAID-1 volumes during your installation, see [Chapter 10](#).
- For guidelines and requirements about creating RAID-1 volumes, see [Chapter 11](#).
- For descriptions of keyword values, see “[fileSYS Profile Keyword \(Creating RAID-1 Volumes\)](#)” on page 144 and “[metadb Profile Keyword \(Creating State Database Replicas\)](#)” on page 150.

EXAMPLE 4-13 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).

EXAMPLE 4-13 Creating RAID-1 Volumes to Mirror Multiple File Systems (Continued)

`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 10](#).
- For guidelines and requirements of creating mirrored file systems, see [Chapter 11](#).
- For descriptions of keyword values, see “`filesys` Profile Keyword (Creating RAID-1 Volumes)” on page 144 and “`metadb` Profile Keyword (Creating State Database Replicas)” on page 150.

EXAMPLE 4-14 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.

- | | |
|----------------------|---|
| <code>fdisk</code> | All <code>fdisk</code> partitions of type DOSOS16 (04 hexadecimal) are deleted from the <code>c0t0d0</code> disk. |
| <code>fdisk</code> | A Solaris <code>fdisk</code> partition is created on the largest contiguous free space on the <code>c0t0d0</code> disk. |
| <code>cluster</code> | The Entire Distribution software group, <code>SUNWCall</code> , is installed on the system. |
| <code>cluster</code> | The system accounting utilities, <code>SUNWCacc</code> , 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 86 and "x86: To Create a Disk Configuration File" on page 88.

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

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

For SPARC based systems:

- Solaris 10 Operating System for SPARC Platforms DVD
- Solaris 10 Software for SPARC Platforms - 1 CD

For x86 based systems:

- Solaris 10 Operating System for x86 Platforms DVD
- Solaris 10 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.

```
Solaris Web Start 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 72](#).

▼ To Test a Profile

- Steps**
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 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 to test an initial installation profile.	Create a temporary Solaris 10 environment to test the profile. For details, see “To Create a Temporary Solaris Environment to Test a Profile” on page 71 . 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.	<code>mount -F nfs server_name:path /tmp/mnt</code>
SPARC: Mount a UFS-formatted diskette.	<code>mount -F ufs /dev/diskette /tmp/mnt</code>
Mount a PCFS-formatted diskette.	<code>mount -F pcfs /dev/diskette /tmp/mnt</code>

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 86.

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 10 software image. You use this option, for example, if the system is using Volume Manager to mount the Solaris 10 Software - 1 CD for your platform.

Note – The `-c` option is not required if you booted from a Solaris 10 Operating System DVD or a Solaris 10 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 `pinstall` is being run, you must specify the path.

Profile Test Examples

The following example shows how to use `pinstall` 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 software is installed. The `basic_prof` profile is located in the `/jumpstart` directory, and the path to the Solaris 10 Software for SPARC Platforms - 1 CD or Solaris 10 Software for x86 Platforms - 1 CD image is specified because Volume Manager is being used.

EXAMPLE 4-15 Profile Test Using a Solaris 10 System

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

The following example shows how to use `pinstall` to test the profile that is named `basic_prof` on a Solaris 10 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 10 Software for SPARC Platforms - 1 CD or Solaris 10 Software for x86 Platforms - 1 CD image that is located in the `/export/install` directory.

EXAMPLE 4-16 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 4-2 describes what the check script does.

TABLE 4-2 What Happens When You Use the check Script

Stage	Description
1	The <code>rules</code> file is checked for syntax. check verifies that the rule keywords are legitimate and that the <i>begin</i> , <i>class</i> , and <i>finish</i> fields are specified for each rule. The <i>begin</i> and <i>finish</i> fields can consist of a minus sign (-) instead of a file name.
2	If no errors are found in the <code>rules</code> file, each profile that is specified in the rules is checked for syntax.
3	If no errors are found, check creates the <code>rules.ok</code> file from the <code>rules</code> 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

- Steps** 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 10 Operating System DVD or on the Solaris 10 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 10 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 10 Operating System DVD or a Solaris 10 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 5](#). You can learn about performing custom JumpStart installations in [Chapter 7](#).

Using Optional Custom JumpStart Features (Tasks)

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

- “Creating Begin Scripts” on page 77
- “Creating Finish Scripts” on page 79
- “Creating a Compressed Configuration File” on page 85
- “Creating Disk Configuration Files” on page 86
- “Using a Site-Specific Installation Program” on page 92

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 165.
- Save begin scripts in the JumpStart directory.

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 5-1 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 165](#).
- 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.

- Steps**
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 5–2 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 5–3](#) shows an example of a finish script that adds packages.
- [Example 5–4](#) shows an example of a finish script that adds patches.

EXAMPLE 5–3 Adding Packages With a Finish Script

```
#!/bin/sh

BASE=/a
MNT=/a/mnt
```

EXAMPLE 5-3 Adding Packages With a Finish Script (Continued)

```
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}
```

- 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 5-4 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.
```

EXAMPLE 5-4 Adding Patches With a Finish Script (Continued)

```
# 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
#
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 5-5](#) customizes the root environment by appending information to the `.cshrc` file in the root (`/`) directory.

EXAMPLE 5-5 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
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 5-6](#).

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 5-6 Setting the System's Root Password With a Finish Script

```
#!/bin/sh
#
# @(#)set_root_pw 1.4 93/12/23 SMI
```

EXAMPLE 5-6 Setting the System's Root Password With a Finish Script (Continued)

```
#
# 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" )
        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 5-1 Solaris Installation Options

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

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

Creating a Compressed Configuration File

Rather than 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 when you issue the `boot` command. As a result, you must compress all of the custom JumpStart configuration files into one file. The compressed configuration file can be one of the following types:

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

▼ To Create a Compressed Configuration File

Steps 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 5-7 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 `pinstall(1M)` from a single system to test profiles against different disk configurations.

▼ SPARC: To Create a Disk Configuration File

- Steps**
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/rdisk/device_name >disk_config_file
```

`/dev/rdisk/device_name` The device name of the system’s disk. *device_name* must be in the form *cwtXdys2* or *cXdys2*.

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

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

```
* /dev/rdisk/c0t0d0s2 partition map
...
* /dev/rdisk/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 5-8 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/rdisk/c0t3d0s2 >104_test
```

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

```
* /dev/rdisk/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
*
*
* Partition  Tag  Flags   First   Sector   Last   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 70 contains information about using disk configuration files to test profiles.

▼ x86: To Create a Disk Configuration File

Steps 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/rdisk/device_name
```

disk_config_file The name of a disk configuration file.

/dev/rdisk/device_name The device name of the `fdisk` layout of the entire disk. *device_name* must be in the form *cwtxdys0* or *cx dys0*.

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

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

/dev/rdisk/device_name The device name of the system's disk. *device_name* must be in the form *cwtxdys2* or *cx dys2*.

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

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

```
* /dev/rdisk/c0t0d0s2 partition map
...
* /dev/rdisk/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 5-9 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/rdisk/c0t0d0p0
```

The `500_test` file looks like the following:

```
* /dev/rdisk/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:  PCILOS
* 4:  DOSOS16
* 5:  EXTLOS
* 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/rdisk/c0t0d0s2 >>500_test
```

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

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

EXAMPLE 5-9 x86: Creating a Disk Configuration File (Continued)

```
*      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/rdisk/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
*
* Partition  Tag  Flags      First  Sector  Last
           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 70 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
```

Creating Custom Rule and Probe Keywords (Tasks)

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

- [“Probe Keywords” on page 93](#)
- [“Creating a custom_probes File” on page 94](#)
- [“Validating the custom_probes File” on page 97](#)

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 167](#).

Creating a custom_probes File

The rule and probe keywords that are described in “Rule Keywords and Values” on page 121 and “Probe Keywords and Values” on page 167 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

- Steps**
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 10 Operating System DVD or on the Solaris 10 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 6-1 custom_probes File

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

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 6-2 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 6-1 describes what the check script does.

TABLE 6-1 What Happens When You Use the check Script

Stage	Description
1	check searches for a <code>custom_probes</code> file.
2	If the file exists, check creates the <code>custom_probes.ok</code> file from the <code>custom_probes</code> 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: <code># version=2 checksum=num</code>

▼ To Validate the `custom_probes` File

- Steps** 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 10 Operating System DVD or on the Solaris 10 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 10 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 10 Operating System DVD or Solaris 10 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.

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 10 software.

- “SPARC: To Perform an Installation or Upgrade With the Custom JumpStart Program” on page 103
- “x86: To Perform an Installation or Upgrade With the Custom JumpStart Program” on page 106

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

TABLE 7-1 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 10 environment.	<i>Solaris 10 Sun Hardware Platform Guide</i> 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 2

TABLE 7-1 Task Map: Setting Up a System for a Custom JumpStart Installation
(Continued)

Task	Description	For Instructions
(Optional) Set system parameters.	You can preconfigure system information to avoid being prompted for the information during the installation or upgrade.	Chapter 4, "Preconfiguring System Configuration Information (Tasks)," in <i>Solaris 10 Installation Guide: Network-Based Installations</i>
Prepare the system for custom JumpStart installation.	Create and validate a rules file and profile files.	Chapter 4
(Optional) Prepare optional custom JumpStart features.	If you are using begin scripts, finish scripts, or other optional features, prepare the scripts or files.	Chapter 5 and Chapter 6
(Optional) Prepare to install the Solaris software from the network.	To install a system from a remote Solaris 10 Operating System DVD or Solaris 10 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 6, "Preparing to Install From the Network With DVD Media (Tasks)," in <i>Solaris 10 Installation Guide: Network-Based Installations</i> Chapter 7, "Preparing to Install From the Network With CD Media (Tasks)," in <i>Solaris 10 Installation Guide: Network-Based Installations</i>
(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 101
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 103

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 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 (Command-Line Interface)” in *Solaris 10 Installation Guide: Solaris Live Upgrade and Upgrade Planning*.

Steps 1. On the install server, create the custom JumpStart `rules` file.

For detailed instructions about creating custom JumpStart files, refer to [Chapter 4](#).

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

For examples of Solaris Flash archive profiles, see “[Profile Examples](#)” on page 61.

From the existing list of custom JumpStart keywords in [Table 9-2](#), the only keywords valid when you install a Solaris Flash archive are the following:

Keyword	Initial Installation	Differential Archive
(required) <code>archive_location</code>	X	X

Keyword	Initial Installation	Differential Archive
fdisk (x86 only)	X	X
filesystem	X	
Note – You cannot set the filesystem 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 127](#).

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” on page 152](#).**

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.**

3. **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 Installation Guide: Network-Based Installations*](#)

- “Adding Systems to Be Installed From the Network With a CD Image” in *Solaris 10 Installation Guide: Network-Based Installations*

4. 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 103.

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

- Steps**
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 10 software, insert the Solaris 10 Operating System for SPARC Platforms DVD or the Solaris 10 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 following command.

```
ok boot cdrom:net - install url:ask dhcp nowin
```

`cdrom` Specifies to boot from a CD or a DVD.

For a system with an older EEPROM, replace `cdrom` with `sd(0,6,2)` to boot from the system’s CD-ROM or DVD-ROM drive.

`net` Specifies to boot from an install server on the network.

`url` Specifies the location of the custom JumpStart files. You can specify a URL for files that are located in the following places:

- Local hard disk

```
file://jumpstart_dir_path/compressed_config_file
```

- NFS server

```
nfs://server_name:IP_address/jumpstart_dir_path/compressed_config_file
```

- 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 after the system boots and connects to the network.

If you bypass the prompt by pressing Return, the installation program interactively configures the network parameters. The installation program then prompts you for the location of the compressed configuration file. If you bypass the prompt by pressing Return, the Solaris installation program begins.

dhcp Specifies to use a DHCP server to obtain network installation information that is needed to boot the system.

If you do not specify to use a DHCP server, the system uses the `/etc/bootparams` file or the name service `bootparams` database.

nowin Specifies not to begin the X program. You do not need to use the X program to perform a custom JumpStart installation, so you can reduce the installation time by using the `nowin` option.

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

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

TABLE 7-2 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 software. This decision determines how to specify the <code>fdisk(1M)</code> keyword in the system's profile.	"x86: <code>fdisk</code> Profile Keyword" on page 139
Check if the system is supported.	Check the hardware documentation for system support in the Solaris 10 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 2
(Optional) Set system parameters.	You can preconfigure system information to avoid being prompted for the information during the installation or upgrade.	Chapter 4, "Preconfiguring System Configuration Information (Tasks)," in <i>Solaris 10 Installation Guide: Network-Based Installations</i>
Prepare the system for custom JumpStart installation.	Create and validate a <code>rules</code> file and profile files.	Chapter 4
(Optional) Prepare optional custom JumpStart features.	If you are using begin scripts, finish scripts, or other optional features, prepare the scripts or files.	Chapter 5 and Chapter 6

TABLE 7-2 x86: Task Map: Setting Up a System for a Custom JumpStart Installation
(Continued)

Task	Description	For Instructions
(Optional) Prepare to install the Solaris software from the network.	To install a system from a remote Solaris 10 Operating System for x86 Platforms DVD or Solaris 10 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 7, "Preparing to Install From the Network With CD Media (Tasks)," in <i>Solaris 10 Installation Guide: Network-Based Installations</i>
(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 101
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" on page 106

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 101.

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

- Steps**
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. **If you are using a profile diskette, insert the profile diskette into the system's diskette drive.**

Note – The profile diskette contains a copy of the Solaris 10 Device Configuration Assistant in addition to profile information. If you are using PXE network boot to boot the system over the network, you must configure your system so that the system boots from the network and not from the diskette.

4. **If you are using the system's DVD-ROM or CD-ROM drive to install the Solaris 10 software, insert the Solaris 10 Operating System for x86 Platforms DVD or the Solaris 10 Software for x86 Platforms - 1 CD in the drive.**
5. **Decide how to boot the system.**
 - If you boot from the Solaris 10 Operating System DVD or the Solaris 10 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.
 - If you boot from a Diskette – Use the profile diskette that you inserted into the drive in [Step 3](#) or insert the Solaris 10 Device Configuration Assistant diskette into the system's diskette drive.

x86 only – You can copy the Device Configuration Assistant software to a diskette from the Solaris 10 Operating System for x86 Platforms DVD or Solaris 10 Software for x86 Platforms - 2 CD by using the procedure in Appendix C, "Preparing to Boot From the Solaris 10 Device Configuration Assistant or the Network (Tasks)," in *Solaris 10 Installation Guide: Network-Based Installations*.

6. **If the system is off, turn the system on. If the system is on, reboot the system.**
The Device Configuration Assistant identifies the system's devices.
7. **On the Boot Solaris screen, select the device from which to boot the system.**
Select DVD, CD, Net, or Disk.

8. At the prompt, perform one of the following:

Select the type of installation you want to perform:

- 1 Solaris Interactive
- 2 Custom JumpStart

Enter the number of your choice followed by the <ENTER> key.

If you enter anything else, or if you wait for 30 seconds, an interactive installation will be started.

To select the custom JumpStart method, perform one of the following actions:

Note – If you do not type 2 or type a boot command within 30 seconds, the Solaris installation program begins. You can stop the timer by typing any key on the command line.

- Type 2 and press Enter.
- To specify the location of the custom JumpStart configuration files, type the following command.

