Installing Oracle® Solaris 11 Systems

Part No: E21798
March 2012 E21798–03

ORACLE®

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS. Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle America, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

Ce logiciel et la documentation qui l'accompagne sont protégés par les lois sur la propriété intellectuelle. Ils sont concédés sous licence et soumis à des restrictions d'utilisation et de divulgation. Sauf disposition de votre contrat de licence ou de la loi, vous ne pouvez pas copier, reproduire, traduire, diffuser, modifier, breveter, transmettre, distribuer, exposer, exécuter, publier ou afficher le logiciel, même partiellement, sous quelque forme et par quelque procédé que ce soit. Par ailleurs, il est interdit de procéder à toute ingénierie inverse du logiciel, de le désassembler ou de le décompiler, excepté à des fins d'interopérabilité avec des logiciels tiers ou tel que prescrit par la loi.

Les informations fournies dans ce document sont susceptibles de modification sans préavis. Par ailleurs, Oracle Corporation ne garantit pas qu'elles soient exemptes d'erreurs et vous invite, le cas échéant, à lui en faire part par écrit.

Si ce logiciel, ou la documentation qui l'accompagne, est concédé sous licence au Gouvernement des Etats-Unis, ou à toute entité qui délivre la licence de ce logiciel ou l'utilise pour le compte du Gouvernement des Etats-Unis, la notice suivante s'applique:

U.S. GOVERNMENT RIGHTS. Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle America, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

Ce logiciel ou matériel a été développé pour un usage général dans le cadre d'applications de gestion des informations. Ce logiciel ou matériel n'est pas conçu ni n'est destiné à être utilisé dans des applications à risque, notamment dans des applications pouvant causer des dommages corporels. Si vous utilisez ce logiciel ou matériel dans le cadre d'applications dangereuses, il est de votre responsabilité de prendre toutes les mesures de secours, de sauvegarde, de redondance et autres mesures nécessaires à son utilisation dans des conditions optimales de sécurité. Oracle Corporation et ses affiliés déclinent toute responsabilité quant aux dommages causés par l'utilisation de ce logiciel ou matériel pour ce type d'applications.

Oracle et Java sont des marques déposées d'Oracle Corporation et/ou de ses affiliés. Tout autre nom mentionné peut correspondre à des marques appartenant à d'autres propriétaires qu'Oracle.

Intel et Intel Xeon sont des marques ou des marques déposées d'Intel Corporation. Toutes les marques SPARC sont utilisées sous licence et sont des marques ou des marques déposées de SPARC International, Inc. AMD, Opteron, le logo AMD et le logo AMD Opteron sont des marques ou des marques déposées d'Advanced Micro Devices. UNIX est une marque déposée d'The Open Group.

Ce logiciel ou matériel et la documentation qui l'accompagne peuvent fournir des informations ou des liens donnant accès à des contenus, des produits et des services émanant de tiers. Oracle Corporation et ses affiliés déclinent toute responsabilité ou garantie expresse quant aux contenus, produits ou services émanant de tiers. En aucun cas, Oracle Corporation et ses affiliés ne sauraient être tenus pour responsables des pertes subies, des coûts occasionnés ou des dommages causés par l'accès à des contenus, produits ou services tiers, ou à leur utilisation.

120305@25097

Contents

Preface

Installing Oracle Solaris 11 Systems provides instructions for installing and configuring the Oracle Solaris operating system (OS) using any of the following methods:

- The Oracle Solaris LiveCD installer
- The Oracle Solaris interactive text installer
- The Oracle Solaris Automated Installer (AI) feature
- The Oracle Solaris SCI Tool interactive system configuration tool
- The `sysconfig`(1M) command line system configuration tool

All cases require access to a package repository on the network to complete the installation.

Who Should Use This Book

This book is for system administrators who want to install the Oracle Solaris 11 OS.

How This Book Is Organized

This book contains the following parts and chapters:

Part I, "Oracle Solaris 11 Installation Options," describes alternative installation methods to help you select the method that best fits your needs.

Part II, "Installing Using Installation Media":

- Chapter 2, "Preparing for the Installation"
- Chapter 3, "Using the LiveCD"
- Chapter 4, "Using the Text Installer"
- Chapter 5, "Automated Installations That Boot From Media "
- Chapter 6, "Unconfiguring or Reconfiguring an Oracle Solaris instance"

Part III, "Installing Using an Install Server," describes automated installations and related processes and tools.

- Chapter 7, "Automated Installation of Multiple Clients," describes how AI performs a hands-free installation of multiple SPARC and x86 client systems in a network.
- Chapter 8, "Setting Up an Install Server," describes how to set up a separate system to manage client installations.
- Chapter 9, "Customizing Installations," describes how to apply client selection criteria to different installation instructions and system configuration instructions so that different client systems are installed and configured differently.
- Chapter 10, "Provisioning the Client System," explains how to create custom installation instructions for different clients.
- Chapter 11, "Configuring the Client System," describes how to specify information needed to configure the client system after installation.
- Chapter 12, "Installing and Configuring Zones," describes how to specify installation and configuration of non-global zones as part of an AI client installation.
- Chapter 13, "Running a Custom Script During First Boot," explains how to create a script that is executed at first boot to perform additional installation or configuration of the client system.
- Chapter 14, "Setting Up Oracle Configuration Manager For Use By AI Client Systems," describes how to ensure that Oracle Configuration Manager works on your AI client installations.
- Chapter 15, "Installing Client Systems," gives the system requirements for AI clients and describes how to associate each client with the correct net image and installation and configuration instructions.
- Chapter 16, "Troubleshooting Automated Installations," discusses some possible failures and how to recover.

Related Information

Oracle Solaris 11 Installation Man Pages provides copies of the `aimanifest`(1M), `distro_const`(1M), `installadm`(1M), `js2ai`(1M), `ai_manifest`(4), and `dc_manifest`(4) man pages.

Creating a Custom Oracle Solaris 11 Installation Image explains how to use the Oracle Solaris Distribution Constructor tool to customize your installation image.

Creating and Administering Oracle Solaris 11 Boot Environments describes how to manage multiple boot environments on your Oracle Solaris system, including non-global zones.

Chapter 6, "Managing Services (Overview)," in *Oracle Solaris Administration: Common Tasks* describes the Oracle Solaris Service Management Facility (SMF) feature. You can use SMF profiles to configure your system.

Adding and Updating Oracle Solaris 11 Software Packages describes the Oracle Solaris Image Packaging System (IPS) feature, and how to find and install IPS packages. The `pkg(5)` man page describes the Image Packaging System in more detail. The `pkg(1)` man page provides more detail about how to find, install, update, and verify IPS packages.

Copying and Creating Oracle Solaris 11 Package Repositories describes how to create a local copy of an Oracle IPS package repository, or how to create your own custom repository.

See the Oracle Solaris 11 System Administration documentation for more information about how to administer Oracle Solaris 11 systems.

See the DHCP section of the isc.org web site for downloads and documentation for the Internet Systems Consortium (ISC) DHCP server.

Transitioning From Oracle Solaris 10 JumpStart to Oracle Solaris 11 Automated Installer provides information to help you migrate from JumpStart to AI, both of which are automated installation features of Oracle Solaris.

Access to Oracle Support

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

Typographic Conventions

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

TABLE P–1 Typographic Conventions

Typeface	Meaning	Example
`AaBbCc123`	The names of commands, files, and directories, and onscreen computer output	Edit your `.login` file. Use `ls -a` to list all files. `machine_name% you have mail.`
`AaBbCc123`	What you type, contrasted with onscreen computer output	`machine_name%` **`su`** `Password:`
aabbcc123	Placeholder: replace with a real name or value	The command to remove a file is `rm` *filename*.

TABLE P–1 Typographic Conventions *(Continued)*

Typeface	Meaning	Example
AaBbCc123	Book titles, new terms, and terms to be emphasized	Read Chapter 6 in the *User's Guide*. A *cache* is a copy that is stored locally. Do *not* save the file. **Note:** Some emphasized items appear bold online.

Shell Prompts in Command Examples

The following table shows the default UNIX system prompt and superuser prompt for shells that are included in the Oracle Solaris OS. Note that the default system prompt that is displayed in command examples varies, depending on the Oracle Solaris release.

TABLE P–2 Shell Prompts

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

PART I

Oracle Solaris 11 Installation Options

◆ ◆ ◆ CHAPTER 1

Overview of Installation Options

The Oracle Solaris software can be installed in a number of different ways depending on your needs. See the following overview of Oracle Solaris installation options.

Comparing Installation Options

The following chart compares the capabilities of the various installation options.

TABLE 1–1 Installation Options

Installation Option	Minimal Preparations	Requires Server	Install to Single or Multiple Systems	Installs Packages from a Package Repository
x86 only: Chapter 3, "Using the LiveCD"	Yes	No, installs from media	Single	No
Chapter 4, "Using the Text Installer"	Yes	No, installs from media	Single	No
"Performing a Text Installation Over the Network" on page 53	No	Yes, retrieves install image from server	Single	Yes
Chapter 5, "Automated Installations That Boot From Media "	No	Server needed if you want to customize install media, but not needed for installation	Single	Yes
Chapter 7, "Automated Installation of Multiple Clients"	No	Yes, server required	Single or multiple	Yes

In addition, you have the option of *Creating a Custom Oracle Solaris 11 Installation Image*, including custom LiveCD images, text installer images, and automated installation images.

Simple, Preset Installations

The GUI installer on the LiveCD and the text installer are simple, preset installer methods.

- Both installers can be used to install Oracle Solaris on the x86 platform. The text installer can also be used to install Oracle Solaris on the SPARC platform.
- Both installers are capable of functioning with a minimum of memory. To check memory requirements, see *Oracle Solaris 11 Release Notes*.
- Both installers enable you to select, create, or modify disk partitions during an installation.

The LiveCD contains a set of software that is appropriate for a desktop or laptop. The text install media contains a smaller set of software that is more appropriate for a general-purpose server system.

The text installer has the following advantages over the GUI installer:

- Enables you to install the operating system on either SPARC or x86 based systems.
- Can be used on systems that do not have, or do not require, graphics cards.
- Can require less memory than the GUI installer, depending on your system's specifications.
- Enables manual configuration of the network and naming services.
- If the network is setup to perform automated installations, you can perform a text installation over the network by setting up an install service on the network and selecting a text installation when the client system boots.

  Note – The package set installed by the text installer is the `solaris-large-server` package set. However, if you use the text installer over the network, a different, smaller package set, `solaris-auto-install`, is installed. After booting into the installed system, you should install the `solaris-large-server` package set.

- In addition to modifying partitions, the text installer enables you to create and modify VTOC slices within the Solaris partition.

For further information about performing a simple installation, see Part II, “Installing Using Installation Media.”

Installations Requiring Server Setup

You can perform a “hands-free” installation of the Oracle Solaris software on single or multiple client systems by using the Automated Installer (AI) feature.

Note – Each system requires network access because the installation process retrieves packages from a networked repository.

To use AI, you must first set up a server on your network. When a client system boots, the system gets installation specifications from the server, retrieves software packages from an Oracle Solaris package repository, and the software is installed on the client system.

AI can perform "hands-free" automatic network installations on both x86 and SPARC based client systems. The install clients can differ in architecture, disk and memory capacity, and other characteristics. The installations can differ in network configuration, packages installed, and other specifications.

For further information, see Part III, "Installing Using an Install Server."

Once you have an AI server set up, you have two additional installation options beyond the "hands-free" network installations.

- You have the option to perform an interactive text installation over the network. The interactive installation allows you to further customize the installation specifications for any particular system.

  For further information, see "Performing a Text Installation Over the Network" on page 53.
- The setup for AI includes downloading AI images and storing them on the network or locally. You can burn the image to removable media, such as a CD, DVD, or for x86 installations, a USB flash drive. Then, you can boot the AI media directly on each of your systems to initiate an automated installation. Installations that use AI media are non-interactive.

  For instructions, see Chapter 5, "Automated Installations That Boot From Media ."

Additional Options

In addition to the installation options already described, you have the following options for installing and modifying the Oracle Solaris operating system.

Create Custom Installation Images
: You can create a preconfigured Oracle Solaris installation image using the distribution constructor tool. The tool takes a customized XML manifest file as input and builds an installation image that is based on the parameters specified in the manifest file. You can build a custom image based on any of the default installation images. For example, you could build a custom text installer image or a custom GUI installer image. For information, see *Creating a Custom Oracle Solaris 11 Installation Image*.

Updating an Installed Oracle Solaris 11 System

You cannot use an installer to update an existing Oracle Solaris 11 installed system. Instead, you need to use `pkg` utility to access package repositories and download new or updated software packages for your system. For further information, see *Adding and Updating Oracle Solaris 11 Software Packages*.

PART II

Installing Using Installation Media

You can install the Oracle Solaris operating system on a single system with a minimal amount of preparation by using either the GUI installer or the text installer. You can perform a text installation locally or over the network. Additionally, if you are using the Automated Installer (AI) feature, you can create an automated installation image, burn it to media, and use that media to install a single system. You, also, have the option to unconfigure and reconfigure an installed system.

See the following:

◆ ◆ ◆ CHAPTER 2

Preparing for the Installation

Before installing your system, review the following information.

System Requirements for LiveCD and Text Installations

The following table outlines requirements for installing the Oracle Solaris 11 release using a LiveCD installation image or a text installation image.

Requirement	Description
Memory	To check the minimum memory requirement for the current release, see *Oracle Solaris 11 Release Notes*. **Note –** The text installer requires less memory than the LiveCD installer. The exact minimum requirement varies depending on system specifications. But, if your system does not have enough memory to run the GUI installer, use the text installer instead.
Disk space	To check the disk space requirements for the current release, see *Oracle Solaris 11 Release Notes*.

Preparing a Boot Environment for Installing Multiple Operating Systems

If you are installing Oracle Solaris as part of a multiple boot environment, review the following specifications for various operating systems.

TABLE 2–1 Multiple Operating System Environments

Existing Operating System	Description
Microsoft Windows	Set up sufficient disk space for installing the Oracle Solaris release. All versions of Oracle Solaris for the x86 platform use the GNU Grand Unified Bootloader (GRUB). Oracle Solaris recognizes Windows and ensures that Windows partitions remain unchanged during an installation. When the installation completes, and the system reboots, the GRUB menu displays both the Windows and the Oracle Solaris boot entries. **Note –** The Oracle Solaris operating system on x86 systems now stores Universal Time Coordinated (UTC) time in the Real Time Clock (RTC) or hardware clock. Previously, the operating system stored local time in RTC on x86 systems.
Linux, or Windows and Linux	If you have the Linux operating system, or both Linux and Windows operating systems, installed on your x86 based system, before installing Oracle Solaris, save a copy of the `menu.lst` file. After the installation, you will need to edit the `menu.lst` file to add the Linux information from the previous installation. For instructions, see "Modifying Boot Entries and Parameters by Editing the menu.lst File" in *Booting and Shutting Down Oracle Solaris on x86 Platforms*. **Note –** When installing Oracle Solaris on a system that also has the Linux operating system installed, the Oracle Solaris partition *must* precede the Linux `swap` partition.
Solaris 10 OS	The installer on the LiveCD cannot be used to install multiple instances of Oracle Solaris. The text installer, however, supports multiple instances of the Oracle Solaris operating system on the same partition, as long as the instances are on different slices. The LiveCD and text installers can be used to replace the Solaris 10 1/06 and later releases on an existing system that has multiple instances of Oracle Solaris installed. **Note –** If you need to preserve a specific Solaris Volume Table of Contents (VTOC) slice in your current operating system, use the text installer.

TABLE 2–1 Multiple Operating System Environments *(Continued)*

Existing Operating System	Description
Extended partitions	If you have another operating system on an extended partition, the existing extended partition does not need to be changed during an installation. You can create, resize, or delete an extended partition when you install Oracle Solaris by using either the LiveCD GUI installer, the text installer, or the Automated Installer. You can also choose to install Oracle Solaris on a logical partition within an extended partition.

Partitioning Your System

This section provides guidelines for partitioning a system prior to installation or during an interactive installation. It also describes how to set up partitions on x86 and Solaris VTOC slices.

Guidelines for Partitioning a System Prior To Installation

When installing Oracle Solaris from the LiveCD ISO image or from the text installer image, you can use the entire disk, or you can install the operating system on an x86 partition. In addition, with the text installer, you can install the operating system on a SPARC slice.

On x86 based systems, the installer uses GRUB, which supports booting multiple operating systems on one or more drives. You can create a partition for installing Oracle Solaris prior to installation as well as during an installation. After partitioning and installing the various operating systems, you can deploy any of them by selecting the appropriate menu entry in the GRUB menu at boot time.

Caution – Remember to back up your system prior to partitioning the hard drive.

You can use the `fdisk` command to create or modify an Oracle Solaris `fdisk` partition. For instructions, see "How to Create a Solaris fdisk Partition" in *Oracle Solaris Administration: Devices and File Systems*. See also the `fdisk`(1M) man page.

Alternately, you can use commercial products or open-source tools to partition your hard drive.

Note – If you create Linux-swap partitions, note that Linux-swap uses the same partition ID that Oracle Solaris uses. During the installation, in the disk partitioning step, you can change the Linux-swap partition to an Oracle Solaris partition.

Guidelines for Partitioning a System During an Interactive Installation

On an x86 based system, you can select, create, or modify partitions during a GUI installation or a text installation. For the text installer *only*, you can select, create, or modify VTOC slices during an interactive installation.

When installing Oracle Solaris, note the following important information about disk partitioning:

- The installation overwrites the whole disk layout, if any of the following is true:
  - The disk table cannot be read.
  - The disk was not previously partitioned.
  - You select the entire disk for the installation.
- If an existing `fdisk` partition is on an Oracle Solaris system and you make no modifications to the existing partitions, the installation overwrites the `fdisk` partition *only*. Other existing partitions are not changed.
- Only one Solaris partition is allowed.
- A Solaris partition must be used for the installation.
- If there is an existing Solaris partition, that partition is selected by default. The partition can be a logical partition within an existing extended partition.
- Changes you make to disk partitioning or slices are not implemented until you finish making the installer panel selections and the installation begins. At any point prior to the installation, you can cancel your changes and restore the original settings.
- If the existing partition table cannot be read, proposed partitioning information is displayed.

Caution – In this case, all of the existing data on the disk is destroyed during the installation.

- During the installation, if you select the Partition the Disk option, the panel displays the existing `fdisk` partitions for the selected disk. Up to four primary partitions are displayed in the same order that they are laid out on the disk. Unused disk space is displayed for these primary partitions. The partition type, current size, and maximum available disk space for each partition are also displayed. If an extended partition exists, its logical partitions are also displayed in the disk layout order within the extended partition.

- An fdisk partition cannot be larger than 2 TB, in order to be used for installing the OS. Disks or partitions that do not have enough space for a successful installation are labeled as such.

x86: Setting Up Partitions During an Interactive Installation

For installations on the x86 platform, you can make changes to disk partitioning by directly editing the entries in the installation screens. As you proceed through the installation, the recommended and minimum sizes for installing the software are also displayed.

The following table describes the disk partitioning options. Use this table to help you determine which option best suits your needs.

TABLE 2–2 Options for Partitioning a Disk During an Interactive Installation

Partitioning Option	Description and User Action (if required)
Use the existing Solaris partition.	This option installs Oracle Solaris 11 operating system on the existing Solaris partition using its current size. Select the Partition a Disk option. No other changes are required.
Create a new Solaris partition.	If there is currently no existing Solaris partition on the system, you can create a new Solaris partition. To do so, select a primary partition or a logical partition and then change its type to Solaris. During an installation, this modification erases the existing partition contents.
Increase the space that is allocated to a Solaris partition and install on that partition.	If enough disk space is available, you can increase the size that is allocated to a Solaris partition before installing the software on that partition. The available space contains any contiguous unused space before or after the selected partition. If you enlarge the partition, unused space after the partition is used first. Then, unused space before the partition is used, which changes the starting cylinder of the selected partition.
Install Oracle Solaris 11 operating system on a different primary partition.	You can install the operating system on a different primary partition. To do so, you must first change the existing Solaris partition type to Unused. You can then select another partition and change its type to Solaris. During an installation, this modification erases the existing partition contents for both the previous Solaris partition and the new Solaris partition.
Create a new Solaris partition within an extended partition.	You can create a new Solaris partition within an extended partition. If a Solaris partition already exists, change its type to Unused. Then, to create a new extended partition, change the partition type to Extended. You can resize the extended partition and then change one of the logical partitions in the extended partition to a Solaris partition. Also, you can enlarge the logical partition up to the size of the extended partition that contains that logical partition.

TABLE 2–2 Options for Partitioning a Disk During an Interactive Installation *(Continued)*

Partitioning Option	Description and User Action (if required)
Delete an existing partition.	You can delete an existing partition by changing its type to Unused. During an installation, the partition is destroyed, and its space is made available when resizing adjacent partitions.

Setting Up VTOC Slices During a Text Installation

For text installations on the SPARC platform, you can modify VTOC slices during the installation. For text installations on the x86 platform, you can modify a slice within a partition if that partition has not already been modified during the installation.

When setting up VTOC slices, keep the following in mind:

- The installer displays the existing slices. The slices are displayed in the order in which they are laid out. The current size and maximum available size for each slice are also displayed.
- Oracle Solaris must be installed in an Oracle ZFS root pool. By default, the slice that contains the root pool is labeled `rpool` by the installer. If you want to install the operating system on a slice that does *not* contain the root pool, change the type for that slice to `rpool` in the installer. During the installation, a ZFS root pool will be created on that slice.

  Note – Because only one ZFS pool can be named `rpool`, if an `rpool` is already on the device, the installer will name any new pool using the format `rpool#`.

- The size of a slice can be increased up to the maximum available size. To make more space available, you can change an adjoining slice to Unused, thereby making its space available to adjacent slices.
- If the slice is not explicitly altered, the content of the slice is preserved during the installation.

The following table describes the options for modifying slices during a text installation.

TABLE 2–3 Options for Modifying VTOC Slices During a Text Installation

Option	Description and User Action (if required)
Use an existing slice.	This option installs the Oracle Solaris 11 operating system on an existing VTOC slice using its current size. Select the target slice, then change its type to `rpool`.
Resize a slice.	You can change the size only of a newly created `rpool` slice. Type the new size in the field.
Create a new slice.	Select an unused slice and change its type. For example, change Unused to `rpool`.

TABLE 2–3 Options for Modifying VTOC Slices During a Text Installation *(Continued)*

Option	Description and User Action (if required)
Delete an existing slice.	Changing the slice type to Unused. During the installation, the slice is destroyed and its space is made available for resizing adjacent slices.

Ensuring That You Have the Proper Device Drivers

Before installing Oracle Solaris, determine whether your system's devices are supported. The Hardware Compatibility Lists (HCL) at `http://www.oracle.com/webfolder/technetwork/hcl/index.html` provides information about hardware that is certified or reported to work with Oracle Solaris. The Solaris on x86 Platforms Device Support tool tells you which Oracle Solaris driver supports the various x86 components.

You can also use the following utilities to determine whether a device driver is available:

- **Oracle Device Detection Tool**

  The Oracle Device Detection Tool reports whether the current release supports the devices that have been detected on your system. This tool runs on many different systems, including several different Solaris 10 releases, Windows, Linux, Mac OS X, and FreeBSD. There is a link to the Oracle Device Detection Tool on the HCL (`http://www.oracle.com/webfolder/technetwork/hcl/index.html`). For instructions on using the tool, see "How to Use the Oracle Device Detection Tool" on page 27.

- **Device Driver Utility**

  The Device Driver Utility provides the same information as the Oracle Device Detection Tool. This utility is available on the desktop for Oracle Solaris 11 systems. And, the utility is available through the text installer menu options.

Note – The Device Driver Utility can require at least 1.5 GB of memory. If your system has an adequate complement of devices to perform an installation, first complete the installation, then boot the installed hard disk before running the Device Driver Utility. Then, the utility can take advantage of the swap space on the installed system.

▼ How to Use the Oracle Device Detection Tool

You can use the Oracle Device Detection Tool to determine whether the current release includes drivers for all of the devices on your system.

1 **In a web browser, go to `http://www.oracle.com/webfolder/technetwork/hcl/hcts/device_detect.html`.**

2. **In the Using the Device Detection Tool section, click the Start Oracle Device Detection Tool option.**
3. **Accept the license agreement.**
4. **Click the ddtool download link.**
5. **Select the Open with JavaWS option, then select Run.**

   The tool runs but it is not installed on your system.
6. **Select the Target Operating System for which you want to check driver availability.**

Tip – For additional information, click the Help button.

Device Driver Utility Overview

An alternative to the Oracle Device Detection Tool, the Device Driver Utility provides information about the devices on your system and the drivers that manage those devices. The utility reports whether the currently booted operating system has drivers for all of the devices that are detected in your system. If a device does not have a driver attached, the Device Driver Utility recommends a driver package to install.

You can also use the Device Driver Utility to submit your system information to the HCL at `http://www.oracle.com/webfolder/technetwork/hcl/index.html`. Your system and its components are then listed on the HCL as "Reported to Work".

This section describes the following tasks:

- "How to Start the Device Driver Utility" on page 28
- "How to Install Missing Drivers" on page 29
- "How to List Your System in the HCL" on page 31

▼ How to Start the Device Driver Utility

The Device Driver Utility runs automatically when you boot an installation image. You can also manually start the Device Driver Utility after you have installed Oracle Solaris.

- **Start the Device Driver Utility by using one of the following methods:**

  - **Boot the LiveCD installation image.**

    When you boot the LiveCD ISO image, the Device Driver Utility runs in the background. If a missing driver is found in an IPS package from the default publisher, the Device Driver

Utility installs that driver package automatically. If any other drivers are missing, the Device Driver Utility prompts you to display the utility window so that you can review the report and install any additional missing drivers.

- **Manually start the Device Driver Utility from the desktop of the LiveCD.**

  To manually start the Device Driver Utility from the desktop of the LiveCD, double-click the Device Driver Utility icon on the desktop. Or, choose Applications → System Tools → Device Driver Utility from the main menu.

- **Boot the Oracle Solaris text installer image.**

  To start the Device Driver Utility from the text installer, choose Install Additional Drivers from the initial menu.

  Note – Automatic networking is set up by default when the text installer boots. If you are using DHCP, no further network setup is necessary to use the Device Driver Utility. If you are not using DHCP, select the Shell option on the initial menu, then use the appropriate commands to manually configure your network settings before using the Device Driver Utility.

- **Start the Device Driver Utility on an installed system.**

  To start the Device Driver Utility from the desktop of an installed system, choose Applications → System Tools → Device Driver Utility from the main menu.

The Device Driver Utility scans your system and then displays a list of the devices that are detected. For each device that is detected, the list displays information such as the manufacturer, the model, and the name of the driver that is currently managing the device.

Next Steps If the utility detects a device that does not have a driver attached, that device is selected on the device list. You can display more information about the device and install the missing driver. See "How to Install Missing Drivers" on page 29.

▼ How to Install Missing Drivers

If the utility detects a device that does not have a driver attached, that device is selected on the device list. You can display more information about the device and install the missing driver.

1. **In the Device Driver Utility list, right-click the device name, then choose Show Details from the popup menu.**

   The Device and Driver Details window is displayed. It shows the device name, vendor name, node name, driver name, and other detailed information about the device.

2. **To display more details about a missing driver, click the Info link for the selected device.**

If no driver is currently managing the device, the Driver column of the device list displays a status for the driver of that device. The missing driver is shown as belonging to one of the following categories:

- `IPS` – One of your configured IPS package repositories.
- `SVR4` – A System V Revision 4 (SVR4) package.
- `DU` – A DU package.
- `UNK` – The Device Driver Utility cannot locate an Oracle Solaris driver for this device.

Tip – For additional information, click the Help button.

---

3 Install the missing driver.

- **For an IPS driver:**

  a. Click the Info link in the corresponding row of the table to display information about the IPS package that contains the driver for the device.

  The text field for the Package radio button is populated with the relevant package information. The correct publisher is specified.

  b. Click the Install button to install the package.

  - **If the Info link lists an IPS package from a publisher that is not configured:**

    i. Select Add Repository from the repositories menu.

    The Repositories manager window is displayed.

    ii. Add the name and URI of the new repository, then click Add.

  - **If the Package field is not populated, type the name of the IPS package from the Info link, then click Install.**

- **For an SVR4 or DU driver:**

  - **If a URL for the package is provided, type the URL in the File/URL field, then click Install.**

  - **If you have a copy of the package on your system, click the Browse button and select the package, then click Install.**

- **If the driver status is displayed as UNK, do the following:**

  a. Select the name of the device that you want this driver to manage.

b. Type the relevant package information in either the Package field or the File/URL field, then click Install.

c. (Optional) To share information about a driver that works for the device, click the Submit button.

Next Steps When you are working in the Device Driver Utility, you can share information with other users about any driver that you've found that works for a particular device. See "How to List Your System in the HCL" on page 31.

How to List Your System in the HCL

You can share information with other users about any driver that you've found that works for a particular device as follows:

1 Start the Device Driver Utility.

See "How to Start the Device Driver Utility" on page 28.

2 To list your system and its components as "Reported to work" on the HCL click the Submit button.

The Submit Information To Hardware Compatibility List window opens. This window displays all of the information that was collected about your system.

a. Select the System Type.

b. Type the appropriate information in any fields that were not automatically populated.

- Manufacturer Name – The name of the system maker, for example, Toshiba, Hewlett-Packard, or Dell.
- The complete model number.

  The BIOS/Firmware Maker is the information on the BIOS Setup screen that is usually displayed while the system is booting.
- The CPU Type – The name of the CPU maker.

c. Provide your name and email address.

d. In the General Notes field, add any additional comments, then click Save. Send the saved file to `device-detect-feedback_ww@oracle.com`.