```
b install url:ask dhcp nowin
```

url Specifies the location of the custom JumpStart files. You can specify a URL for files that are located in the following places:

- Local hard disk

```
file://jumpstart_dir_path/compressed_config_file
```

- NFS server

```
nfs://server_name:IP_address/jumpstart_dir_path/compressed_config_file
```

- 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 after the system boots and connects to the network.
- If you bypass the prompt by pressing Return, the installation program interactively configures the network parameters. The installation program then prompts you for the location of the compressed configuration file. If you bypass the prompt by pressing Return, the Solaris installation program begins.
- `dhcp` Specifies to use a DHCP server to obtain network installation information that is needed to boot the system.
- If you do not specify to use a DHCP server, the system uses the `/etc/bootparams` file or the name service `bootparams` database.
- `nowin` Specifies not to begin the X program. You do not need to use the X program to perform a custom JumpStart installation, so you can reduce the installation time by using the `nowin` option.

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

10. 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`

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.

- “Sample Site Setup” on page 111
- “Create an Install Server” on page 113
- “x86: Create a Boot Server for Marketing Systems” on page 114
- “Create a JumpStart Directory” on page 115
- “Share the JumpStart Directory” on page 115
- “SPARC: Create the Engineering Group’s Profile” on page 115
- “x86: Create the Marketing Group’s Profile” on page 116
- “Update the rules File” on page 117
- “Validate the rules File” on page 117
- “SPARC: Set Up Engineering Systems to Install From the Network” on page 118
- “x86: Set Up Marketing Systems to Install From the Network” on page 118
- “SPARC: Boot the Engineering Systems and Install Solaris 10 Software” on page 119
- “x86: Boot the Marketing Systems and Install Solaris 10 Software” on page 120

Sample Site Setup

Figure 8-1 shows the site setup for this example.

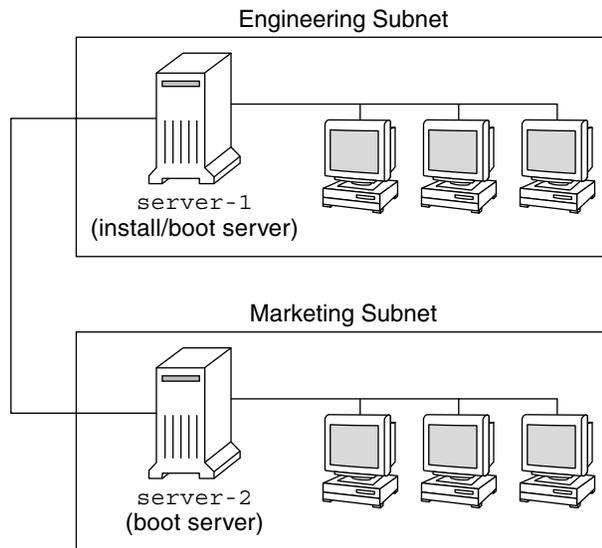


FIGURE 8-1 Sample Site Setup

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 software from the network.

Create an Install Server

Because the groups need to install Solaris 10 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 10 Software CDs and the Solaris 10 Languages CD
- Solaris 10 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 8-1 SPARC: Copying the Solaris 10 CDs

Insert the Solaris 10 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 10 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 10 Software you want to install.

Insert the SPARC: Solaris 10 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
```

EXAMPLE 8-2 x86: Copying the Solaris 10 CDs

Insert the Solaris 10 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
```

Insert the Solaris 10 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
```

EXAMPLE 8-2 x86: Copying the Solaris 10 CDs (Continued)

Repeat the previous command for each Solaris 10 Software you want to install.

Insert the Solaris 10 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
```

EXAMPLE 8-3 SPARC: Copying the Solaris 10 DVD

Insert the Solaris 10 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 8-4 x86: Copying the Solaris 10 DVD

Insert the Solaris 10 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 10 Operating System for x86 Platforms DVD or the Solaris 10 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 10 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 10 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 10 Operating System DVD image or from the Solaris 10 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 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.

<code>install_type</code>	The installation is to be treated as an initial installation, as opposed to an upgrade.
<code>system_type</code>	The engineering systems are standalone systems.
<code>partitioning</code>	The JumpStart software uses default disk partitioning for installing Solaris software on the engineering systems.
<code>cluster</code>	The Developer System Support software group is to be installed.
<code>filesys</code>	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 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.

<code>install_type</code>	The installation is to be treated as an initial installation, as opposed to an upgrade.
<code>system_type</code>	The marketing systems are standalone systems.
<code>partitioning</code>	The JumpStart software is to use default disk partitioning for installing Solaris on the marketing systems.
<code>cluster</code>	The End User Solaris Software Group is to be installed.
<code>package</code>	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 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 software by using the `eng_prof` profile. The systems on the 255.222.44.0 network are to be installed with the Solaris 10 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 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 9-1](#) 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 10 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.
- 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 10 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:

<code>-d</code>	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. <code>-d</code> is optional for network boot clients that do not use PXE network boot.
<code>-s</code>	Specifies the install server (<code>server-1</code>) and the path to the Solaris 10 software (<code>/export/install/x86_10</code>).
<code>-c</code>	Specifies the server (<code>server-1</code>) and path (<code>/jumpstart</code>) to the JumpStart directory.
<code>host-mkt1</code>	The name of a system in the marketing group.
<code>host-mkt2</code>	The name of another system in the marketing group.
<code>sun4u</code>	Specifies the platform group of the systems that use <code>server-1</code> as an install server. The platform group is for Ultra 5 systems.
<code>SUNW.i86pc</code>	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.
<code>i86pc</code>	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 10 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 10 Software

You can boot the system from one of the following:

- Solaris 10 Software for x86 Platforms - 1 CD
- Solaris 10 Operating System for x86 Platforms DVD
- The network by using PXE network boot
- The profile diskette
- The Solaris 10 Device Configuration Assistant diskette

Solaris 10 is automatically installed on the marketing group's systems.

Custom JumpStart (Reference)

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

- “Rule Keywords and Values” on page 121
- “Profile Keywords and Values” on page 125
- “Custom JumpStart Environment Variables” on page 165
- “Probe Keywords and Values” on page 167

Rule Keywords and Values

Table 9-1 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 56.

TABLE 9-1 Descriptions of Rule Keywords and Values

Keyword	Value	Matches
any	minus sign (-)	Anything. The any keyword always succeeds.
arch	<i>processor_type</i> Valid values for <i>processor_type</i> are the following: <ul style="list-style-type: none">■ SPARC: <code>sparc</code>■ x86: <code>i386</code>	A system’s processor type. The <code>uname -p</code> command reports the system’s processor type.

TABLE 9-1 Descriptions of Rule Keywords and Values (Continued)

Keyword	Value	Matches
disksize	<p><i>actual_disk_name size_range</i></p> <p><i>actual_disk_name</i> – A disk name in the form <i>cxydz</i>, such as <code>c0t3d0</code> or <code>c0d0</code>, or the special word <code>rootdisk</code>. If <code>rootdisk</code> is used, the disk to be matched is determined in the following order:</p> <ul style="list-style-type: none"> ■ SPARC: The disk that contains the preinstalled boot image, which is a new SPARC based system with factory JumpStart installed ■ The <code>c0t3d0s0</code> disk, if the disk exists ■ The first available disk that is searched in kernel probe order <p><i>size_range</i> – The size of the disk, which must be specified as a range of Mbytes (<i>x-x</i>).</p> <p>Note – When calculating <i>size_range</i>, 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 <i>size_range</i> equal to 530–550.</p>	<p>The name and size of a system’s disk in Mbytes.</p> <p>Example:</p> <pre>disksize c0t3d0 250-300</pre> <p>In the example, the JumpStart program attempts to match a system disk that is named <code>c0t3d0</code>. The disk can hold between 250 and 300 Mbytes of information.</p> <p>Example:</p> <pre>disksize rootdisk 750-1000</pre> <p>In the example, the JumpStart program attempts to match a disk in the following order:</p> <ol style="list-style-type: none"> 1. A system disk that contains a preinstalled boot image 2. The <code>c0t3d0s0</code> disk, if the disk exists 3. The first available disk that can hold between 750 Mbytes and 1 Gbyte of information
domainname	<i>actual_domain_name</i>	<p>A system’s domain name, which controls how a name service determines information.</p> <p>If you have a system already installed, the <code>domainname</code> command reports the system’s domain name.</p>
hostaddress	<i>actual_IP_address</i>	A system’s IP address.
hostname	<i>actual_host_name</i>	<p>A system’s host name.</p> <p>If you have a system that is already installed, the <code>uname -n</code> command reports the system’s host name.</p>

TABLE 9-1 Descriptions of Rule Keywords and Values (Continued)

Keyword	Value	Matches
installed	<p><i>slice version</i></p> <p><i>slice</i> – A disk slice name in the form <i>cwtxdysz</i>, such as <i>c0t3d0s5</i>, or the special words <i>any</i> or <i>rootdisk</i>. If <i>any</i> is used, the JumpStart program attempts to match all of the system’s disks in kernel probe order. If <i>rootdisk</i> is used, the disk to be matched is determined in the following order:</p> <ul style="list-style-type: none"> ■ SPARC: The disk that contains the preinstalled boot image, which is a new SPARC based system with factory JumpStart installed ■ The <i>c0t3d0s0</i> disk, if the disk exists ■ The first available disk that is searched in kernel probe order <p><i>version</i> – A version name or the special words <i>any</i> or <i>upgrade</i>. If <i>any</i> is used, any Solaris or SunOS release is matched. If <i>upgrade</i> is used, any Solaris 7 or compatible release that can be upgraded is matched.</p> <p>If the JumpStart program finds a Solaris release but is unable to determine the version, the version that is returned is <i>SystemV</i>.</p>	<p>A disk that has a root (/) file system that corresponds to a particular version of Solaris software.</p> <p>Example:</p> <pre>installed c0t3d0s1 Solaris_10</pre> <p>In the example, the JumpStart program attempts to match a system that has a Solaris root (/) file system on <i>c0t3d0s1</i>.</p>
karch	<p><i>actual_platform_group</i></p> <p>Valid values are <i>sun4u</i>, <i>i86pc</i>, and <i>prep</i>. A list of systems and their corresponding platform group is presented in the <i>Solaris 10 Sun Hardware Platform Guide</i> at http://docs.sun.com.</p>	<p>A system’s platform group.</p> <p>If you have a system that is already installed, the <i>arch -k</i> command or the <i>uname -m</i> command reports the system’s platform group.</p>
memsize	<p><i>physical_mem</i></p> <p>The value must be a range of Mbytes, <i>x-x</i>, or a single Mbyte value.</p>	<p>A system’s physical memory size in Mbytes.</p> <p>Example:</p> <pre>memsize 64-128</pre> <p>The example tries to match a system with a physical memory size between 64 and 128 Mbytes.</p> <p>If you have a system that is already installed, the output of the <i>prtconf</i> command, line 2, reports the system’s physical memory size.</p>

TABLE 9-1 Descriptions of Rule Keywords and Values (Continued)

Keyword	Value	Matches
model	<i>actual_platform_name</i>	<p>A system's platform name. See the <i>Solaris 10 Sun Hardware Platform Guide</i> at http://docs.sun.com for a list of valid platform names.</p> <p>To find the platform name of an installed system, use the <code>uname -i</code> command or the output of the <code>prtconf</code> command, line 5.</p> <p>Note – If the <i>actual_platform_name</i> contains spaces, you must replace spaces with underscores (_).</p> <p>Example:</p> <pre>SUNW,Sun_4_50</pre>
network	<i>network_num</i>	<p>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.</p> <p>Example:</p> <pre>network 192.168.2.0</pre> <p>The example tries to match a system with a 192.168.2.8 IP address, if the subnet mask is 255.255.255.0.</p>
osname	<i>Solaris_x</i>	<p>A version of Solaris software that is already installed on a system.</p> <p>Example:</p> <pre>osname Solaris_10</pre> <p>In the example, the JumpStart program attempts to match a system with the Solaris 10 OS already installed.</p>

TABLE 9-1 Descriptions of Rule Keywords and Values (Continued)

Keyword	Value	Matches
probe	<i>probe_keyword</i>	<p>A valid probe keyword or a valid custom probe keyword.</p> <p>Example:</p> <pre>probe disks</pre> <p>The example returns the size of a system's disks in Mbytes and in kernel probe order, for example, <code>c0t3d0s1, c0t4d0s0</code>, on a SPARC based system. The JumpStart program sets the <code>SI_DISKLIST</code>, <code>SI_DISKSIZE</code>, <code>SI_NUMDISKS</code>, and <code>SI_TOTALDISK</code> environment variables.</p> <p>Note – The <code>probe</code> keyword is unique in that the keyword does not attempt to match an attribute and run a profile. The <code>probe</code> keyword returns a value. Consequently, you cannot specify begin scripts, profiles, and finish scripts with the <code>probe</code> rule keyword.</p> <p>Probe keywords are described in Chapter 6.</p>
totaldisk	<p><i>size_range</i></p> <p>The value must be specified as a range of Mbytes (<i>x-x</i>).</p> <p>Note – When calculating <i>size_range</i>, 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 <i>size_range</i> equal to 530-550.</p>	<p>The total disk space on a system in Mbytes. The total disk space includes all the operational disks that are attached to a system.</p> <p>Example:</p> <pre>totaldisk 300-500</pre> <p>In the example, the JumpStart program tries to match a system with a total disk space between 300 and 500 Mbytes.</p>

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 59](#).

Profile Keywords Quick Reference

Table 9-2 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.

TABLE 9-2 Profile Keywords Overview

Profile Keyword	Installation Scenarios				
	Standalone System (Nonnetworked)	Standalone System (Networked) or Server	OS Server	Upgrade	Upgrade With Disk Space Reallocation
archive_location (installing Solaris Flash archives)	✓	✓			
backup_media					✓
boot_device	✓	✓	✓		
bootenv createbe	✓	✓	✓		
client_arch			✓		
client_root			✓		
client_swap			✓		
cluster (adding software groups)	✓	✓	✓		
cluster (adding or deleting clusters)	✓	✓	✓	✓	✓
dontuse	✓	✓	✓		
fdisk (x86 only)	✓	✓	✓		
filesystem (mounting remote file systems)		✓	✓		
filesystem (creating local file systems)	✓	✓	✓		
filesystem (creating mirrored file systems)	✓	✓	✓		
forced_deployment (installing Solaris Flash differential archives)	✓	✓			
geo	✓	✓	✓	✓	✓
install_type	✓	✓	✓	✓	✓

TABLE 9-2 Profile Keywords Overview (Continued)

Profile Keyword	Installation Scenarios				
	Standalone System (Nonnetworked)	Standalone System (Networked) or Server	OS Server	Upgrade	Upgrade With Disk Space Reallocation
layout_constraint					✓
local_customization (installing Solaris Flash archives)	✓	✓			
locale	✓	✓	✓	✓	✓
metadb (creating state database replicas)	✓	✓	✓		
no_master_check (installing Solaris Flash differential archives)	✓	✓			
no_content_check (installing Solaris Flash differential archives)	✓	✓			
num_clients			✓		
package	✓	✓	✓	✓	✓
partitioning	✓	✓	✓		
patch	✓	✓	✓	✓	✓
root_device	✓	✓	✓	✓	✓
system_type	✓	✓	✓		
usedisk	✓	✓	✓		

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 128

- “Archive Stored on an HTTP or HTTPS Server” on page 128
- “Archive Stored on an FTP Server” on page 129
- “Archive Stored on a Local Tape” on page 130
- “Archive Stored on a Local Device” on page 131
- “Archive Stored on a Local File” on page 132

location Specifics for locations are noted in the following sections.

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 9-1 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.

<i>path</i>	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.
<i>filename</i>	The name of the Solaris Flash archive file.
<i>optional_keywords</i>	The optional keywords that you can specify when you retrieve a Solaris Flash archive from an HTTP server.

TABLE 9-3 Optional Keywords to Use With `archive_location` HTTP

Keyword	Value Definition
<code>auth basic user_name password</code>	<p>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.</p> <p>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.</p>
<code>timeout min</code>	<p>The <code>timeout</code> 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 <code>timeout</code> value of 0 (zero), the connection is not reopened.</p> <ul style="list-style-type: none"> ■ 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.
<code>proxy host:port</code>	The <code>proxy</code> 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 <code>proxy</code> keyword.

EXAMPLE 9-2 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
```

<i>user_name:password</i>	The user name and password that you need to access the FTP server in the profile file.
<i>server_name</i>	The name of the server where you stored the archive.
<i>port</i>	A is an optional port. <i>port</i> 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.
<i>path</i>	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.
<i>filename</i>	The name of the Solaris Flash archive file.
<i>optional_keywords</i>	The optional keywords that you can specify when you retrieve a Solaris Flash archive from an FTP server.

TABLE 9-4 Optional Keywords to Use With `archive_location` FTP

Keyword	Value Definition
<code>timeout min</code>	The <code>timeout</code> 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 <code>timeout</code> value of 0 (zero), the connection is not reopened. <ul style="list-style-type: none"> ■ 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.
<code>proxy host:port</code>	The <code>proxy</code> 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 <code>proxy</code> keyword.

EXAMPLE 9-3 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 9-4 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 9-5 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
```

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 10 Software - 1 CD or from the Solaris 10 Operating System DVD. The system cannot access `/net` or any other automounted directory when it is booted from the Solaris 10 Software - 1 CD or from the Solaris 10 Operating System DVD.

filename The name of the Solaris Flash archive file.

EXAMPLE 9-6 Archive Stored on a Local File

```
archive_location local_file /archives/usrarchive
```

backup_media Profile Keyword

```
backup_media type path
```

Note – 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 <i>type</i> Value	Valid <i>path</i> Value	Specification
local_tape	/dev/rmt/ <i>n</i>	A local tape drive on the system that is being upgraded. <i>path</i> must be the character (raw) device path for the tape drive. <i>n</i> is the number of the tape drive.
local_diskette	/dev/rdiskette <i>n</i>	A local diskette drive on the system that is being upgraded. <i>path</i> must be the character (raw) device path for the diskette drive. <i>n</i> is the number of the diskette drive. Diskettes that you use for the backup must be formatted.
local_filesystem	/dev/dsk/ <i>cwt.xdysz</i> <i>/file_system</i>	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. <i>path</i> can be a block device path for a disk slice. For example, the <i>tx</i> in /dev/dsk/ <i>cwt.xdysz</i> might not be needed. Or, <i>path</i> can be the absolute path to a file system that is mounted by the <i>/etc/vfstab</i> file.
remote_filesystem	<i>host</i> : <i>/file_system</i>	An NFS file system on a remote system. <i>path</i> must include the name or IP address of the remote system, <i>host</i> , and the absolute path to the NFS file system, <i>file_system</i> . The NFS file system must have read/write access.
remote_system	<i>user@host</i> : <i>/directory</i>	A directory on a remote system that can be reached by a remote shell, <i>rsh</i> . The system that is being upgraded must have access to the remote system through the remote system's <i>.rhosts</i> file. <i>path</i> must include the name of the remote system <i>host</i> and the absolute path to the directory <i>directory</i> . If a user login ID <i>user</i> is not specified, <i>root</i> is used by default.

EXAMPLE 9-7 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

`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: `cwtxdysz` or `cxdysz` The disk slice where the JumpStart program places the root (/) file system, for example, `c0t0d0s0`.

x86: `cwtxdy` or `cxdy` 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 9-8 boot_device Profile Keyword

```
boot_device c0t0d0s2 update
```

bootenv createbe Profile Keyword

```
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/cwtxdysz*.
- *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 4-11
For background about using Solaris Live Upgrade that creates, upgrades, and activates inactive boot environments	Chapter 4, “Solaris Live Upgrade (Overview),” in <i>Solaris 10 Installation Guide: Solaris Live Upgrade and Upgrade Planning</i>
For background about using a Solaris Flash archive	Chapter 1, “Solaris Flash (Overview),” in <i>Solaris 10 Installation Guide: Solaris Flash Archives (Creation and Installation)</i>

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 10 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 9-9 `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 9-5](#) shows how the size of swap is determined during a custom JumpStart installation.

TABLE 9-5 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 9-5](#).

Note – Physical memory plus swap space must total a minimum of 32 Mbytes.

`cluster` Profile Keyword (Adding Software Groups)

```
cluster group_name
```

`cluster` designates the software group to add to the system. 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” on page 27](#).

cluster Profile Keyword (Adding or Deleting Clusters)

`cluster cluster_name add_delete_switch`

`cluster` designates whether a cluster is to be added or deleted from the software group that is to be installed on the system.

<i>cluster_name</i>	The name of the cluster that must be in the form <code>SUNWCname</code> .
<i>add_delete_switch</i>	An optional keyword that indicates whether to add or delete the cluster that is specified. Use the value <code>add</code> or <code>delete</code> . If you do not specify <code>add</code> or <code>delete</code> , <code>add</code> 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.

dontuse Profile Keyword

`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 `cxydz` or `cydz`, for example, `c0t0d0`.

Note – You cannot specify the `dontuse` keyword and the `usedisk` keyword in the same profile.

x86: fdisk Profile Keyword

`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:

- `cxydz` or `cydz` – A specific disk, for example, `c0t3d0`.
- `rootdisk` – The variable that contains the value of 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 163.
- `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`.

- `0xHH` – 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	<i>DDD</i>	<i>HH</i>
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)

`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.

<i>server</i>	The name of the server where the remote file system is located, followed by a colon.
<i>path</i>	The remote file system's mount-point name. For example, <code>/usr</code> or <code>/export/home</code>
<i>server_address</i>	The IP address of the server that is specified in <i>server:path</i> . If a name service is not running on the network, the <i>server_address</i> value can be used to populate the <code>/etc/hosts</code> 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 name 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 9-10 `filesys` 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.

<i>slice</i>	Use one of the following values:	
	any	The JumpStart program places the file system on any disk.
	<i>cwtxdysz</i> or <i>cxdysz</i>	The disk slice where the JumpStart program places the file system, for example, <code>c0t0d0s0</code> or <code>c0d0s0</code> .
	<i>rootdisk.sn</i>	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

Note – You cannot specify any when *size* is `existing`, `all`, `free`, `start:size`, or `ignore`.

163. The *sn* suffix indicates a specific slice on the disk.

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 *cwt.xdysz*. 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 `cwtxdysz`.

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).

filesystem Profile Keyword (Creating RAID-1 Volumes)

`filesystem 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.

<i>name</i>	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, <code>d100</code> . 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” on page 182.
<i>slice</i>	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 <code>cwtxdysz</code> , for example <code>c0t0d0s0</code> or <code>c0t0d0s5</code> . 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.
<i>size</i>	This value specifies the size, in Mbytes, of the file system.
<i>file_system</i>	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 <code>root (/)</code> , <code>/usr</code> , and <code>/var</code> , you can also specify swap as the file system.
<i>optional_parameters</i>	One or more mount options, which is the same as the <code>-o</code> option of the <code>mount(1M)</code> command. The mount options are added to the <code>/etc/vfstab</code> entry for the specified <i>file_system</i> . If you need to specify more than one mount option, the mount options must be separated by commas and no spaces, for example, <code>ro,quota</code> .

For more information about creating mirrored file systems during your installation, see [Chapter 10](#).

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

`geo locale`

`geo` designates the regional locale or locales that you want to install on a system or to add when upgrading a system. Values you can specify for `locale` are listed in the following table.

Value	Description
<code>N_Africa</code>	Northern Africa, including Egypt
<code>C_America</code>	Central America, including Costa Rica, El Salvador, Guatemala, Mexico, Nicaragua, Panama
<code>N_America</code>	North America, including Canada, United States
<code>S_America</code>	South America, including Argentina, Bolivia, Brazil, Chile, Colombia, Ecuador, Paraguay, Peru, Uruguay, Venezuela
<code>Asia</code>	Asia, including Japan, Republic of Korea, People's Republic of China, Taiwan, Thailand
<code>Ausi</code>	Australasia, including Australia, New Zealand
<code>C_Europe</code>	Central Europe, including Austria, Czech Republic, Germany, Hungary, Poland, Slovakia, Switzerland
<code>E_Europe</code>	Eastern Europe, including Albania, Bosnia, Bulgaria, Croatia, Estonia, Latvia, Lithuania, Macedonia, Romania, Russia, Serbia, Slovenia, Turkey
<code>N_Europe</code>	Northern Europe, including Denmark, Finland, Iceland, Norway, Sweden
<code>S_Europe</code>	Southern Europe, including Greece, Italy, Portugal, Spain

Value	Description
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

`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*:

<code>initial_install</code>	Specifies to perform an initial installation of the Solaris OS
<code>upgrade</code>	Specifies to perform an upgrade of the Solaris OS
<code>flash_install</code>	Specifies to install a Solaris Flash archive that overwrites all files
<code>flash_update</code>	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*

Note – You can use `layout_constraint` only for the upgrade option when you need to reallocate disk space.

`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
If you do not specify the <code>layout_constraint</code> keyword	The JumpStart program lays out the disk as follows: <ul style="list-style-type: none"> ■ 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 <code>/etc/vfstab</code> file are marked changeable. ■ Remaining file systems are marked fixed because auto-layout cannot change the file systems.
If you specify one or more <code>layout_constraint</code> keywords	The JumpStart program lays out the disk as follows: <ul style="list-style-type: none"> ■ File systems that require more space for the upgrade are marked changeable. ■ File systems for which you specified a <code>layout_constraint</code> 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 <code>layout_constraint</code> keyword to change the <code>minimum_size</code> 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.

<i>slice</i>	Specifies the file system's disk slice on which to specify the constraint. You must specify the system's disk slice in the form <code>cwtxdysz</code> or <code>cxdysz</code> .		
<i>constraint</i>	Use one of the following constraints for the specified file system: <table> <tbody> <tr> <td><code>changeable</code></td> <td>Auto-layout can move the file system to another location and it can change the file system size. The <code>changeable</code> constraint can only be specified on</td> </tr> </tbody> </table>	<code>changeable</code>	Auto-layout can move the file system to another location and it can change the file system size. The <code>changeable</code> constraint can only be specified on
<code>changeable</code>	Auto-layout can move the file system to another location and it can change the file system size. The <code>changeable</code> 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.

<code>movable</code>	Auto-layout can move the file system to another slice on the same disk or different disk. The file system size remains the same.
<code>available</code>	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 <code>available</code> constraint can only be specified on file systems that are not mounted by the <code>/etc/vfstab</code> file.
<code>collapse</code>	Auto-layout moves and collapses the specified file system into the parent file system. You can use the <code>collapse</code> option to reduce the number of file systems on a system as part of the upgrade. For example, if a system has the <code>/usr</code> and <code>/usr/share</code> file systems, collapsing the <code>/usr/share</code> file system moves the file system into <code>/usr</code> , the parent file system. You can specify the <code>collapse</code> constraint only on file systems that are mounted by the <code>/etc/vfstab</code> 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 9-11 `layout_constraint` Profile Keyword

```
layout_constraint c0t3d0s1 changeable 200
```

```
layout_constraint c0t3d0s4 movable
```

EXAMPLE 9-11 layout_constraint Profile Keyword (Continued)

```
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 Installation Guide: Solaris Flash Archives (Creation and Installation)*.

locale Profile Keyword

```
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 `local` 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.