Using Oracle Configuration Manager

Upon rebooting after an installation from a LiveCD or after a text installation, anonymous system configuration information is uploaded to Oracle Support by the Oracle Configuration Manager. My Oracle Support receives information about the installed system's configuration, but does not receive any of your customer information.

Specifically, during the first reboot, an Oracle Configuration Manager service runs for the first time and attempts to register the system with the registration server. If this registration succeeds, an upload of the anonymous configuration information is performed. Also, upon successful registration, an internal scheduler is started. Thereafter, configuration data is uploaded under control of the scheduler. On subsequent reboots, configuration data is not sent as part of service startup. The service recognizes that the system is already registered and simply launches the scheduler. Scheduling may be tuned by using `/usr/sbin/emCCR`. See the `emCCR`(1M) man page and the *Oracle Configuration Manager Installation and Administration Guide*.

When performing an installation from a LiveCD or a text installation, you have the following options.

- Allow the default anonymous registration of your configuration information to occur automatically.
- Disable Oracle Configuration Manager functions at the end of the installation, before rebooting the system.

  For disabling instructions, see the following:

  - LiveCD installation: Step 13
  - Text installation: Step 16

Note – If you are using an install server for purposes such as creating an installation image and burning it to media or initiating a text installation over the network, you have different options for administering the Oracle Configuration Manager. See Chapter 14, "Setting Up Oracle Configuration Manager For Use By AI Client Systems."

Whether you chose to allow or cancel the default anonymous registration, you can still choose to register or re-register your system later with the Oracle Configuration Manager.

- If you already registered anonymously, you may choose to provide your My Oracle Solaris (MOS) account information as part of registration later in order to facilitate future support. When customer configuration data is uploaded on a regular basis, customer support representatives can analyze this data and provide better service. For example, when you log a service request, the support representative can associate the configuration data directly with that service request. The customer support representative can then view the list of your systems and solve problems accordingly.
- If you cancelled the default registration, you may choose to register anonymously or to perform a full customer registration later.
- If automatic registration was unable to complete due to a network proxy requirement, you may register anonymously or with your MOS account information.

You can register by using the `configCCR` utility (`/usr/sbin/configCCR`) in interactive mode. After completing registration, you can enable the service as follows:

```
# svcadm enable system/ocm
```

For further information, see the following:

- Chapter 5, "Working With Oracle Configuration Manager," in *Oracle Solaris Administration: Common Tasks*
- `configCCR`(1M) man page
- *Oracle Configuration Manager Installation and Administration Guide*

◆ ◆ ◆ CHAPTER 3

Using the LiveCD

This chapter describes how to perform installations using a LiveCD image.

Installing With the GUI installer

When installing Oracle Solaris 11 software, consider the following information:

- See "System Requirements for LiveCD and Text Installations" on page 21.
- If you are installing Oracle Solaris on an x86 based system that will have more than one operating system installed in it, you can partition your disk during the installation process. See the instructions for using the GUI installer or the text installer.

  Note – The installer on the LiveCD ISO image is for x86 platforms only.

  Alternatively, you can use the `fdisk` command or a third-party partitioning tool to create a new partition or make adjustments to preexisting partitions prior to an installation. See "Guidelines for Partitioning a System Prior To Installation" on page 23.

  For more information about preparing an environment for the installation of specific operating systems, see "Preparing a Boot Environment for Installing Multiple Operating Systems" on page 22.
- The GUI installer cannot upgrade your operating system. However, after you have installed the Oracle Solaris 11 release, you can update all of the packages on your system that have available updates by using the Image Packaging System. See *Adding and Updating Oracle Solaris 11 Software Packages*.
- The GUI installer can perform an initial installation on the whole disk or on an Oracle Solaris x86 partition on the disk.

Caution – The installation overwrites all of the software and data on the targeted device.

Default Settings With GUI Installer

The default network and security settings used by the GUI installer on the LiveCD are as follows:

- Oracle Solaris is automatically networked by using DHCP, with Domain Name System (DNS) resolution.

  The DNS domain and server Internet Protocol (IP) addresses are retrieved from the DHCP server.
- Automatic networking enables IPv6 autoconfiguration on active interfaces.
- The NFSv4 domain is dynamically derived.

▼ How to Perform a GUI Installation

1 Complete any preliminary tasks.

a. If you do not have the LiveCD, download the LiveCD ISO image.

To download the Oracle Solaris LiveCD ISO image, go to `http://www.oracle.com/technetwork/server-storage/solaris11/downloads/index.html`.

Note – Alternately, if you want to burn the image to a USB flash drive, download a USB image.

After you download the image, do one of the following:

- **Copy the image to removable media, such as a CD, DVD, or USB flash drive.**

  Note – For USB images, you need the `usbcopy` utility, in order to copy the image to a USB flash drive. You can add this utility to your system by installing the `pkg:/install/distribution-constructor` package.

- **Save the image to your system and then run it in a virtual machine.**

b. **Check the requirements and limitations for running the installer on your system:**

   i. **Verify that your system meets all of the necessary system requirements.**
   See "System Requirements for LiveCD and Text Installations" on page 21.

   ii. **Verify that you have all of the necessary device drivers.**
   See "Ensuring That You Have the Proper Device Drivers" on page 27.

c. **Choose one of the following options for installing the Oracle Solaris 11 release:**

   - **If the Oracle Solaris 11 release is the only operating system that is to be installed on your system, see "Installing With the GUI installer" on page 35.**

   - **If you are setting up an environment that supports the installation of multiple operating systems:**

     i. **Review the specifications in "Preparing a Boot Environment for Installing Multiple Operating Systems" on page 22.**

     ii. **Back up your system.**

     iii. **If you need to partition your system prior to the installation, see "Partitioning Your System" on page 23.**

     Note – If you have previously installed the Linux operating system, you will need to save a copy of the existing `menu.lst` file to a USB drive for use after the installation. For instructions, see "Modifying Boot Entries and Parameters by Editing the menu.lst File" in *Booting and Shutting Down Oracle Solaris on x86 Platforms*.

2 **Insert the installation media and boot the system.**

On the LiveCD, when the GRUB menu is displayed, the default entry is automatically used unless you select another option.

Note – If your system's graphics card is not supported by the LiveCD, or your system does not have a graphics card, the system boots in console mode when you insert the LiveCD. In this case, you cannot perform a GUI installation. See "What to Do If Your System Boots in Console Mode" on page 41.

- If you are prompted to log in to the LiveCD, the user name and password are both `jack`.
- The `root` password is `solaris`.

3 **Make keyboard and language selections or accept the default English options.**

Note – The language and keyboard selections set the defaults for the installer and for the installed system. You can modify the locale on the login panel for the installed system.

4 **Install any missing drivers that are required for installation.**

When you boot the LiveCD, if any drivers are missing, a prompt is displayed. Follow the instructions for accessing the Device Driver Utility to locate and install any drivers that are required for the installation.

5 **On the LiveCD desktop, double-click the Install Oracle Solaris icon to start the GUI installer.**

6 **In the Welcome panel, select Next.**

Note – You can review the Release Notes for the current release on this panel.

7 **In the Disk panel, if there are multiple installation targets shown, select an installation target or accept the default. Then, specify whether to install the operating system on the whole disk or on a partition on the disk.**

Optionally, you can modify the partition layout. For instructions, see the "Guidelines for Partitioning a System During an Interactive Installation" on page 24.

At any point during this phase of the installation, you can revert to the original settings.

Caution – If the existing partition table cannot be read, the panel displays proposed partitioning. In this instance, all of the data on the disk is destroyed during the installation.

8 **Select the target time zone. Then adjust date and time to match your current local time.**

The installer uses the time zone from the system's internal settings as the initial default, if possible. When you select your location on the map, the installer uses that information to set the date, time and time zone.

9 **Complete the user settings.**

- Type a user name and password.

  To complete the user account setup, a login name and password are required. The login name must begin with a letter and can contain only letters and numbers.

Note – The user account that you create will have administrative privileges.

On an installed system, the initial `root` password defaults to the user account password that you enter here. The first time you use the `root` password, you will be prompted to change the password.

- Type a computer name or accept the default. This field cannot be blank.

10 Review the installation specifications.

Review the specifications in the Installation Summary panel. If necessary, go back and make any required changes before starting the installation.

11 Install the system using the specifications you have provided.

The Oracle Solaris installation process begins.

Caution – Do not interrupt an installation that is in progress. An incomplete installation can leave a disk in an indeterminate state.

12 Review the installation logs.

The Installation Results panel provides access to installation logs that you can review.

13 (Optional) If you want to cancel anonymous registration of the installed system with Oracle Configuration Manager, perform the following steps to mount the newly-created boot environment and add an "opt-out" file to that boot environment before rebooting the system.

Caution – By default, the system configuration of the installed system is sent to the Oracle Configuration Manager. This is an anonymous registration with no customer information provided.

The anonymous registration will be automatic upon reboot after the initial installation, but you may cancel the registration per the following directions after the installation and prior to rebooting the installed system.

If you do not opt-out at installation time you may still suspend the service at any later time.

For further information, see "Using Oracle Configuration Manager" on page 32.

a. Before rebooting the installed system, press Quit to exit the installer.

b. Open a terminal window.

c. Assume the root role.

Note – Prior to rebooting the newly-installed system, the default root password is "solaris," even if you changed the computer name in the User panel.

d. Mount the newly-created boot environment as in the following example.

```
# beadm mount solaris /a
```

The boot environment name prior to rebooting is, by default, "solaris."

e. In an editor, create a new file in the mounted boot environment and name the file `/a/etc/svc/profile/site/ocm.xml`.

For example, type the following:

```
# vi /a/etc/svc/profile/site/ocm.xml
```

f. Enter the following contents into the file, save the file, and exit the file.

```
<?xml version='1.0'?>
<!DOCTYPE service_bundle SYSTEM '/usr/share/lib/xml/dtd/service_bundle.dtd.1'>
<service_bundle type='profile' name='ocm'
 xmlns:xi='http://www.w3.org/2003/XInclude'>
   <service name='system/ocm' type='service' version='1'>
      <instance name='default' enabled='false'/>
   </service>
</service_bundle>
```

This file disables the default SMF service and changes the property to "opt-out" from an anonymous registration.

g. Unmount the boot environment, as shown in this sample command.

```
# beadm unmount solaris
```

h. Exit the terminal.

Note – After the installation and reboot, you can choose to register your system by removing this file then enabling the service as follows:

```
# svcadm enable system/ocm
```

This command performs an anonymous registration.

If you wish to associate the system's configuration data with your MOS account, or if your site requires use of a network proxy, you must use the `configCCR` command. See "Using Oracle Configuration Manager" on page 32.

14 Reboot the system, or quit the installer and shut down the system.

After a successful installation, reboot the system or exit the installer and shut down the system.

Eject the LiveCD as the next system boot begins. Or, select the "Boot from Hard Disk" option in the GRUB menu.

If the installation fails, you can view the installation log and exit the installer.

What to Do If Your System Boots in Console Mode

If your system's graphics card is not supported by the LiveCD, or your system does not have a graphics card, the system boots in console mode when you insert the LiveCD. In this case, you cannot perform a GUI installation.

Your two alternatives are as follows:

- Use the text installer image instead of the LiveCD ISO image.

  You can run the text installer on the local console without network access. See Chapter 4, "Using the Text Installer."
- Perform a remote installation using the following procedure.

Note – If you use this option, you do not need to download the text installer image. However, note that this option requires remote `ssh` access and a target system that has an X server running.

▼ How to Install Oracle Solaris From the LiveCD If Your System Boots in Console Mode

Before You Begin For this procedure, two networked systems are required: the system on which the LiveCD was booted (target system) and a remote system, from which the installation will be performed. Both systems must have network access. It is not required that the two systems be on the same subnet. However, the target system must be reachable from the remote system. Also, the remote system must be running an OS that supports a graphical desktop.

1. **On the system to be installed, insert the LiveCD, then boot the system.**

2. **At the console login, type the default login and password.**

   The default user login and password for Oracle Solaris is `jack`.

3. **Become the `root` user.**

   ```
   $ su root
   Password: solaris
   ```

   The `root` password is `solaris`.

4 **Enable the service for the `ssh` remote login program.**

```
# svcadm enable ssh:default
```

5 **Display the IP address that is assigned by DHCP to the target system.**

```
# ifconfig -a
```

6 **On the remote system, open a terminal window, then type:**

```
$ ssh -X ip-address-of-target -l jack
```

where *ip-address-of-target* is the output of the `ifconfig -a` command that you ran on the target system.

Running this command on the remote system opens a secure shell, enabling you to access the target system to use the GUI installer.

7 **Assume the `root` role.**

```
$ su root
Password: solaris
```

Note – The default root password prior to installation is "solaris."

---

8 **Run the GUI installer:**

```
# /usr/bin/gui-install
```

Note – Installer graphic display may be imperfect using this method.

---

9 **After the installation completes, reboot the target system.**

See Also See *Oracle Solaris Administration: Common Tasks* for information about the following topics:

- Managing user accounts and groups
- Booting and shutting down a system
- Managing services
- Managing hardware faults
- Managing system processes
- Troubleshooting general system problems such as the following:
  - What to do if rebooting fails
  - What to do if you forgot the root password
  - What to do if a system hangs

Adding Software After LiveCD Installation

To add software packages after you have installed the operating system, use the `pkg` commands as described in the `pkg`(1) man page and in Chapter 12, "Managing Software Packages (Tasks)," in *Oracle Solaris Administration: Common Tasks*. Or, you can use the Oracle Solaris Package Manager GUI tool to install additional software. On the desktop menu, go to System>Administration>Package Manager.

Use the `pkg` commands or the Package Manager tool to find the names of packages you might want to install, get more information about the packages, and install the packages.

Optionally, you can install into a new boot environment, so that you can continue to use your current image if the new installation has problems.

With the `pkg install` command, you should use the `-nv` option first to see what the package installation will look like prior to actually installing the packages. After you have identified the packages you want to install and examined the output from the `pkg install` command with the `-nv` option, issue a command similar to the following to install additional software.

```
# pkg install --require-new-be --be-name newBEname packagename
```

This sample command include options to require creation of a new boot environment, and specifies a package to be installed.

If you do not have a GUI desktop and you want to install the Oracle Solaris desktop, install the `solaris-desktop` package.

◆ ◆ ◆ CHAPTER 4

Using the Text Installer

You can perform an interactive text installation on individual SPARC and x86 client systems. Additionally, if you have set up your network for automated installations, you can perform a text installation over the network.

Installing With the Text Installer

When installing the Oracle Solaris 11 release, consider the following information:

- See "System Requirements for LiveCD and Text Installations" on page 21.
- If you are installing Oracle Solaris on an x86 based system that will have more than one operating system installed in it, you can partition your disk during the installation process.

  Alternatively, you can use the `fdisk` command or a third-party partitioning tool to create a new partition or make adjustments to preexisting partitions prior to an installation. See "Guidelines for Partitioning a System Prior To Installation" on page 23.

  For more information about preparing an environment for the installation of specific operating systems, see "Preparing a Boot Environment for Installing Multiple Operating Systems" on page 22.
- The Oracle Solaris 11 installers cannot upgrade your operating system. However, after you have installed the Oracle Solaris 11 release, you can update all of the packages on your system that have available updates by using the Image Packaging System. See *Adding and Updating Oracle Solaris 11 Software Packages*.
- The text installer can perform an initial installation on the whole disk, an Oracle Solaris x86 partition, or a SPARC slice.

Caution – The installation overwrites all of the software and data on the targeted device.

- The LiveCD contains a set of software that is appropriate for a desktop or laptop. The text install media contains a smaller set of software that is more appropriate for a general-purpose server system. In particular, the text installer does not install the GNOME desktop. To install additional packages after an installation with the text installer, see "Adding Software After Text Installation" on page 53.

Networking Configuration With Text Installer

The networking panel in the text installer provide users with the following options.

- Automatically – Configures target system with automatic NCP, similar to the LiveCD installer's method.
- Manually – Selects "DefaultFixed" NCP and provides for static IPv4 configuration of one network interface (NIC). IPv4 default route and IPv6 autoconfiguration are enabled for that chosen NIC. This option also provides for manual configuration of DNS, NIS and LDAP naming services.
- None – Selects "DefaultFixed" NCP and configures loopback interfaces only.

▼ How to Perform a Text Installation

1 Complete any preliminary tasks.

a. If you do not have the text installer image, download the image.

To download the Oracle Solaris text installer ISO image, go to `http://www.oracle.com/technetwork/server-storage/solaris11/downloads/index.html`.

Note – If you want to burn the image to a USB flash drive, download a USB image.

After you download the image, do one of the following:

- **Copy the image to removable media, such as a CD, DVD, or USB flash drive.**

  Note – For USB images, you need the `usbcopy` utility, in order to copy the image to a USB flash drive. You can add this utility to your system by installing the `pkg:/install/distribution-constructor` package.

- **Save the image to your system and then run it in a virtual machine.**

b. Check the requirements and limitations for running the installer on your system:

i. Verify that your system meets all of the necessary system requirements.

See "System Requirements for LiveCD and Text Installations" on page 21.

ii. Verify that you have all of the necessary device drivers.

See "Ensuring That You Have the Proper Device Drivers" on page 27.

c. If you are setting up an environment that supports the installation of multiple operating systems:

i. Review the specifications in "Preparing a Boot Environment for Installing Multiple Operating Systems" on page 22.

ii. Back up your system.

iii. If you need to partition your system prior to the installation:

Review the guidelines in Chapter 2, "Preparing for the Installation." In particular, if you are planning to set up and install Oracle Solaris on a partition or slice and have not done so yet, review the information in "Guidelines for Partitioning a System Prior To Installation" on page 23.

Note – If you have previously installed the Linux operating system, you will need to save a copy of the existing `menu.lst` file to a USB drive for use after the installation. For instructions, see "Modifying Boot Entries and Parameters by Editing the menu.lst File" in *Booting and Shutting Down Oracle Solaris on x86 Platforms*.

---

2 Insert the installation media, boot the system, then make any preliminary keyboard and language selections.

Note – The language and keyboard selections set the defaults for the installer and for the installed system.

---

3 (Optional) In install required drivers, select option #2 on the installation menu.

For instructions on using the Device Driver Utility, see "How to Start the Device Driver Utility" on page 28. After you have installed the drivers, restart the text installation and return to the installation menu.

4 Initiate the installation by selecting the first option on the installation menu.

```
Welcome to the Oracle Solaris xxx installation menu

1 Install Oracle Solaris
2 Install Additional Drivers
```

```
3 Shell
4 Terminal type (currently sun-color)
5 Reboot

Please enter a number [1]:
```

Note – Use the keyboard to navigate through the installer panels. You cannot use a mouse. See the key commands listed on each panel, and see the online help for further information.

5. **Continue past the welcome panel.**

6. **In the Disks panel, if there is more than one target disk listed, select a target disk or accept the default.**

7. **Choose whether to install the operating system on the whole disk or on a partition or a slice on the disk.**

   - The whole disk
   - An x86 partition
   - A SPARC slice

8. **(Optional) In the series of target selection panels, you have the option to modify the partition or slice layout.**

   At any point as you complete the installation panels, you can revert to the original settings.

Caution – If the existing partition table cannot be read, the panel displays proposed partitioning. In this instance, all of the data on the disk is destroyed during the installation.

For detailed partitioning instructions, see "Guidelines for Partitioning a System During an Interactive Installation" on page 24, or see the online help in the installer.

9. **Enter a computer name to identify the system on the network.**

10. **Specify how the wired ethernet network connection should be configured by selecting one of the following options.**

    - **To use DHCP to configure the network connection, select Automatically.**

      The installer continues to the Time Zone panels.

    - **To provide networking specifications, select Manually and continue as follows:**

      a. **If there is more than one interface, select a connection to be configured.**

      b. **Type the connection settings or accept the default information detected and provided by the installer.**

Note – The IP address and netmask are required fields. The router is an optional field.

c. Specify whether the system should use the DNS name service.

d. If you selected Configure DNS, continue with the following steps.

i. Type at least one IP address for the DNS server or servers to be used by the system.

ii. Provide at least one domain name to be searched when a DNS query is made.

e. Specify whether the system should use either the LDAP name services, a NIS name service, or None.

If you selected DNS in the previous step, LDAP or NIS would be set up as alternate name services in addition to DNS. If you did not select DNS in the previous step, LDAP or NIS would be set up as the only name service.

If you will be configuring LDAP on the system without an LDAP profile, select None instead of selecting LDAP. Then, configure LDAP manually after the installation is complete.

Note – If no network naming services are selected, network names can be resolved by using standard name source files such as `/etc/hosts`(4). For further information, see the `nsswitch.conf`(4) man page.

f. Provide the domain where the system resides for the alternate name service you selected.

Note – To determine the domain name, check with your system administrator. Or, use the `domainname` command on a previously-installed system.

g. If you selected LDAP as the only name service or as an additional name service with DNS, provide the LDAP specifications as follows.

i. Specify the LDAP profile to be used to configure the LDAP name service on the system.

ii. Type the IP address for the LDAP profile server.

iii. Provide an LDAP search base or accept the default search base.

iv. Specify whether LDAP proxy bind information will be provided.

Note – If the profile specifies a proxy credential level, and the authentication method is not None, then you must provide the proxy bind information. If you omit that information, LDAP will not be initialized.

v. If necessary, provide the LDAP proxy bind distinguished name and proxy bind password.

h. If you selected NIS as the only name service, or as an additional name service with DNS, provide the NIS specifications.

You can either let the software search for a name server, or you can specify a name server. Select one of the following two choices.

- **Select Find One.**

  Note – The software can only find a name server if that server is on the local subnet.

- **Select Specify One and type the name server's host name or IP address in the subpanel.**

After completing the series of networking configuration panels, the installer displays a series of time zone panels and a Date and Time panel.

- **To specify that the network is not configured during the installation, select None.**

  The install continues to the Time Zone panels.

11 In the series of time zone panels, select a time zone first, then adjust the date and time to match your local time.

Note – The default is for the GMT time zone to be configured.

12 Complete the User panel.

You are not required to create a user account. But, you must create a root password.

- **If you create a user account in this panel, you need to provide both the user's password and a root password.**

  In this case, root will be a role assigned to the user.

  To create a user account, type a username and password. The name must begin with a letter and can contain only letters and numbers.

- **If you do not create a user account, you still need to provide a root password.**

  In this case, root will be a regular user.

13 **Review the installation specifications.**

Review the specifications in the Installation Summary panel. If necessary, go back and make any required changes before starting the installation.

14 **Install the system using the specifications you have provided.**

The Oracle Solaris installation process begins.

Caution – Do not interrupt an installation that is in progress. An incomplete installation can leave a disk in an indeterminate state.

15 **Review the installation logs.**

The Installation Results panel provides access to installation logs that you can review.

16 **(Optional) If you want to cancel anonymous registration of the installed system with Oracle Configuration Manager, perform the following steps to mount the newly-created boot environment and add an "opt-out" file to that boot environment before rebooting the system.**

Caution – By default, the system configuration of the installed system is sent to the Oracle Configuration Manager. This is an anonymous registration with no customer information provided.

The anonymous registration will be automatic upon reboot after the initial installation, but you may cancel the registration per the following directions after the installation and prior to rebooting the installed system.

If you do not opt-out at installation time you may still suspend the service at any later time.

For further information, see "Using Oracle Configuration Manager" on page 32.

a. **Before rebooting the installed system, press F9 to exit the installer.**

b. **In the installation menu, select '3' to open a shell.**

c. **Assume the root role.**

d. **Mount the newly-created boot environment as in the following example:**

```
# beadm mount solaris /a
```

Note – The boot environment name prior to rebooting is, by default, "solaris," even if you changed the computer name in the installer panels.

e. In an editor, create a new file in the mounted boot environment and name the file `/a/etc/svc/profile/site/ocm.xml`.

For example, type the following:

```
# vi /a/etc/svc/profile/site/ocm.xml
```

f. Enter the following contents into the file, save the file, and exit the file.

```
<?xml version='1.0'?>
<!DOCTYPE service_bundle SYSTEM '/usr/share/lib/xml/dtd/service_bundle.dtd.1'>
<service_bundle type='profile' name='ocm'
 xmlns:xi='http://www.w3.org/2003/XInclude'>
   <service name='system/ocm' type='service' version='1'>
      <instance name='default' enabled='false'/>
   </service>
</service_bundle>
```

This file disables the default SMF service and changes the property to "opt-out" from an anonymous registration.

g. Unmount the boot environment, as shown in this sample command.

```
# beadm unmount solaris
```

h. Exit the shell.

```
# exit
```

Note – After installation and reboot, you can choose to register your system by removing this file then enabling the service as follows:

```
# svcadm system/ocm enable
```

This command performs an anonymous registration.

If you wish to associate the system's configuration data with your MOS account, or if your site requires use of a network proxy, you must use the `configCCR` command. See "Using Oracle Configuration Manager" on page 32.

---

17 Reboot or go to a shell and shut down the system.

See Also See *Oracle Solaris Administration: Common Tasks* for information about the following topics:

- Managing user accounts and groups
- Booting and shutting down a system
- Managing services
- Managing hardware faults
- Managing system processes
- Troubleshooting general system problems such as the following:

- What to do if rebooting fails
- What to do if you forgot the root password
- What to do if a system hangs

Adding Software After Text Installation

To add software packages after you have installed the operating system, use the `pkg` commands as described in the `pkg`(1) man page and in Chapter 12, "Managing Software Packages (Tasks)," in *Oracle Solaris Administration: Common Tasks*.

Use the `pkg` commands or the Package Manager tool to find the names of packages you might want to install, get more information about the packages, and install the packages.

Optionally, you can install into a new boot environment, so that you can continue to use your current image if the new installation has problems.

With the `pkg install` command, you should use the `-nv` option first to see what the package installation will look like prior to actually installing the packages. After you have identified the packages you want to install and examined the output from the `pkg install` command with the `-nv` option, issue a command similar to the following to install additional software.

```
# pkg install packagename
```

Replace the *packagename* variable with the name of the package you want to install.

Alternately, you can use the following sample command to create a new backup boot environment and to specify a package to be installed.

```
# pkg install --require-new-be --be-name newBEname packagename
```

If you do not have a GUI desktop and you want to install the Oracle Solaris desktop, install the `solaris-desktop` package.

Performing a Text Installation Over the Network

If you have set up your system to perform automated installations over the network, you also have the option of performing an interactive text installation over the network. Although you can install only a single system at a time with this option, you have the opportunity to customize each installation by using the interactive selections to modify the installation specifications.

▼ How to Perform a Text Installation Over the Network

1 **Download an AI client image and create an install service based on that image.**

For instructions, see Part III, "Installing Using an Install Server."

2 **Boot the client system over the network as follows.**

- **For SPARC clients, type the following at the OBP prompt:**

```
# boot net:dhcp
```

- **For x86 clients, select 1 from the installation menu.**

```
Welcome to the Oracle Solaris xxx installation menu

1 Install Oracle Solaris
2 Install Additional Drivers
3 Shell
4 Terminal type (currently sun-color)
5 Reboot

Please enter a number [1]:
```

3 **Complete the text installation of the client system.**

For instructions, see "How to Perform a Text Installation" on page 46.

Note – The package set installed by the text installer is the `solaris-large-server` package set. However, the text installer over the network is actually an automated installation. Automated installations are designed to download as much of the needed software as possible from IPS repositories. When you use the text installer over the network, a smaller package set, `solaris-auto-install`, is installed by default.

This installed system will be very minimal. After booting into the installed system, you should probably install the `solaris-large-server` package set and, optionally, install a desktop as follows.

```
# pkg install solaris-desktop

# pkg install solaris-large-server
```

◆ ◆ ◆ CHAPTER 5

Automated Installations That Boot From Media

You can initiate an automated installation of the Oracle Solaris 11 OS on a SPARC system or an X86 system by booting an AI Image on media rather than booting over the network. This chapter discusses reasons to boot an AI client from media and how to perform the installation in that mode.

Overview of Installation Using AI Media

Installation using AI media enables you to accomplish the following optional tasks:

- Install the system that will be your AI install server.
- Install a SPARC system that does not have WAN boot capability.
- Troubleshoot an ailing system. Boot the system from the removable media and then inspect the installed system and run diagnostics.

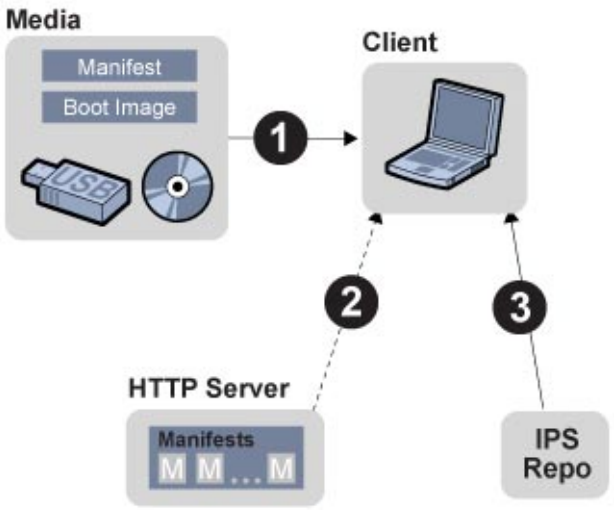
Installation using AI media has the following characteristics:

- You do not need to set up an install server or an install service.
- The system does not need to be able to boot over the network.

Installing Using AI Media

You can boot an AI image from a CD, DVD, or USB device to initiate a hands-free installation of only that system. An AI manifest provides installation instructions. The system to be installed must have network access. To complete the installation, software packages are retrieved from an IPS repository on the Internet or on the local network. Review the default AI manifest as described in "Creating a Custom AI Manifest" on page 57.

FIGURE 5–1 AI Install Using Media



System Requirements for Installing Using AI Media

Both SPARC and x86 systems must meet the following requirements.