<code>slice</code>	You must specify the disk slice on which you want the custom JumpStart program to place the state database replica. The <code>slice</code> value must follow the format <code>cwtxdysz</code> .
<code>size size-in-blocks</code>	The <code>size</code> optional keyword enables you to specify the size, in blocks, of the state database replica to be created. If you do not specify a <code>size</code> value, the custom JumpStart program uses a default size of 8192 blocks for the state database replica.
<code>count number-of-replicas</code>	You can specify the number of state database replicas you are creating by setting the optional <code>count</code> keyword value in your profile. If you do not specify a <code>count</code> 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”](#) on page 180.

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 101.

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 101.

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

`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 `SUNWname`. 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 SUNWname. 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 9–12 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
```

<i>package_name</i>	Specifies the package name in the form <i>SUNWname</i> . To view detailed information about packages and their names, on an installed system, use the <code>pkginfo -l</code> command.
<i>server_name</i>	Specifies the name of the server where you stored the package.
<i>port</i>	Specifies an optional port. <i>port</i> 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.
<i>path</i>	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.
<i>optional_keywords</i>	Specifies the optional keywords to use when you retrieve a package from an HTTP server.

TABLE 9-6 Optional package Keywords to Use With HTTP

Keyword	Value Definition
<i>timeout min</i>	The <code>timeout</code> 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 <code>timeout</code> 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.
<i>proxy host:port</i>	The <code>proxy</code> 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 <code>proxy</code> keyword.

EXAMPLE 9-13 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 9-14 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
```

<i>package_name</i>	Specifies the package name in the form <code>SUNWname</code> . To view detailed information about packages and their names, on an installed system, use the <code>pkginfo -l</code> command.
<i>device</i>	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 <code>/dev/dsk/</code> to the path.
<i>path</i>	Specifies the path to the Solaris package, relative to the root (<code>/</code>) file system on the device you specified.
<i>file_system_type</i>	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 9-15 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 9-16 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 `SUNWname`. 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 10 Software - 1 CD or from the Solaris 10 Operating System DVD. The system cannot access `/net` when it is booted from the Solaris 10 Software - 1 CD or from the Solaris 10 Operating System DVD.

EXAMPLE 9-17 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:

<code>default</code>	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 <code>filesys</code> keywords. <code>rootdisk</code> is selected first. The JumpStart program uses additional disks if the specified software does not fit on <code>rootdisk</code> .
----------------------	--

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` 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_directory [retry n]
```

<i>patch_id_list</i>	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.
<i>patch_file</i>	A file with a list of patches that is found in the <i>patch_location</i> . The patches are installed in the order specified in the file.
<i>server_name</i>	Specifies the name of the server where you stored the patches.
<i>patch_directory</i>	Specifies the location of the patch directory on the specified server. The patches must be in standard patch format.
<i>retry n</i>	Is an optional keyword. <i>n</i> is the maximum number of times the install utility attempts to mount the directory.

EXAMPLE 9-18 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 9-19 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
```

<i>patch_id_list</i>	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.
----------------------	--

<i>patch_file</i>	A file with a list of patches that is found in the <i>patch_location</i> . The patches are installed in the order specified in the file.
<i>server_name</i>	Specifies the name of the server where you stored the patch.
<i>port</i>	Specifies an optional port. <i>port</i> 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.
<i>patch_directory</i>	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.
<i>optional_keywords</i>	Specifies the optional keywords to use when you retrieve a patch from an HTTP server.

TABLE 9-7 Optional patch Keywords to Use With HTTP

Keyword	Value Definition
<i>timeout min</i>	The <i>timeout</i> 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 <i>timeout</i> 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.
<i>proxy host:port</i>	The <i>proxy</i> 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 <i>proxy</i> keyword.

EXAMPLE 9-20 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 9-21 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
```

<i>patch_id_list</i>	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.
<i>patch_file</i>	A file with a list of patches that is found in the <i>patch_location</i> . The patches are installed in the order specified in the file.
<i>device</i>	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 <code>/dev/dsk/</code> to the path.
<i>path</i>	Specifies the path to the Solaris patch, relative to the root (<code>/</code>) file system on the device you specified.
<i>file_system_type</i>	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 9-22 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 `/Solaris10/patches` from the local device `c0t6d0s0`. The patch file determines the order of patches to be installed.

```
patch patch_cal_device c0t6d0s0 /Solaris_10/patches
```

EXAMPLE 9-23 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 package 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 10 Software - 1 CD or from the Solaris 10 Operating System DVD. The system cannot access `/net` when it is booted from the Solaris 10 Software - 1 CD or from the Solaris 10 Operating System DVD.

EXAMPLE 9-24 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 9-25 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 `package` 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.

root_device Profile Keyword

`root_device slice`

`root_device` designates the system's root disk. "How the System's Root Disk Is Determined" on page 163 contains additional information.

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 `cwt.xdysz` or `cxdysz`.

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 9-26 `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 9-8 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 9-8 How JumpStart Determines a System's Root Disk (Initial Installation)

Stage	Action
1	If the <code>root_device</code> keyword is specified in the profile, the JumpStart program sets <code>rootdisk</code> to the root device.

TABLE 9-8 How JumpStart Determines a System's Root Disk (Initial Installation)
(Continued)

Stage	Action
2	If <code>rootdisk</code> is not set and the <code>boot_device</code> keyword is specified in the profile, the JumpStart program sets <code>rootdisk</code> to the boot device.
3	If <code>rootdisk</code> is not set and a <code>filesys cwtxdysz size /</code> entry is specified in the profile, the JumpStart program sets <code>rootdisk</code> to the disk that is specified in the entry.
4	If <code>rootdisk</code> is not set and a <code>rootdisk.sn</code> 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 <code>rootdisk</code> to the found disk.
5	If <code>rootdisk</code> is not set and <code>partitioning existing</code> 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 <code>rootdisk</code> to the found disk.
6	If <code>rootdisk</code> is not set, the JumpStart program sets <code>rootdisk</code> 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

`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 `cxydz` or `cydz`, 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_DISKSIZE`, 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 9-9 describes these variables and their values.

TABLE 9-9 Installation Environment Variables

Environment Variable	Value
<code>SI_ARCH</code>	The hardware architecture of the install client. The <code>SI_ARCH</code> variable is set when the <code>arch</code> keyword is used in the <code>rules</code> file.
<code>SI_BEGIN</code>	The name of the begin script, if one is used.
<code>SI_CLASS</code>	The name of the profile that is used to install the install client.
<code>SI_DISKLIST</code>	A comma-separated list of disk names on the install client. The <code>SI_DISKLIST</code> variable is set when the <code>disksize</code> keyword is used and matched in the <code>rules</code> file. The <code>SI_DISKLIST</code> and <code>SI_NUMDISKS</code> variables are used to determine the physical disk to use for the <code>rootdisk</code> . <code>rootdisk</code> is described in “How the System’s Root Disk Is Determined” on page 163.
<code>SI_DISKSIZE</code>	A comma-separated list of disk sizes on the install client. The <code>SI_DISKSIZE</code> variable is set when the <code>disksize</code> keyword is used and matched in the <code>rules</code> file.
<code>SI_DOMAINNAME</code>	The domain name. The <code>SI_DOMAINNAME</code> variable is set when the <code>domainname</code> keyword is used and matched in the <code>rules</code> file.
<code>SI_FINISH</code>	The name of the finish script, if one is used.

TABLE 9-9 Installation Environment Variables (Continued)

Environment Variable	Value
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 163.
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 10 Operating System DVD or the Solaris 10 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.
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 9–10 describes each rule keyword and its equivalent probe keyword.

Note – Always place probe keywords at or near the beginning of the rules file.

TABLE 9–10 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 <code>SI_ARCH</code> .
disksize	disks	Returns the size of a system's disks in Mbytes in kernel probe order, c0t3d0s0, c0t3d0s1, c0t4d0s0. <code>disksize</code> sets <code>SI_DISKLIST</code> , <code>SI_DISKSIZE</code> , <code>SI_NUMDISKS</code> , and <code>SI_TOTALDISK</code> .
domainname	domainname	Returns a system's NIS or NIS+ domain name or blank and sets <code>SI_DOMAINNAME</code> . The <code>domainname</code> keyword returns the output of <code>domainname(1M)</code> .
hostaddress	hostaddress	Returns a system's IP address, the first address that is listed in the output of <code>ifconfig(1M) -a</code> that is not <code>lo0</code> , and sets <code>SI_HOSTADDRESS</code> .
hostname	hostname	Returns a system's host name that is the output from <code>uname(1) -n</code> and sets <code>SI_HOSTNAME</code> .
installed	installed	Returns the version name of the Solaris OS that is installed on a system and sets <code>SI_ROOTDISK</code> and <code>SI_INSTALLED</code> . If the JumpStart program finds a Solaris release but is unable to determine the version, the version that is returned is <code>SystemV</code> .
karch	karch	Returns a system's platform group, for example <code>i86pc</code> or <code>sun4u</code> , and sets <code>SI_KARCH</code> . For a list of platform names, see <i>Solaris 10 Sun Hardware Platform Guide</i> at http://docs.sun.com .
memsize	memsize	Returns the size of physical memory on a system in Mbytes and sets <code>SI_MEMSIZE</code> .
model	model	Returns a system's platform name and sets <code>SI_MODEL</code> . For a list of platform names, see the <i>Solaris 10 Sun Hardware Platform Guide</i> at http://docs.sun.com .

TABLE 9-10 Descriptions of Probe Keywords (Continued)

Rule Keyword	Equivalent Probe Keyword	Description of Probe Keyword
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 <code>ifconfig(1M) -a</code> that is not <code>lo0</code> . The <code>network</code> keyword sets <code>SI_NETWORK</code> .
osname	osname	Returns the version and operating system name of the Solaris OS that is found on a CD and sets <code>SI_OSNAME</code> . If the JumpStart program finds a Solaris release but is unable to determine the version, the version that is returned is <code>SystemV</code> .
	rootdisk	Returns the name and size in Mbytes of a system's root disk and sets <code>SI_ROOTDISK</code> .
totaldisk	totaldisk	Returns the total disk space on a system (in Mbytes) and sets <code>SI_TOTALDISK</code> . The total disk space includes all of the operational disks that are attached to a system.

PART III Using RAID-1 Volumes

This part provides an overview of Solaris Volume Manager components that can be used in a Solaris installation or upgrade. This part also includes guidelines and requirements that are necessary to use RAID-1 volumes.

Creating RAID-1 Volumes (Mirrors) During Installation (Overview)

This section discusses the advantages of creating mirrored file systems. The section also describes the Solaris Volume Manager components that are required to create mirrored file systems.

This chapter describes the following topics.

- “Why Use RAID-1 Volumes?” on page 171
- “How Do RAID-1 Volumes Work?” on page 172
- “Overview of Solaris Volume Manager Components” on page 174
- “Example of RAID-1 Volume Disk Layout” on page 177

For additional information about how to create mirrored file systems with Solaris Live Upgrade, see “General Guidelines When Creating RAID-1 Volumes (Mirrored) File Systems” in *Solaris 10 Installation Guide: Solaris Live Upgrade and Upgrade Planning*.

For additional information about how to create mirrored file systems with the custom JumpStart installation method, see “`filesys` Profile Keyword (Creating RAID-1 Volumes)” on page 144 and “`metadb` Profile Keyword (Creating State Database Replicas)” on page 150.

Why Use RAID-1 Volumes?

During the installation or upgrade, you can create RAID-1 volumes to duplicate your system data over multiple physical disks. By duplicating your data over separate disks, you can protect your data from disk corruption or a disk failure.

The Solaris custom JumpStart and Solaris Live Upgrade installation methods use the Solaris Volume Manager technology to create RAID-1 volumes that mirror a file system. Solaris Volume Manager provides a powerful way to reliably manage your disks and data by using volumes. Solaris Volume Manager enables concatenations,

stripes, and other complex configurations. The custom JumpStart and Solaris Live Upgrade installation methods enable a subset of these tasks, such as creating a RAID-1 volume for the root (/) file system. You can create RAID-1 volumes during your installation or upgrade, eliminating the need to create them after the installation.

Installation Program	Limitations
The custom JumpStart program	Supports RAID-0 and RAID-1 volumes, but does not support other Solaris Volume Manager components, such as RAID-5 volumes Supports the creation of RAID-1 volumes during an initial installation only
Solaris Live Upgrade	Supports RAID-0 and RAID-1 volumes, but does not support other Solaris Volume Manager components, such as RAID-5 volumes Supports the creation of RAID-1 volumes during an upgrade

For detailed information about complex Solaris Volume Manager software and components, see *Solaris Volume Manager Administration Guide*.

How Do RAID-1 Volumes Work?

Solaris Volume Manager uses virtual disks to manage physical disks and their associated data. In Solaris Volume Manager, a virtual disk is called a *volume*. A *volume* is a name for a group of physical slices that appear to the system as a single, logical device. Volumes are actually pseudo, or virtual, devices in standard UNIX[®] terms.

A volume is functionally identical to a physical disk in the view of an application or a file system (such as UFS). Solaris Volume Manager converts I/O requests that are directed at a volume into I/O requests to the underlying member disks.

Solaris Volume Manager volumes are built from slices (disk partitions) or from other Solaris Volume Manager volumes.

You use volumes to increase performance and data availability. In some instances, volumes can also increase I/O performance. Functionally, volumes behave the same way as slices. Because volumes look like slices, they are transparent to end users, applications, and file systems. Like physical devices, you can use Solaris Volume Manager software to access volumes through block or raw device names. The volume name changes, depending on whether the block or raw device is used.

The custom JumpStart installation method and Solaris Live Upgrade support the use of block devices to create mirrored file systems. See [“RAID Volume Name Requirements and Guidelines for Custom JumpStart and Solaris Live Upgrade”](#) on page 182 for details about volume names.

When you create RAID-0 volumes (single-slice concatenations) and RAID-1 volumes, Solaris Volume Manager duplicates data on the concatenations (submirrors), and treats the submirrors as one volume.

Figure 10–1 shows a mirror that duplicates the root (/) file system over two physical disks.

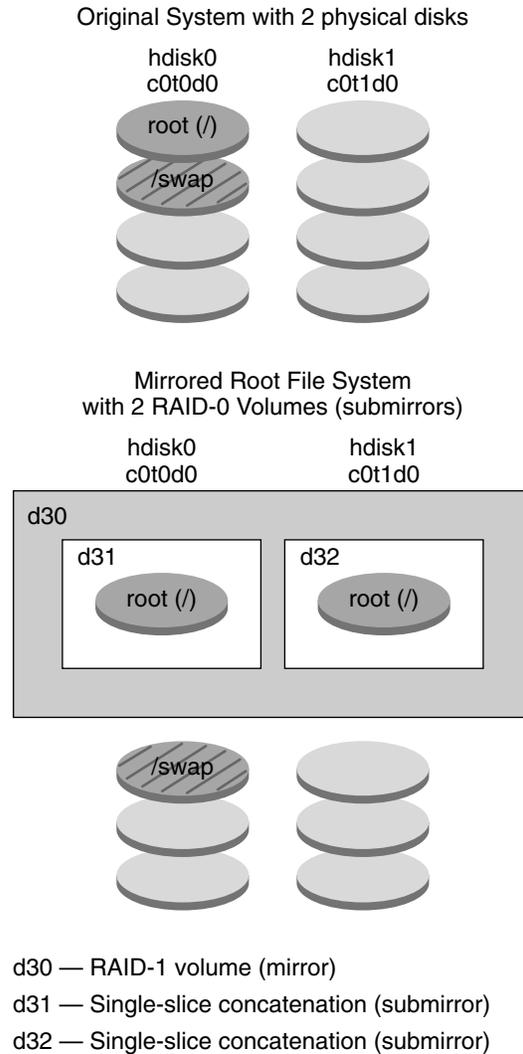


FIGURE 10–1 Creating RAID-1 Volumes on the Root (/) File System on Two Disks

Figure 10–1 shows a system with the following configuration.

- The root file system (/) on `hdisk0` is included in the single-slice concatenation that is named `d31`.
 - A single-slice concatenation that is named `d32` is created on the hard disk that is named `hdisk1`.
 - The mirror that is named `d30` consists of the submirrors that are named `d31` and `d32`.
 - The mirror duplicates the data in the root file system on both submirrors.
-

Overview of Solaris Volume Manager Components

The custom JumpStart installation method and Solaris Live Upgrade enable you to create the following components that are required to replicate data.

- State database and state database replicas (metadbs)
- Single-slice concatenations (submirrors)
- RAID-1 volumes (mirrors)

This section briefly describes each of these components. For complete information about these components, see *Solaris Volume Manager Administration Guide*.

State Database and State Database Replicas

The *state database* is a database that stores information on a physical disk about the state of your Solaris Volume Manager configuration. The state database records and tracks changes that are made to your configuration. Solaris Volume Manager automatically updates the state database when a configuration or state change occurs. Creating a new volume is an example of a configuration change. A submirror failure is an example of a state change.

The state database is actually a collection of multiple, replicated database copies. Each copy, referred to as a *state database replica*, ensures that the data in the database is always valid. Having copies of the state database protects against data loss from single points of failure. The state database tracks the location and status of all known state database replicas.

Solaris Volume Manager cannot operate until you have created the state database and its state database replicas. A Solaris Volume Manager configuration must have an operating state database.

When you set up your configuration, you can locate the state database replicas on either of the following:

- Dedicated slices
- (Solaris Live Upgrade only) Slices that will later become part of volumes

You can keep more than one copy of a state database on one slice. However, you might make the system more vulnerable to a single point of failure by placing state database replicas on a single slice.

The state database replicas ensure that the data in the state database is always valid. When the state database is updated, each state database replica is also updated. The updates occur one at a time to protect against corruption of all updates if the system crashes.

If your system loses a state database replica, Solaris Volume Manager must identify which state database replicas still contain valid data. Solaris Volume Manager determines this information by using a *majority consensus algorithm*. This algorithm requires that a majority (half + 1) of the state database replicas be available and in agreement before any of them are considered valid. Because of this majority consensus algorithm, you must create at least three state database replicas when you set up your disk configuration. A consensus can be reached if at least two of the three state database replicas are available.

Each state database replica occupies 4 Mbytes (8192 disk sectors) of disk storage by default. Replicas can be stored on the following devices:

- A dedicated local disk slice
- (Solaris Live Upgrade only) A local slice that will be part of a volume
- (Solaris Live Upgrade only) A local slice that will be part of a UFS logging device

Replicas cannot be stored on the root (/), swap, or /usr slices, or on slices that contain existing file systems or data. After the replicas have been stored, volumes or file systems can be placed on the same slice.

When using custom JumpStart or Solaris Live Upgrade to install RAID-1 volumes, review these guidelines and requirements [“State Database Replicas Guidelines and Requirements” on page 180](#)

For more detailed information about the state database and state database replicas [Solaris Volume Manager Administration Guide](#)

RAID-0 Volumes (Concatenations)

The custom JumpStart and Solaris Live Upgrade installation methods enable you to create RAID-0 volumes. A RAID-0 volume single-slice concatenation is a volume whose data is organized serially and adjacently across components, forming one logical storage unit. The custom JumpStart installation method and Solaris Live Upgrade do not enable you to create stripes or other complex Solaris Volume Manager volumes.

During the installation or upgrade, you can create RAID-1 volumes (mirrors) and attach RAID-0 volumes to these mirrors. The RAID-0 volumes that are *mirrored* are called *submirrors*. A mirror is made of one or more RAID-0 volumes. After the installation, you can manage the data on separate RAID-0 submirror volumes by administering the RAID-1 mirror volume through the Solaris Volume Manager software.

The custom JumpStart installation method enables you to create a mirror that consists of up to two submirrors. Solaris Live Upgrade enables you to create a mirror that consists of up to three submirrors. Practically, a two-way mirror is usually sufficient. A third submirror enables you to make online backups without losing data redundancy while one submirror is offline for the backup.

For RAID-0 volume planning information	“RAID-1 and RAID-0 Volume Requirements and Guidelines” on page 181
For RAID-0 volumes details	<i>Solaris Volume Manager Administration Guide</i>

RAID-1 Volumes (Mirrors)

A RAID-1 volume, or *mirror*, is a volume that maintains identical copies of the data in RAID-0 volumes (single-slice concatenations.) Using RAID-1 volumes to mirror file systems requires an investment in disks. You need at least twice as much disk space as the amount of data. Because Solaris Volume Manager software must write to all RAID-0 volumes, duplicating the data can also increase the time that is required for write requests to be written to disk.

With RAID-1 volumes, data can be read from both RAID-0 volumes simultaneously (either volume can service any request), providing improved performance. If one physical disk fails, you can continue to use the mirror with no loss in performance or loss of data.

After you configure a RAID-1 volume, the volume can be used just as if it were a physical slice.

You can duplicate any file system, including existing file systems. You can also use a RAID-1 volume for any application, such as a database.

For RAID-1 volume planning information	“RAID-1 and RAID-0 Volume Requirements and Guidelines” on page 181
For RAID-1 volume details	<i>Solaris Volume Manager Administration Guide</i>

Example of RAID-1 Volume Disk Layout

The following figure shows a RAID-1 volume that duplicates the root file system (/) over two physical disks. State database replicas (metadbs) are placed on both disks.

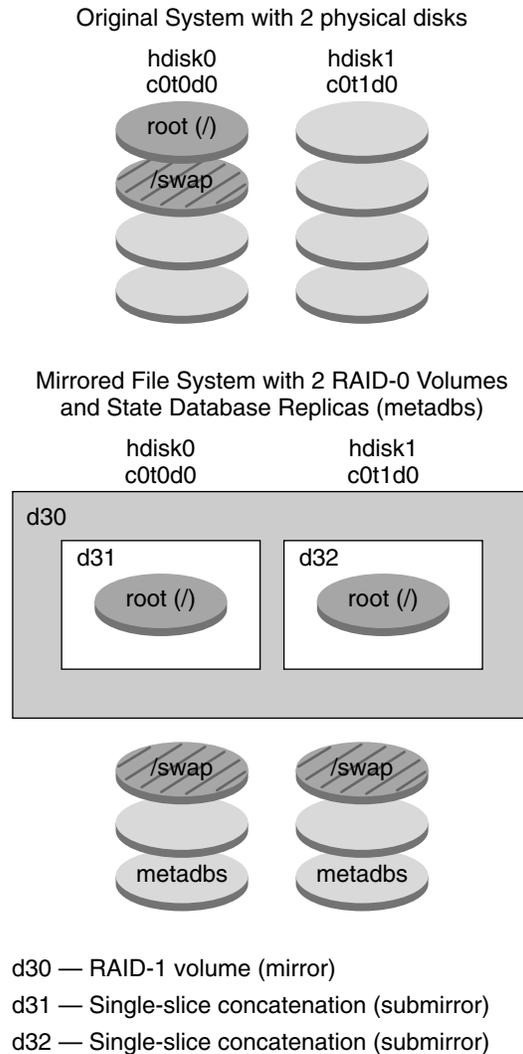


FIGURE 10-2 RAID-1 Volume Disk Layout

Figure 10-2 shows a system with the following configuration.

- The root file system (/) on `hdisk0` is included in the single-slice concatenation that is named `d31`.
- A single-slice concatenation that is named `d32` is created on the hard disk that is named `hdisk1`.
- The mirror that is named `d30` consists of the submirrors that are named `d31` and `d32`.
- The mirror duplicates the data in the root file system on both submirrors.
- State database replicas are created on slices both `hdisk0` and `hdisk1`.

For an example profile that uses the custom JumpStart installation method to create this configuration

[Example 4-12](#)

For instructions about how to create RAID-1 volumes with Solaris Live Upgrade

*“To Create a Boot Environment With RAID-1 Volumes (Mirrors) (Command-Line Interface)” in *Solaris 10 Installation Guide: Solaris Live Upgrade and Upgrade Planning**

Creating RAID-1 Volumes (Mirrors) During Installation (Planning)

This chapter describes the requirements and guidelines that are necessary to create RAID-1 volumes with the custom JumpStart or Solaris Live Upgrade installation methods.

This chapter describes the following topics.

- [“System Requirement” on page 179](#)
- [“State Database Replicas Guidelines and Requirements” on page 180](#)
- [“RAID-1 and RAID-0 Volume Requirements and Guidelines” on page 181](#)
- [“How Booting Into Single-User Mode Affects RAID-1 Volumes” on page 185](#)

For additional information about planning to create mirrored file systems with the Solaris Live Upgrade installation method, see [“General Guidelines When Creating RAID-1 Volumes \(Mirrored\) File Systems”](#) in *Solaris 10 Installation Guide: Solaris Live Upgrade and Upgrade Planning*.

For instructions about how to create mirrored file systems with the custom JumpStart installation method, see [“filesys Profile Keyword \(Creating RAID-1 Volumes\)” on page 144](#) and [“metadb Profile Keyword \(Creating State Database Replicas\)” on page 150](#).

System Requirement

To create RAID-1 volumes to duplicate data on specific slices, the disks that you plan to use must be directly attached and available to the system during the installation.

State Database Replicas Guidelines and Requirements

You should distribute state database replicas across slices, drives, and controllers, to avoid single points of failure. You want a majority of replicas to survive a single component failure. If you lose a replica, when a device fails, for example, the failure might cause problems with running Solaris Volume Manager software or when rebooting the system. Solaris Volume Manager software requires at least half of the replicas to be available to run, but a majority (half plus one) to reboot into multiuser mode.

For detailed instructions about creating and administering state database replicas, see *Solaris Volume Manager Administration Guide*.

Selecting Slices for State Database Replicas

Before selecting slices for state database replicas, consider the following guidelines and recommendations.

- You should create state database replicas on a dedicated slice of at least 4 Mbytes per replica. If necessary, you could create state database replicas on a slice that is to be used as part of a RAID-0 or RAID-1 volume. You must create the replicas before you add the slice to the volume.
- By default, the size of a state database replica is 4 Mbytes or 8192 disk blocks. Because your disk slices might not be that small, you can resize a slice to hold the state database replica. For information about resizing a slice, see Chapter 12, “Administering Disks (Tasks),” in *System Administration Guide: Devices and File Systems*.
- You can create state database replicas on slices that are not in use. The part of a slice that is reserved for the state database replica should not be used for any other purpose.
- You cannot create state database replicas on existing file systems, or the root (/), /usr, and swap file systems. If necessary, you can create a new slice (provided a slice name is available) by allocating space from swap and then put state database replicas on that new slice.
- When a state database replica is placed on a slice that becomes part of a volume, the capacity of the volume is reduced by the space that is occupied by the replica or replicas. The space that is used by a replica is rounded up to the next cylinder boundary and this space is skipped by the volume.

Choosing the Number of State Database Replicas

Before choosing the number of state database replicas, consider the following guidelines.

- A minimum of 3 state database replicas are recommended, up to a maximum of 50 replicas per Solaris Volume Manager disk set. The following guidelines are recommended:
 - For a system with only a single drive: put all three replicas in one slice.
 - For a system with two to four drives: put two replicas on each drive.
 - For a system with five or more drives: put one replica on each drive.
- Additional state database replicas can improve the mirror's performance. Generally, you need to add two replicas for each mirror you add to the system.
- If you have a RAID-1 volume that is to be used for small-sized random I/O (for example, for a database), consider your number of replicas. For best performance, ensure that you have at least two extra replicas per RAID-1 volume on slices (and preferably on disks and controllers) that are unconnected to the RAID-1 volume.

Distributing State Database Replicas Across Controllers

If multiple controllers exist, replicas should be distributed as evenly as possible across all controllers. This strategy provides redundancy if a controller fails and also helps balance the load. If multiple disks exist on a controller, at least two of the disks on each controller should store a replica.

RAID-1 and RAID-0 Volume Requirements and Guidelines

When you are working with RAID-1 volumes (mirrors) and RAID-0 volumes (single-slice concatenations), consider the following guidelines.

Custom JumpStart and Solaris Live Upgrade Guidelines

The custom JumpStart installation method and Solaris Live Upgrade support a subset of the features that are available in the Solaris Volume Manager software. When you create mirrored file systems with these installation programs, consider the following guidelines.

Installation Program	Supported Feature	Unsupported Feature
Custom JumpStart and Solaris Live Upgrade	RAID-0 volume is supported, but only as a single-slice concatenation.	In Solaris Volume manager a RAID-0 volume can refer to disk stripes or disk concatenations. You cannot create RAID-0 stripe volumes during the installation or upgrade.
Custom JumpStart	File systems that are created as RAID-1 volumes do not need to be created before installation.	
Custom JumpStart	You can create up to two RAID-0 volumes (submirrors) for each RAID-1 volume. Two submirrors usually provide sufficient data redundancy for most applications, and the disk drive costs are less expensive.	More than two RAID-0 volumes are not supported.
Solaris Live Upgrade	You can create up to three RAID-0 volumes (submirrors) for each RAID-1 volume. Three submirrors enable you to take a submirror offline and perform a backup while maintaining the two remaining submirrors for continued data redundancy.	More than three RAID-0 volumes are not supported.

RAID Volume Name Requirements and Guidelines for Custom JumpStart and Solaris Live Upgrade

Observe the following rules when assigning names for volumes.

- Use a naming method that maps the slice number and disk number to volume numbers.
- Volume names must begin with the letter `d` followed by a number, for example, `d0`.
- Solaris Volume Manager has 128 default volume names from 0–127. The following list shows some example volume names.
 - Device `/dev/md/dsk/d0` – block volume `d0`
 - Device `/dev/md/dsk/d1` – block volume `d1`
- Use ranges for each particular type of volume. For example, assign numbers 0–20 for RAID-1 volumes, and 21–40 for RAID-0 volumes.

- Instead of specifying the full volume name, such as `/dev/md/dsk/d1`, you can often use an abbreviated volume name, such as `d1`.

RAID Volume Naming Conventions for Solaris Live Upgrade

You can abbreviate the names of physical disk slices and Solaris Volume Manager volumes. The abbreviation is the shortest name that uniquely identifies a device. Examples follow.

- A Solaris Volume Manager volume can be identified by its *dnum* designation, so that, for example, `/dev/md/dsk/d10` becomes simply `d10`.
- If a system has a single controller and multiple disks, you might use `t0d0s0`, but with multiple controllers use `c0t0d0s0`.

When you use the Solaris Live Upgrade to create RAID-1 volumes (mirrors) and RAID-0 volumes (submirrors), you can let the software detect and assign volume names, or you can assign the names. If you let the software detect the names, the software assigns the first mirror or submirror name that is available. If you assign mirror names, assign names ending in zero so that the installation can use the names ending in 1 and 2 for submirrors. If you assign submirror names, assign names ending in 1 or 2. If you assign numbers incorrectly, the mirror might not be created. For example, if you specify a mirror name with a number that ends in 1 or 2 (`d1` or `d2`), Solaris Live Upgrade fails to create the mirror if the mirror name is a duplicate of a submirror's name.

In this example, Solaris Live Upgrade assigns the volume names. The RAID-1 volumes `d0` and `d1` are the only volumes in use. For the mirror `d10`, Solaris Live Upgrade chooses `d2` for the submirror for the device `c0t0d0s0` and `d3` for the submirror for the device `c1t0d0s0`.