TABLE 5–1 System Requirements for Installation Using AI Media

Requirement	Specifications
Memory	To check the minimum memory requirement for the current release, see *Oracle Solaris 11 Release Notes*.
Disk Space	To check the disk space requirements for the current release, see *Oracle Solaris 11 Release Notes*.
Network Access	The system to be installed must be able to access the following resources during the installation: ■ A DHCP server that provides network configuration information ■ An IPS repository that contains the packages to be installed on the client system If you create a custom AI manifest, the system must be able to access that manifest on an HTTP server.

▼ How To Install Using AI Media

1. **Download the AI boot image.**

   To download the AI boot image, go to the following Internet location:
   `http://www.oracle.com/technetwork/server-storage/solaris11/downloads/index.html`

   - **SPARC systems – Download the SPARC AI `.iso` file.**
   - **x86 systems – Download the x86 AI `.iso` file or the x86 AI `.usb` file**

2. **Review the default AI manifest.**

   You can use the default manifest that is provided in the AI image, or you can create a custom manifest and provide the location of this custom manifest when the client boots. See "Creating a Custom AI Manifest" on page 57.

3. **Create bootable media.**

   - **SPARC and x86 ISO images – Burn the `.iso` file to a CD or DVD.**
   - **x86 USB images – Use the `usbcopy` utility, in order to copy the image to a USB flash drive.**

     Note – You can add this utility to your system by installing the `pkg:/install/distribution-constructor` package.

4. **Boot from the media.**

   Boot the system from the device that contains the boot image. See "Booting a SPARC System From AI Media" on page 58 and "Booting an x86 System From AI Media" on page 59 for instructions about how to specify the default AI manifest or a custom AI manifest.

   A "hands-free" installation is performed. After the installation, the SCI Tool starts and asks you to provide configuration information for the system.

5. **Provide configuration information in the SCI Tool panels.**

   See "Creating a Configuration Profile Using the SCI Tool" on page 67.

Creating a Custom AI Manifest

You can install the system using the installation specifications in the AI manifest provided in the AI boot image, or you can create custom installation specifications. If you create a custom AI manifest, store the manifest on an HTTP server, and provide the location of the manifest when you boot the system to be installed.

If you download the `.iso` AI image, you can use the following commands to inspect the AI manifest in that image. In this example, `/tmp` is the directory where you downloaded the AI image, and `/home/username` is the directory where you want to copy and edit the AI manifest. The AI manifest is in `auto-install/default.xml` in the image.

```
# lofi_dev=$(/usr/sbin/lofiadm -a /tmp/sol-11-ai-sparc.iso)
# /usr/sbin/mount -o ro -F hsfs ${lofi_dev} /mnt
# cp /mnt/auto_install/manifest/default.xml /home/username/custom.xml
# /usr/sbin/umount /mnt
# /usr/sbin/lofiadm -d ${lofi_dev}
```

Review your copy of the default manifest file (`/home/username/custom.xml` in this example), and decide whether these specifications are satisfactory for this installation.

Alternatively, you can use the manifest shown in "Default AI Manifest" on page 113 as the base to create a custom manifest.

To change installation specifications such as target disk or additional packages to install, see the `ai_manifest`(4) man page.

Note – You also have the option to preset Oracle Configuration Manager settings. See Chapter 14, "Setting Up Oracle Configuration Manager For Use By AI Client Systems."

When you are finished modifying the AI manifest, copy the custom manifest to an HTTP server. Note the URL to the custom AI manifest so that you can provide that URL when you boot the system to be installed. For example, the URL might be `http://example.com/custom.xml`.

Booting a SPARC System From AI Media

You can specify the default AI manifest or a custom AI manifest when you boot the system from the AI media.

Use the Default AI Manifest

To use the default AI manifest that is in the AI boot image, type the following command at the OBP prompt:

```
ok> boot cdrom - install
```

The automated installation proceeds, using the specifications in the default manifest.

Use a Custom AI Manifest

To use a custom AI manifest, type the following command at the OBP prompt:

```
ok> boot cdrom - install aimanifest=prompt
```

The following prompt displays:

```
Enter the URL for the AI manifest [HTTP, default]:
```

Type the URL to your custom manifest. For example, type `http://example.com/custom.xml`.

The automated installation proceeds, using the specifications in the custom manifest.

Boot a SPARC Image Without Installing

You might want to boot from media but not install. For example, you might want to troubleshoot or examine the system.

To boot the AI image but not start an automated installation, use the following command:

```
ok> boot cdrom
```

The system boots and a login panel displays, but the installation does not begin.

Booting an x86 System From AI Media

On an x86 system, choose an automated installation option from the GRUB menu. The GRUB menu selection or boot command that you use specifies whether the installation will use the default manifest on the media or a custom manifest that you have stored on an HTTP server.

Your GRUB menu selections should look similar to the following example:

```
GNU GRUB version 0.97 (639K lower / 2078660K upper memory)

Oracle Solaris 11 Automated Install custom
Oracle Solaris 11 Automated Install
Oracle Solaris 11 Automated Install custom ttya
Oracle Solaris 11 Automated Install custom ttyb
Oracle Solaris 11 Automated Install ttya
Oracle Solaris 11 Automated Install ttyb
Boot from Hard Disk

Use the arrow keys to select which entry is highlighted.
Press enter to boot the selected OS, 'e' to edit the
commands before booting, or 'c' for a command-line.
```

Use the Default AI Manifest

To use the default AI manifest that is in the AI boot image, use the arrow keys to choose one of the following options:

```
Oracle Solaris 11 Automated Install
Oracle Solaris 11 Automated Install ttya
Oracle Solaris 11 Automated Install ttyb
```

The `ttya` option sends the screen output during the installation to serial console `ttya (COM1)`. The `ttyb` option sends the screen output during the installation to serial console `ttyb (COM2)`.

The automated installation proceeds, using the specifications in the default manifest.

Use a Custom AI Manifest

To use a custom AI manifest, choose one of the following options:

```
Oracle Solaris 11 Automated Install custom
Oracle Solaris 11 Automated Install custom ttya
Oracle Solaris 11 Automated Install custom ttyb
```

When you select one of these custom options, the following prompt displays:

```
Enter the URL for the AI manifest [HTTP, default]:
```

Type the URL to your custom manifest. For example, type `http://example.com/custom.xml`.

The automated installation proceeds, using the specifications in the custom manifest.

Boot an x86 Image Without Installing

You might want to boot from media but not install. For example, you might want to troubleshoot or examine the system.

In general, if `install=true` is specified in the kernel line for the GRUB entry that you use, the installation automatically begins. If you want to boot the x86 system without immediately starting an automated installation, examine the GRUB menu entry that you plan to choose. If `install=true` is specified in the kernel line for that GRUB entry, edit the line to remove `install=true`. Then, when you choose that option, the system boots and a login screen displays but the installation does not begin.

Viewing the Installation Log Files

When the automated installation is complete, the output states whether the installation succeeded or failed.

- If the installation failed, you can review the installation log at `/system/volatile/install_log`.
- If the installation succeeded, you can find the log at `/system/volatile/install_log` before you reboot the system or at `/var/sadm/system/logs/install_log` after you reboot.

◆ ◆ ◆ CHAPTER 6

Unconfiguring or Reconfiguring an Oracle Solaris instance

An **Oracle Solaris instance** is created and configured during installation. An Oracle Solaris instance is defined as a boot environment in either a global or a non-global zone. This chapter describes how to unconfigure and reconfigure an Oracle Solaris instance.

Functional Groupings

When you unconfigure or reconfigure an Oracle Solaris instance, several predefined subsystems are affected. These subsystems are referred to as functional groupings.

The overall grouping for an instance is called “system”.

The following table lists the configurable functional groupings that exist in an Oracle Solaris instance.

TABLE 6–1 Functional Groupings

Grouping	Components	Unconfigured State
`system`	full system	Compilation of below
`identity`	system nodename	Unknown
`kdb_layout`	keyboard	U.S. English
`network`	network	No network
`location`	timezone	`UTC`
	locale	C locale
`users`	root	Empty root password
	initial user account	Remove user account

TABLE 6–1 Functional Groupings *(Continued)*

Grouping	Components	Unconfigured State
`naming_services`	DNS, NIS and LDAP clients, nsswitch	No network naming services

Using the sysconfig Utility

You can use the `sysconfig` utility to perform the following configuration tasks on an Oracle Solaris instance.

- To unconfigure an Oracle Solaris instance in a global or non-global zone and leave it in an unconfigured state, use the `sysconfig unconfigure` command.

  See "Unconfiguring an Oracle Solaris Instance" on page 62.
- To reconfigure an Oracle Solaris instance in a global or non-global zone, use the `sysconfig configure` command
  - If you specify an existing configuration profile with the command, a non-interactive configuration is performed.
  - If you do not specify an existing configuration profile with the command, an interactive SCI Tool runs. The SCI Tool enables you to provide specific configuration information for that Solaris instance.

  See "Configuring a System" on page 63.
- You can use the `sysconfig create-profile` command to create a new system configuration profile. See "Creating a Configuration Profile Using the SCI Tool" on page 67.

The `sysconfig` command affects all functional groupings in the Solaris instance. For detailed instructions, see the `sysconfig`(1M) man page.

Note – You must become the root role to use the `sysconfig` utility.

Unconfiguring an Oracle Solaris Instance

If you want to unconfigure a previously configured Solaris instance and leave it in an unconfigured state, use the unconfigure command. All the functional groupings will be unconfigured.

Use the `sysconfig unconfigure` command as in the following example.

```
# sysconfig unconfigure -g system
```

This example unconfigures the instance.

Note – If the `-g` option is not specified, confirmation will be requested before the system is unconfigured.

Alternately, you can unconfigure the system and shutdown the system as follows.

```
# sysconfig unconfigure -s
```

For further information, see the `sysconfig`(1M) man page.

Configuring a System

You can use the `sysconfig configure` command to configure or reconfigure an Oracle Solaris instance in a global or non-global zone. This configuration can occur either interactively or non-interactively.

- You can use the `-c` option in the `sysconfig configure` utility to specify an existing system configuration profile. If the utility is run with that option, then the utility reads the configuration specifications in the existing profile and uses those specifications to configure the system non-interactively.

  For example, the following command specifies that the system be configured using the existing configuration profile named myprofile.xml.

  ```
  # sysconfig configure -c myprofile.xml
  ```

  Note – The `-c` option specifies a profile or a directory of profiles. All profiles must include the .xml file extension.

  For information about system configuration (SC) profiles, see Chapter 11, "Configuring the Client System."
- If the `sysconfig configure` command is invoked without a specified profile, the SCI Tool is automatically activated.

  The SCI Tool supports configuration of freshly installed or unconfigured systems. You can use this tool to provide system configuration for newly created non-global zones or other unconfigured systems. The SCI Tool consists of a series of interactive text panels that ask for configuration information. See "How to Reconfigure Using the SCI Tool" on page 64.

  Note – The series screens in the SCI Tool also run automatically as part of a text installation.

Alternately, you can run the SCI Tool to create a new system configuration profile based on the configuration specifications entered in the SCI Tool screens. See "Creating a Configuration Profile Using the SCI Tool" on page 67.

▼ How to Reconfigure Using the SCI Tool

1 **Become the root role.**

Note – If you are working in a non-global zone, log into the zone as the root role as follows:

```
# zlogin -C -e ^ ZONENAME
```

---

2 **Run the `sysconfig configure` command without specifying a profile.**

```
# sysconfig configure
```

The SCI Tool is displayed. The following steps provide instructions for completing the series of interactive panels in the SCI Tool.

Note – Use the function keys to navigate through the SCI Tool panels. You cannot use a mouse. Refer to the function key references on each panel and to the online help as needed.

---

3 **Continue past the initial Welcome panel.**

4 **Enter a name to identify the system on the network.**

5 **Specify how the wired ethernet network connection should be configured by selecting one of the following options.**

- **To use DHCP to configure the network connection, select Automatically.**

  The SCI Tool continues to the Time Zone panels.

- **To provide networking specifications, select Manually and continue as follows:**

  a. **If there is more than one interface, select a connection to be configured.**

  b. **Type the connection settings or accept the default information detected and provided by the SCI Tool.**

  Note – The IP address and netmask are required fields. The router is an optional field.

  ---

  c. **Specify whether the system should use the DNS name service.**

d. If you selected Configure DNS, continue with the following steps.

i. Type at least one IP address for the DNS server or servers to be used by the system.

ii. Provide at least one domain name to be searched when a DNS query is made.

e. Specify whether the system should use either the LDAP name services, a NIS name service, or None.

If you selected DNS in the previous step, LDAP or NIS would be set up as alternate name services in addition to DNS. If you did not select DNS in the previous step, LDAP or NIS would be set up as the only name service.

If you will be configuring LDAP on the system without an LDAP profile, select None instead of selecting LDAP. Then, configure LDAP manually after the SCI Tool process is complete.

Note – If no network naming services are selected, network names can be resolved by using standard name source files such as `/etc/hosts`(4). For further information, see the `nsswitch.conf`(4) man page.

---

f. Provide the domain where the system resides for the alternate name service you selected.

Note – To determine the domain name, check with your system administrator. Or, use the `domainname` command on a previously-installed system.

---

g. If you selected LDAP as the only name service or as an additional name service with DNS, provide the LDAP specifications as follows.

i. Specify the LDAP profile to be used to configure the LDAP name service on the system.

ii. Type the IP address for the LDAP profile server.

iii. Provide an LDAP search base or accept the default search base.

iv. Specify whether LDAP proxy bind information will be provided.

Note – If the profile specifies a proxy credential level, and the authentication method is not None, then you must provide the proxy bind information. If you omit that information, LDAP will not be initialized.

---

v. **If necessary, provide the LDAP proxy bind distinguished name and proxy bind password.**

h. **If you selected NIS as the only name service, or as an additional name service with DNS, provide the NIS specifications.**

You can either let the software search for a name server, or you can specify a name server. Select one of the following two choices.

- **Select Find One.**

Note – The software can only find a name server if that server is on the local subnet.

- **Select Specify One and type the name server's host name or IP address in the subpanel.**

After completing the series of networking configuration panels, the SCI Tool displays a series of time zone panels and a Date and Time panel.

- **To specify that the network is not configured during the installation, select None.**

The SCI Tool continues to the Time Zone panels.

6 **In the series of time zone panels, select a time zone first, then adjust the date and time to match your local time.**

Note – The default is for the GMT time zone to be configured.

7 **Complete the User panel.**

You are not required to create a user account. But, you must create a root password.

- **If you create a user account in this panel, you need to provide both the user's password and a root password.**

In this case, root will be a role assigned to the user.

To create a user account, type a username and password. The name must begin with a letter and can *only* contain letters and numbers.

- **If you do not create a user account, you still need to provide a root password.**

In this case, root will be a regular user.

8 **Review the configuration settings.**

- **If the settings are correct, apply the configuration to the system.**

- **If the settings are not correct, press the Back key as often as necessary to return to the panel with the incorrect information, make changes, and continue through the panels again.**

Creating a Configuration Profile Using the SCI Tool

You can run the SCI Tool to generate a new system configuration profile based on the configuration specifications entered in the SCI Tool panels. The default location for the new profile is `/system/volatile/profile/sc_profile.xml`.

To create a new configuration profile, use the `sysconfig create-profile` command. A profile will be created, but the configuration will not be applied to the system.

The SCI Tool creates the new configuration profile based on the specifications that you provide in the SCI Tool panels. The new profile is stored in the default location. You can use that new profile to configure a system as shown in the following example.

```
# sysconfig configure -g system -c /system/volatile/profile/sc_profile.xml
```

The `-g` option is used to specify a specific functional grouping that should be configured. In this example, the full system will be configured. For a list of the functional groupings, see Table 6–1.

The following example uses the `sysconfig create-profile -o` option to specify a different output file location when creating the profile. Then, the `sysconfig configure -c` option points to that profile location to reconfigure a system.

```
# sysconfig create-profile -o /tmp/myprofile.xml
# sysconfig configure -g system -c /tmp/myprofile.xml
```

Note – You must include the `.xml` extension for the configuration profile, in order successfully use that profile for reconfiguration.

For further information, see the `sysconfig`(1M) man page. Also, see Chapter 11, "Configuring the Client System."

PART III

Installing Using an Install Server

This section describes automated installation of client systems over a network.

◆ ◆ ◆ CHAPTER 7

Automated Installation of Multiple Clients

Use the Automated Installer (AI) to install the Oracle Solaris 11 operating system (OS) on multiple client systems in a network. AI performs a hands-free installation of both SPARC and x86 systems. All installations require access to a software package repository on the network.

What Is an Automated Installation?

AI automates the installation of the Oracle Solaris 11 OS on SPARC and x86 clients over the network. The clients can be customized with installation parameters such as disk layout and software selection and with system configuration parameters such as host name, network configuration, and user accounts. Customizations can be made on a client-by-client basis and can be scaled for large environments.

An automated installation of a client over the network consists of the following high-level steps:

1. The client system boots over the network and gets its network configuration and the location of the install server from the DHCP server.
2. The install server provides a boot image to the client.
3. Characteristics of the client determine which installation instructions and which system configuration instructions are used to install the client.
4. The Oracle Solaris 11 OS is installed on the client, pulling packages from the package repository specified by the installation instructions in the AI install service.

How Do I Use the Automated Installer?

To use AI to install client systems over the network, you must set up DHCP and set up an AI install service on an install server. See Chapter 8, "Setting Up an Install Server." AI uses DHCP to provide the IP address, subnet mask, router, DNS server, and the location of the install server to the client machine to be installed. The DHCP server and AI install server can be the same machine or two different machines.

The client machines you want to install must be able to access an Oracle Solaris Image Packaging System (IPS) software package repository. The IPS package repository can be on the install server, on another server on the local network, or on the Internet.

An AI install service includes a SPARC or x86 network boot image (net image), one or more installation instruction files (AI manifests), and zero or more system configuration instruction SMF profile files. The net image is not a complete installation. Client machines must access an IPS package repository to complete their installations. Each client uses only one AI manifest. Different clients can use different AI manifests. The AI manifest specifies one or more IPS package repositories where the client retrieves the packages needed to complete the installation. The AI manifest also includes the names of additional packages to install and information such as target device and partition information. See Chapter 10, "Provisioning the Client System," for information about customizing AI manifests, either prior to booting the client or dynamically at client installation time. You can also specify instructions for configuring the client. See Chapter 11, "Configuring the Client System," for information about system configuration profiles. See Chapter 13, "Running a Custom Script During First Boot," for information about how to perform further installation and configuration at first boot of the client.

If two client machines have different architectures or need to be installed with different versions of the Oracle Solaris 11 OS, then create two AI install services, and associate each install service with the appropriate image source for the architecture and OS version you want to install. When the first install service of a particular architecture is created on an install server, a copy of that service, `default-i386` or `default-sparc`, is automatically created. This default service is used for all installations to clients of that architecture that are not explicitly associated with a different install service with the `create-client` subcommand.

If two client machines need to be installed with the same version of the Oracle Solaris 11 OS but need to be installed differently in other ways, then create two AI manifests for the AI install service. The different AI manifests can specify different packages to install or a different slice as the install target, for example.

If client systems need to have different configurations applied, then create multiple system configuration profiles for the install service. The different system configuration profiles can specify different network or locale setup or unique host name and IP address, for example.

The installation begins when you boot the client. When the client boots, DHCP directs the client to the AI install server, and the client accesses the correct install service and the correct AI manifest and system configuration profiles within that service. Chapter 15, "Installing Client

Systems," explains how a client is associated with a particular install service. Chapter 9, "Customizing Installations," explains how a client identifies the correct AI manifest and system configuration profiles to use.

If adequate system configuration instructions have not been provided, an interactive tool prompts for system configuration information at first boot after installation. See Chapter 11, "Configuring the Client System," for information and examples of system configuration profiles. See "Configuring a System" on page 63 for information about the interactive configuration tool.

If you have specified installation of non-global zones, those zones are configured and installed at first boot after installation. See Chapter 12, "Installing and Configuring Zones," for information about how to specify configuration and installation of non-global zones as part of AI client installation.

Also at first boot, Oracle Configuration Manager attempts to collect system configuration information and send the information to Oracle Support. See Chapter 14, "Setting Up Oracle Configuration Manager For Use By AI Client Systems."

Automated Installer Use Cases

The following use cases describe the primary distinct ways to use AI. These use cases do not build on each other. Instead, each case describes a separate feature of AI, and all behavior that is not part of that feature is the same as in the minimum case. You probably will use a combination of the features described in these use cases.

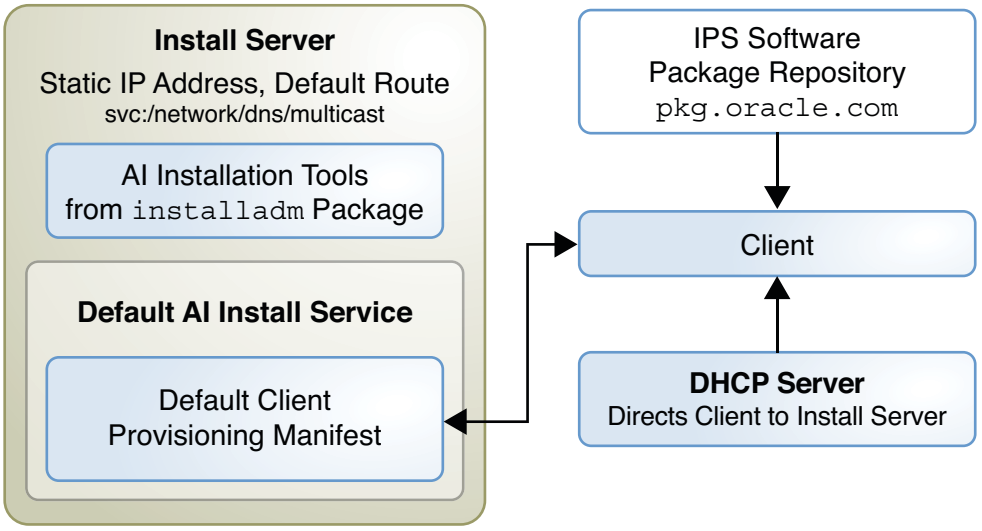
Minimum Requirements for AI Use

The minimum you have to do to use AI is create one install service. In this minimal scenario, all clients are the same architecture and are to be installed with the same version of the Oracle Solaris OS. The installations use the default AI manifest, which specifies the most recent version of the OS available from the default IPS package repository on the Internet.

1. Make sure the install server has a static IP address and default route.
2. Install the installation tools package, `install/installadm`.
3. Run the `installadm create-service` command.

   When the first install service for a particular architecture is created on an install server, a copy of that service, `default-i386` or `default-sparc`, is automatically created. This default service is used for all installations on clients of that architecture that are not explicitly associated with a different install service with the `create-client` subcommand.
4. Make sure the clients can access a DHCP server.
5. Make sure the necessary information is available in the DHCP configuration to boot the service.

6. Make sure the clients can access an IPS software package repository. To use the default IPS package repository, the clients must be able to access the Internet.
7. Network boot the client.



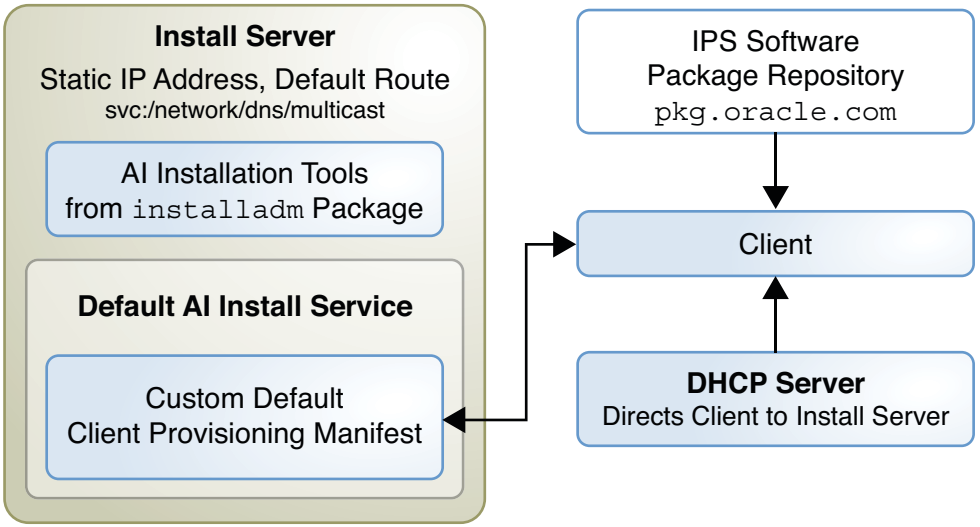
When you network boot the client, the following steps are performed:

1. The client gets the install server address from the DHCP server.
2. The client uses the `default-`*arch* install service if the architecture matches.
3. The client uses the default AI manifest of the `default-`*arch* install service, installing software packages from the IPS package repository over the network.
4. When the client boots after installation, an interactive tool prompts for system configuration information because no system configuration profile is provided.

Customize Installation Instructions

To specify installation parameters such as the target disk for installation, partition or mirror configuration, or additional software packages to install, provide a customized AI manifest. Perform the following steps before you boot the client, in addition to the minimum required steps:

1. Create a new AI manifest, or write a script that dynamically creates a custom AI manifest at client installation time. See Chapter 10, “Provisioning the Client System.”
2. Run the `installadm create-manifest` command to add the new manifest or script to the `default-`*arch* install service. Specify criteria for the client to select this manifest or script.



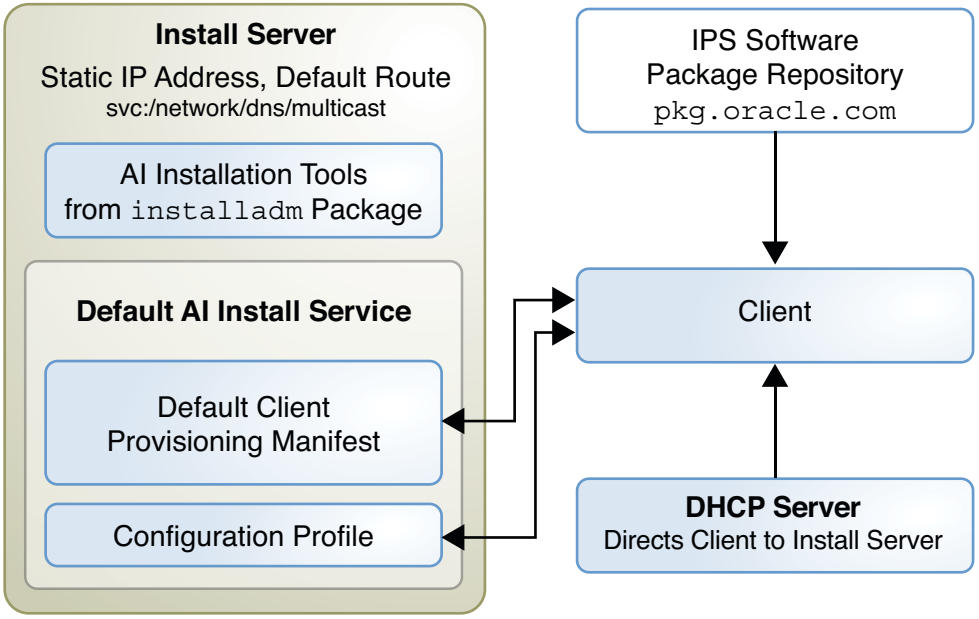
When you network boot the client, the following steps are performed:

1. The client gets the install server address from the DHCP server.
2. The client uses the `default-`*arch* install service if the architecture matches.
3. The client is directed to the correct AI manifest by criteria specified to `create-manifest`. If no criteria match, the client uses the default manifest for this service.
4. The client is provisioned according to the selected AI manifest.
5. When the client boots after installation, an interactive tool prompts for system configuration information because no system configuration profile is provided.

Provide System Configuration Instructions

To specify system configuration parameters such as time zone, user accounts, and networking, provide a Service Management Facility (SMF) system configuration profile file. Perform the following steps before you boot the client, in addition to the minimum required steps:

1. Create a system configuration profile as described in Chapter 11, "Configuring the Client System."
2. Run the `installadm create-profile` command to validate the profile, add the profile to the `default-`*arch* install service, and specify criteria to select which clients should use this system configuration profile. If no criteria are specified, the profile is used by all clients of the service.



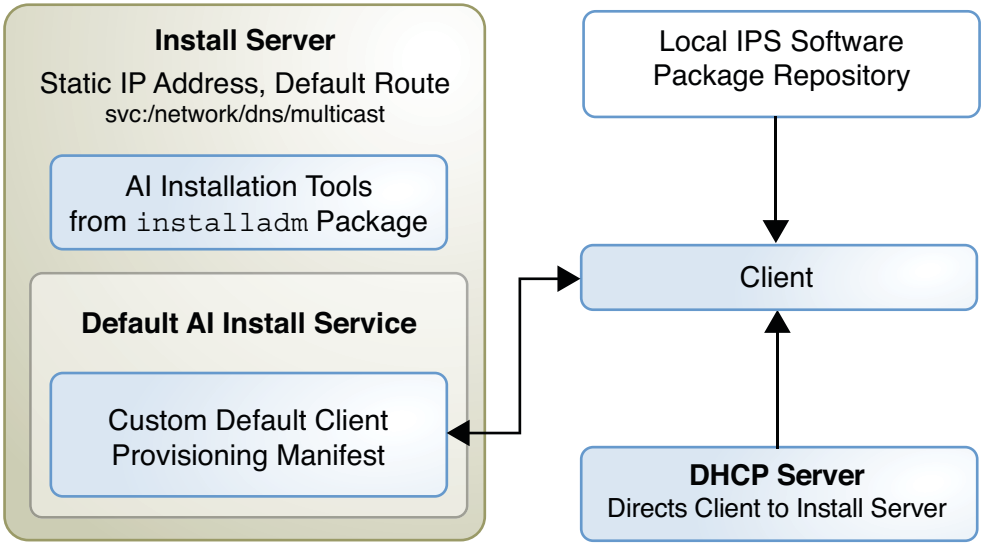
When you network boot the client, the following steps are performed:

1. The client gets the install server address from the DHCP server.
2. The client uses the `default-`*arch* install service if the architecture matches.
3. The client uses the default AI manifest of the `default-`*arch* install service, installing software packages from the IPS package repository over the network.
4. The client is directed to the correct system configuration profile by criteria specified to `create-profile` for the `default-`*arch* install service.
5. The client is configured according to the selected configuration profile. If no configuration profile is selected because criteria do not match, the interactive configuration tool starts.

Provide a Local IPS Package Repository

You might want to use a local package repository rather than an Internet package repository to improve data transfer performance, because clients do not have Internet access, or for other reasons. Perform the following steps before you boot the client, in addition to the minimum required steps:

1. Make a local copy of an IPS package repository and make the repository accessible to client systems. See *Copying and Creating Oracle Solaris 11 Package Repositories* for instructions.
2. Customize the default AI manifest to specify the new repository as a software source. Export and edit the default manifest, and run the `installadm update-manifest` command to replace the default AI manifest in the `default-`*arch* install service with the edited manifest. See Chapter 10, "Provisioning the Client System," for instructions.



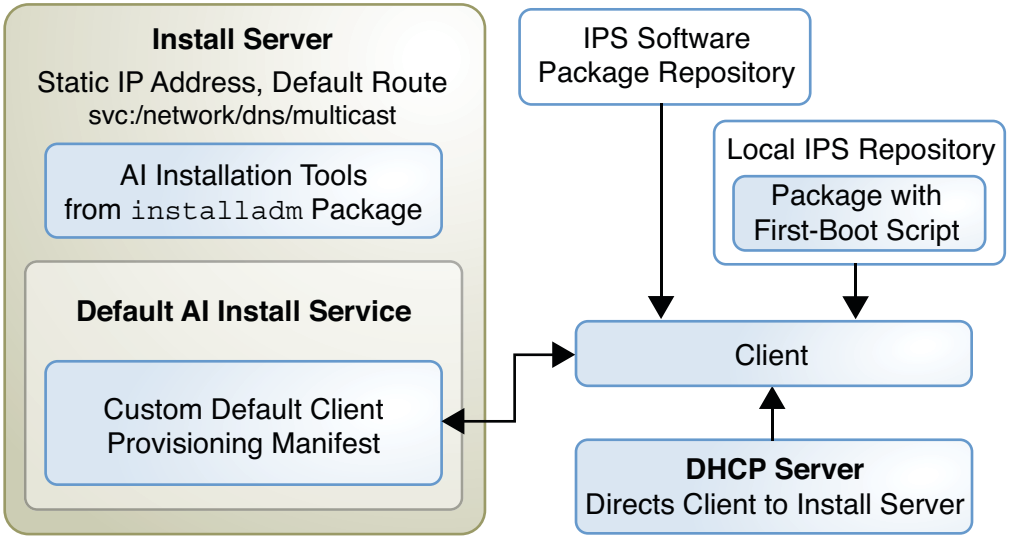
When you network boot the client, the following steps are performed:

1. The client gets the install server address from the DHCP server.
2. The client uses the `default-`*arch* install service if the architecture matches.
3. The client is provisioned according to the customized AI manifest, using the local IPS package repository.
4. When the client boots after installation, an interactive tool prompts for system configuration information because no system configuration profile is provided.

Provide a Custom First Boot Script

To include configuration that cannot be expressed in an AI manifest or system configuration profile, you can include a script that runs at first boot. Perform the following steps before you boot the client, in addition to the minimum required steps. See Chapter 13, "Running a Custom Script During First Boot," for detailed information about these steps.

1. Create a script to run at first boot of the client.
2. Create a run-once SMF service to run the script.
3. Create an IPS package for the service and script, and add the package to a local IPS repository.
4. Make the repository accessible to client systems.
5. Customize the default AI manifest to specify the new repository as a software source and specify the new package to be installed. Export and edit the default manifest, and run the `installadm update-manifest` command to replace the default AI manifest in the `default-`*arch* install service with the edited manifest. See Chapter 10, "Provisioning the Client System," for instructions.



When you network boot the client, the following steps are performed:

1. The client gets the install server address from the DHCP server.
2. The client uses the `default-`*arch* install service if the architecture matches.
3. The client is provisioned according to the customized AI manifest, including installing the custom package with the first-boot script.

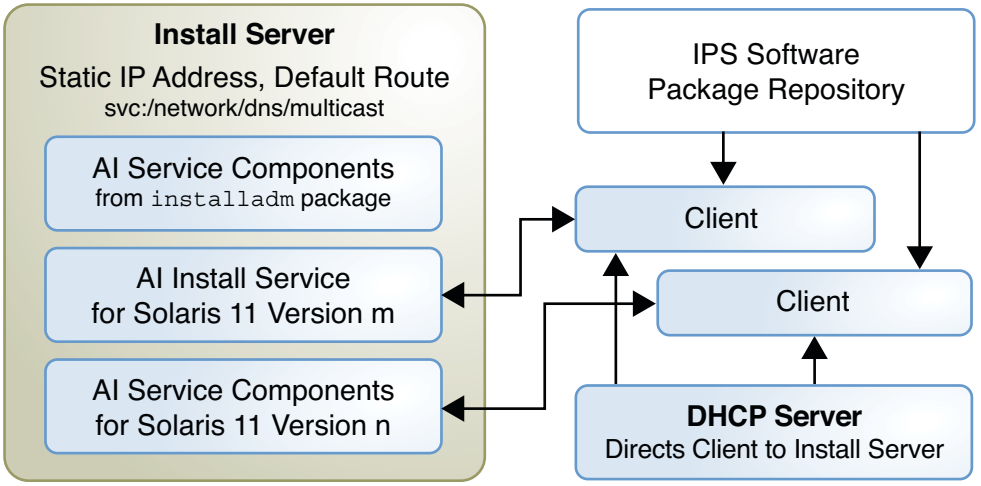
4. When the client boots after installation, an interactive tool prompts for system configuration information because no system configuration profile is provided.
5. When the client boots after installation, the custom run-once first-boot service runs and executes the custom script.

Provide Additional AI Install Services

To install on a different client architecture, or to install a different version of the Oracle Solaris 11 OS, create an additional AI install service as described in Chapter 8, "Setting Up an Install Server." Perform the following steps before you boot the client, in addition to the minimum required steps:

1. Run the `installadm create-service` command and specify a source that corresponds to the architecture and OS version that you want to install.
2. If this is the first install service for a different architecture, a copy of that service, `default-`*arch*, is automatically created. This default service is used for all installations on clients of that architecture that are not explicitly associated with a different install service with the `create-client` subcommand.

   If this new install service is for the same architecture as the existing install service, run the `installadm create-client` command to direct the client to this new install service instead of to the default service for this architecture.



When you network boot the client, the following steps are performed:

1. The client gets the install server address from the DHCP server.
2. The client is directed to the new install service by `create-client`, or the client is directed to the default install service if `create-client` was not run for this client.
3. The client is provisioned according to the default AI manifest for the selected install service.
4. When the client boots after installation, an interactive tool prompts for system configuration information because no system configuration profile is provided.

◆ ◆ ◆ CHAPTER 8

Setting Up an Install Server

To install clients over the network, AI requires a separate system to be an install server. On the install server, create an AI install service to provide a net image and instructions for installing the Oracle Solaris 11 OS on different clients.

AI Server Setup Task Map

The following task map summarizes the steps to set up an AI install server.

TABLE 8–1 Task Map

Task	Reference
Check whether the server meets the minimum hardware requirements to be an AI install server.	See "AI Server Hardware Requirements" on page 82.
Configure the AI install server to use a static IP address and default route. Optionally, enable the `svc:/network/dns/multicast` SMF service. Make sure the AI install server can access an IPS software package repository.	See "AI Server Software Requirements" on page 82.
Install the AI tool set.	See "Install the AI Installation Tools" on page 83.
Set up an install service.	See "Create an AI Install Service" on page 86. You need a separate install service for each architecture you want to install and for each different version of the operating system you want to install.

Install Server Requirements

Any system that meets these requirements can be used as an AI install server, including laptops, desktops, virtual machines, and enterprise servers. The install server can be either an x86 machine or a SPARC machine. An x86 install server can install both SPARC and x86 clients, and a SPARC install server can install both SPARC and x86 clients.

AI Server Hardware Requirements

The following requirements assume the Oracle Solaris 11 OS is already installed. If you need to install or update the Oracle Solaris 11 OS on your AI install server, check Chapter 4, "Using the Text Installer," and Chapter 3, "Using the LiveCD," for memory and disk space requirements.

Memory The minimum requirement to operate as an AI install server is 1 GB of memory.

Disk space Additional disk space required to operate as an AI install server depends on how many install services you set up. You need a separate install service for each different client architecture that you plan to install and for each different version of the Oracle Solaris 11 OS that you plan to install on client systems. Each net image is approximately 300-400 MB.

AI Server Software Requirements

Operating system Install the Oracle Solaris 11 OS on the AI server. To install the Oracle Solaris 11 on the AI server, see Part II, "Installing Using Installation Media."

Static IP address Configure the AI server to use a static IP address. See "How to Configure an IP Interface" in *Oracle Solaris Administration: Network Interfaces and Network Virtualization*.

Default router Ensure that your AI server has a default route set by using the `netstat`(1M) command to show network status. If your AI server does not have a default route set, you can set a static default route by populating the `/etc/defaultrouter`(4) file with the IP address of a static default route for your server's network.

Multicast DNS Optional: Enable the `svc:/network/dns/multicast` SMF service. Use the `svcs`(1) command to check the status of the service, and then use the `svcadm`(1M) command to enable the service if necessary. See Example 8–1.

Software package repository	Ensure that the install server can access an IPS software package repository. AI requires the `install/installadm` package.
DHCP	Set up DHCP. The AI server can also be the DHCP server. Alternatively, you can use a DHCP server that is already set up in this network. You need different DHCP configurations for each client architecture. "Create an Install Service Including Local DHCP Setup" on page 90 shows an example of setting up DHCP on the install server. For more information about DHCP configuration, see Part II, "DHCP," in *Oracle Solaris Administration: IP Services*.

EXAMPLE 8–1 Enable Multicast DNS

The following commands check the status of the `svc:/network/dns/multicast` SMF service and then enable the service.

```
# svcs /network/dns/multicast
STATE          STIME    FMRI
disabled       10:19:28 svc:/network/dns/multicast:default
# svcadm enable /network/dns/multicast
# svcs /network/dns/multicast
STATE          STIME    FMRI
online         13:28:30 svc:/network/dns/multicast:default
```

Install the AI Installation Tools

The AI installation tools package provides the `installadm`(1M) command that enables you to create and maintain AI install services.

The `installadm` command enables you to accomplish the following tasks:

- Create and enable install services.
- Set up and update a DHCP server.
- Add custom client installation and configuration instructions.
- Set criteria for clients to use custom installation and configuration instructions.

See "Maintain an Install Server" on page 92 and *Oracle Solaris 11 Installation Man Pages* for more information about the `installadm` command.

To install the tools package, your AI install server must be able to access an Oracle Solaris Image Packaging System (IPS) software package repository. Make sure you are connected to the Internet or to a local IPS package server that contains the `install/installadm` package.

Use the pkg `list` command to determine whether the `installadm` package is already installed on this system.

```
$ pkg list installadm
pkg list: no packages matching 'installadm' installed
```

Use the `-a` option to make sure your IPS package repository contains the `installadm` package.

```
$ pkg list -a installadm
NAME (PUBLISHER)                                      VERSION                    IFO
install/installadm                                    0.5.11-0.175.0.0.0.0.1345 ---
```

If more than one publisher is defined for this image, use the `-v` option to show which publisher provides the `installadm` package.

```
$ pkg list -av installadm
FMRI                                                                             IFO
pkg://solaris/install/installadm@0.5.11,5.11-0.175.0.0.0.0.1345:20110815T024057Z ---
```

Use the pkg `publisher` command to show the origin for the publisher. In this example, a local copy of the `solaris` repository has been made.

```
# pkg publisher
PUBLISHER                 TYPE    STATUS  URI
solaris                   origin  online  file:///export/Solaris11/
example.com  (non-sticky) origin  online  http://pkg.example.com/
```

Use the pkg `install` command to install the `installadm` package.

```
# pkg install install/installadm
    Packages to install:  1
Create boot environment: No
     Services to change:  2

DOWNLOAD                        PKGS       FILES    XFER (MB)
Completed                        1/1       66/66      0.3/0.3

PHASE                             ACTIONS
Install Phase                     119/119

PHASE                               ITEMS
Package State Update Phase            1/1
Image State Update Phase              2/2

PHASE                               ITEMS
Reading Existing Index                8/8
Indexing Packages                     1/1
Deleting content cache
```

```
$ pkg info installadm
          Name: install/installadm
       Summary: installadm utility
   Description: Automatic Installation Server Setup Tools
```

```
      Category: System/Administration and Configuration
         State: Installed
     Publisher: solaris
       Version: 0.5.11
 Build Release: 5.11
        Branch: 0.175.0.0.0.0.1345
Packaging Date: Mon Aug 15 02:40:57 2011
          Size: 967.86 kB
          FMRI: pkg://solaris/install/installadm@0.5.11,5.11-0.175.0.0.0.0.1345:20110815T024057Z
```

Configure the Install Server

This section describes some of the configuration you might want to perform on the install server to prepare for AI client installations.

Configure a Multihomed Install Server

By default, the AI install server is configured to serve install clients on all networks that the server is connected to if the server is multihomed. To modify this configuration, adjust the `all_services/networks` and `all_services/exclude_networks` properties of the `svc:/system/install/server:default` SMF service.

The value of the `all_services/networks` property is a list of networks in CIDR format (for example, 192.168.56.0/24). The value of the `all_services/exclude_networks` property is a boolean true/false that specifies how the `all_services/networks` property is processed. If `exclude_networks` is false, the AI install server only serves the networks listed in the `networks` property. If `exclude_networks` is true, the AI install server does not serve the networks listed in the `networks` property.

The following commands reconfigure an AI install server that is connected to three networks to serve installations on only one network. In this example, the multihomed AI install server is connected to the following three networks: 192.168.56.0/24, 205.10.11.0/24, and 205.10.12.0/24. Run the following commands to serve installations on only the 192.168.56.0/24 network:

```
# svccfg -s system/install/server:default \
setprop all_services/networks = 192.168.56.0/24
# svcadm refresh system/install/server:default
```

Configure the Web Server Host Port

An AI server hosts install services using a web server. By default, the web server is hosted on port 5555. To customize the port that hosts the install services web server, configure the

`all_services/port` property of the `svc:/system/install/server:default` SMF service. The following commands configure the AI server to host install services from port 7000:

```
# svccfg -s system/install/server:default setprop all_services/port = 7000
# svccfg refresh system/install/server:default
```

Note – Customize the `port` property before creating any install services. If the `port` property is modified after install services are created, those existing install services will no longer function properly and will need to be deleted and recreated.

Create an AI Install Service

An install server can have more than one install service. Create a separate install service for each client hardware architecture and each different version of the Oracle Solaris 11 OS that you want to install.

Use the `installadm create-service` command to create an AI install service. Give the service a meaningful name, and specify the path where you want the service created. Specify the source of the network boot image (net image) package or ISO file.

When an AI install service is created, the AI SMF service, `system/install/server`, is enabled if it was not already enabled. The install service image is mounted at `/etc/netboot/`*svcname*. For SPARC install services, the `wanboot.conf` file is at the root of the install service image. For x86 install services, the `menu.lst` GRUB menu is at the root of the install service image.

When the first install service for a particular architecture is created on an install server, an alias of that service, `default-i386` or `default-sparc`, is automatically created. This default service is a complete service, with its own manifests and profiles. This default service is used for all installations on clients of that architecture that were not explicitly associated with a different install service with the `create-client` subcommand.

To change which service the `default-`*arch* service aliases, set the `aliasof` property using the `set-service` subcommand. Manifests and profiles that were added to either service remain the same after resetting an alias. The only change is which net image the service uses. See "Modifying Install Service Properties" on page 94 for more information about setting the `aliasof` property.

If a `default-`*arch* alias is changed to a new install service and a local ISC DHCP configuration is found, this default alias boot file is set as the default DHCP server-wide boot file for that architecture.

If a local ISC DHCP server is already configured when a new `default-`*arch* alias is created, the default boot file for this architecture is set to the boot file of this new alias.

Each service, including the `default-`*arch* service, includes a default AI manifest in *imagepath*`/auto_install/manifest`. This manifest can be copied to another file which can be edited and then added to an install service with the `create-manifest` subcommand. See "Customizing an XML AI Manifest File" on page 118.

The `installadm create-service` command also provides a net image on a web server running on port 5555. For example, the web server address might be `http://`*aiserver*`:5555/export/aiserver/s11-ai-x86/s11-x86`.

For information about all options, see "Create an Install Service" on page 92 or the `installadm(1M)` man page.

```
installadm create-service [-n svcname]
    [-s FMRI_or_ISO] [-d imagepath]
```

`-n` *svcname*
: If you do not provide a name for the install service, a default name is assigned.

`-s` *FMRI_or_ISO*
: The *FMRI* is the identifier of the IPS AI net image package, which is `install-image/solaris-auto-install` in the Oracle Solaris 11 release. The *ISO* is the path name of the AI net image ISO file.

  If you do not specify *FMRI_or_ISO*, the newest version of the `install-image/solaris-auto-install` package is installed from the first publisher in the `pkg publisher` list that provides that package.

  To install a different version of the package, or to install the package from a different publisher, specify the version or publisher in the *FMRI*. For example, specify `pkg://`*publisher*`/install-image/solaris-auto-install` or `pkg://`*publisher*`/install-image/solaris-auto-install@`*version*. Use the `-p` option to specify the particular package repository.

`-d` *imagepath*
: The *imagepath* is the location of the new install service. The `install-image/solaris-auto-install` package is installed to this location, or the specified ISO file is expanded at this location.

  If you do not specify *imagepath*, the service is created at `/export/auto_install/`*svcname*, and you are prompted to confirm that you want to use the automatically generated location. Specify the `-y` option to suppress this prompt.

The `create-service` command can set up DHCP on the AI install server as shown in "Create an Install Service Including Local DHCP Setup" on page 90. See Part II, "DHCP," in *Oracle Solaris Administration: IP Services* if you want to set up a separate DHCP server or configure an existing DHCP server for use with AI. The DHCP server must be able to provide DNS information to the systems to be installed.

Create an Install Service Without DHCP Setup

In the examples in this section, DHCP is already set up on a different server or will be set up later. If the `create-service` command does not detect that ISC DHCP is running on this server, the output of the command displays instructions for configuring DHCP. In these examples, the `create-service` command provides the boot file required for DHCP configuration.

Create a SPARC Install Service Using an ISO File

This example creates an AI install service for SPARC clients using a net image from an ISO file.

```
# installadm create-service -n s11-sparc \
-s /var/tmp/images/sparc/sol-11-dev-170-ai-sparc.iso \
-d /install/images/s11-sparc

Creating service: s11-sparc

Setting up the target image at /install/images/s11-sparc ...
Service discovery fallback mechanism set up
Creating SPARC configuration file
Refreshing install services

Creating default-sparc alias.

No local DHCP configuration found. This service is the default alias
for all SPARC clients. If not already in place, the following should
be added to the DHCP configuration:
        Boot file            : http://10.80.238.5:5555/cgi-bin/wanboot-cgi

Service discovery fallback mechanism set up
Creating SPARC configuration file
Refreshing install services
```

The following operations are performed as a result of executing the above `installadm create-service` command.

1. The install service is named `s11-sparc`.
2. The install service target directory, `/install/images/s11-sparc`, is created.
3. The ISO file, `/var/tmp/images/sparc/sol-11-dev-170-ai-sparc.iso`, is unpacked into the net image location, `/install/images/s11-sparc`.
4. The `wanboot.conf` file for this service is generated at `/install/images/s11-sparc/wanboot.conf`.
5. The AI SMF service, `system/install/server`, is refreshed to mount `/install/images/s11-sparc` as `/etc/netboot/s11-sparc`.
6. Since this is the first SPARC install service created on this install server, the `default-sparc` service alias is automatically created. The image from `s11-sparc` is used by the alias, so `/install/images/s11-sparc` is also mounted as `/etc/netboot/default-sparc`.

7. The configuration file `/etc/netboot/wanboot.conf` is symbolically linked to `/etc/netboot/default-sparc/wanboot.conf`. The configuration file `/etc/netboot/system.conf` is symbolically linked to `/etc/netboot/default-sparc/system.conf`.
8. The boot file required for DHCP configuration, `http://10.80.238.5:5555/cgi-bin/wanboot-cgi`, is provided.
9. If a local ISC DHCP server is already configured, the boot file of the new `default-sparc` alias is set as the default boot file for all SPARC clients. This is true regardless of whether the `-i` and `-c` options are used.

Create an x86 Install Service Using an IPS Package

This example creates an AI install service for x86 clients using a net image from an IPS package. This command also illustrates default behavior when options are not specified since this command only provides the install service name option. In addition to the boot file required for DHCP configuration, this command also provides the boot server IP required for DHCP configuration.

```
# installadm create-service -n s11-i386 -y

Creating service from: pkg:/install-image/solaris-auto-install
Download: install-image/solaris-auto-install ...  Done
Install Phase ...  Done
Package State Update Phase ...  Done
Image State Update Phase ...  Done
Reading Existing Index ...  Done
Indexing Packages ...  Done

Creating service: s11-i386

Image path: /export/auto_install/s11-i386

Refreshing install services

Creating default-i386 alias.

No local DHCP configuration found. This service is the default
alias for all PXE clients. If not already in place, the following should
be added to the DHCP configuration:
        Boot server IP       : 10.134.125.136
        Boot file            : default-i386/boot/grub/pxegrub

Refreshing install services
```

The following operations are performed as a result of executing the above `installadm create-service` command.

1. The install service is named `s11-i386`.
2. Because no net image source option is specified, the newest version of the `install-image/solaris-auto-install` package is obtained from the first publisher in the install server publisher list that provides this package.
3. Because no net image destination is specified with the `-d` option, the image is created in the default directory, `/export/auto_install/s11-i386`. Because the `y` option is specified, the prompt to confirm that this default destination is acceptable is suppressed.
4. The `install-image/solaris-auto-install` package is installed into the net image location, `/export/auto_install/s11-i386`.

   By default, the variant of the `install-image/solaris-auto-install` package that is installed matches the architecture of the AI install server. In this example, the install server is an x86 system. If you wanted to create a SPARC install service on this server, you would need to use the `-a` option. See "Create an Install Service" on page 92 for information about the `-a` option.
5. The `pxegrub` menu is created at `/export/auto_install/s11-i386/menu.lst`.
6. The AI SMF service, `system/install/server`, is refreshed to mount `/export/auto_install/s11-i386` as `/etc/netboot/s11-i386`.
7. Since this is the first x86 install service created on this install server, the `default-i386` service alias is automatically created. The image from `s11-i386` is used by the alias, so `/export/auto_install/s11-i386` is also mounted as `/etc/netboot/default-i386`.
8. The boot server IP required for DHCP configuration provided. The boot file required for DHCP configuration, `default-i386/boot/grub/pxegrub`, is provided.
9. If a local ISC DHCP server is already configured, the boot file of the new `default-i386` alias is set as the default boot file for all x86 clients. This is true regardless of whether the `-i` and `-c` options are used.

Create an Install Service Including Local DHCP Setup

You can use the `installadm create-service` command to set up a DHCP server on this AI install server. The following example creates an install service for x86 clients where the network consists of a single subnet and the install server also acts as the DHCP server for the network, using DNS to resolve host names. This install service serves twenty IP addresses (`-c`), starting from 10.80.239.150 (`-i`). If a DHCP server is not yet configured, an ISC DHCP server is configured. If an ISC DHCP server is already configured, that DHCP server is updated.

Note that when `-i` and `-c` arguments are provided and DHCP is configured, no binding exists between the install service being created and the IP range. When `-i` and `-c` are passed, the IP

range is set up, a new DHCP server is created if needed, and that DHCP server remains up and running for all install services and all clients to use. The network information provided to the DHCP server has no specific bearing on the service being created.

If the IP range requested is not on a subnet that the install server has direct connectivity to and the install server is multihomed, the `-B` option is used to provide the address of the boot file server (usually an IP address on this system). This should only be necessary when multiple IP addresses are configured on the install server and DHCP relays are employed. In all other configurations, the software can determine this automatically.

```
# installadm create-service -n s11-x86 \
-s /var/tmp/images/i386/sol-11-dev-171-ai-x86.iso \
-d /install/images/s11-x86 \
-i 10.80.239.150 -c 20

Creating service from: /var/tmp/images/i386/sol-11-dev-171-ai-x86.iso
Setting up the image ...

Creating service: s11-x86

Image path: /install/images/s11-x86

Starting DHCP server...
Adding IP range to local DHCP configuration

Refreshing install services

Creating default-i386 alias.

Setting the default PXE bootfile in the local DHCP configuration to
'default-i386/boot/grub/pxegrub'

Refreshing install services
```

The following operations are performed as a result of executing the above `installadm create-service` command.

1. The install service is named `s11-x86`.
2. The install service target directory, `/install/images/s11-x86`, is created.
3. The ISO file, `/var/tmp/images/i386/sol-11-dev-171-ai-x86.iso`, is unpacked into the net image location, `/install/images/s11-x86`.
4. The `pxegrub` menu is created at `/install/images/s11-x86/menu.lst`.
5. The AI SMF service, `system/install/server`, is refreshed to mount `/install/images/s11-x86` as `/etc/netboot/s11-x86`.
6. Since this is the first x86 install service created on this install server, the `default-i386` service alias is automatically created. The image from `s11-x86` is used by the alias, so `/install/images/s11-x86` is also mounted as `/etc/netboot/default-i386`.

7. A DHCP service is created if necessary, and IP addresses 10.80.239.150 through 10.80.239.169 are provisioned. If DHCP service is already set up on this server, the `-i` and `-c` options update the DHCP server with new IP addresses for this service.
8. The `default-i386/boot/grub/pxegrub` boot file is added to the local DHCP configuration as the default boot file for PXE clients.

Maintain an Install Server

After you have set up an AI install server, you might want to perform some of the following tasks. For complete information, see the `installadm(1M)` man page.

- "Add, Modify, or Delete an Install Service" on page 92
- "Associate Clients With Install Services" on page 95
- "Associate Client-Specific Installation Instructions With Install Services" on page 97
- "List All Install Services on the Install Server" on page 103
- "List Clients Associated With Install Services" on page 103
- "List All AI Manifests and System Configuration Profiles" on page 104

Add, Modify, or Delete an Install Service

You need a separate install service for each different client architecture that you plan to install and for each different version of the Oracle Solaris 11 OS that you plan to install on client systems.

Create an Install Service

Use the following command to create an install service. See "Create an AI Install Service" on page 86 for examples.

```
installadm create-service [-n svcname] [-s FMRI_or_ISO]
    [-p prefix=origin] [-a architecture]
    [-d imagepath] [-y] [-t existing_service]
    [-i dhcp_ip_start] [-c count_of_ipaddr]
    [-b boot_property=value,...] [-B server_ipaddr]
```

`-n` *svcname*
: The *svcname* can consist of alphanumeric characters, underscores (_), and hyphens (-). The first character of *svcname* cannot be a hyphen. If you do not provide a name for the install service, a default name is assigned.

`-s` *FMRI_or_ISO*
: This option specifies the source of the net boot image. The *FMRI* is the identifier of the IPS AI net image package, which is

`install-image/solaris-auto-install` in the Oracle Solaris 11 release. If you are using an AI net image ISO file, specify the path name of the net image ISO file.

If you do not specify *FMRI_or_ISO*, the newest version of the `install-image/solaris-auto-install` package is installed from the first publisher in the `pkg publisher` list that provides that package.

To install a different version of the package, or to install the package from a different publisher, specify the version or publisher in the *FMRI*. For example, specify `pkg://`*publisher*`/install-image/solaris-auto-install` or `pkg://`*publisher*`/install-image/solaris-auto-install@`*version*. Use the `-p` option to specify the particular package repository.

`-p` *prefix=origin*

This option specifies the IPS package repository from where you want to retrieve `install-image/solaris-auto-install` package. The *prefix* is the publisher name and the *origin* is the URI, as in `solaris=http://pkg.oracle.com/solaris/release/`.

If `-s` and `-p` are not specified, then the newest version of the `install-image/solaris-auto-install` package is installed from the first publisher in the `pkg publisher` list that provides that package.

`-a` *architecture*

This option is used only when the net image source is an IPS package. The *architecture* specifies the architecture of the clients to be installed. You can specify either `i386` or `sparc`.

When creating a service from an IPS package, the variant of the package that is installed by default is the variant that matches the architecture of the system where the service is created. For example, if your AI install server is x86, the variant of the `solaris-auto-install` package that `create-service` installs by default is the `i386` variant. If you are creating a service to install SPARC clients, specify `-a sparc` to install the `sparc` variant of the `solaris-auto-install` package in the install service.

`-d` *imagepath*

The *imagepath* is the location of the new install service. If you do not specify *imagepath*, the service is created at `/export/auto_install/`*svcname*, and you are prompted to confirm that you want to use the automatically generated location. Specify the `-y` option to suppress this prompt.