```
lucreate -n newbe -m /:d10:mirror,ufs -m /:c0t0d0s0:attach -m
/:c1t0d0s0:attach
```

In this example, the volume names are assigned in the command. For the mirror `d10`, `d11` is the name for the submirror for the device `c0t0d0s0` and `d12` is the name for the submirror for the device `c1t0d0s0`.

```
lucreate -n newbe -m /:d10:mirror,ufs -m /:c0t0d0s0,d11:attach -m
/:c1t0d0s0,d12:attach
```

For detailed information about Solaris Volume Manager naming requirements, see *Solaris Volume Manager Administration Guide*.

RAID Volume Naming Conventions for Custom JumpStart

When you use the custom JumpStart installation method to create RAID-1 volumes (mirrors) and RAID-0 volumes (submirrors), you can let the software detect and assign volume names to mirrors, or you can assign the names in the profile. If you let the software detect the names, the software assigns the first volume number that is

available. If you assign names in the profile, assign mirror names ending in zero so that the installation can use the names ending in 1 and 2 for submirrors. If you assign numbers incorrectly, the mirror might not be created. For example, if you specify a mirror name with a number that ends in 1 or 2 (d1 or d2), JumpStart fails to create the mirror if the mirror name is a duplicate of a submirror's name. In the following profile example, the mirror is assigned the first volume numbers that are available. If the next available mirror ending in zero is d10, then the names d11 and d12 are assigned to the submirrors.

```
fileSYS          mirror c0t0d0s1 /
```

In the following profile example, the mirror number is assigned in the profile as d30. The submirror names are assigned by the software, based on the mirror number and the first available submirrors. In this example, the submirrors are named d31 and d32.

```
fileSYS          mirror:d30 c0t1d0s0 c0t0d0s0 /
```

For detailed information about Solaris Volume Manager naming requirements, see *Solaris Volume Manager Administration Guide*.

Guidelines for Selecting Disks and Controllers

When you choose the disks and controllers that you want to use to mirror a file system, consider the following guidelines.

- Use components that are on different controllers to increase the number of simultaneous reads and writes that can be performed.
- Keep the slices of different submirrors on different disks and controllers. Data protection is diminished considerably if slices of two or more submirrors of the same mirror are on the same disk.
- Organize submirrors across separate controllers, because controllers and associated cables tend to fail more often than disks. This practice also improves mirror performance.
- Use the same type of disks and controllers in a single mirror. Particularly in old SCSI storage devices, different models or brands of disk or controller can have widely varying performance. Mixing the different performance levels in a single mirror can cause performance to degrade significantly.

Guidelines for Selecting Slices

When you choose the slices that you want to use to mirror a file system, consider the following guidelines.

- Any file system, including root (/), swap, and /usr, can use a mirror. Any application, such as a database, also can use a mirror.

- Make sure that your submirror slices are of equal size. Submirrors of different sizes result in unused disk space.
- If you have a mirrored file system in which the first submirror attached does not start on cylinder 0, all additional submirrors you attach must also not start on cylinder 0. If you attempt to attach a submirror starting on cylinder 0 to a mirror in which the original submirror does not start on cylinder 0, the following error message is displayed:

```
can't attach  
labeled submirror to an unlabeled mirror
```

You must ensure that all submirrors you plan to attach to a mirror either all start on cylinder 0, or that none of them start on cylinder 0.

Starting cylinders do not have to be identical across all submirrors, but all submirrors must either include or not include cylinder 0.

How Booting Into Single-User Mode Affects RAID-1 Volumes

If a system with mirrors for root (`/`), `/usr`, and `swap` is booted into single-user mode, the system indicates that these mirrors are in need of maintenance. When you view these mirrors with the `metastat` command, these mirrors, and possibly all mirrors on the system, appear in the “Needing Maintenance” state.

Though this situation appears to be potentially dangerous, do not be concerned. The `metasync -r` command, which normally occurs during boot to resynchronize mirrors, is interrupted when the system is booted into single-user mode. After the system is rebooted, the `metasync -r` command runs and resynchronizes all mirrors.

If this interruption is a concern, run the `metasync -r` command manually.

For more information about the `metasync`, see the `metasync(1M)` man page, and *Solaris Volume Manager Administration Guide*.

PART **IV** Appendixes

This part contains troubleshooting and reference information.

Troubleshooting (Tasks)

This chapter contains a list of specific error messages and general problems you might encounter when installing Solaris 10 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.

- “Problems With Setting Up Network Installations” on page 189
- “Problems With Booting a System” on page 190
- “Initial Installation of the Solaris OS” on page 196
- “Upgrading the Solaris OS” on page 198

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 name service.

Description: Add the host *host_name* to the name service and execute the `add_install_client` command again.

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 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 10 Operating System DVD or the Solaris 10 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 10 Operating System DVD or Solaris 10 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 10 Operating System DVD or the Solaris 10 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 software was installed (either through the Solaris installation program or custom JumpStart), no boot disk was selected. You now must use the Solaris 10 Device Configuration Assistant diskette or edit the BIOS to boot the system.

Solution: Follow these instructions:

- Insert the Solaris 10 Device Configuration Assistant diskette into the system's boot diskette drive (usually drive A). For information about accessing the Solaris 10 Device Configuration Assistant diskette, see Appendix C, "Preparing to Boot From the Solaris 10 Device Configuration Assistant or the Network (Tasks)," in *Solaris 10 Installation Guide: Network-Based Installations*.
- If you cannot use the bootable media, 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
```

For x86 based systems, when the installation program prompts you to "Select type of installation," type the following command.

```
b - -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 10 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 IDE BIOS primary drive on your system was not detected by the Solaris 10 Device Configuration Assistant diskette during the pre-booting phase. (*x86 based systems only*)

Solution: Follow these instructions:

- If you are using old drives, they might be unsupported. Check your hardware manufacturer's documentation.
- Make sure the ribbon and power cables are connected correctly. Check the manufacturer's documentation.
- If only one drive is attached to the controller, designate the drive as the master drive by setting jumpers. Some drives have different jumper settings for a single master, as opposed to a master operating with a slave. Connect the drive to the connector at the end of the cable to reduce signal ringing that occurs when an unused connector is dangling at the end of the cable.
- If two drives are attached to the controller, designate one drive as the master by setting jumpers (or as a master operating with a slave), and set the second drive as a slave by setting jumpers.
- If one drive is a hard disk and the second a CD-ROM drive, designate one drive as the slave drive by setting jumpers. You can designate either physical drive as the slave drive.
- If problems persist with two drives on a single controller, attach one drive at a time to verify that each drive works. Designate the drive as master or single master by setting jumpers, and use the drive connector at the end of the IDE ribbon cable to attach the drive. Verify that each drive works, then set the jumpers for the drives back to a master and slave configuration.
- If the drive is a disk drive, use the BIOS setup utility to ensure that the drive type (which indicates the number of cylinders, heads, and sectors) is configured correctly. Some BIOS software might have a feature that automatically detects the drive type.
- If the drive is a CD-ROM drive, use the BIOS setup screen to configure the drive type as a CD-ROM drive, provided the BIOS software offers this capability.

- For many systems, IDE CD-ROM drives are only recognized by MS-DOS if an MS-DOS CD-ROM driver has been installed. Try another drive.

The IDE disk or CD-ROM drive on your system was not found by the Solaris 10 Device Configuration Assistant diskette during the pre-booting phase. (*x86 based systems only*)

Solution: Follow these instructions:

- If disks are disabled in the BIOS, use the Solaris 10 Device Configuration Assistant diskette to boot from the hard disk. For information about accessing the Solaris 10 Device Configuration Assistant, see Appendix C, "Preparing to Boot From the Solaris 10 Device Configuration Assistant or the Network (Tasks)," in *Solaris 10 Installation Guide: Network-Based Installations*.
- If the system has no disks, it might be a diskless client.

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 name 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 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 10 Operating System DVD, either the DVD or a copy of the DVD image on the install server
- Solaris 10 Software - 1 CD image, either the Solaris 10 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 10 Operating System DVD or Solaris 10 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 10 Operating System DVD image or Solaris 10 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+ name 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 name service. If the system's host name is listed in the NIS or NIS+ name 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.

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 10 Operating System DVD, the Solaris 10 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/Solaris_10/SUNWxxx/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 10 Device Configuration Assistant diskette 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 10 Device Configuration Assistant diskette.

▼ 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.

Steps 1. Boot to the installation media in single-user mode.

```
# b -s
```

2. Start the `format(1M)` program.

```
# format
```

3. Specify the IDE disk drive on which you want to perform a surface analysis.

```
# cxdy
```

`cx` Is the controller number

`dy` Is the device number

4. Determine if you have an `fdisk` partition.

- If a Solaris `fdisk` partition already exists, proceed to [Step 5](#).
- If a Solaris `fdisk` partition does not exist, use the `fdisk` command to create a Solaris partition on the disk.

```
format> fdisk
```

5. To begin the surface analysis, type:

```
format> analyze
```

6. Determine the current settings, type:

```
analyze> config
```

7. (Optional) To change settings, type:

```
analyze> setup
```

8. 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.

9. To exit the analysis, type:

```
analyze> quit
```

10. Determine if you want to specify blocks to remap.

- If no, go to [Step 11](#).
- If yes, type:

```
format> repair
```

11. To exit the format program, type:

```
quit
```

12. To restart the media in multiuser mode, type:

```
ok b
```

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/bzczt` not found

Cause: Solaris Live Upgrade fails because of needing a patch cluster.

Solution: A patch is needed to install Solaris Live Upgrade. Go to sunsolve.sun.com for the patch.

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 Solaris 10 Software - 1 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 10 Operating System DVD, a net installation image, or JumpStart.

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” on page 31](#) 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.

- Steps**
1. **Reboot the system from the Solaris 10 Operating System DVD, the Solaris 10 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.

▼ 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.

- Steps**
1. **Create an inactive boot environment. See “Creating a New Boot Environment” in *Solaris 10 Installation Guide: Solaris Live Upgrade and Upgrade Planning*.**

2. 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 /.alt.12345
```

- b. Change to the directory that contains the `vfstab`, for example:

```
# cd /.alt.12345/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 /.alt.12345/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. Change directories to the `install-db` file on the inactive boot environment, for example:

```
# cd /.alt.12345/etc
```

- j. Create the Veritas `install-db` file, for example:

```
# touch vx/reconfig.d/state.d/install-db
```

k. Unmount the inactive boot environment.

```
# luumount inactive_boot_environment_name mount_point
```

3. Upgrade the inactive boot environment. See Chapter 7, “Upgrading With Solaris Live Upgrade (Tasks),” in *Solaris 10 Installation Guide: Solaris Live Upgrade and Upgrade Planning*.

4. Activate the inactive boot environment. See “Activating a Boot Environment” in *Solaris 10 Installation Guide: Solaris Live Upgrade and Upgrade Planning*.

5. Shut down the system.

```
# init 0
```

6. 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.

7. 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 VRTSvman VRTSvmdev
```

8. Restore the original vfstab and system files:

```
# cp /etc/vfstab.original /etc/vfstab  
# cp /etc/system.original /etc/system
```

9. 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 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 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 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 10 Operating System DVD

To install the software from a net installation image or from the Solaris 10 Operating System DVD over the network, follow these steps.

- 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 x86 boot partition and the Solaris partition.

▼ To Install From the Solaris 10 Software - 1 CD or From a Network Installation Image

To use the Solaris installation program to install from the Solaris 10 Software - 1 CD or from a network installation image on a boot server, follow these steps.

- 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.**

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 Solaris OS”](#) on page 205
- [“Preventing User Interaction When Installing or Upgrading”](#) on page 209

Preventing Modification of the Current Solaris OS

Following the requirements in this section keeps the currently running Solaris OS unaltered.