`-y`	Specify the `-y` option to suppress the prompt to confirm using an automatically generated *imagepath*.
`-t` *existing_service*	Designates the new service as an alias, which shares the net image of the *existing_service* service but has its own manifests, profiles, and clients.
`-i` *dhcp_ip_start*	This option specifies the starting IP address in a range to be added to the local DHCP configuration. The number of IP addresses is provided by the `-c` option. If a local ISC DHCP configuration does not exist, an ISC DHCP server is started.
`-c` *count_of_ipaddr*	Sets up a total number of IP addresses in the DHCP configuration equal to the value of the *count_of_ipaddr*. The first IP address is the value of *dhcp_ip_start* that is provided by the `-i` option.
`-b` *boot_property=value*,...	For x86 services only. This option sets a property value in the service-specific `menu.lst` file in the service image. Use this option to set boot properties that are specific to this service. This option can accept multiple comma-separated *boot_property=value* pairs.
`-B` *server_ipaddr*	Use this option to provide the IP address of the boot server from which clients should request boot files. This option is required only if this IP address cannot be determined by other means.

Modifying Install Service Properties

Use the `installadm set-service` command to specify a property and value to set for the *svcname* install service.

```
installadm set-service -o prop=value svcname
```

The *prop=value* pair must be one of the following:

`aliasof=`*another_svcname*
: Changes the install service that the *svcname* service is an alias of.

  Setting this property changes the *svcname* service to be an alias of the *another_svcname* service. The *svcname* service must already be an alias. The `default-`*arch* install services are aliases. A service created using the `-t` option of `create-service` is an alias. Use the `installadm list` command as shown in "List All Install Services on the Install Server" on page 103 to confirm that *svcname* is an alias.

  Manifests, profiles, and client bindings that were added to either *svcname* or *another_svcname* remain the same after resetting the alias. The only change is which net image the *svcname* service uses.

Manifests and profiles that were added to *svcname* prior to setting the alias are revalidated when the alias is reset since the AI and SMF DTDs associated with the new net image could be different. This validation is the same validation that is performed by `create-manifest` and `create-profile`, described below.

`default-manifest=`*manifest_or_script_name*
Designates a particular manifest or script that is already registered with a given service to be the default manifest or script for that service. Use the following command to show a list of manifests and scripts registered with this service.

```
$ installadm list -n svcname -m
```

Rename an Install Service

Use the following command to rename *svcname* to *newsvcname*.

```
installadm rename-service svcname newsvcname
```

The *newsvcname* can consist of alphanumeric characters, underscores (_), and hyphens (-). The first character of *newsvcname* cannot be a hyphen.

Enable or Disable an Install Service

Use the following command to enable the *svcname* install service.

```
installadm enable svcname
```

Use the following command to disable the *svcname* install service.

```
installadm disable svcname
```

Delete an Install Service

Use the following command to delete the *svcname* install service.

```
installadm delete-service [-r] [-y] svcname
```

This command deletes the AI manifests and system configuration profiles, the net image, and the web server configuration for the *svcname* install service. If the service is a default alias and a local ISC DHCP configuration exists, the boot file associated with this service is removed from the ISC DHCP configuration.

Use the `-r` option to remove any clients associated with this service and any services aliased to this service. Use the `-y` option to suppress confirmation prompts.

Associate Clients With Install Services

The `installadm create-client` command associates a client with a specific install service. See "Setting Up an Install Client" on page 183 for more examples and sample output.

Add a Client To an Install Service

Use the `installadm create-client` command to associate the *macaddr* client with the *svcname* install service and provide custom client settings for x86 clients. To find the MAC address of a system, use the `dladm` command as described in *Oracle Solaris Administration: Network Interfaces and Network Virtualization* and in the `dladm`(1M) man page.

```
installadm create-client [-b property=value,...]
    -e macaddr -n svcname
```

If the client is an x86 system and a local ISC DHCP configuration exists, the client is configured in the ISC DHCP configuration.

For x86 client systems, use the `-b` option to set boot properties in the client-specific `menu.lst` file in `/etc/netboot`.

The following command adds the client with MAC address `00:14:4f:a7:65:70` to the `s11-sparc` install service.

```
# installadm create-client -e 00:14:4f:a7:65:70 -n s11-sparc
```

The following example adds an x86 client and redirects installation output to a serial console.

```
# installadm create-client -e c0ffeec0ffee -n s11-x86 -b 'console=ttya'
```

Associate a Client With a Different Install Service

A client can be associated with only one install service. If you run the `installadm create-client` command more than once and specify the same MAC address each time, that client is associated only with the install service that was specified last.

Delete a Client From an Install Service

Use the `installadm delete-client` command to delete the *macaddr* client from its associated install service.

```
installadm delete-client macaddr
```

If the client is an x86 system and a local ISC DHCP configuration exists, the client is unconfigured in the ISC DHCP configuration.

The following command deletes the client with MAC address `00:14:4f:a7:65:70`. You do not need to specify the service name since a client can be associated with only one install service.

```
# installadm delete-client 00:14:4f:a7:65:70
```

Associate Client-Specific Installation Instructions With Install Services

You can specify multiple sets of installation instructions for each install service, and you can specify which instruction set to use for each client.

Add an AI Manifest

Use the `installadm create-manifest` command to add the *manifest_or_script_filename* custom AI manifest to the *svcname* install service.

```
installadm create-manifest -n svcname
    -f manifest_or_script_filename [-m manifest_or_script_name]
    [-c criteria=value|list|range...
    | -C criteriafile] [-d]
```

The *manifest_or_script_filename* can be an AI manifest XML file, or it can be a derived manifests script. See Chapter 10, "Provisioning the Client System." The `create-manifest` subcommand validates XML manifest files before adding them to the install service. To validate derived manifests script files, use the `aimanifest validate` command as shown in "Add a Derived Manifests Script To an Install Service" on page 133.

The *manifest_or_script_name* is the name displayed by the `installadm list` command. See "List All AI Manifests and System Configuration Profiles" on page 104. If *manifest_or_script_name* is not provided, the *manifest_or_script_name* is the value of the `name` attribute of the `ai_instance` element, if present, or the base name of the *manifest_or_script_filename*.

Use the `-d` option to make this AI manifest the default AI manifest. The default manifest is the manifest used by any clients that do not match criteria specified for any other manifests in this install service. If `-d` is specified, then `-c` and `-C` are ignored for the purpose of manifest selection. The previous default AI manifest for this service becomes inactive if it has no client criteria. If the previous default manifest has criteria, it remains active and its associated criteria become effective.

If `-d` is not specified, then either `-c` or `-C` must be specified to define which clients should use this AI manifest to complete their installation. If `-d`, `-c`, and `-C` are all not specified, then this manifest is added to the service but is inactive: No clients can use it.

If you want certain clients to use this AI manifest, first make sure those clients will use the install service specified in this `create-manifest` command. Any client systems that have not been explicitly associated with a particular install service by using the `create-client` command will use the appropriate `default-`*arch* install service. You can add customized AI manifests to the `default-`*arch* install service, or you can add customized AI manifests to a different service and then use `create-client` to make sure clients use that service.

The `-c` option specifies client selection criteria on the command line. The `-C` option specifies criteria in an XML file. The value of *criteriafile* is a full path and file name. See Chapter 9, "Customizing Installations," for a list of criteria keywords with command line and file examples.

The `installadm create-manifest` command verifies that criteria of the same type do not overlap. For example, if one criteria specification matches IP addresses from 10.0.0.0 to 10.255.255.255, `installadm` exits with an error if you try to add a criteria specification that matches IP address 10.10.10.10. For more information about criteria specifications, see Chapter 9, "Customizing Installations."

The following command adds the `manifest_t200.xml` manifest to the `s11-sparc` install service. The `-c` option specifies that any clients that are using this install service and identify themselves as Sun Fire T200 servers are assigned the `manifest_t200.xml` installation instructions.

```
# installadm create-manifest -f ./mymanifests/manifest_t200.xml \
-m t200 -n s11-sparc -c platform="SUNW,Sun-Fire-T200"
```

The following command is equivalent to the preceding command if the content of the `criteria_t200.xml` file is as shown.

```
# installadm create-manifest -f ./mymanifests/manifest_t200.xml \
-m t200 -n s11-sparc -C ./mymanifests/criteria_t200.xml
```

Following is the content of the `criteria_t200.xml` file.

```
<ai_criteria_manifest>
    <ai_criteria name="platform">
        <value>SUNW,Sun-Fire-T200</value>
    </ai_criteria>
</ai_criteria_manifest>
```

Update an AI Manifest

Use the `installadm update-manifest` command to replace the contents of the *manifest_or_script_name* AI manifest with the *manifest_or_script_filename* AI manifest for the *svcname* install service. Criteria, default status, and *manifest_or_script_name* are not changed as a result of the update.

```
installadm update-manifest -n svcname
    -f manifest_or_script_filename [-m manifest_or_script_name]
```

The `update-manifest` subcommand validates XML manifest files before adding them to the install service. To validate derived manifests script files, use the `aimanifest validate` command as shown in "Add a Derived Manifests Script To an Install Service" on page 133.

The *manifest_or_script_name* manifest must already exist in the *svcname* service. Use the `installadm list` command to confirm. See "List All AI Manifests and System Configuration Profiles" on page 104.

If *manifest_or_script_name* is not specified, then the manifest that is replaced is identified in one of the following ways:

- The `name` attribute of the `ai_instance` element in the *manifest_or_script_filename* manifest, if this attribute is specified and if the value of this attribute matches the *manifest_or_script_name* of an existing manifest for this install service.
- The base name of the *manifest_or_script_filename* manifest if this name matches the *manifest_or_script_name* of an existing manifest for this install service.

The following command updates the content of the `t200` manifest in the `s11-sparc` service with the content of `./mymanifests/manifest_newt200.xml`. The name of the manifest in `installadm list` is still `t200`.

```
# installadm update-manifest -n s11-sparc \
-f ./mymanifests/manifest_newt200.xml -m t200
```

Delete an AI Manifest

Use the `installadm delete-manifest` command to remove the *manifest_or_script_name* AI manifest from the *svcname* install service. The *manifest_or_script_name* is the manifest name that the `installadm list` command returns. See "List All AI Manifests and System Configuration Profiles" on page 104.

```
installadm delete-manifest -m manifest_or_script_name -n svcname
```

You cannot delete the default AI manifest.

The following command removes the `t200` AI manifest from the `s11-sparc` install service.

```
# installadm delete-manifest -m t200 -n s11-sparc
```

Associate Client-Specific Configuration Instructions With Install Services

You can specify multiple sets of system configuration instructions for each install service. Multiple system configuration profiles can be associated with each client.

Add a System Configuration Profile

Use the `installadm create-profile` command to add the *profile_filename* system configuration profile to the *svcname* install service.

```
installadm create-profile -n svcname
    -f profile_filename... [-p profile_name]
    [-c criteria=value|list|range... | -C criteriafile]
```

Multiple system configuration profiles can be specified in one `create-profile` command because a single client can use multiple configuration profiles. The same client selection criteria, or overlapping criteria, or no criteria can be specified for multiple profiles. When no criteria are specified, the profile is used by all clients that use this install service.

The `create-profile` subcommand validates system configuration profiles before adding them to the install service. To validate profiles under development, see the `validate` subcommand below.

The *profile_filename* can contain replacement tags that get their values from criteria specified in the `create-profile` command or from environment variables. See Chapter 11, "Configuring the Client System."

The *profile_name* is the name displayed by the `installadm list` command. See "List All AI Manifests and System Configuration Profiles" on page 104. If *profile_name* is not provided, the *profile_name* is the base name of the *profile_filename*. The `-p` option is not valid when more than one *profile_filename* is specified.

The `-c` option specifies client selection criteria on the command line. The `-C` option specifies criteria in an XML file. The value of *criteriafile* is a full path and file name. See Chapter 9, "Customizing Installations," for a list of criteria keywords with command line and file examples.

If you want certain clients to use this system configuration profile, first make sure those clients will use the install service specified in this `create-profile` command. Any client systems that have not been explicitly associated with a particular install service by using the `create-client` command will use the appropriate `default-`*arch* install service. You can add customized system configuration profiles to the `default-`*arch* install service, or you can add customized configuration profiles to a different service and then use `create-client` to make sure clients use that service.

The following command adds the `profile_t200.xml` profile to the `s11-sparc` install service. The `-c` option specifies that any clients that are using this install service and identify themselves as Sun Fire T200 servers are assigned the `profile_t200.xml` system configuration instructions.

```
# installadm create-profile -f ./mymanifests/profile_t200.xml \
-p t200 -n s11-sparc -c platform="SUNW,Sun-Fire-T200"
```

Validate a System Configuration Profile

Use the `installadm validate` command to validate system configuration profiles for syntactic correctness.

```
installadm validate -n svcname -P profile_filename... | -p profile_name...
```

Use the `-P` option to validate profiles that have not been added to the install service. The *profile_filename* is a full path name to the file.

Use the `-p` option to validate profiles that have already been added to the *svcname* install service using the `create-profile` subcommand as shown in "List All AI Manifests and System Configuration Profiles" on page 104. The `create-profile` subcommand validates system configuration profiles before adding them to the install service. The `validate -p` subcommand verifies that the profile has not become corrupted since it was added.

The *svcname* is required for both *profile_filename* and *profile_name* profiles. The service name is required for profiles that have not yet been added to an install service because the `service_bundle`(4) DTD might be different in different versions of the OS. An install service might be defined to install a different version of the OS than the version your install server is running. The profile must be validated against the DTD that will be in use on the client being installed.

Validated profiles are output to `stdout`. Errors are listed to `stderr`.

Delete a System Configuration Profile

Use the `installadm delete-profile` command to remove the *profile_name* system configuration profile from the *svcname* install service. The *profile_name* is the profile name that the `installadm list` command returns. See "List All AI Manifests and System Configuration Profiles" on page 104.

```
installadm delete-profile -p profile_name... -n svcname
```

The following command removes the `t200` system configuration profile from the `s11-sparc` install service.

```
# installadm delete-profile -p t200 -n s11-sparc
```

Export an AI Manifest or a System Configuration Profile

Use the `installadm export` command to copy the contents of the specified AI manifests or system configuration profiles from the *svcname* install service to the *pathname* file or directory.

```
installadm export -n svcname
    -m manifest_or_script_name... -p profile_name...
    [-o pathname]
```

If *pathname* is not specified, the manifest and profile contents go to `stdout`. If only one input file is specified, *pathname* can be a file name. If more than one input file is specified, *pathname* must be a directory.

The *manifest_or_script_name* can be an XML AI manifest or a derived manifests script. See Chapter 10, "Provisioning the Client System," for information about creating manifests and derived manifests scripts.

Use the `installadm export` command to:

- Check the specifications in the manifests and profiles.
- Modify an existing manifest or profile. Use an existing manifest or profile as a base for creating a new manifest or profile.

Modify Criteria for an AI Manifest or a System Configuration Profile

Use the `installadm set-criteria` command to update the client criteria associated with an AI manifest or system configuration profiles that you already added to the *svcname* install service using `create-manifest` or `create-profile`.

```
installadm set-criteria -m manifest_or_script_name -p profile_name... -n svcname
    -c criteria=value|list|range... | -C criteriafile |
    -a criteria=value|list|range...
```

Zero or one manifest can be specified along with zero or any number of profiles on the same `set-criteria` command line. The *manifest_or_script_name* and *profile_name* names are the names that the `installadm list` command returns. See "List All AI Manifests and System Configuration Profiles" on page 104.

Use the `-c` or `-C` options to replace the criteria for these existing manifests and profiles with the new criteria specified. Use the `-a` option to retain the existing criteria and add the specified criteria. See Chapter 9, "Customizing Installations," for more information about specifying criteria.

The following command adds a memory criteria specification to a manifest that was originally added to this service with a platform criteria specification.

```
# installadm set-criteria -m t200 -n s11-sparc -a mem="4096-unbounded"
```

The result of the criteria specified with `create-manifest` and added with `set-criteria` is that the manifest is used by any client that is using this install service that is a Sun Fire T200 server and that has at least 4 Gbytes of memory.

You could achieve this same result by using the `-C` option instead of the `-a` option with the following `criteria_t200.xml` file.

```
<ai_criteria_manifest>
    <ai_criteria name="platform">
        <value>SUNW,Sun-Fire-T200</value>
    </ai_criteria>
    <ai_criteria name="mem">
        <range>
            4096
            unbounded
```

```
        </range>
    </ai_criteria>
</ai_criteria_manifest>
```

Show Information About Install Services

Use the `installadm list` command to show information about install services.

```
installadm list [-n svcname] [-c] [-m] [-p]
```

List All Install Services on the Install Server

The following command displays all of the install services on this server. In this example, two enabled install services are found. Disabled services have a Status value of `off`. Recall that the first service created for a given architecture is the default service for clients of that architecture. See "Add, Modify, or Delete an Install Service" on page 92.

```
$ installadm list

Service Name  Alias Of  Status Arch  Image Path
------------  --------  ------ ----  ----------
default-i386  s11-x86   on     x86   /install/images/s11_x86
default-sparc s11-sparc on     Sparc /install/images/s11_sparc
s11-sparc               on     Sparc /install/images/s11_sparc
s11-x86                 on     x86   /install/images/s11_x86
```

Show Information for a Specified Install Service

The following command displays information about the install service specified by the `-n` option:

```
$ installadm list -n s11-sparc

Service Name  Alias Of  Status Arch  Image Path
------------  --------  ------ ----  ----------
s11-sparc               on     Sparc /install/images/s11_sparc
```

List Clients Associated With Install Services

The following command lists all the clients that are associated with the install services on this install server. The clients were associated with the install services by using the `installadm create-client` command. See "Add a Client To an Install Service" on page 96.

```
$ installadm list -c

Service Name Client Address    Arch  Image Path
------------ --------------    ----  ----------
s11-sparc    00:14:4F:A7:65:70 Sparc /install/images/s11_sparc
s11-x86      08:00:27:8B:BD:71 x86   /install/images/s11_x86
             01:C2:52:E6:4B:E0 x86   /install/images/s11_x86
```

List Clients Associated With a Specific Install Service

The following command lists all the clients that have been added to the specified install service. In the following example, one client is associated with the `s11-sparc` install service.

```
$ installadm list -c -n s11-sparc

Service Name Client Address    Arch  Image Path
------------ --------------    ----  ----------
s11-sparc    00:14:4f:a7:65:70 Sparc /install/images/s11_sparc
```

Show Information About Customized Installations

The commands in this section show which AI manifests and system configuration profiles are associated with a particular install service. These commands also show which client criteria are associated with each manifest and profile.

List All AI Manifests and System Configuration Profiles

The following command lists all AI manifests, derived manifests scripts, and system configuration profiles for all install services on this install server. The Manifest/Profile column displays the internal name of the manifest, script, or profile.

```
# installadm list -m -p

Service Name  Manifest
------------  --------
s11-sparc     t200
s11-x86       ipv4
              mem1

Service Name  Profile
------------  --------
s11-sparc     mac1
              t200
s11-x86       mac2
              mac3
              ipv4
              mem1
```

List Manifests and Profiles Associated With a Specified Install Service

The following example shows all AI manifests, derived manifests scripts, and system configuration profiles associated with the install service `s11-sparc`. The Manifest/Profile column displays the internal name of the manifest, script, or profile. The Criteria column shows the associated client criteria.

The `orig_default` manifest is the original default AI manifest that was part of the install service when the install service was created. The `mem1` manifest was created with memory criteria and also with the `-d` option to make it the new default manifest for this service. Because `mem1` is the

default manifest, its criteria are ignored. If another manifest is created as the default manifest, then the `mem1` criteria are used to select clients to use this manifest. The original default manifest is inactive because it has no associated criteria to determine which clients should use it. Only the default manifest can have no associated criteria. A client that does not match the criteria to use any other manifest uses the default manifest. See Chapter 9, "Customizing Installations," for more information about selecting an AI manifest.

```
# installadm list -m -p -n s11-sparc

Manifest      Status    Criteria
--------      ------    --------
orig_default  Inactive  None
mem1          Default   (Ignored: mem = 2048 - 4095)
t200                    platform = SUNW,Sun-Fire-T200
                        mem = 4096-unbounded

Profile  Criteria
-------  --------
mac1     mac  = 01:C2:52:E6:4B:E0
         hostname = server1
         ipv4 = 192.168.168.251
t200     platform = SUNW,Sun-Fire-T200
         mem = 4096-unbounded
```

Administering the AI SMF Service

On the AI server, the SMF service `svc:/system/install/server:default` is the service that represents the overall state of the AI server application and all install services.

EXAMPLE 8–2 Enabling the AI SMF Service

The AI SMF service is enabled when you run the `installadm create-service` command. The AI SMF service also is enabled when you run any other `installadm` command that affects existing install services. To manually enable the AI SMF service, run the following command:

```
# svcadm enable svc:/system/install/server:default
```

The AI SMF service goes into maintenance mode if no install services are currently enabled on the install server or if a problem occurs that requires attention.

EXAMPLE 8–3 Disabling the AI SMF Service

To disable the AI SMF service, run the following command:

```
# svcadm disable svc:/system/install/server:default
```

Do not disable the AI SMF service if any AI install service is still enabled. See "List All Install Services on the Install Server" on page 103 for information about how to see whether any install services are enabled.

◆ ◆ ◆ CHAPTER 9

Customizing Installations

To customize an installation, customize the installation instructions and the system configuration instructions. Then specify client criteria to match the customized installation and configuration instructions with clients identified by that criteria.

An AI install service includes one or more installation instructions files (AI manifests) and zero or more configuration instructions files (SMF system configuration profiles). Each client uses one and only one AI manifest. Each client can use any number of system configuration profiles. If a client system does not use any configuration profile, then an interactive tool opens on that client at first boot after that client installation to complete the configuration of that client.

Matching Clients With Installation and Configuration Instructions

When you use AI, you first set up a DHCP server and an install server. The install server has at least one AI boot image and an AI install service that is associated with that boot image. When a client boots, DHCP directs the client to the install server.

The client uses the default install service for that client architecture, or uses an assigned install service. The install service uses the methods described in this chapter to match the client with the correct installation and configuration instructions to use.

To define installations that use different boot images (a SPARC image and an x86 image, or different Oracle Solaris versions), create a separate service for each image.

To assign a client to a specific install service, add that client to the install service. See Chapter 15, "Installing Client Systems." Specify the MAC address of the client and the name of the install service for this client to use. When the client with this MAC address boots, DHCP directs the client to the install server, and the client uses the specified install service. To find the MAC address of a system, use the `dladm` command as described in *Oracle Solaris Administration: Network Interfaces and Network Virtualization* and in the `dladm`(1M) man page.

To define more than one type of installation for one net image, create additional AI manifests and create system configuration profiles. Add the new AI manifests and configuration profiles to the AI install service for that net image. Specify criteria that define which clients should use which AI manifest and which system configuration profiles. See "Associate Client-Specific Installation Instructions With Install Services" on page 97.

To create custom AI manifests, see Chapter 10, "Provisioning the Client System." To create system configuration profiles, see Chapter 11, "Configuring the Client System."

Selecting the AI Manifest

Each client uses one and only one AI manifest to complete its installation. The AI manifest is selected for a client according to the following algorithm:

- If no custom AI manifests are defined for this install service, the default AI manifest is used. The default AI manifest is not associated with any client criteria. See "Default AI Manifest" on page 113 for an example of a default AI manifest.
- If custom AI manifests are defined for this install service but the client does not match criteria for any custom AI manifest, then the client uses the default AI manifest.
- If the client matches criteria that have been specified for a custom AI manifest, the client uses that custom manifest.

  If client characteristics match criteria for multiple AI manifests, the client characteristics are evaluated in the order shown in Table 9–1 to select the manifest for the installation. The `installadm` tool verifies that criteria of the same type do not overlap. See "Add an AI Manifest" on page 97.

  Multiple non-overlapping criteria are used in the order specified in the table below. For example, if one criteria specification matches the client's MAC address and another criteria specification matches the same client's IP address, the manifest associated with the MAC address criteria specification is used, because `mac` is higher priority for selection than `ipv4`.

EXAMPLE 9–1 Matching Clients With AI Manifests

In the following example, two custom AI manifests have been added to the same install service. The client criteria associated with those manifests are as shown.

The `manifest_x86.xml` AI manifest was added to the service with the following criteria file that specifies client architecture:

```
<ai_criteria_manifest>
    <ai_criteria name="arch">
        <value>i86pc</value>
    </ai_criteria>
</ai_criteria_manifest>
```

The `manifest_mac1.xml` AI manifest was added to the service with the following criteria file that specifies a client MAC address:

EXAMPLE 9–1 Matching Clients With AI Manifests *(Continued)*

```
<ai_criteria_manifest>
    <ai_criteria name="mac">
        <value>00:14:4f:a7:65:70</value>
    </ai_criteria>
</ai_criteria_manifest>
```

If an x86 client is being installed, it is assigned `manifest_x86.xml`.

If a SPARC client with MAC address `00:14:4f:a7:65:70` is being installed, it is assigned `manifest_mac1.xml`.

If a SPARC system with some other MAC address is being installed, it is assigned the default AI manifest.

Selecting System Configuration Profiles

The same criteria keywords are used for selecting system configuration profiles for a client as are used for selecting an AI manifest. The `hostname` criteria keyword can be used only for configuration profiles, not for AI manifests. See Table 9–1.

More than one system configuration profile can be selected for any particular client. No algorithm is needed to narrow the selection to one profile.

If client characteristics match criteria for multiple system configuration profiles, all matching configuration profiles are applied to configure the system. For example, if one criteria specification matches the client's host name and another criteria specification matches the same client's memory size, both configuration profiles are used to configure that client.

Selection Criteria

The following table shows the criteria keywords that can be used to indicate which clients should use a particular AI manifest or system configuration profile. The examples column shows some possible values. The criteria keywords and values can be used with the following `installadm` subcommands: `create-manifest`, `create-profile`, and `set-criteria`.

Specify criteria keywords and values on the command line by using the `-c` option.

```
-c criteria=value|list|range
-c mac="aa:bb:cc:dd:ee:ff"
-c zonename="zone1 zone2"
-c mem="2048-unbounded"
```

Criteria can also be specified in `ai_criteria` elements in an XML file. The content of this file should be only criteria specifications. Use the `-C` option to name the criteria file on the command line. Examples are shown in the table.

TABLE 9–1 Criteria Keywords and Criteria Hierarchy

Criteria Name	Description	Command Line and XML File Examples

`mac`	Hexadecimal MAC address with colon (:) separators, or range of MAC addresses	**CLI, single MAC address:**
		`-c mac="0:14:4F:20:53:97"`
		CLI, range of MAC addresses:
		`-c mac=0:14:4F:20:53:94-0:14:4F:20:53:A0`
		XML, single MAC address:
		`<ai_criteria name="mac">`
		`  <value>0:14:4F:20:53:97</value>`
		`</ai_criteria>`
		XML, range of MAC addresses:
		`<ai_criteria name="mac">`
		`  <range>`
		`  0:14:4F:20:53:94`
		`  0:14:4F:20:53:A0`
		`  </range>`
		`</ai_criteria>`

`ipv4`	IP version 4 network address, or range of IP addresses	**CLI, single IP address:**
		`-c ipv4="10.6.68.127"`
		CLI, range of IP addresses:
		`-c ipv4="10.6.68.1-10.6.68.200"`
		XML, single IP address:
		`<ai_criteria name="ipv4">`
		`  <value>10.6.68.127</value>`
		`</ai_criteria>`
		XML, range of IP addresses:
		`<ai_criteria name="ipv4">`
		`  <range>`
		`  10.6.68.1`
		`  10.6.68.200`
		`  </range>`
		`</ai_criteria>`

TABLE 9–1 Criteria Keywords and Criteria Hierarchy *(Continued)*

Criteria Name	Description	Command Line and XML File Examples

`platform`	Platform name returned by `uname -i`
	Values include:
	`i86pc`
	`SUNW,SPARC-Enterprise`
	`SUNW,Sun-Fire-T200`
		CLI:
		`-c platform="SUNW,Sun-Fire-T200"`
		XML:
		`<ai_criteria name="platform">`
		`<value>SUNW,Sun-Fire-T200</value>`
		`</ai_criteria>`

`arch`	Architecture returned by `uname -m`
	Values: `i86pc`, `sun4u`, or `sun4v`
		CLI:
		`-c arch="i86pc"`
		XML:
		`<ai_criteria name="arch">`
		`<value>i86pc</value>`
		`</ai_criteria>`

`cpu`	CPU class returned by `uname -p`
	Values: `i386` or `sparc`
		CLI:
		`-c cpu="sparc"`
		XML:
		`<ai_criteria name="cpu">`
		`<value>sparc</value>`
		`</ai_criteria>`

`network`	IP version 4 network number, or a range of network numbers	**CLI, single IP address:**
		`-c network="10.0.0.0"`
		CLI, range of IP addresses:
		`-c network="11.0.0.0-12.0.0.0"`
		XML, single IP address:
		`<ai_criteria name="network">`
		`<value>10.0.0.0</value>`
		`</ai_criteria>`
		XML, range of IP addresses:
		`<ai_criteria name="network">`
		`<range>`
		`11.0.0.0`
		`12.0.0.0`
		`</range>`
		`</ai_criteria>`

TABLE 9–1 Criteria Keywords and Criteria Hierarchy *(Continued)*

Criteria Name	Description	Command Line and XML File Examples

`mem`	Memory size in megabytes returned by `prtconf`, or a range of memory size
	
	The `unbounded` keyword indicates no upper limit in a range.
		CLI, one memory size:
		`-c mem="4096"`
		CLI, range of memory size:
		`-c mem="2048-unbounded"`
		XML, one memory size:
		`<ai_criteria name="mem">`
		`  <value>4096</value>`
		`</ai_criteria>`
		XML, range of memory size:
		`<ai_criteria name="mem">`
		`  <range>`
		`  2048`
		`  unbounded`
		`  </range>`
		`</ai_criteria>`

`zonename`	Name or list of names of `zones`(5) zones as shown by `zoneadm list`. See Chapter 12, "Installing and Configuring Zones."	**CLI, single zone name:**
		`-c zonename="myzone"`
		CLI, list of zone names:
		`-c zonename="zoneA zoneB zoneC"`
		XML, single zone name:
		`<ai_criteria name="zonename">`
		`  <value>myzone</value>`
		`</ai_criteria>`
		XML, list of zone names:
		`<ai_criteria name="zonename">`
		`  <value>zoneA zoneB zoneC</value>`
		`</ai_criteria>`

TABLE 9–1 Criteria Keywords and Criteria Hierarchy *(Continued)*

Criteria Name	Description	Command Line and XML File Examples

`hostname`	Client host name or list of client host names.
	
	This criteria keyword can be used only for system configuration profiles, not for AI manifests.
		CLI, single host name:
		
		`-c hostname="host3"`
		
		CLI, list of host names:
		
		`-c hostname="host1 host2 host6"`
		
		XML, single host name:
		
		`<ai_criteria name="hostname">`
		`  <value>host3</value>`
		`</ai_criteria>`
		
		XML, list of host names:
		
		`<ai_criteria name="hostname">`
		`  <value>host1 host2 host6</value>`
		`</ai_criteria>`

Default AI Manifest

When you create a new install service, *install_service_image_path*`/auto_install/manifest/default.xml` is the initial default AI manifest for that install service.

This default AI manifest is shown below. This default manifest might be slightly different in different install images.

The `target` section in the default manifest defines ZFS file systems, or datasets, to be created. The default manifest does not define a target disk for the installation. Refer to the `ai_manifest(4)` man page for a description of how the default target location for the installation is determined when no target disk is specified in the manifest.

The `destination` section can be used to specify which locales to install. Facet specifications can be used in the manifest to limit which locales should be installed, which can save time and space if you do not need all locales. If no facets are specified, then facets for all locales default to true. Refer to the `ai_manifest`(4) man page for more information about setting image facets and attributes.

The software installation instructions specify the default IPS package repository and install the following two packages:

- The `entire` package is required. This incorporation package constrains system packages being installed to compatible versions. Proper system update and correct package selection depend on the presence of this incorporation. Do not remove the installation of this package from your AI manifest, and do not uninstall this package after installation.
- The `solaris-large-server` package is a group package of tools and device drivers that you might want in most environments that you install. This package installs many network and storage drivers, Python libraries, Perl, and much more. For a complete list of packages that are included in the `solaris-large-server` group package, use the `pkg contents` command as described in "Listing All Installable Packages In a Group Package" in *Adding and Updating Oracle Solaris 11 Software Packages*.

  For information about how to find the names of other packages that you might want to install, see *Adding and Updating Oracle Solaris 11 Software Packages*.

```
<?xml version="1.0" encoding="UTF-8"?>
<!--

 Copyright (c) 2008, 2011, Oracle and/or its affiliates. All rights reserved.

-->
<!DOCTYPE auto_install SYSTEM "file:///usr/share/install/ai.dtd.@DTD_VERSION_AI@">
<auto_install>
  <ai_instance name="default">
    <target>
      <logical>
        <zpool name="rpool" is_root="true">
          <filesystem name="export" mountpoint="/export"/>
          <filesystem name="export/home"/>
          <be name="solaris"/>
        </zpool>
      </logical>
    </target>
    <software type="IPS">
      <destination>
        <image>
          <!-- Specify locales to install -->
          <facet set="false">facet.locale.*</facet>
          <facet set="true">facet.locale.de</facet>
          <facet set="true">facet.locale.de_DE</facet>
          <facet set="true">facet.locale.en</facet>
          <facet set="true">facet.locale.en_US</facet>
          <facet set="true">facet.locale.es</facet>
          <facet set="true">facet.locale.es_ES</facet>
          <facet set="true">facet.locale.fr</facet>
          <facet set="true">facet.locale.fr_FR</facet>
          <facet set="true">facet.locale.it</facet>
          <facet set="true">facet.locale.it_IT</facet>
          <facet set="true">facet.locale.ja</facet>
          <facet set="true">facet.locale.ja_*</facet>
          <facet set="true">facet.locale.ko</facet>
          <facet set="true">facet.locale.ko_*</facet>
```

```
            <facet set="true">facet.locale.pt</facet>
            <facet set="true">facet.locale.pt_BR</facet>
            <facet set="true">facet.locale.zh</facet>
            <facet set="true">facet.locale.zh_CN</facet>
            <facet set="true">facet.locale.zh_TW</facet>
          </image>
        </destination>
        <source>
          <publisher name="solaris">
            <origin name="http://pkg.oracle.com/solaris/release"/>
          </publisher>
        </source>
        <!--
          By default the latest build available, in the specified IPS
          repository, is installed.  If another build is required, the
          build number has to be appended to the 'entire' package in the
          following form:

              <name>pkg:/entire@0.5.11-0.build#</name>
        -->
        <software_data action="install">
          <name>pkg:/entire@latest</name>
          <name>pkg:/group/system/solaris-large-server</name>
        </software_data>
      </software>
    </ai_instance>
</auto_install>
```

◆ ◆ ◆ CHAPTER 10

Provisioning the Client System

When you create an AI install service, you get a default AI manifest that specifies how to provision the clients. The AI manifest is an XML file that specifies where to install the operating system and what software packages to install. You can also specify disk configuration such as striping, mirroring, and partitioning. See the `ai_manifest(4)` man page and the *install_service_image_path*`/auto_install/manifest/ai_manifest.xml` sample AI manifest for information about the XML elements in an AI manifest.

This chapter explains how you can create custom AI manifests for particular clients.

- Create a custom XML AI manifest file. This method is best suited to an environment where few systems require custom provisioning. Most systems to be installed have identical or similar hardware and will be provisioned identically.
- Write a script that dynamically creates an AI manifest for each client at installation time. Use this method to create a custom installation for each client, based on client characteristics discovered at installation time.

Any particular install service can include both XML manifest files and scripts for generating manifest files. Any particular client uses only one AI manifest, either static or generated by a script. Which manifest or script a particular client uses depends on the criteria specified when the manifest or script is added to the install service. If the client does not match any criteria to use a custom manifest or script, the default manifest is used. Any manifest or script in a service can be designated to be the default for that service.

Customizing an XML AI Manifest File

To create and apply a custom XML AI manifest file, follow these steps:

1. Copy an existing AI manifest. When you create an AI install service, you get a default AI manifest. See Chapter 8, "Setting Up an Install Server," for information about creating an install service.

   Use the `list` subcommand to see what AI manifests you already have associated with a particular install service.

   ```
   $ installadm list -m -n s11-x86

   Manifest          Status   Criteria
   -----------       -------  --------
   orig_default      Default  None
   ```

   Use the `installadm export` command to copy this default manifest or any other AI manifest that has been added to this service.

   ```
   # installadm export -n s11-x86 -m orig_default -o mem1.xml
   ```

   A copy of `orig_default` is now in the file `mem1.xml`.

2. Modify `mem1.xml`, adding tags and values according to the information in the `ai_manifest(4)` man page.

3. Add the new AI manifest to the appropriate AI install service, specifying criteria that define which clients should use these installation instructions.

   ```
   # installadm create-manifest -n s11-x86 -f ./mem1.xml -m mem1 \
   -c mem="2048-unbounded"
   ```

   You can specify multiple `-c` options or one `-C` file. See Chapter 9, "Customizing Installations," and the `set-criteria` subcommand for information about specifying client criteria.

   ```
   $ installadm list -n s11-x86 -m

   Manifest          Status   Criteria
   -----------       -------  --------
   orig_default      Default  None
   mem1                       mem  = 2048 MB - unbounded
   ```

You can designate any manifest file or derived manifests script to be the default manifest or script for a service. To change the default among manifests and scripts that you have already added to the service, use the `-o` option with the `set-service` subcommand.

```
# installadm set-service -o default-manifest=mem1 s11-x86
# installadm list -n s11-x86 -m

Manifest          Status    Criteria
-----------       -------   --------
orig_default      Inactive  None
mem1              Default   (Ignored: mem  = 2048 MB - unbounded)
```

In this example, the original default is now inactive because it has no criteria to specify which clients should use it. Only the default manifest or script can have no client selection criteria and still be active.

If you want to add a new default manifest or script for this service, use the `-d` option with `create-manifest` and do not specify any client criteria.

```
# installadm create-manifest -n s11-x86 -d \
-f ./region1.xml -m region1
# installadm list -n s11-x86 -m

Manifest          Status    Criteria
-----------       -------   --------
orig_default      Inactive  None
mem1                        mem  = 2048 MB - unbounded
region1           Default   None
```

If you want to change the content of a manifest or script that has already been added to an install service, use the `installadm update-manifest` command. Criteria, default status, and *manifest_or_script_name* are not changed as a result of the update.

```
# installadm update-manifest -n s11-x86
    -f ./newregion1.xml -m region1
```

The `create-manifest` and `update-manifest` subcommands validate XML manifest files before adding them to the install service. AI syntactically validates the AI manifests at client installation time.

Note – If an invalid manifest is provided to a client, the automated installation aborts. To investigate the cause of the validation failure, see the `/system/volatile/install_log` on the client.

See "Maintain an Install Server" on page 92 for more information about the `list`, `export`, `create-manifest`, `set-criteria`, `update-manifest`, and `set-service` subcommands.

Creating an AI Manifest at Client Installation Time

An alternative to creating custom AI manifests prior to client installation is to write a script that dynamically creates an AI manifest for each client at client installation time. The script can query environment variables and other client configuration information to create a custom AI manifest for each client. Because the manifest is based on attributes of each client discovered at installation time, the manifest is called a *derived manifest*.

A derived manifest is especially useful if you have a large number of systems that can be installed almost identically so that the AI manifests for these systems have relatively small differences. Create an AI manifest that specifies the installation parameters that are common to

this group of systems. Using this common manifest as a base, create a derived manifests script that adds the parameters that are different for each client to the common manifest when each client is installed. For example, a derived manifests script can detect the number and size of disks attached to each client system and modify the AI manifest at client installation time to specify a custom disk layout for each client.

To create and apply a derived manifests script, follow these steps:

1. Identify an existing AI manifest to use as a base manifest to modify.

   To develop and test your script, you can work with a local copy. At installation time, the base manifest must be accessible by each client that uses this derived manifests script.
2. Write a script to dynamically modify the base manifest at installation time based on attributes of the client being installed.
3. Add the derived manifests script to the appropriate AI install service, specifying criteria that define which clients should use this script to create their installation instructions at installation time.

AI executes the script at client installation time to produce an instance of an AI manifest. AI syntactically validates the resultant manifest.

Note – If a manifest is not created or the derived manifest does not validate, the client installation aborts. To investigate the cause of the validation failure, see the `/system/volatile/install_log` on the client.

If the client installation is successful, the derived manifest is copied to `/var/sadm/system/logs/derived/manifest.xml`on the client, and the script used to derive the manifest is copied to `/var/sadm/system/logs/derived/manifest_script`.

Create a Derived Manifests Script

In general, a derived manifests script retrieves information from the client and uses that information to modify a base AI manifest to create a custom AI manifest just for this client. A derived manifests script can also combine multiple partial AI manifests. The final derived manifest must be complete and must pass validation.

A derived manifests script can be any kind of script that is supported in the image. For example, `ksh93` and `python` are in the image by default. If you want to use another kind of script, make sure the required support is in the image.

Retrieve Client Attributes

The derived manifests script can run commands to read system attributes. AI runs the script as role `aiuser`. The `aiuser` role has all the privileges of a non-privileged user plus the following additional privileges:

```
solaris.network.autoconf.read
solaris.smf.read*
```

The `aiuser` role is non-privileged except that it can read more information from the system than other non-privileged users. The `aiuser` role cannot change the system.

For information about roles, profiles, and privileges, see Part II, "Roles, Rights Profiles, and Privileges," in *Oracle Solaris Administration: Security Services*.

In addition to using commands to read system attributes, attributes of the client are available through the environment variables shown in the following table.

TABLE 10–1 Client Attribute Environment Variables

Environment Variable Name	Description
`SI_ARCH`	The architecture of the client to be installed. Equivalent to the output of `uname -p`.
`SI_CPU`	The ISA or processor type of the client to be installed. Equivalent of the output of `uname -p`.
`SI_NUMDISKS`	The number of disks on the client.
`SI_DISKNAME_#`	A flat set of variables representing the `ctds` names of the disks found on the client. There will exist `SI_NUMDISKS` number of `SI_DISKNAME_#` variables, where the # is replaced by an integer starting at 1, up to `SI_NUMDISKS`. This set of variables correlates with the set of variables described by `SI_DISKSIZE_#`.
`SI_DISKSIZE_#`	A flat set of variables representing the disk sizes of the disks found on the client. There will exist `SI_NUMDISKS` number of `SI_DISKSIZE_#`variables, where the # is replaced by an integer starting at 1, up to `SI_NUMDISKS`. This set of variables correlates with the set of variables described by `SI_DISKNAME_#`. The sizes are integer numbers of megabytes.
`SI_HOSTADDRESS`	The IP address of the client as set in the install environment.
`SI_HOSTNAME`	The host name of the client as set in the install environment.
`SI_KARCH`	The kernel architecture of the client. Equivalent to the output of `uname -m`.
`SI_INSTALL_SERVICE`	The name of the install service used to obtain the manifest script. This environment variable has a value only for network boots, not for media boots.
`SI_MANIFEST_SCRIPT`	The URL of the manifest script.
`SI_MEMSIZE`	The amount of physical memory on the client. The size is an integer number of megabytes.
`SI_MODEL`	The model name of the client. Equivalent to the output of `uname -i`.
`SI_NATISA`	The native instruction set architecture of the client. Equivalent to the output of `isainfo -n`.
`SI_NETWORK`	The network number of the client. The network number is (`IP_ADDR` & *netmask*).

TABLE 10–1 Client Attribute Environment Variables *(Continued)*

Environment Variable Name	Description
SI_PLATFORM	The platform of the client. Equivalent to the output of uname -i.

Customize the AI Manifest

To add or modify XML elements in an AI manifest, use the /usr/bin/aimanifest command.

The minimum file that aimanifest can modify must contain both of the following pieces:

- A !DOCTYPE reference to a DTD that is valid for the XML manifest being developed.
- The root element for this DTD.

The following example shows the minimum base manifest file for an AI manifest, including specifying the AI DTD file for the install service where this derived manifests script will be added:

```
<!DOCTYPE auto_install SYSTEM "file:///image_path/auto_install/ai.dtd.#">
<auto_install/>
```

The # is an integer such as 1. The *image_path* is the path returned by the following command, where *service_name* is the name of the install service where this derived manifests script will be added:

```
$ installadm list -n service_name
```

Use the load subcommand of the aimanifest command to load a base manifest before any other aimanifest call in the derived manifests script. Any files that you load must be accessible by the client at client installation time. For example, you could load a manifest from *image_path*/auto_install/manifest/ in the target install service.

The examples in this chapter load the file /usr/share/auto_install/manifest/default.xml. The sample manifests in /usr/share/auto_install/manifest/ could be different from the manifests in the target install service. In production work, you should not load manifests from /usr/share/auto_install/manifest/.

The load subcommand can also be used to load or insert partial manifests.

Use the add subcommand to add new elements. Use the set subcommand to add element attributes or change element or attribute values. See the aimanifest(1M) man page for details. See the aimanifest(1M) man page and the example scripts below for examples of using the aimanifest command.

If a value specified in an aimanifest command contains a special character such as a forward slash (/) or single or double quotation marks, then that value must be enclosed in single or double quotation marks. The quotation marks might need to be escaped with a preceding backslash character (\) according to the rules of the shell used, so that the shell does not remove or interpret the quotation marks.

The following example returns the action of the `software_data` element that contains the package name `pkg:/entire`. In this example, quotation marks are needed around `pkg:/entire` because the forward slash character is a special character. The backslash characters are needed to escape the quotation marks if this command is invoked in a shell script such as a `ksh93` script.

```
/usr/bin/aimanifest get software_data[name=\"pkg:/entire\"]@action
```

Tip – As a best practice, set up a trap to stop on error.

The following partial script is a good model for a derived manifests script.

```
#!/bin/ksh93

SCRIPT_SUCCESS=0
SCRIPT_FAILURE=1

function handler
{
    exit $SCRIPT_FAILURE
}

trap handler ERR

/usr/bin/aimanifest load baseAImanifest.xml

# Customize AI manifest. For example:
/usr/bin/aimanifest load -i manifest_fragment.xml
/usr/bin/aimanifest set origin@name file:///net/myserver/myrepo/repo.redist

exit $SCRIPT_SUCCESS
```

Example Derived Manifests Scripts

This section shows how to write derived manifests scripts to determine client attributes and use that information to customize the AI manifest. These examples do not necessarily include all the information required to produce a valid AI manifest.

To try these examples, perform the following setup steps:

1. Set the `AIM_MANIFEST` environment variable to a location where the script will develop the AI manifest.

   The `$AIM_MANIFEST` file is rewritten for each `aimanifest` command that modifies the file. Each invocation of `aimanifest` with the `load`, `add`, or `set` subcommand opens, modifies, and saves the `AIM_MANIFEST` file. If `AIM_MANIFEST` is not set, `aimanifest` commands fail.

2. Set the `AIM_LOGFILE` environment variable to a location where the script can write verbose information and error messages.

   The `aimanifest` command logs the name of the subcommand, argument values, and return status of each `aimanifest` call to the screen and to the `$AIM_MANIFEST_LOG` file if set.

3. Make sure the `aimanifest` command is available on the system where you run the script. If the `aimanifest` command is not available, install the `auto-install-common` package.
4. Set environment variables. These examples demonstrate using environment variables to retrieve information about the client. To try these examples, you must set values for these environment variables.

   When you install a system using AI, the environment variables shown in Table 10–1 have values and are available for a derived manifests script to use.

EXAMPLE 10–1 Specify Disk Partitioning Based on Disk Size

This example customizes the AI manifest to use only half of the target disk for a Solaris `fdisk` partition if the size of the disk is greater than 1 TB. Try setting `SI_DISKSIZE_1` to less than 1 TB and then greater than 1 TB for different runs of this script.

```
#!/bin/ksh93

SCRIPT_SUCCESS=0
SCRIPT_FAILURE=1

function handler
{
    exit $SCRIPT_FAILURE
}

trap handler ERR

/usr/bin/aimanifest load /usr/share/auto_install/manifest/default.xml

# Assume there is only one disk on the system for this example.
if [[ $SI_DISKSIZE_1 -gt "1048576" ]] ; then
    typeset -i PARTN_SIZE=$SI_DISKSIZE_1/2

    # Default action is to create.
    /usr/bin/aimanifest add \
        /auto_install/ai_instance/target/disk/partition@name 1
    /usr/bin/aimanifest add \
        /auto_install/ai_instance/target/disk/partition[@name=1]/size@val \
        ${PARTN_SIZE}mb
else
    /usr/bin/aimanifest add \
        /auto_install/ai_instance/target/disk/partition@action \
        use_existing_solaris2
fi
exit $SCRIPT_SUCCESS
```

For some clients, the following elements are added to `$AIM_MANIFEST`:

```
<target>
  <disk>
    <partition action="use_existing_solaris2"/>
  </disk>
</target>
```

EXAMPLE 10–1 Specify Disk Partitioning Based on Disk Size *(Continued)*

For other clients, elements similar to the following are added to `$AIM_MANIFEST`, depending on the value of `SI_DISKSIZE_1`:

```
<target>
  <disk>
    <partition name="1">
      <size val="524288mb"/>
    </partition>
  </disk>
</target>
```

EXAMPLE 10–2 Specify the Root Pool Layout Based on the Existence of Additional Disks

This example customizes the AI manifest to configure a mirror of the root pool if a second disk exists, and configure a three-way mirror if a third disk exists. Set `SI_NUMDISKS` and `SI_DISKNAME_1` before you run the script. Set `SI_DISKNAME_2`, `SI_DISKNAME_3`, and any others as necessary, depending on the value you set for `SI_NUMDISKS`. These environment variables will be set and available to derived manifests scripts during AI installations.

This example demonstrates using the `aimanifest` return path (`-r` option). See the `aimanifest(1M)` man page for more information about the return path.

```
#!/bin/ksh93

SCRIPT_SUCCESS=0
SCRIPT_FAILURE=1

function handler
{
    exit $SCRIPT_FAILURE
}

trap handler ERR

/usr/bin/aimanifest load /usr/share/auto_install/manifest/default.xml

# Use the default if there is only one disk.
if [[ $SI_NUMDISKS -ge 2 ]] ; then
    typeset -i disk_num

    # Turn on mirroring. Assumes a root zpool is already set up.
    vdev=$(/usr/bin/aimanifest add -r \
        target/logical/zpool[@name=rpool]/vdev@name mirror_vdev)
    /usr/bin/aimanifest set ${vdev}@redundancy mirror

    for ((disk_num = 1; disk_num <= $SI_NUMDISKS; disk_num++)) ; do
        eval curr_disk="$"SI_DISKNAME_${disk_num}
        disk=$(/usr/bin/aimanifest add -r target/disk@in_vdev mirror_vdev)
        /usr/bin/aimanifest set ${disk}@in_zpool rpool
        /usr/bin/aimanifest set ${disk}@whole_disk true
        disk_name=$(/usr/bin/aimanifest add -r \
            ${disk}/disk_name@name $curr_disk)
```

EXAMPLE 10–2 Specify the Root Pool Layout Based on the Existence of Additional Disks *(Continued)*

```
        /usr/bin/aimanifest set ${disk_name}@name_type ctd
    done
fi
exit $SCRIPT_SUCCESS
```

For a system with two disks named `c0t0d0` and `c0t1d0`, the output of this example is the following XML:

```
<target>
  <disk in_vdev="mirror_vdev" in_zpool="rpool" whole_disk="true">
    <disk_name name="c0t0d0" name_type="ctd"/>
  </disk>
  <disk in_vdev="mirror_vdev" in_zpool="rpool" whole_disk="true">
    <disk_name name="c0t1d0" name_type="ctd"/>
  </disk>
  <logical>
    <zpool name="rpool">
      <vdev name="mirror_vdev" redundancy="mirror"/>
    </zpool>
  </logical>
</target>
```

EXAMPLE 10–3 Specify a Mirrored Configuration If at Least Two Disks of a Specified Size Are Present

This example customizes the AI manifest to specify a mirrored configuration if the system has at least two 200 GB disks. Use the first two disks found that are at least two 200 GB. Set `SI_NUMDISKS`, `SI_DISKNAME_1`, and `SI_DISKSIZE_1` in your test environment before you run the script. Also set `SI_DISKNAME_2`, `SI_DISKSIZE_2`, and any others as necessary, depending on the value you set for `SI_NUMDISKS`. These environment variables will be set and available to derived manifests scripts during AI installations.

This example shows how to modify a node when more than one node with the same path is present. The shell implementation uses the return path (`-r`) option of `aimanifest` to return the path to a specific node, and uses that path to make additional modifications to the same node. The Python implementation demonstrates the use of subpathing (using `[]` inside a node path) to make additional modifications to the same node.

```
#!/bin/ksh93

SCRIPT_SUCCESS=0
SCRIPT_FAILURE=1

function handler
{
    exit $SCRIPT_FAILURE
}

trap handler ERR

# Find the disks first.
typeset found_1
```

EXAMPLE 10–3 Specify a Mirrored Configuration If at Least Two Disks of a Specified Size Are Present *(Continued)*

```
typeset found_2
typeset -i disk_num

for ((disk_num = 1; disk_num <= $SI_NUMDISKS; disk_num++)) ; do
    eval curr_disk="$"SI_DISKNAME_${disk_num}
    eval curr_disk_size="$"SI_DISKSIZE_${disk_num}
    if [[ $curr_disk_size -ge "204800" ]] ; then
        if [ -z $found_1 ] ; then
            found_1=$curr_disk
        else
            found_2=$curr_disk
            break
        fi
    fi
done

# Now, install them into the manifest.
# Let the installer take the default action if two large disks are not found.

/usr/bin/aimanifest load /usr/share/auto_install/manifest/default.xml

if [[ -n $found_2 ]] ; then
    # Turn on mirroring.
    vdev=$(/usr/bin/aimanifest add -r \
        /auto_install/ai_instance/target/logical/zpool/vdev@redundancy mirror)
    /usr/bin/aimanifest set ${vdev}@name mirror_vdev
    disk=$(/usr/bin/aimanifest add -r \
        /auto_install/ai_instance/target/disk@in_vdev mirror_vdev)
    disk_name=$(/usr/bin/aimanifest add -r ${disk}/disk_name@name $found_1)
    /usr/bin/aimanifest set ${disk_name}@name_type ctd

    disk=$(/usr/bin/aimanifest add -r \
        /auto_install/ai_instance/target/disk@in_vdev mirror_vdev)
    disk_name=$(/usr/bin/aimanifest add -r ${disk}/disk_name@name $found_2)
    /usr/bin/aimanifest set ${disk_name}@name_type ctd
fi

exit $SCRIPT_SUCCESS
```

The following script is a Python version of the preceding Kornshell version.

```
import os
from subprocess import call, check_call, CalledProcessError

SCRIPT_SUCCESS = 0
SCRIPT_FAILURE = 1

def main():

    # Find the disks first.
    found_1 = ""
    found_2 = ""

    for disk_num in range(1, SI_NUMDISKS + 1):
```

EXAMPLE 10–3 Specify a Mirrored Configuration If at Least Two Disks of a Specified Size Are Present *(Continued)*

```
        curr_disk_var = "SI_DISKNAME_" + str(disk_num)
        curr_disk = os.environ[curr_disk_var]
        curr_disk_size_var = "SI_DISKSIZE_" + str(disk_num)
        curr_disk_size = os.environ[curr_disk_size_var]
        if curr_disk_size >= "204800":
            if not len(found_1):
                found_1 = curr_disk
            else:
                found_2 = curr_disk
                break

    # Now, write the disk specifications into the manifest.
    # Let the installer take the default action if two large disks are not found.

    try:
        subprocess.check_call(["/usr/bin/aimanifest", "load",
            "/usr/share/auto_install/manifest/default.xml"])
    except CalledProcessError as err:
        sys.exit(err.returncode)

    if len(found_2):
        try:            subprocess.check_call(["/usr/bin/aimanifest", "set",
               "target/logical/zpool/vdev@redundancy", "mirror"])
            subprocess.check_call(["/usr/bin/aimanifest", "set",
               "target/logical/zpool/vdev[@redundancy='mirror']@name", "mirror_vdev"])

            subprocess.check_call(["/usr/bin/aimanifest", "add",
                "target/disk/disk_name@name", "found_1"])
            subprocess.check_call(["/usr/bin/aimanifest", "set",
                "target/disk/disk_name[@name='" + found_1 + "']" + "@name_type", "ctd"])
            subprocess.check_call(["/usr/bin/aimanifest", "set",
                "target/disk[disk_name@name='" + found_1 + "']" + "@in_vdev", "mirror_vdev"])

            subprocess.check_call(["/usr/bin/aimanifest", "add",
                "target/disk/disk_name@name", "found_2"])
            subprocess.check_call(["/usr/bin/aimanifest", "set",
                "target/disk/disk_name[@name='" + found_2 + "']" + "@name_type", "ctd"])
            subprocess.check_call(["/usr/bin/aimanifest", "set",
                "target/disk[disk_name@name='" + found_2 + "']" + "@in_vdev", "mirror_vdev"])
        except CalledProcessError as err:
            sys.exit(err.returncode)

    sys.exit(SCRIPT_SUCCESS)

if __name__ == "__main__":
    main()
```

EXAMPLE 10–4 Specify Packages To Install Based on IP Address

This example customizes the AI manifest to install one package if the IP address of the client is in a specified range, and install a different package if the IP address of the client is in a different range. Set `SI_HOSTADDRESS` in your test environment before you run the script. This environment variable will be set and available to derived manifests scripts during AI installations.

EXAMPLE 10–4 Specify Packages To Install Based on IP Address *(Continued)*

```
#!/bin/ksh93

SCRIPT_SUCCESS=0
SCRIPT_FAILURE=1

function handler
{
    exit $SCRIPT_FAILURE
}

trap handler ERR

/usr/bin/aimanifest load /usr/share/auto_install/manifest/default.xml

# First determine which range the host IP address of the client is in.
echo $SI_HOSTADDRESS | sed 's/\./ /g' | read a b c d

# Assume all systems are on the same class A and B subnets.

# If the system is on class C subnet = 100, then install the /pkg100 package.
# If the system is on class C subnet = 101, then install the /pkg101 package.
# Otherwise, do not install any other additional package.

if ((c == 100)) ; then
    /usr/bin/aimanifest add \
    software/software_data[@action='install']/name pkg:/pkg100
fi
if ((c == 101)) ; then
    /usr/bin/aimanifest add \
    software/software_data[@action='install']/name pkg:/pkg101
fi

exit $SCRIPT_SUCCESS
```

EXAMPLE 10–5 Specify the Target Disk Must Be a Certain Size

This example customizes the AI manifest to only install on a disk that is at least 50 GB. Ignore smaller disks. Set `SI_NUMDISKS`, `SI_DISKNAME_1`, and `SI_DISKSIZE_1` in your test environment before you run the script. Also set `SI_DISKNAME_2`, `SI_DISKSIZE_2`, and any others as necessary, depending on the value you set for `SI_NUMDISKS`. These environment variables will be set and available to derived manifests scripts during AI installations.

```
#!/bin/ksh93

SCRIPT_SUCCESS=0
SCRIPT_FAILURE=1

function handler
{
    exit $SCRIPT_FAILURE
}

trap handler ERR
```

EXAMPLE 10–5 Specify the Target Disk Must Be a Certain Size *(Continued)*

```
/usr/bin/aimanifest load /usr/share/auto_install/manifest/default.xml

typeset found
typeset -i disk_num
for ((disk_num = 1; disk_num <= $SI_NUMDISKS; disk_num++)) ; do
    eval curr_disk="$"SI_DISKNAME_${disk_num}
    eval curr_disk_size="$"SI_DISKSIZE_${disk_num}
    if [[ $curr_disk_size -ge "512000" ]] ; then
        found=$curr_disk
        /usr/bin/aimanifest add \
            /auto_install/ai_instance/target/disk/disk_name@name $found
        break
    fi
done

if [[ -z $found ]] ; then
    exit $SCRIPT_FAILURE
fi

exit $SCRIPT_SUCCESS
```

EXAMPLE 10–6 Script With Incorrect Manifest Specifications

The script in this example contains errors.

```
#!/bin/ksh93

SCRIPT_SUCCESS=0
SCRIPT_FAILURE=1

function handler
{
    exit $SCRIPT_FAILURE
}

trap handler ERR

/usr/bin/aimanifest load /usr/share/auto_install/manifest/default.xml

/usr/bin/aimanifest set \
    software[@type="IPS"]/software_data/name pkg:/driver/pcmcia
/usr/bin/aimanifest set \
    software/software_data[@name=pkg:/driver/pcmcia]@action uninstall

return $SCRIPT_SUCCESS
```

EXAMPLE 10–6 Script With Incorrect Manifest Specifications *(Continued)*

This example has three problems with writing to `$AIM_MANIFEST`.