Using Absolute Paths

For an installation of an operating system to be successful, packages must recognize and correctly interact with 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 on an alternative root (`/`) 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 by 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	<code>if [-f \${PKG_INSTALL_ROOT}\ /etc/myproduct.conf] ; then</code>	<code>if [-f /etc/myproduct.conf] ; \ then</code>
Removing a file	<code>/bin/rm -f \${PKG_INSTALL_ROOT}\ /etc/myproduct.conf</code>	<code>/bin/rm -f /etc/myproduct.conf</code>
Changing a file	<code>echo "test=no" > \${PKG_INSTALL_ROOT}\ /etc/myproduct.conf</code>	<code>echo "test=no" > \ /etc/myproduct.conf</code>

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 Solaris OS to prevent modifying the Solaris 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	Solaris Live Upgrade Impact	Non-global Zones Impact
Scripts must be written in Bourne shell (<code>/bin/sh</code>). Bourne shell is the interpreter that is used by the <code>pkgadd</code> 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 <code>ps</code> or <code>truss</code> , 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 <code>expr</code> , <code>cp</code> , and <code>ls</code> and other commands that facilitate shell scripting.	X	X

TABLE B-2 Guidelines for Creating Scripts (Continued)

Guidelines	Solaris Live Upgrade Impact	Non-global Zones Impact
Any commands that a script invokes must be available in all releases from Solaris 7 or later, since a package must run on all of those releases. Therefore, you cannot use commands that were added or removed after the Solaris 7 release. For example, the <code>pgrep</code> command cannot be used, because this command was added in the Solaris 8 release and not available in the Solaris 7 release.	X	
To verify that a specific command or option is supported in a Solaris 7 or later release, see the specific version of <i>Solaris Reference Manual AnswerBook</i> on http://docs.sun.com .		

Maintaining Diskless Client Compatibility

Packages must not execute commands that the package itself delivers. This requirement 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 (`/`) 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. The command 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 ${TEMPDIR} SUNWvsvm \
2>&1 > /dev/null | grep -v O_RDONLY | grep -v \
'open("${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
- 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.

```
mail=
instance=overwrite
partial=nocheck
runlevel=nocheck
idepend=nocheck
space=ask
setuid=nocheck
conflict=nocheck
action=nocheck
basedir=default
```

For More 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 <i>Application Packaging Developer's Guide</i>
For basic information about adding and removing packages and the installation administration file	Chapter 15, "Managing Software (Overview)," in <i>System Administration Guide: Basic Administration</i>

For detailed information about specific commands that are referenced in this appendix	<code>dircmp(1)</code> , <code>fssnap(1M)</code> , <code>ps(1)</code> , or <code>truss(1)</code> <code>pkgadd(1M)</code> , <code>pkgchk(1M)</code> , or <code>pkgrm(1M)</code>
For an overview of Solaris Live Upgrade	<i>Solaris 10 Installation Guide: Solaris Live Upgrade and Upgrade Planning</i>
For an overview of custom JumpStart	Chapter 3
For an overview of Solaris Zones	Chapter 16, "Introduction to Solaris Zones," in <i>System Administration Guide: Solaris Containers—Resource Management and Solaris Zones</i>

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	<p>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.</p> <p>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.</p>
arrow keys	One of the four directional keys on the numeric keypad.
begin script	A user-defined Bourne shell script, specified within the <code>rules</code> 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 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.

bootlog-cgi	The CGI program that enables a web server to collect and store remote client-booting and installation console messages during a WAN boot installation.
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.
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	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 installed 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 <i>software groups</i> , which are each composed of clusters and <i>packages</i> .
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 <i>RAID-0 volume</i> .
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 <code>vfstab</code> of the active and inactive boot environments. Examples are <code>root (/)</code> , <code>/usr</code> , <code>/var</code> , and <code>/opt</code> . 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 <code>rules</code> 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 <code>SI_</code> 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 <i>rules file</i> .
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 <i>disc</i> .
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 <code>pfinstall</code> 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: <code>tundra.mpk.ca.us</code>). 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 10 release.
Entire Solaris Software Group Plus OEM Support	A software group that contains the entire Solaris 10 release, plus additional hardware support for OEMs. This software group is recommended when installing Solaris software on SPARC based servers.
/etc	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	A file system on an OS server that is shared with other systems on a network. For example, the <code>/export</code> 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 <code>/export</code> file system on an OS server to boot and run.
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 <code>fdisk</code> partition on an x86 based system. x86 based systems allow up to four different <code>fdisk</code> partitions on a disk. These partitions can be used to hold individual operating systems. Each operating system must be located on a unique <code>fdisk</code> partition. A system can only have one Solaris <code>fdisk</code> 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 <code>rules</code> 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 <i>Solaris Zones</i> and <i>non-global zone</i> .
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 known as a *media server*). You can create an install server by copying the Solaris DVD or CD images to the server's hard disk.

IP address (Internet protocol address) In TCP/IP, a unique 32-bit number that identifies each host in a network. An IP address consists of four numbers that are separated by periods (192.168.0.0, for example). Most often, each part of the IP address is a number between 0 and 255. However, the first number must be less than 224 and the last number cannot be 0.

IP addresses are logically divided into two parts: the network (similar to a telephone area code), and the local system on the network (similar to a phone number). The numbers in a Class A IP address, for example, represent "network.local.local.local" and the numbers in a Class C IP address represent "network.network.network.local."

Class	Range (xxx is a number 0 to 255)	Number of Available IP Addresses
Class A	1.xxx.xxx.xxx - 126.xxx.xxx.xxx	Over 16 million
Class B	128.0.xxx.xxx - 191.255.xxx.xxx	Over 65,000
Class C	192.0.0.xxx - 223.255.255.xxx	256

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.

IPv6 is described in more detail in Part I, "Introducing System Administration: IP Services," in *System Administration Guide: IP Services*.

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	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 in the view 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 <i>install server</i> .
metadevice	See <i>volume</i> .
miniroot	The smallest possible bootable Solaris root file system. A miniroot contains a kernel and just enough software to install the Solaris environment on a hard disk. The miniroot is the file system that is copied to a machine in the initial installation.
mirror	See <i>RAID-1 volume</i> .

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, <code>/usr</code>).
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 name service to systems on a network.
name 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 name service, the system information can be maintained, managed, and accessed on a network-wide basis. Without a name service, each system has to maintain its own copy of the system information in the local <code>/etc</code> files. Sun supports the following name services: LDAP, NIS, and NIS+.
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.
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 <i>name server</i> and an <i>install server</i> .
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 <i>Solaris Zones</i> and <i>global zone</i> .
nonnetworked systems	Systems that are not connected to a network or do not rely on other systems.
/opt	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 (<code>/export/root</code> , <code>/export/swap</code>).
package	A collection of software that is grouped into a single entity for modular installation. The Solaris software is divided into <i>software groups</i> , which are each composed of <i>clusters</i> 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 <code>i86pc</code> and <code>sun4u</code> .
platform name	The output of the <code>uname -i</code> command. For example, the platform name for the Ultra 60 is <code>SUNW,Ultra-60</code> .
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—a <code>sun4u</code> SPARC based system, for example—the Power Management software is 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.
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 <i>rule</i> .
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 <i>rules file</i> .

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.
private key	The decryption key used in public-key encryption.
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-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-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.
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)	In a hierarchy of items, the one item from which all other items are descended. The root item has nothing above it in the hierarchy. / is the base directory from which all other directories stem, directly or indirectly. The root directory 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.
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 <i>profile</i> .
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 <i>must</i> 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 <code>/export/home</code> and <code>/swap</code> . 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 <code>vfstab</code> 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.
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 based 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, known as 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 <i>archive</i> .
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 the 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 <i>global zone</i> and <i>non-global zone</i> .
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 <i>RAID-0 volume</i> .
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.
swap space	A slice or file that temporarily holds the contents of a memory area till it can be loaded back into memory. Also called the <i>/swap</i> or <i>swap</i> file system.
sysidcfg file	A file in which you specify a set of special system configuration keywords that preconfigure a system.
system configuration file	(<i>system.conf</i>) A text file in which you specify the locations of the <i>sysidcfg</i> 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	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 <i>truststore</i> 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 saves 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 <i>protocol://machine:port/document</i> . A sample URL is <code>http://www.example.com/index.html</code> .
/usr	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	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.
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 in the view of an application or file system. In some command-line utilities, a volume is called a metadvice. 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 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.
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.
zone	See <i>non-global zone</i>

Index

Numbers and Symbols

- #
 - in profiles, 60
 - in rules files, 56
- = (equal sign) in profile field, 78
- ! (exclamation mark) rule field, 57
- && (ampersands) rule field, 57
- (/) file systems
 - value set by JumpStart, 163

A

- add_install_client command, JumpStart
 - directory access, 49
- adding
 - clusters when upgrading, 138
 - packages and patches with a finish script, 80
 - packages from software groups, 152
 - rules to rules file, 57
- alternative installation programs, 92
- ampersands (&&) rule field, 57
- AND rule field, 57
- any
 - probe keyword, description and values, 168
 - rule keyword, description and values, 121, 167
- arch probe keyword, 167
- arch rule keyword, 121, 167
- archive
 - JumpStart profile example, 64, 65, 66, 67
 - keywords, custom JumpStart, 127-132
- archive_location keyword, 127-132

- auto_install_sample directory
 - check script, 75, 97
 - copying files to JumpStart directory, 48, 52, 55
 - set_root_pw finish script, 83

B

- b option of setup_install_server command, 115
- backslash in rules files, 57
- backup_media keyword, 132-133
- begin.log file, 78
- begin rule field, description, 57
- begin scripts
 - creating derived profiles with, 78, 79
 - overview, 77
 - permissions, 78
 - rule field, 57
 - site-specific installation programs, 92
- boot: cannot open /kernel/unix message, 190
- boot_device keyword, 134
- boot-disk partition layout, new default (x86 based systems), 36
- bootenv createbe keyword, 135
- booting into single-user mode, 185
- bootparams file
 - enabling JumpStart directory access, 50
 - updating, 196
- Bourne shell scripts in rule fields, 57

C

- c option
 - pfinstall command, 74
 - add_install_client command, 118, 119
- Can't boot from file/device message, 190
- CHANGE DEFAULT BOOT DEVICE
 - message, 197
- changing directories
 - image of Solaris SPARC software on local disk, 52
 - to image of Solaris software on local disk, 48
 - to image of Solaris x86 based software on local disk, 55
 - to JumpStart directory, 75, 97
- check script
 - custom_probes file validation, 97, 98
 - custom_probes.ok file creation, 97
 - derived profiles and, 79
 - rules file validation, 75, 76, 98
 - rules.ok file creation, 75
 - testing rules, 76, 97
- client_arch keyword, 136
- CLIENT MAC ADDR error message, 196
- client_root profile keyword, 136
- clock gained xxx days message, 190
- cluster profile keyword
 - description and values, 137-138, 138
 - examples, 61
- comments
 - in profiles, 60
 - in rules files, 56
- concatenation
 - conceptual overview, 175
 - definition, 175
 - information for creating, 181
- configuring, creating disk configuration files, 86
- copying
 - JumpStart directory files, 80
 - JumpStart installation files, 48, 52, 55
 - Solaris 10 Device Configuration Assistant diskette, 53
- Core Solaris Software Group, 137-138
- Core System Support Software Group
 - description, 27-29
 - size, 28
- CPUs (processors)
 - probe keywords, 167
- CPUs (processors) (Continued)
 - rule keywords, 121, 167
- creating
 - custom_probes.ok file, 97
 - disk configuration files, 86
 - JumpStart directory, on server, 47
 - local file systems, 142-144
 - profiles
 - derived, 78
 - description, 59
 - RAID-1 volumes, 144-145
 - rules file, 56
 - rules.ok file, 75, 97
 - UFS, 52
- .cshrc file, 83
- custom JumpStart installation, 99
 - booting and installing, 99
 - description, 42
 - examples, 111, 120
 - booting and installing, 119
 - check script, 117
 - eng_profile creation, 115
 - engineering systems setup, 118
 - JumpStart directory, 115
 - marketing_profile creation, 116
 - marketing systems setup, 114, 118
 - networked, 42
 - nonnetworked, 40
 - RAID-1 volume profiles, 68, 69
 - rules file editing, 117
 - site setup, 111, 112
 - Solaris Flash profile, 64, 66, 67
 - standalone system, 40
 - WAN boot installation profile, 65
 - optional features, 77
 - begin scripts, 77, 79
 - finish scripts, 79, 83
 - overview, 77
 - site-specific installation programs, 92
 - overview, 42
 - preparing, 42, 76
 - profile keywords, 126
 - tip line connection requirements, 103, 107
- custom_probes file
 - naming, 94
 - requirements, 94
 - testing custom_probes, 97
 - validating by using check, 97, 98

custom_probes.ok file
 creating, 97
 description, 97

D

dd command, 54
defaults
 derived profile name, 78
 partitioning
 designating disks, 164
 excluding disks, 139
 software group installed, 138
deleting, clusters when upgrading, 138
derived profiles, 78, 79
Developer Solaris Software Group, 137-138
 description, 27-29
 profile example, 61
 size, 28
dfstab file, 48, 115
directories
 changing
 to image of Solaris software on local
 disk, 48
 to image of Solaris *SPARC* software on
 local disk, 52
 to image of Solaris x86 based software on
 local disk, 55
 to JumpStart directory, 75, 97
 JumpStart
 adding files, 80
 copying files, 80
 copying installation files, 48, 52, 55
 creating directory, 115
 creating for systems, 51
 permissions, 47, 51
 rules file example, 56
 sharing directory, 48, 115
disk configuration files
 creating
 SPARC based systems, 86
 x86 based systems, 88
 description, 71, 86
disk space
 planning, 25-29
 requirements, for software groups, 28

diskettes
 copying Solaris 10 Device Configuration
 Assistant diskette, 53
 formatting, 51, 54
 JumpStart directory access, 49
 x86: JumpStart directory, 51
diskless clients
 platforms, 136
 swap space, 136
disks probe keyword, description and
 values, 167
disksize rule keyword, description and
 values, 122, 167
display
 tip line connection requirements, 103, 107
domainname probe keyword, 167
domainname rule keyword, 122, 167
domains
 probe keyword, 167
 rule keyword, 122, 167
dontuse profile keyword, 139, 164

E

End User Solaris Software Group, 137-138
 description, 27-29
 size, 28
eng_profile example, 115
Entire Solaris Software Group, 137-138
 description, 27-29
 size, 28
Entire Solaris Software Group Plus OEM
 Support, 137-138
 description, 27-29
 size, 28
equal sign (=) in profile field, 78
/etc/bootparams file
 enabling JumpStart directory access, 50, 196
/etc/dfs/dfstab file, 48, 115
/etc/mnttab file, 51
exclamation mark (!) rule field, 57

F

failed upgrade, rebooting problems, 200
fdformat command, 51, 54

- fdisk command, 89
- fdisk profile keyword
 - description and values, 139-141
 - example, 61
- files and file systems
 - begin script output, 78
 - copying
 - JumpStart directory files using finish scripts, 80
 - JumpStart installation files, 48, 52, 55
 - Solaris 10 Device Configuration Assistant diskette, 53
 - creating
 - local file systems, 142-144
 - RAID-1 volumes, 144-145
 - finish script output, 79
 - mounting remote file systems, 141-142
 - UFS creation, 52
- filesys keyword, 142-144, 144-145
- filesys profile keyword
 - description and values, 141-142
 - examples, 61
- finish.log file, 79
- finish rule field, description, 58
- finish scripts
 - adding packages and patches, 80
 - customizing the root environment, 83
 - rule field, 58
 - setting the system's root password, 83
- formatting diskettes, 51

G

- geo keyword, 146
- getfile: RPC failed: error 5: RPC Timed out message, 50

H

- hard disks
 - mounting, 141-142
 - partitioning
 - designating for partitioning default, 164
 - examples, 61
 - excluding for partitioning default, 139
 - profile keyword, 157

- hard disks (Continued)
 - rootdisk values, 163
 - size
 - probe keywords, 167, 168
 - root space, 136
 - rule keywords, 122, 125, 167, 168
 - swap space
 - diskless client, 136
 - maximum size, 137
 - profile examples, 42, 61
- hostaddress probe keyword, 167
- hostaddress rule keyword, 122, 167
- hostname probe keyword, description and values, 167
- hostname rule keyword
 - description and values, 122, 167
 - example, 121-125

I

- install_config command, 50
- install_type keyword, 147
- install_type profile keyword
 - examples, 61
 - requirement, 60, 61
 - testing profiles, 74-75
- installation
 - compared to upgrade, 19
 - disk space recommendations, 25-29
 - overview of tasks, 15
- installed probe keyword, description and values, 167
- installed rule keyword, description and values, 123, 167
- IP addresses
 - probe keyword, 167
 - rule keyword, 122, 167

J

- JumpStart directory
 - adding files with finish scripts, 80
 - copying files
 - installation files, 48, 52, 55
 - using finish scripts, 80

JumpStart directory (Continued)

- creating
 - diskette for SPARC based systems, 51
 - diskette for x86 based systems, 51, 53
 - example, 115
 - server, 47
- permissions, 47, 51
- rules file example, 56
- sharing, 47, 115

K

- karch probe keyword, 167
- karch rule keyword, 123, 167
- keywords
 - probe, 93
 - Solaris Flash archives, custom
 - JumpStart, 127-132

L

- layout_constraint keyword, 147-150
- le0: No carrier - transceiver cable problem
 - message, 190
- locale keyword, 150
- log files
 - begin script output, 78
 - finish script output, 79
- logical AND rule field, 57

M

- majority consensus algorithm, 175
- marketing_profile example, 116
- matching
 - derived profiles, 78
 - order for rules, 58, 101, 106
 - rootdisk values, 163
- memory
 - probe keyword, 167
 - rule keyword, 123, 167
 - swap space size and, 137
- memsize probe keyword, description and values, 167

- memsize rule keyword, description and values, 123, 167
- metadb profile keyword, 150-151
- metadevice, *See* volume
- microprocessors
 - probe keywords, 167
 - rule keywords, 121, 167
- mirror, *See* RAID-1 volume
- mnttab file, 51
- model probe keyword, description and values, 167
- model rule keyword, description and values, 124, 167
- mounting
 - begin script caution, 78
 - by Solaris installation, 79
 - remote file systems, 141-142
- multiple lines in rules files, 57

N

- names/naming
 - custom_probes file, 94
 - derived profile names, 78
 - host name, 122, 167
 - rules file, 56, 57
 - system model names, 124, 167
- network installation, custom JumpStart
 - installation, example, 42
- network number, 124, 168
- network probe keyword, description and values, 168
- network rule keyword, description and values, 124, 168
- No carrier - transceiver cable problem
 - message, 190
- no_master_check keyword, 152
- noneuclidean profile keyword, 152
- Not a UFS filesystem message, 190

O

- osname probe keyword, 168
- osname rule keyword, 124, 168
- output files
 - begin script log, 78

output files (Continued)
finish script log, 79

P

-p option of check script, 76, 97

packages

adding

with a finish script, 80

with chroot, 82

administration file, 77

requirements when using custom

JumpStart, 205

Solaris Live Upgrade

requirements, 205

partitioning

examples, 61

excluding disks, 139

fdisk partitions, 61, 139-141

profile keyword, 157, 164

partitioning keyword, 157

password, root, 83

patches

adding

with a finish script, 80

with chroot, 82

paths, check script, 76, 97

permissions

begin scripts, 78

finish scripts, 79

JumpStart directory, 47, 51

pfinstall command, 70

planning

disk space, 25-29

initial installation compared to upgrade, 19

installing over the network, 18

overview of tasks, 15

platforms

diskless client, 136

matching system attributes and profiles, 58,
101, 106

probe keywords, 167

rule keywords, 123, 167

system model names, 124, 167

preparing for installation, with custom

JumpStart, 42, 76

probe keywords

arch, 167

disks, 167

domainname, 167

hostaddress, 167

hostname, 167

installed, 167

karch, 167

memsize, 167

model, 167

network, 168

osname, 168

rootdisk, 168

totaldisk, 168

probe rule keyword, description and
values, 125

processors

probe keywords, 167

rule keywords, 121, 167

profile keywords, 126, 164

archive_location, 127-132

backup_media, 132-133

boot_device, 134

bootenv createbe, 135

case sensitivity, 126

client_arch, 136

client_root, 136

client_swap, 136

cluster

description and values, 137-138, 138

examples, 61

creating state database replicas
(meatball), 150-151

dontuse

description and values, 139

usedisk and, 164

fdisk

description and values, 139-141

example, 61

filesystems

description and values, 141-142

examples, 61

local file systems, 142-144

RAID-1 volumes, 144-145

remote file systems, 141-142

forced_deployment, description and
values, 146

profile keywords (Continued)

- geo
 - description and values, 146
 - install_type
 - description and values, 147
 - examples, 61
 - requirement, 60, 61
 - layout_constraint, description and values, 147-150
 - local_customization, description and values, 150
 - locale, description and values, 150
 - metadb
 - description and values, 150-151
 - examples, 61
 - no_master_check, description and values, 152
 - noneuclidean, 152
 - partitioning
 - description and values, 157
 - designating disks, 164
 - examples, 61
 - excluding disks, 139
 - root_device, 163
 - system_type
 - description and values, 164
 - examples, 61
 - usedisk, description and values, 164
- ## profiles
- comments in, 60
 - creating, 59
 - derived profiles, 78, 79
 - description, 59
 - examples, 61
 - eng_profile, 115
 - marketing_profile, 116
 - Solaris Flash, 64, 66, 67
 - WAN boot installation, 65
 - matching systems to, 58, 101, 106
 - naming, 60
 - requirements, 56, 60
 - rule field, 57
 - testing, 74-75
- ## prtvtoc command
- SPARC: creating disk configuration file, 87
 - x86: disk configuration file creation, 89

R

- r option of check script, 76, 97
- RAID-0 volume
 - conceptual overview, 175
 - definition, 175
 - guidelines, 181-185
- RAID-1 volume
 - and disk geometries, 181
 - conceptual overview, 176
 - definition, 176
 - guidelines, 181-185
 - information for creating, 181
- Reduced Network Support Software Group, 137-138
 - description, 27-29
 - size, 28
- release of Solaris software
 - installed probe keyword, 167
 - installed rule keyword, 123, 167
 - osname probe keyword, 168
 - osname rule keyword, 124, 168
- remote file systems, mounting, 141-142
- replica, 174
- requirements
 - custom_probes file, 94
 - disk space, 25-29
 - memory, 23
 - profiles, 56, 60
- root (/) file systems, package requirements for an inactive boot environment, 205
- root (/) file systems, profile example, 42
- root_device keyword, 163
- root environment, customizing with a finish script, 83
- root password, setting with a finish script, 83
- rootdisk
 - definition, 163
 - slice value for filesystems, 142
 - value set by JumpStart, 163
- RPC failed: error 5: RPC Timed out message, 50
- RPC Timed out message, 50, 195
- rule_keyword rule field, 57
- rule keywords, 121
 - any, description and values, 121, 167
 - arch, 121, 167
 - disksize, description and values, 122, 167
 - domainname, 122, 167
 - hostaddress, 122, 167

- rule keywords (Continued)
 - hostname, 121-125, 167
 - installed, description and values, 123, 167
 - karch, 123, 167
 - memsize, 123, 167
 - model, 124, 167
 - network, 124, 168
 - osname, 124, 168
 - probe, 125
 - totaldisk, 125, 168
- rule_value rule field, 57
- rules
 - derived profiles, 78, 79
 - examples, 58
 - field descriptions, 57, 58
 - matching order, 58, 101, 106
 - multiple line rules, 57
 - rootdisk matching rules, 163
 - syntax, 57
 - testing validity, 76, 97
- rules file
 - adding rules, 57
 - comments in, 56
 - creating, 56
 - custom JumpStart example, 117
 - description, 56
 - example, 56
 - multiple line rules, 57
 - naming, 56, 57
 - syntax, 57
 - testing rules, 76
 - validating by using check, 76
 - custom JumpStart example, 117
 - derived profiles and, 79
- rules.ok file
 - creating, 75
 - description, 75
- rules.ok file, matching order for rules, 58
- rules.ok file
 - matching order for rules, 101, 106

S

- s option of add_install_client command, 119
- scripts
 - begin scripts, 77, 79, 92
 - Bourne shell scripts in rule fields, 57

- scripts (Continued)
 - finish scripts, 79, 83, 92
- security, root password, 83
- servers
 - JumpStart directory creation, 47
 - root space, 136
- Service partition, preserving during installation (x86 based systems), 36
- set_root_pw finish script, 83
- share command
 - sharing JumpStart directory, 48, 115
- shareall command, 48, 115
- sharing JumpStart directory, 48, 115
- SI_PROFILE environment variable, 78
- site-specific installation programs, 92
- size
 - hard disk
 - probe keywords, 167, 168
 - root space, 136
 - rule keywords, 122, 125, 167, 168
 - memory, 123, 167
 - swap space
 - diskless client, 136
 - maximum size, 137
 - profile examples, 42
 - tip line connection display dimensions, 103, 107
- slices
 - probe keyword, 167
 - profile examples, 61
 - rule keyword, 123, 167
- software groups
 - descriptions, 28
 - for profiles, 137-138
 - profile examples, 61
 - sizes, 28
 - upgrading, 138
- Solaris 10 Device Configuration Assistant
 - diskette, 53
- Solaris software
 - groups, 137-138
 - profile examples, 61
 - upgrading, 138
 - release or version
 - installed probe keyword, 167
 - installed rule keyword, 123, 167
 - osname probe keyword, 168
 - osname rule keyword, 124, 168

- Solaris Volume Manager, creating volumes
 - during custom JumpStart, example, 61
- standalone systems
 - custom JumpStart installation example, 40
 - profile examples, 61
- starting, check script, 75, 76
- state database
 - conceptual overview, 174-175
 - definition, 174-175
- state database replicas, 174
 - basic operation, 175
 - creating multiple on a single slice, 180
 - definition, 174
 - location, 180, 181
 - minimum number, 181
 - usage, 175
- stty command, 103, 107
- submirror, *See* RAID-0 volume
- SUNWCall group, 137-138
- SUNWCprog group, 137-138
- SUNWCreq group, 137-138
- SUNWCrnet group, 137-138
- SUNWCuser group, 137-138
- SUNWCXall group, 137-138
- swap file systems
 - diskless client swap space, 136
 - memory size and, 137
 - profile examples, 42
 - size determination, 137
- system_type profile keyword
 - description and values, 164
 - examples, 61

T

- testing
 - profiles, 71, 74-75
 - validating custom_probes files
 - testing custom_probes, 97
 - using check, 97
 - validating rules files
 - custom JumpStart example, 117
 - derived profiles and, 79
 - testing rules, 76
 - using check, 75, 76, 98
- timed out RPC error, 195
- tip line connection display requirements, 107

- tip line connection requirements, 103
- token ring card, booting error with, 195
- totaldisk probe keyword, 168
- totaldisk rule keyword, 125, 168
- transceiver cable problem message, 190
- troubleshooting
 - booting from network with DHCP, 196
 - booting from wrong server, 196
 - general installation problems
 - booting from the network with DHCP, 196
 - booting the system, 196

U

- UFS, 52
- Unknown client error message, 189
- upgrade
 - compared to initial installation, 19
 - custom JumpStart installation, 99
 - disk space recommendations, 25-29
 - failed upgrade, 200
 - overview of tasks, 15
 - profile keywords, 138, 147, 157
- usedisk profile keyword, description and values, 164

V

- validating
 - custom_probes file
 - testing, 97
 - using check, 98
 - rules files
 - custom JumpStart example, 117
 - derived profiles and, 79
 - testing rules, 76
 - using check, 75, 76, 98
- /var/sadm/system/logs/begin.log file, 78
- /var/sadm/system/logs/finish.log file, 79
- variables
 - SI_PROFILE, 78
 - SYS_MEMSIZE, 73

- version of Solaris software
 - installed probe keyword, 167
 - installed rule keyword, 123, 167
 - osname probe keyword, 168
 - osname rule keyword, 124, 168
- volcheck command, 51, 53, 54
- volume
 - conceptual overview, 173
 - naming conventions, 182
 - uses, 172
 - virtual disk, 172
- Volume Manager
 - copying, 51, 54
 - Solaris 10 Device Configuration Assistant diskette, 53
 - Solaris 10 Device Configuration Assistant image, 54

W

- WARNING: CHANGE DEFAULT BOOT DEVICE, 197
- WARNING: clock gained xxx days message, 190
- wrapping lines in rules files, 57