1. The `set` subcommand of `aimanifest` can change the value of an existing element or attribute or create a new attribute. The `set` subcommand cannot create a new element. The first `set` subcommand attempts to modify an existing package name in the manifest, instead of creating a new package name. If more than one package name already exists in the manifest, an ambiguity error results because which package to modify cannot be determined. The first `set` subcommand in this example should have been an `add` subcommand.
2. In the second `set` subcommand in this example, an element `name` with value `pkg:/driver/pcmcia` is specified with a preceding @ sign. Attribute values are specified with a preceding @ sign. Element values are not.
3. The value `pkg:/driver/pcmcia` should be enclosed in quotation marks. Values with slashes or other special characters must be quoted.

The following lines should replace the two `set` lines in this example:

```
/usr/bin/aimanifest add \
    software[@type="IPS"]/software_data@action uninstall
/usr/bin/aimanifest add \
    software/software_data[@action=uninstall]/name pkg:/driver/pcmcia
```

These two `add` subcommands add the following lines to the end of the `software` section of the manifest that is being written:

```
<software_data action="uninstall">
  <name>pkg:/driver/pcmcia</name>
</software_data>
```

Testing Derived Manifests Scripts

To test your derived manifests script, run the script in an environment similar to the AI installation environment.

1. Set up a base AI manifest for the script to modify.
   a. Make sure the first `aimanifest` command in your script is an `aimanifest load` command. Make sure the file being loaded contains a `<!DOCTYPE>` definition that specifies the appropriate DTD to use for AI manifest validation for the target install service. The following example shows the minimum base manifest file for an AI manifest, including specifying the AI DTD file for the install service where this derived manifests script will be added:

      ```
      <!DOCTYPE auto_install SYSTEM "file:///image_path/auto_install/ai.dtd.#">
      <auto_install/>
      ```

The # is an integer such as 1. The *image_path* is the path returned by the following command, where *service_name* is the name of the install service where this derived manifests script will be added:

```
$ installadm list -n service_name
```

b. Set `AIM_MANIFEST` to a location where the script will develop the AI manifest. This location must be writable by the non-privileged user `aiuser`.

Note – When AI is doing the installation, `AIM_MANIFEST` does not need to be set. AI sets a default value.

---

2. Set `AIM_LOGFILE` to a location where the script can write verbose information and error messages. This location must be writable by the non-privileged user `aiuser`.

Note – When AI is doing the installation, `AIM_LOGFILE` does not need to be set. This log information is part of the larger installation log, `/system/volatile/install_log`.

---

3. Make sure the `aimanifest` command is available on the system where you test the script. If the `aimanifest` command is not available, install the `auto-install-common` package.
4. Make sure you are able to assume the root role. From the root role, you can assume the `aiuser` role without specifying a password.

```
$ su
Password:
# su aiuser -c ./script
#
```

AI executes the derived manifests script as role `aiuser`. To approximate the AI installation environment, assume the `aiuser` role to run the script. If you run the script as a user with different privileges than the `aiuser` role has, some operations in the script might have different results.

5. Set environment variables in the test environment with values that represent the client systems that will be installed using this derived manifests script. The sample file `/usr/share/auto_install/derived_manifest_test_env.sh` can be used as a template. Change the values as applicable.

When AI is doing the installation, the environment variables shown in Table 10–1 have values and are available for a derived manifests script to use.

The intended client system might be very different from the install server or other system where you might test the derived manifests script. Commands that you call in the script might be unavailable or might be a different version with different behavior. The systems might be different architectures or have different number and sizes of disks. Setting environment variables in the test environment as described above addresses some of these differences.

Use the following method to test the derived manifests script on one of the intended client systems:

1. Boot an AI image on that client system in "Text Installer and command line" mode.
2. Select "Shell" from the installer initial menu.
3. Use `wget` or `sftp` to copy your script from the AI install server.
4. Use one of the following methods to debug the script:
   - Run the script manually.
   - Use the following command to run AI in a test mode:

     ```
     $ auto-install -m script -i
     ```

     Inspect the AI log file `/system/volatile/install_log`. The log file should contain the following line to indicate the script validates:

     ```
     Derived Manifest Module: XML validation completed successfully
     ```
5. Copy the script back to the install server.

Add a Derived Manifests Script To an Install Service

Add a script to an AI install service the same way you add an XML manifest to the install service. Use the same options to specify criteria to select which clients will use this script to create a manifest for their installation. You can update a script just as you can update an XML manifest. A script can be set to be the default manifest for the service. Scripts are shown when you list manifests associated with a service. The contents of a script can be exported just as an XML manifest can be exported.

When you add an XML manifest to an install service, the manifest is validated. When you add a script to an install service, the script is not validated.

Validate a derived AI manifest before adding the script to an install service.

1. Run the script in an environment similar to the intended client system.
2. Use the `validate` subcommand on the resulting manifest.

   ```
   $ /usr/bin/aimanifest validate
   ```

   Messages are displayed only if the validation fails.

Add the script to the appropriate AI install service, specifying criteria that define which clients should use these installation instructions.

```
# installadm create-manifest -n s11-x86 -f ./mac1.ksh -m mac1 \
-c mac=BB:AA:AA:AA:AA:AA
```

You can specify multiple `-c` options or one `-C` file. See also the `set-criteria` subcommand. See Chapter 9, "Customizing Installations," for information about specifying client criteria.

See "Maintain an Install Server" on page 92 for information about the `list`, `export`, `create-manifest`, `set-criteria`, `update-manifest`, and `set-service` subcommands.

◆ ◆ ◆ CHAPTER 11

Configuring the Client System

This chapter describes how to specify information needed to configure the client system after installation. You can specify configuration of anything that is configurable via `smf`(5) properties.

Providing Configuration Profiles

System configuration profiles specify client system configuration as a set of configuration parameters in the form of a Service Management Facility (SMF) profile. The system configuration profile sets SMF properties for appropriate SMF services.

System configuration profiles are applied during the first boot of the system after AI installation. SMF services responsible for particular configuration areas process SMF properties and configure the system accordingly.

Each client can use any number of system configuration profiles. For example, a client might be assigned one profile that provides just the host name and IP address for that client. The same client and many other clients might be assigned other profiles that set more broadly applicable property values.

If no system configuration profile is provided for a particular client, the interactive configuration tool opens on that client. See "Configuring a System" on page 63 for information about how to use the configuration tool interactively.

Creating System Configuration Profiles

Use one of the following methods to create a system configuration profile:

- Run the interactive configuration tool and save the output to a file. The following command creates a valid profile in `sc.xml` from responses you enter interactively.

  ```
  # sysconfig create-profile -o sc.xml
  ```

See "Creating a Configuration Profile Using the SCI Tool" on page 67 for information about using the configuration tool to produce a profile file.

- Create the system configuration profile manually, using the property specifications shown in "Specifying Configuration in a System Configuration Profile" on page 137 and "Example System Configuration Profiles" on page 148.

  Include the following lines in every system configuration profile:

  ```
  <!DOCTYPE service_bundle SYSTEM "/usr/share/lib/xml/dtd/service_bundle.dtd.1">
  <service_bundle type="profile" name="sysconfig">
    <!-- service, property_group, property, and propval specifications -->
  </service_bundle>
  ```

  If you specify a service or property that does not apply, that specification is ignored.

  Do not specify any particular property more than one time.

A system configuration profile can express property and attribute values in two ways. One profile can use both methods.

- Values can be entered explicitly before the profile is added to the install service, using the property specifications shown in this chapter.
- A system configuration profile can include variables that are replaced with valid values when the profile is added to the install service. See "Using System Configuration Profile Templates" on page 146.

Validating System Configuration Profiles

Use the `installadm validate` command to validate system configuration profiles under development for syntactic correctness. The install service you plan to add this profile to must already exist. See "Validate a System Configuration Profile" on page 100 for more information about the `validate` subcommand.

To validate a profile template, as described in "Using System Configuration Profile Templates" on page 146, provide the variable values in environment variables.

```
# export AI_HOSTNAME=server1;export AI_IPV4=10.0.0.1;export AI_NETWORK=10.0.0.0
# installadm validate -n svc1 -P /export/hostIPnet.xml
```

Adding System Configuration Profiles To an Install Service

Use the `installadm create-profile` command to add a system configuration profile to an install service. The `create-profile` subcommand validates profiles before adding them to the install service.

Specify criteria so that appropriate clients select that configuration profile. If no criteria are specified, all clients use this profile.

A single client can match and use more than one system configuration profile. Make sure that no client uses a set of profiles such that a particular property is specified more than one time. If a client receives more than one specification for any particular property, even if the value of the property is the same in each specification, the behavior of the SMF service being configured is undefined.

If a client does not match any criteria specified for any system configuration profile in the install service, the interactive configuration tool opens on that client.

Use the `installadm list` command to list profiles that have been added to a given install service and list the criteria that are specified for each profile.

You can use the `installadm set-criteria` command to change or add to the client selection criteria specified for a profile.

Use the `installadm export` command to retrieve a copy of the contents of a profile that has been added to an install service. You could modify that copy to create another profile.

See "Maintain an Install Server" on page 92 and the `installadm(1M)` man page for more information about the `create-profile`, `list`, `set-criteria`, and `export` subcommands.

Specifying Configuration in a System Configuration Profile

You can specify configuration of anything that is configurable via `smf(5)` properties. For example, the system configuration profile can configure a root account, an initial user, keyboard layout, terminal type, an IPv4 network interface (static or DHCP) and default route, an IPv6 network interface (static or `addrconf`) and default route, and DNS (name server list, search list, domain). If you specify a service or property that does not apply, that specification is ignored. Do not specify any particular property more than one time.

If you are not sure what SMF properties you need to specify, you can use the `describe` subcommand of the `svccfg` command to display a description of the property groups and properties of a service, including possible settings. See "Property Inspection and Modification Subcommands" on the `svccfg(1M)` man page.

```
svccfg -s FMRI describe [-v] [-t] [propertygroup/property]
```

A property group or specific property can be queried by specifying either the property group name, or the property group name and property name, separated by a slash (/), as an argument.

The `-v` option gives all information available, including descriptions for current settings, constraints, and other possible setting choices.

The `-t` option shows only the template data for the selection (see `smf_template(5)`), and does not display the current settings for property groups and properties.

```
$ svccfg -s name-service/switch describe config
config                      application
    Name service switch configuration data as described in nsswitch.conf(4).
config/value_authorization  astring             solaris.smf.value.name-service.switch
config/default              astring             files
    Default configuration database entry.
config/host                 astring             "files dns mdns"
    Override configuration for host database lookups. (both IPv4 and IPv6 hosts)
config/printer              astring             "user files"
    Override configuration for printer database lookups.
$ svccfg -s name-service/switch describe -v config
config                      application
    name: config
    type: application
    required: true
    target: this
    description: Name service switch configuration data as described in nsswitch.conf(4).
config/value_authorization  astring             solaris.smf.value.name-service.switch
config/default              astring             files
    type: astring
    required: true
    Default configuration database entry.
    visibility: readwrite
    minimum number of values: 1
    maximum number of values: 1
  value: files
...
$ svccfg -s name-service/switch describe -t config
name: config
type: application
    Name service switch configuration data as described in nsswitch.conf(4).
  name: default
  type: astring
    Default configuration database entry.
  name: host
  type: astring
    Override configuration for host database lookups. (both IPv4 and IPv6 hosts)
  name: password
  type: astring
    Override configuration for passwd database lookups. Also used with the shadow and user_attr databases.
  name: group
  type: astring
    Override configuration for group database lookups.
  name: network
  type: astring
    Override configuration for network database lookups.
...
$ svccfg -s system/config-user describe root_account
root_account                     application
root_account/expire              astring
root_account/password            astring
root_account/read_authorization  astring           solaris.smf.read.system-config
root_account/stability           astring           Evolving
root_account/type                astring
```

Root and User Accounts

Use the `sysconfig create-profile` command with the `users` grouping to generate a valid profile that configures the root user and initial user.

```
# sysconfig create-profile -g users -o sc_users.xml
```

The `svc:/system/config-user` SMF service configures user and root accounts. This service recognizes two property groups:

- The `root_account` property group includes SMF properties that configure the root account.
- The `user_account` property group includes SMF properties that configure user accounts.

Tip – One method of generating encrypted passwords for the Oracle Solaris OS is to create a user of the intended name and password, copy the password from the `/etc/shadow` file between the first and second colons of the user's record, and add that information into the `password` values in the manifest.

Configuring the Root Account

The `root_account` property group contains the following properties.

TABLE 11–1 root_account Property Group Properties

Property	Type	Required	Description
`password`	`astring`	yes	Encrypted root password. If you do not provide a root password, the root password is empty.
`type`	`astring`	no	Account type: `normal` or `role`. The default is `normal`.
`expire`	`string`	no	Expiration date for login. If set to 0 (zero), the user will be forced to change the root password at the next login.

EXAMPLE 11–1 Configuring the Root Account Only With Password Expired

```
<service name="system/config-user" version="1" type="service">
    <instance name="default" enabled="true">
        <property_group name="root_account" type="application">
            <propval name="password" value="encrypted_password"/>
            <propval name="type" value="normal"/>
            <propval name="expire" value="0"/>
        </property_group>
    </instance>
</service>
```

Configuring a User Account

The `user_account` property group contains the following properties.

TABLE 11–2 user_account Property Group Properties

Property	Type	Required	Description
`login`	`astring`	`yes`	User's login.
`password`	`astring`	`yes`	Encrypted user password.
`description`	`astring`	`no`	Usually the user's full name.
`shell`	`astring`	`no`	Full path name of the program used as the user's shell on login.
`uid`	`count`	`no`	UID of the new user. The default UID is 101.
`gid`	`count`	`no`	User's primary group membership. The default GID is 10.
`type`	`astring`	`no`	Account type: `normal` or `role`. The default is `normal`.
`profiles`	`astring`	`no`	One or more comma-separated execution profiles defined in `prof_attr`(4).
`roles`	`astring`	`no`	One or more comma-separated roles defined in `user_attr`(4).
`sudoers`	`astring`	`no`	Entry added to the `sudoers`(4) file along with the `login`.
`expire`	`astring`	`no`	Expiration date for the login. If set to 0 (zero), the user will be forced to change the password at the next login.
`home_zfs_dataset`	`astring`	`no`	User's home directory ZFS dataset. The default is *root_pool*`/export/home/`*login*.
`home_mountpoint`	`astring`	`no`	User's home directory mount point. The default is `/export/home/`*login*

System Identity

Use the `sysconfig create-profile` command with the `identity` grouping to generate a valid profile that configures the system node name.

```
# sysconfig create-profile -g identity -o sc_identity.xml
```

The `svc:/system/identity:node` SMF service sets the system host name. The `node` is the instance of `svc:/system/identity`.

The `identity` property group contains the following properties.

TABLE 11–3 identity Property Group Properties

Property	Type	Required	Description
nodename	astring	no	System host name. The default is unknown.

EXAMPLE 11–2 Configuring the Host Name

This example sets the system host name to solaris.

```
<service name="system/identity" version="1" type="service">
    <instance name="node" enabled="true">
        <property_group name="config" type="application">
            <propval name="nodename" value="solaris"/>
        </property_group>
    </instance>
</service>
```

Time Zone and Locale

Use the sysconfig create-profile command with the location grouping to generate a valid profile that configures the time zone and locale.

```
# sysconfig create-profile -g location -o sc_location.xml
```

The svc:/system/timezone SMF service sets the time zone for the system.

The timezone property group contains the following properties.

TABLE 11–4 timezone Property Group Properties

Property	Type	Required	Description
localtime	astring	no	System time zone. The default is UTC.

EXAMPLE 11–3 Configuring the Time Zone

This example sets the time zone to Central European Time/Prague, CZ.

```
<service name='system/timezone' version='1'>
  <instance name='default' enabled='true'>
    <property_group name='timezone'>
      <propval name='localtime' value='Europe/Prague'/>
    </property_group>
  </instance>
</service>
```

The svc:/system/environment:init SMF service sets the locale for the system.

The environment property group can define following environment variables. See the environ(5) man page for information about environment variables.

TABLE 11–5 environment Property Group Properties

Environment Variable	Type	Required	Default Value
LC_CTYPE	astring	no	C
LC_NUMERIC	astring	no	C
LC_TIME	astring	no	C
LC_COLLATE	astring	no	C
LC_MONETARY	astring	no	C
LC_MESSAGES	astring	no	C
LC_ALL	astring	no	C
LANG	astring	no	C

EXAMPLE 11–4 Configuring the Locale

This example sets the locale to Czech language (cs) and Czech Republic (CZ).

```
<service name='system/environment' version='1'>
  <instance name='init' enabled='true'>
    <property_group name='environment'>
      <propval name='LC_ALL' value='cs_CZ.UTF-8'/>
    </property_group>
  </instance>
</service>
```

Terminal Type and Keyboard Layout

EXAMPLE 11–5 Configuring Terminal Type

The svc:/system/console-login SMF service configures terminal type. See the ttymon(1M) man page for definition of related SMF properties.

This example sets the terminal type to vt100.

```
<service name="system/console-login" version="1" type="service">
    <instance name="default" enabled="true">
        <property_group name="ttymon" type="application">
            <propval name="terminal_type" value="vt100"/>
        </property_group>
    </instance>
</service>
```

EXAMPLE 11–6 Configuring Keyboard Layout

Use the `sysconfig create-profile` command with the `kdb_layout` grouping to generate a valid profile that configures the keyboard layout.

```
# sysconfig create-profile -g kdb_layout -o sc_kdb.xml
```

The `svc:/system/keymap` SMF service configures keyboard layout. See the `kbd`(1) man page for definition of related SMF properties.

This example sets the keyboard layout to Czech.

```
<service name='system/keymap' version='1' type='service'>
    <instance name='default' enabled='true'>
        <property_group name='keymap' type='system'>
            <propval name='layout' value='Czech'/>
        </property_group>
    </instance>
</service>
```

Static Network Configuration

Use the `sysconfig create-profile` command with the `network` grouping to generate a valid profile that configures the network.

```
# sysconfig create-profile -g network -o sc_network.xml
```

The `svc:/network/install` SMF service configures an initial physical network interface. This service is initially disabled with property values that do not result in any system configuration.

The `svc:/network/install` service supports configuring one IPv4 interface and one IPv6 interface and, optionally, a default route reachable by these interfaces. The service defines two property groups: one property group for an IPv4 interface and one for an IPv6 interface. The service uses its properties and `ipadm`(1M) to configure the network interfaces. Similarly, the service uses its properties and `route`(1M) to define a default route.

See the examples in "Specifying Static Network Configuration" on page 149.

The `install_ipv4_interface` property group contains the following properties.

TABLE 11–6 install_ipv4_interface Property Group Properties

Property	Type	Required	Description
`name`	`astring`	`yes`	Name of the network interface.
`address_type`	`astring`	`yes`	Value used to construct the `-T` option for the `ipadm`(1M) `create-addr` subcommand. Valid values are `static` or `dhcp`.

TABLE 11–6 install_ipv4_interface Property Group Properties *(Continued)*

Property	Type	Required	Description
`static_address`	`net_address_v4`	`no`	Only required with an `address_type` of `static`. Used to construct the local address for the `ipadm`(1M) `create-addr` subcommand.
`dhcp_wait`	`astring`	`no`	Only applies with an `address_type` of `dhcp`. If defined, this property is used to construct the `-w` *seconds* (or `forever`) portion of the `ipadm`(1M) `create-addr` subcommand.
`default_route`	`net_address_v4`	`no`	Used to define a default route using `route`(1M). **`# /usr/sbin/route \`** **`-p add default`** *default-route* **`\`** **`-ifp`** *ifname* The value of *ifname* is the interface name portion of the `name` property.

The `install_ipv6_interface` property group contains the following properties.

TABLE 11–7 install_ipv6_interface Property Group Properties

Property	Type	Required	Description
`name`	`astring`	`yes`	Name of the network interface.
`address_type`	`astring`	`yes`	Value used to construct the `-T` option for the `ipadm`(1M) `create-addr` subcommand. Valid values are `static` or `addrconf`.
`static_address`	`net_address_v6`	`no`	Only required with an `address_type` of `static`. Used to construct the local address for the `ipadm`(1M) `create-addr` subcommand.
`interface_id`	`net_address_v6`	`no`	Only applies with an `address_type` of `addrconf`. Used to construct the `-i` *interface_id* portion of the `ipadm`(1M) `create-addr` subcommand.
`stateless`	`astring`	`no`	Only applies with an `address_type` of `addrconf`. Used to construct the `-p stateless=yes|no` portion of the `ipadm`(1M) `create-addr` subcommand.
`stateful`	`astring`	`no`	Only applies with an `address_type` of `addrconf`. Used to construct the `-p stateful=yes|no` portion of the `ipadm`(1M) `create-addr` subcommand.

TABLE 11–7 install_ipv6_interface Property Group Properties *(Continued)*

Property	Type	Required	Description
`default_route`	`net_address_v6`	`no`	Used to define a default route using `route`(1M). **`# /usr/sbin/route \`** **`-p add default`** *default-route* **`\`** **`-ifp`** *ifname* The value of *ifname* is the interface name portion of the `name` property.

The `svc:/network/dns/client` service supports the configuration of a DNS client. The service defines one property group: `config`. The service uses its properties to construct a DNS `resolv.conf(4)` file.

The `config` property group contains the following properties.

TABLE 11–8 config Property Group Properties

Property	Type	Required	Description
`domain`	`astring`	`no`	Local domain name. Used to construct the `domain` directive in `resolv.conf`(4).
`nameserver`	`net_address_list`	`yes`	List of IPv4 and IPv6 addresses. Used to construct the `nameserver` directives in `resolv.conf`(4).
`search`	`astring_list`	`no`	List of domain values for the search list for host name lookup. Used to construct the `search` directive in `resolv.conf`(4).

Name Service Configuration

Use the `sysconfig create-profile` command with the `naming_services` grouping to generate a valid profile that configures DNS, NIS, and LDAP clients and name service switch.

```
# sysconfig create-profile -g naming_services -o sc_ns.xml
```

The `svc:/network/dns/client` SMF service configures an initial DNS client configuration. This service is initially disabled with property values that do not result in any system configuration. See the examples in "Specifying Name Service Configuration" on page 152.

Using System Configuration Profile Templates

Profiles can contain variables to retrieve configuration parameters from the install server environment or from criteria specified in the `create-profile` subcommand. In this way, a single profile file can set different configuration parameters on different clients. See Table 6–1 for a list of variables you can use.

In the following example profile, named `hostIPnet.xml`, `AI_HOSTNAME` is a placeholder for the client system's host name, `AI_IPV4` is a placeholder for the client system's IP address, and `AI_NETWORK` is a placeholder for the client system's IP version 4 network number.

```
<?xml version='1.0'?>
<!DOCTYPE service_bundle SYSTEM "/usr/share/lib/xml/dtd/service_bundle.dtd.1">
<service_bundle type="profile" name="sysconfig">
  <service name="system/identity" version="1" type="service">
    <instance name="node" enabled="true">
      <property_group name="config" type="application">
        <propval name="nodename" value="{{AI_HOSTNAME}}"/>
      </property_group>
      <property_group name="install_ipv4_interface" type="application">
        <propval name="name" value="net0/v4"/>
        <propval name="address_type" value="static"/>
        <propval name="static_address" type="net_address_v4" value="{{AI_IPV4}}/8"/>
        <propval name="default_route" type="net_address_v4" value="{{AI_NETWORK}}"/>
      </property_group>
    </instance>
  </service>
</service_bundle>
```

The following commands create customized system configuration profiles in the install service without changing the input `hostandIP.xml` file.

```
# installadm create-profile -n svc1 -f /export/hostIPnet.xml \
-p server1 -c hostname="server1" -c ipv4="10.0.0.1" -c network="10.0.0.0"
# installadm create-profile -n svc1 -f /export/hostIPnet.xml \
-p server2 -c hostname="server2" -c ipv4="10.0.0.2" -c network="10.0.0.0"
$ installadm list -n svc1 -p
Profile  Criteria
-------  --------
server1  hostname = server1
         ipv4 = 10.0.0.1
         network = 10.0.0.0
server2  hostname = server2
         ipv4 = 10.0.0.2
         network = 10.0.0.0
```

While the `hostandIP.xml` file remains unchanged, the `server1` and `server2` profiles that are internal to the `svc1` install service are customized. For example, the `server1` profile has the following content:

```
# installadm export -n svc1 -p server1
<?xml version='1.0'?>
<!DOCTYPE service_bundle SYSTEM "/usr/share/lib/xml/dtd/service_bundle.dtd.1">
```

```
<service_bundle type="profile" name="sysconfig">
  <service name="system/identity" version="1" type="service">
    <instance name="node" enabled="true">
      <property_group name="config" type="application">
        <propval name="nodename" value="server1"/>
      </property_group>
      <property_group name="install_ipv4_interface" type="application">
        <propval name="name" value="net0/v4"/>
        <propval name="address_type" value="static"/>
        <propval name="static_address" type="net_address_v4" value="10.0.0.1/8"/>
        <propval name="default_route" type="net_address_v4" value="10.0.0.0"/>
      </property_group>
    </instance>
  </service>
</service_bundle>
```

This same result can be achieved by setting environment variables. You need to specify one `-c` option that identifies this client uniquely, such as MAC address or host name, so that these configuration values are applied only to that client.

```
# export AI_HOSTNAME=server1;export AI_IPV4=10.0.0.1;export AI_NETWORK=10.0.0.0
# installadm create-profile -n svc1 -f /export/hostIPnet.xml \
-p server1 -c mac="aa:bb:cc:dd:ee:ff"
```

The following table shows the variables that can be used as placeholders in template profiles. These variables can also be specified as environment variables.

TABLE 11–9 Variables for System Configuration Template Profiles

Variable Name	Criteria Name	Description
`AI_ARCH`	`arch`	Kernel architecture from `uname -m`
`AI_CPU`	`cpu`	Processor type from `uname -p`
`AI_HOSTNAME`	`hostname`	Client DNS name
`AI_IPV4`	`ipv4`	IP version 4 network address, or range of IP addresses
`AI_MAC`	`mac`	Hexadecimal MAC address with colon (:) separators, or range of MAC addresses
`AI_MEM`	`mem`	Memory size in megabytes returned by `prtconf`, or a range of memory size
`AI_NETWORK`	`network`	IP version 4 network identifier, or a range of network identifiers
`AI_SERVICE`		Install service name
`AI_ZONENAME`	`zonename`	Name of a `zones`(5) zone as shown by `zoneadm list`

Example System Configuration Profiles

The examples in this section are complete system configuration profiles that can be added to an install service using the `installadm create-profile` command.

Sample System Configuration Profile

This section shows a sample system configuration profile that you might want to use as a base to modify. This sample is available at `/usr/share/auto_install/sc_profiles/sc_sample.xml`. After you have created an install service, this sample configuration profile is available at *image_path*`/auto_install/sc_profiles/sc_sample.xml`.

```
<?xml version='1.0'?>
<!--
Copyright (c) 2011, Oracle and/or its affiliates. All rights reserved.
-->

<!--
Sample system configuration profile for use with Automated Installer

Configures the following:
* User account name 'jack', password 'jack', GID 10, UID 101, root role, bash shell
* 'root' role with password 'solaris'
* Keyboard mappings set to US-English
* Timezone set to UTC
* Network configuration is automated with Network Auto-magic
* DNS name service client is enabled

See installadm(1M) for usage of 'create-profile' subcommand.
-->

<!DOCTYPE service_bundle SYSTEM "/usr/share/lib/xml/dtd/service_bundle.dtd.1">
<service_bundle type="profile" name="system configuration">
    <service name="system/config-user" version="1">
      <instance name="default" enabled="true">
        <property_group name="user_account">
          <propval name="login" value="jack"/>
          <propval name="password" value="9Nd/cwBcNWFZg"/>
          <propval name="description" value="default_user"/>
          <propval name="shell" value="/usr/bin/bash"/>
          <propval name="gid" value='10'/>
          <propval name="type" value="normal"/>
          <propval name="roles" value="root"/>
          <propval name="profiles" value="System Administrator"/>
        </property_group>
        <property_group name="root_account">
            <propval name="password" value="encrypted_password"/>
            <propval name="type" value="role"/>
        </property_group>
      </instance>
    </service>

    <service version="1" name="system/identity">
```

```
      <instance enabled="true" name="node">
        <property_group name="config">
           <propval name="nodename" value="solaris"/>
        </property_group>
      </instance>
    </service>

    <service name="system/console-login" version="1">
      <instance name='default' enabled='true'>
        <property_group name="ttymon">
          <propval name="terminal_type" value="sun"/>
        </property_group>
      </instance>
    </service>

    <service name='system/keymap' version='1'>
      <instance name='default' enabled='true'>
        <property_group name='keymap'>
          <propval name='layout' value='US-English'/>
        </property_group>
      </instance>
    </service>

    <service name='system/timezone' version='1'>
      <instance name='default' enabled='true'>
        <property_group name='timezone'>
          <propval name='localtime' value='UTC'/>
        </property_group>
      </instance>
    </service>

    <service name='system/environment' version='1'>
      <instance name='default' enabled='true'>
        <property_group name='environment'>
          <propval name='LC_ALL' value='en_US.UTF-8'/>
        </property_group>
      </instance>
    </service>

    <service name="network/physical" version="1">
      <instance name="default" enabled="true">
          <property_group name='netcfg' type='application'>
              <propval name='active_ncp' type='astring' value='Automatic'/>
          </property_group>
      </instance>
    </service>
</service_bundle>
```

Specifying Static Network Configuration

This sample profile is available at `/usr/share/auto_install/sc_profiles/static_network.xml`.

This example profile configures the following parameters:

- `bge0` with IPv4 static address 10.0.0.10 and netmask 255.0.0.0
- 10.0.0.1 IPv4 default route
- `bge1` with IPv6 `addrconf` address type
- DNS 8.8.8.8 nameserver
- `example1.com` as local DNS domain name
- `example2.com` and `example3.com` as DNS search list for host name lookup

The netmask is specified with the notation *IPaddress*/*netmask*, where *netmask* is a number that specifies the number of high-order bits of the netmask.

Value of *netmask*	**Netmask Example**
8	255.0.0.0
16	255.255.0.0
24	255.255.255.0

```
<!DOCTYPE service_bundle SYSTEM "/usr/share/lib/xml/dtd/service_bundle.dtd.1">
<service_bundle type="profile" name="system configuration">
    <service name="system/config-user" version="1">
      <instance name="default" enabled="true">
        <property_group name="user_account">
          <propval name="login" value="jack"/>
          <propval name="password" value="9Nd/cwBcNWFZg"/>
          <propval name="description" value="default user"/>
          <propval name="shell" value="/usr/bin/bash"/>
          <propval name="gid" value='10'/>
          <propval name="type" value="normal"/>
          <propval name="roles" value="root"/>
          <propval name="profiles" value="System Administrator"/>
        </property_group>
        <property_group name="root_account">
            <propval name="password" value="encrypted_password"/>
            <propval name="type" value="role"/>
        </property_group>
      </instance>
    </service>

    <service version="1" name="system/identity">
      <instance enabled="true" name="node">
        <property_group name="config">
           <propval name="nodename" value="solaris"/>
        </property_group>
      </instance>
    </service>

    <service name="system/console-login" version="1">
      <instance name='default' enabled='true'>
        <property_group name="ttymon">
          <propval name="terminal_type" value="sun"/>
        </property_group>
```

```
  </instance>
</service>

<service name='system/keymap' version='1'>
  <instance name='default' enabled='true'>
    <property_group name='keymap'>
      <propval name='layout' value='US-English'/>
    </property_group>
  </instance>
</service>

<service name='system/timezone' version='1'>
  <instance name='default' enabled='true'>
    <property_group name='timezone'>
      <propval name='localtime' value='UTC'/>
    </property_group>
  </instance>
</service>

<service name='system/environment' version='1'>
  <instance name='default' enabled='true'>
    <property_group name='environment'>
      <propval name='LC_ALL' value='en_US.UTF-8'/>
    </property_group>
  </instance>
</service>

<service name="network/physical" version="1">
    <instance name="default" enabled="true">
      <property_group name='netcfg' type='application'>
          <propval name='active_ncp' type='astring' value='DefaultFixed'/>
      </property_group>
    </instance>
</service>

<service name='network/install' version='1' type='service'>
    <instance name='default' enabled='true'>
        <property_group name='install_ipv4_interface' type='application'>
            <propval name='name' type='astring' value='net0/v4'/>
            <propval name='address_type' type='astring' value='static'/>
            <propval name='static_address' type='net_address_v4' value='x.x.x.x/n'/>
            <propval name='default_route' type='net_address_v4' value='x.x.x.x'/>
        </property_group>

        <property_group name='install_ipv6_interface' type='application'>
            <propval name='name' type='astring' value='net0/v6'/>
            <propval name='address_type' type='astring' value='addrconf'/>
            <propval name='stateless' type='astring' value='yes'/>
            <propval name='stateful' type='astring' value='yes'/>
        </property_group>
    </instance>
</service>

<service name='network/dns/client' version='1'>
    <property_group name='config'>
        <property name='nameserver'>
            <net_address_list>
                <value_node value='x.x.x.x'/>
            </net_address_list>
```

```
            </property>
            <property name='search'>
                <astring_list>
                    <value_node value='example.com'/>
                </astring_list>
            </property>
        </property_group>
        <instance name='default' enabled='true'/>
    </service>

    <service version="1" name="system/name-service/switch">
        <property_group name="config">
            <propval name="default" value="files"/>
            <propval name="host" value="files dns mdns"/>
            <propval name="printer" value="user files"/>
        </property_group>
        <instance enabled="true" name="default"/>
    </service>

    <service version="1" name="system/name-service/cache">
        <instance enabled="true" name="default"/>
    </service>
</service_bundle>
```

Specifying Name Service Configuration

You can use the sample profiles in this section as templates to create your own profiles, or you can use the `sysconfig` tool with the `naming_services` grouping to produce a profile based on your responses to prompts. See "Creating a Configuration Profile Using the SCI Tool" on page 67 and the `sysconfig(1M)` man page for more information about using `sysconfig` to create a system configuration profile.

Configuring Name Service NIS

EXAMPLE 11–7 Enable NIS For a Specified Domain

This example profile performs the following configuration:

- Enables NIS for `my.domain.com`
- Uses broadcasting to discover the NIS server, which must be on the same subnet
- Enables the name service cache service, which is required

```
<?xml version="1.0"?>
<!DOCTYPE service_bundle SYSTEM "/usr/share/lib/xml/dtd/service_bundle.dtd.1">
<!--
 Copyright (c) 2010, Oracle and/or its affiliates. All rights reserved.
 -->
<service_bundle type='profile' name='default'>
    <service name='network/nis/domain' type='service' version='1'>
    <property_group name='config' type='application'>
        <propval name='domainname' type='hostname' value='my.domain.com'/>
    </property_group>
```

EXAMPLE 11–7 Enable NIS For a Specified Domain *(Continued)*

```
    <instance name='default' enabled='true' />
    </service>
    <service name='network/nis/client' type='service' version='1'>
    <property_group name='config' type='application'>
        <propval name='use_broadcast' type='boolean' value='true'/>
    </property_group>
    <instance name='default' enabled='true' />
    </service>
    <service name='system/name-service/switch' type='service' version='1'>
    <property_group name='config' type='application'>
        <propval name='default' type='astring' value='files nis'/>
        <propval name='printer' type='astring' value='user files nis'/>
        <propval name='netgroup' type='astring' value='nis'/>
    </property_group>
    <instance name='default' enabled='true' />
    </service>
    <service name='system/name-service/cache' type='service' version='1'>
    <instance name='default' enabled='true' />
    </service>
</service_bundle>
```

EXAMPLE 11–8 Configure NIS and Disable DNS

This example profile performs the following configuration:

- Configures name service NIS with automatic broadcasting for a NIS server, which must be on the same subnet
- Configures the NIS domain `my.domain.com`
- Enables the name service cache service, which is required
- Disables the DNS name service

```
<?xml version='1.0'?>
<!DOCTYPE service_bundle SYSTEM "/usr/share/lib/xml/dtd/service_bundle.dtd.1">
<service_bundle type="profile" name="sysconfig">
  <!-- service name-service/switch below for NIS only - (see nsswitch.conf(4)) -->
  <service version="1" type="service" name="system/name-service/switch">
    <property_group type="application" name="config">
      <propval type="astring" name="default" value="files nis"/>
      <propval type="astring" name="printer" value="user files nis"/>
      <propval type="astring" name="netgroup" value="nis"/>
    </property_group>
    <instance enabled="true" name="default"/>
  </service>
  <!-- service name-service/cache must be present along with name-service/switch -->
  <service version="1" type="service" name="system/name-service/cache">
    <instance enabled="true" name="default"/>
  </service>
  <!-- if no DNS, must be explicitly disabled to avoid error msgs -->
  <service version="1" type="service" name="network/dns/client">
    <instance enabled="false" name="default"/>
  </service>
  <service version="1" type="service" name="network/nis/domain">
```

EXAMPLE 11–8 Configure NIS and Disable DNS *(Continued)*

```
    <property_group type="application" name="config">
      <propval type="hostname" name="domainname" value="my.domain.com"/>
    </property_group>
    <instance enabled="true" name="default"/>
  </service>
  <!-- configure the NIS client service to broadcast the subnet for a NIS server -->
  <service version="1" type="service" name="network/nis/client">
    <property_group type="application" name="config">
      <propval type="boolean" name="use_broadcast" value="true"/>
    </property_group>
    <instance enabled="true" name="default"/>
  </service>
</service_bundle>
```

EXAMPLE 11–9 Configure NIS

The following profile configures name service NIS with server IP address 10.0.0.10 and domain `mydomain.com`. The NIS server is not required to be on the same subnet when the server IP address is explicitly specified.

```
<?xml version='1.0'?>
<!DOCTYPE service_bundle SYSTEM "/usr/share/lib/xml/dtd/service_bundle.dtd.1">
<service_bundle type="profile" name="sysconfig">
    <!-- name-service/switch below for NIS only - (see nsswitch.conf(4)) -->
  <service version="1" type="service" name="system/name-service/switch">
    <property_group type="application" name="config">
      <propval type="astring" name="default" value="files nis"/>
      <propval type="astring" name="printer" value="user files nis"/>
      <propval type="astring" name="netgroup" value="nis"/>
    </property_group>
    <instance enabled="true" name="default"/>
  </service>
  <!-- name-service/cache must be present along with name-service/switch -->
  <service version="1" type="service" name="system/name-service/cache">
    <instance enabled="true" name="default"/>
  </service>
  <!-- if no DNS, must be explicitly disabled to avoid error msgs -->
  <service version="1" type="service" name="network/dns/client">
    <instance enabled="false" name="default"/>
  </service>
  <service version="1" type="service" name="network/nis/domain">
    <property_group type="application" name="config">
      <propval type="hostname" name="domainname" value="mydomain.com"/>
      <!-- Note: use property with net_address_list and value_node as below -->
      <property type="net_address" name="ypservers">
        <net_address_list>
          <value_node value="10.0.0.10"/>
        </net_address_list>
      </property>
    </property_group>
    <!-- configure default instance separate from property_group -->
    <instance enabled="true" name="default"/>
  </service>
  <!-- enable the NIS client service -->
```

EXAMPLE 11–9 Configure NIS *(Continued)*

```
  <service version="1" type="service" name="network/nis/client">
    <instance enabled="true" name="default"/>
  </service>
</service_bundle>
```

EXAMPLE 11–10 Enable NIS and DNS For a Specified Domain

This example configures both DNS and NIS name services:

- Specifies multiple DNS name servers
- Specifies a DNS domain search list
- Specifies a NIS domain
- Specifies broadcasting to discover the NIS server

```
<?xml version="1.0"?>
<!DOCTYPE service_bundle SYSTEM "/usr/share/lib/xml/dtd/service_bundle.dtd.1">
<!--
 Copyright (c) 2010, Oracle and/or its affiliates. All rights reserved.
-->
<service_bundle type='profile' name='default'>
    <service name='network/dns/client' type='service' version='1'>
    <property_group name='config' type='application'>
        <propval name='domain' type='astring' value='us.oracle.com'/>
        <property name='nameserver' type='net_address'>
            <net_address_list>
                <value_node value='130.35.249.52' />
                <value_node value='130.35.249.41' />
                <value_node value='130.35.202.15' />
            </net_address_list>
        </property>
        <property name='search' type='astring'>
            <astring_list>
                <value_node value='us.oracle.com oracle.com oraclecorp.com' />
            </astring_list>
        </property>
    </property_group>
    <instance name='default' enabled='true' />
    </service>
    <service name='network/nis/domain' type='service' version='1'>
    <property_group name='config' type='application'>
        <propval name='domainname' type='hostname' value='mydomain.com'/>
    </property_group>
    <instance name='default' enabled='true' />
    </service>
    <service name='network/nis/client' type='service' version='1'>
    <property_group name='config' type='application'>
        <propval name='use_broadcast' type='boolean' value='true'/>
    </property_group>
    <instance name='default' enabled='true' />
    </service>
    <service name='system/name-service/switch' type='service' version='1'>
    <property_group name='config' type='application'>
        <propval name='default' type='astring' value='files nis'/>
        <propval name='host' type='astring' value='files dns'/>
```

EXAMPLE 11–10 Enable NIS and DNS For a Specified Domain *(Continued)*

```
        <propval name='printer' type='astring' value='user files nis'/>
        <propval name='netgroup' type='astring' value='nis'/>
    </property_group>
    <instance name='default' enabled='true' />
    </service>
    <service name='system/name-service/cache' type='service' version='1'>
    <instance name='default' enabled='true' />
    </service>
</service_bundle>
```

Configuring Name Service DNS

EXAMPLE 11–11 Configure DNS With Search List

The following example profile configures the following parameters:

- Name service DNS
- Server IP addresses 1.1.1.1 and 2.2.2.2
- Domain `dom.ain.com`

```
<?xml version='1.0'?>
<!DOCTYPE service_bundle SYSTEM "/usr/share/lib/xml/dtd/service_bundle.dtd.1">
<service_bundle type="profile" name="sysconfig">
  <!-- name-service/switch below for DNS only - (see nsswitch.conf(4)) -->
  <service version="1" type="service" name="system/name-service/switch">
    <property_group type="application" name="config">
      <propval type="astring" name="default" value="files"/>
      <propval type="astring" name="host" value="files dns"/>
      <propval type="astring" name="printer" value="user files"/>
    </property_group>
    <instance enabled="true" name="default"/>
  </service>
  <!-- name-service/cache must be present along with name-service/switch -->
  <service version="1" type="service" name="system/name-service/cache">
    <instance enabled="true" name="default"/>
  </service>
  <service version="1" type="service" name="network/dns/client">
    <property_group type="application" name="config">
      <!-- Note: use property with net_address_list and value_node as below -->
      <property type="net_address" name="nameserver">
        <net_address_list>
          <value_node value="1.1.1.1"/>
          <value_node value="2.2.2.2"/>
        </net_address_list>
      </property>
      <!-- Note: use property with astring_list and value_node,
           concatenating search names, as below -->
      <property type="astring" name="search">
        <astring_list>
          <value_node value="dom.ain.com ain.com"/>
        </astring_list>
      </property>
    </property_group>
```

EXAMPLE 11–11 Configure DNS With Search List *(Continued)*

```
    <instance enabled="true" name="default"/>
  </service>
</service_bundle>
```

Configuring Name Service LDAP

EXAMPLE 11–12 Configure LDAP and LDAP Search Base

This example profile configures the following parameters:

- Name service LDAP with server IP address 10.0.0.10
- Domain `my.domain.com` specified in service `system/nis/domain`
- LDAP search base, which is required, `dc=my,dc=domain,dc=com`

```
<?xml version='1.0'?>
<!DOCTYPE service_bundle SYSTEM "/usr/share/lib/xml/dtd/service_bundle.dtd.1">
<service_bundle type="profile" name="sysconfig">
  <service version="1" type="service" name="system/name-service/switch">
    <property_group type="application" name="config">
      <propval type="astring" name="default" value="files ldap"/>
      <propval type="astring" name="printer" value="user files ldap"/>
      <propval type="astring" name="netgroup" value="ldap"/>
    </property_group>
    <instance enabled="true" name="default"/>
  </service>
  <service version="1" type="service" name="system/name-service/cache">
    <instance enabled="true" name="default"/>
  </service>
  <service version="1" type="service" name="network/dns/client">
    <instance enabled="false" name="default"/>
  </service>
  <service version="1" type="service" name="network/ldap/client">
    <property_group type="application" name="config">
      <propval type="astring" name="profile" value="default"/>
      <property type="host" name="server_list">
        <host_list>
          <value_node value="10.0.0.10"/>
        </host_list>
      </property>
      <propval type="astring" name="search_base" value="dc=my,dc=domain,dc=com"/>
    </property_group>
    <instance enabled="true" name="default"/>
  </service>
  <service version="1" type="service" name="network/nis/domain">
    <property_group type="application" name="config">
      <propval type="hostname" name="domainname" value="my.domain.com"/>
    </property_group>
    <instance enabled="true" name="default"/>
  </service>
</service_bundle>
```

EXAMPLE 11–13 Configure LDAP With a Secure LDAP Server

This example profile configures the following parameters:

- Name service LDAP with server IP address 10.0.0.10
- Domain `my.domain.com` specified in service `system/nis/domain`
- LDAP search base, which is required, `dc=my,dc=domain,dc=com`
- LDAP proxy bind distinguished name `cn=proxyagent,ou=profile,dc=my,dc=domain,dc=com`
- LDAP proxy bind password, encrypted as a security measure. You can find the encrypted value by using one of the following methods:
  - Take the `bind_passwd` property value from `sysconfig create-profile`.
  - Take the value from the SMF configuration on the LDAP server.

```
<?xml version='1.0'?>
<!DOCTYPE service_bundle SYSTEM "/usr/share/lib/xml/dtd/service_bundle.dtd.1">
<service_bundle type="profile" name="sysconfig">
  <service version="1" type="service" name="system/name-service/switch">
    <property_group type="application" name="config">
      <propval type="astring" name="default" value="files ldap"/>
      <propval type="astring" name="printer" value="user files ldap"/>
      <propval type="astring" name="netgroup" value="ldap"/>
    </property_group>
    <instance enabled="true" name="default"/>
  </service>
  <service version="1" type="service" name="system/name-service/cache">
    <instance enabled="true" name="default"/>
  </service>
  <service version="1" type="service" name="network/dns/client">
    <instance enabled="false" name="default"/>
  </service>
  <service version="1" type="service" name="network/ldap/client">
    <property_group type="application" name="config">
      <propval type="astring" name="profile" value="default"/>
      <property type="host" name="server_list">
        <host_list>
          <value_node value="10.0.0.10"/>
        </host_list>
      </property>
      <propval type="astring" name="search_base" value="dc=my,dc=domain,dc=com"/>
    </property_group>
    <property_group type="application" name="cred">
      <propval type="astring" name="bind_dn" value="cn=proxyagent,ou=profile,dc=my,dc=domain,dc=com"/>
      <!-- note that the password below is encrypted -->
      <propval type="astring" name="bind_passwd" value="{NS1}c2ab873ae7c5ceefa4b9"/>
    </property_group>
    <instance enabled="true" name="default"/>
  </service>
  <service version="1" type="service" name="network/nis/domain">
    <property_group type="application" name="config">
      <propval type="hostname" name="domainname" value="my.domain.com"/>
    </property_group>
    <instance enabled="true" name="default"/>
```

EXAMPLE 11–13 Configure LDAP With a Secure LDAP Server *(Continued)*

```
  </service>
</service_bundle>
```

Using DNS With LDAP

DNS name service can be used in conjunction with LDAP name service. A typical usage is for DNS to resolve node names (including the LDAP server name), and for LDAP to resolve all other names. The service `system/name-service/switch` is used to specify DNS for node name search and LDAP to resolve other names, as shown in the first `service` element in this example:

```
<?xml version='1.0'?>
<!DOCTYPE service_bundle SYSTEM "/usr/share/lib/xml/dtd/service_bundle.dtd.1">
<service_bundle type="profile" name="sysconfig">
  <service version="1" type="service" name="system/name-service/switch">
    <property_group type="application" name="config">
      <propval type="astring" name="default" value="files ldap"/>
      <propval type="astring" name="host" value="files dns"/>
      <propval type="astring" name="printer" value="user files ldap"/>
    </property_group>
    <instance enabled="true" name="default"/>
  </service>
  <service version="1" type="service" name="system/name-service/cache">
    <instance enabled="true" name="default"/>
  </service>
  <service version="1" type="service" name="network/dns/client">
    <property_group type="application" name="config">
      <property type="net_address" name="nameserver">
        <net_address_list>
          <value_node value="10.0.0.10"/>
        </net_address_list>
      </property>
      <propval type="astring" name="domain" value="my.domain.com"/>
      <property type="astring" name="search">
        <astring_list>
          <value_node value="my.domain.com"/>
        </astring_list>
      </property>
    </property_group>
    <instance enabled="true" name="default"/>
  </service>
  <service version="1" type="service" name="network/ldap/client">
    <property_group type="application" name="config">
      <propval type="astring" name="profile" value="default"/>
      <property type="host" name="server_list">
        <host_list>
          <!-- here, DNS is expected to resolve the LDAP server by name -->
          <value_node value="ldapserver.my.domain.com"/>
        </host_list>
      </property>
      <propval type="astring" name="search_base" value="dc=my,dc=domain,dc=com"/>
    </property_group>
    <instance enabled="true" name="default"/>
  </service>
```

```
  <service version="1" type="service" name="network/nis/domain">
    <property_group type="application" name="config">
      <propval type="hostname" name="domainname" value="my.domain.com"/>
    </property_group>
    <instance enabled="true" name="default"/>
  </service>
</service_bundle>
```

Using NIS With DNS

NIS can be used in conjunction with DNS in a similar way.

```
<?xml version='1.0'?>
<!DOCTYPE service_bundle SYSTEM "/usr/share/lib/xml/dtd/service_bundle.dtd.1">
<service_bundle type="profile" name="sysconfig">
  <service version="1" type="service" name="system/name-service/switch">
    <property_group type="application" name="config">
      <propval type="astring" name="default" value="files nis"/>
      <propval type="astring" name="host" value="files dns"/>
      <propval type="astring" name="printer" value="user files nis"/>
    </property_group>
    <instance enabled="true" name="default"/>
  </service>
  <service version="1" type="service" name="system/name-service/cache">
    <instance enabled="true" name="default"/>
  </service>
  <service version="1" type="service" name="network/dns/client">
    <property_group type="application" name="config">
      <property type="net_address" name="nameserver">
        <net_address_list>
          <value_node value="10.0.0.10"/>
        </net_address_list>
      </property>
      <propval type="astring" name="domain" value="my.domain.com"/>
      <property type="astring" name="search">
        <astring_list>
          <value_node value="my.domain.com"/>
        </astring_list>
      </property>
    </property_group>
    <instance enabled="true" name="default"/>
  </service>
  <service version="1" type="service" name="network/nis/domain">
    <property_group type="application" name="config">
      <propval type="hostname" name="domainname" value="my.domain.com"/>
    </property_group>
    <instance enabled="true" name="default"/>
  </service>
  <service version="1" type="service" name="network/nis/client">
    <property_group type="application" name="config">
      <propval type="boolean" name="use_broadcast" value="true"/>
    </property_group>
    <instance enabled="true" name="default"/>
  </service>
</service_bundle>
```

◆ ◆ ◆ CHAPTER 12

Installing and Configuring Zones

This chapter describes how to specify installation and configuration of non-global zones as part of an AI client installation.

How AI Installs Non-Global Zones

Non-global zones are installed and configured on the first reboot after the global zone is installed.

1. When a system is installed using AI, non-global zones can be installed on that system by using the `configuration` element in the AI manifest. See "Zone Specification in the Global Zone AI Manifest" on page 162 for information about the `configuration` element.
2. When the system first boots after the global zone installation, the zones self-assembly SMF service (`svc:/system/zones-install:default`) configures and installs each non-global zone defined in the global zone AI manifest. See "Non-Global Zone Configuration and Installation Data" on page 163 for information about the data used to install the non-global zones.
3. If the zone is configured with `autoboot=true`, the `system/zones-install` service boots the zone after the zone is installed.

   Labeled zones can be created and installed using the `system/zones-install` service. Labeled zones are autobooted only if the zone is configured with `autoboot=true` and the global zone is also labeled. After AI has installed the global zone and the `system/zones-install` service has created and installed the labeled non-global zones, you can make the necessary changes to make the global zone labeled. On reboot of the system, the `svc:/system/zones:default` service boots any labeled zones that have set `autoboot=true`.

The `system/zones-install` service remains online but will not process new configuration information until restarted. You should not disable or enable the `system/zones-install` service. You should only restart this service.

To monitor non-global zone installation, monitor the `system/zones-install` service or the output of `zoneadm list -cv`.

Zones are not installed if any of the following errors occurs:

- A zone `config` file is not syntactically correct.
- A collision exists among zone names, zone paths, or delegated ZFS datasets in the set of zones to be installed
- Required datasets are not configured in the global zone.

Zone Specification in the Global Zone AI Manifest

Use the `configuration` element in the AI manifest for the client system to specify non-global zones. Use the `name` attribute of the `configuration` element to specify the name of the zone. Use the `source` attribute to specify the location of the `config` file for the zone. The `source` location can be any `http://` or `file://` location that the client can access during installation.

The following sample AI manifest specifies two non-global zones:

```
<!DOCTYPE auto_install SYSTEM "file:///usr/share/install/ai.dtd.1">
<auto_install>
  <ai_instance>
    <target>
      <logical>
        <zpool name="rpool" is_root="true">
          <filesystem name="export" mountpoint="/export"/>
          <filesystem name="export/home"/>
          <be name="solaris"/>
        </zpool>
      </logical>
    </target>
    <software type="IPS">
      <source>
        <publisher name="solaris">
          <origin name="http://pkg.oracle.com/solaris/release"/>
        </publisher>
      </source>
      <software_data action="install">
        <name>pkg:/entire@latest</name>
        <name>pkg:/group/system/solaris-large-server</name>
      </software_data>
    </software>

    <configuration type="zone" name="zone1" source="http://server/zone1/config"/>
    <configuration type="zone" name="zone2" source="file:///net/server/zone2/config"/>

  </ai_instance>
</auto_install>
```

Non-Global Zone Configuration and Installation Data

The following files are used to configure and install non-global zones:

`config` file

Required. The `config` file is the zone's configuration in file form from the output of the `zonecfg export` command.

The location of the `config` file is specified by the `source` attribute of the `configuration` element in the AI manifest. AI copies this `config` file onto the installed client system to be used to configure the zone.

AI manifest

Optional. This AI manifest for zone installation specifies packages to be installed in the zone, along with publisher information and certificate and key files as necessary. See "Non-Global Zone AI Manifest" on page 164 for information about creating a custom AI manifest for a zone.

To provide a custom AI manifest for a zone, add the manifest to the install service that is installing the global zone. In the `create-manifest` command, specify the `zonename` criteria keyword with the names of all zones that should use this AI manifest.

If you do not provide a custom AI manifest for a non-global zone, the default AI manifest for zones is used as shown in Example 12–1.

Configuration profile

Optional. You can provide zero or more configuration files for a non-global zone. These configuration profiles are similar to the system configuration profiles for configuring the global zone. See Chapter 11, "Configuring the Client System," for information about system configuration profile files. You might want to provide configuration profile files to specify zone configuration such as users and the root password for the zone administrator. See "Non-Global Zone Configuration Profiles" on page 166 for an example configuration profile for a non-global zone.

To provide configuration profile files for a zone, add the configuration profiles to the install service that is installing the global zone. In the `create-profile` command, specify the `zonename` criteria keyword with the names of all zones that should use this configuration profile.

If you do not provide any configuration profile files, the system configuration interactive tool runs and queries for required data on first boot of the zone. See "Configuring a System" on page 63 for information about using the interactive configuration tool.

The following example adds the `/tmp/zmanifest.xml` AI manifest to the `s11-sparc` install service and specifies that `zone1` and `zone2` should use this manifest.

```
# installadm create-manifest -n s11-sparc -f /tmp/zmanifest.xml \
-m zmanifest -c zonename="zone1 zone2"
```

The following example adds the `/tmp/z1profile.xml` configuration profile to the `s11-sparc` install service and specifies that `zone1` and `zone2` should use this profile.

```
# installadm create-profile -n s11-sparc -f /tmp/z1profile.xml \
-p z1profile -c zonename="zone1 zone2"
```

The following example adds the `/tmp/z2profile.xml` configuration profile to the `s11-sparc` install service and specifies that `zone2` should use this profile.

```
# installadm create-profile -n s11-sparc -f /tmp/z2profile.xml \
-p z2profile -c zonename=zone2
```

The following example shows the AI manifests and configuration profiles that have been added to the `s11-sparc` install service.

```
# installadm list -n s11-sparc -m -p

Manifest         Status   Criteria
--------         ------   --------
orig_default     Default  None
line1-netra2000           mac       = 00:14:4F:2D:7A:DC
zmanifest                 zonename  = zone1 zone2

Profile    Criteria
-------    --------
z1profile  zonename  = zone1 zone2
z2profile  zonename  = zone2
```

Non-Global Zone AI Manifest

This AI manifest for non-global zone installation is similar to the AI manifest for installing the global zone. See the `ai_manifest(4)` man page for information about AI manifest elements and attributes.

Do not use the following elements or attributes in a non-global zone AI manifest:

- The `auto_reboot` attribute of the `ai_instance` element
- The `http_proxy` attribute of the `ai_instance` element
- The `disk` child element of the `target` element
- The `noswap` attribute of the `logical` element
- The `nodump` attribute of the `logical` element
- The `configuration` element

Only the `logical` child element of the `target` element can be used in a non-global zone AI manifest. The `logical` section defines additional file systems, or datasets.

In the `zpool` element of the `logical` element, only the `filesystem` and be child elements can be used in a non-global zone AI manifest.

The only value supported for the `type` attribute of the `software` element is IPS, which is the default value.

EXAMPLE 12–1 Default Zone AI Manifest

The following file shows the default AI manifest for non-global zones. This manifest is used if you do not provide a custom AI manifest for a zone. This manifest is available at `/usr/share/auto_install/manifest/zone_default.xml`.

```
<?xml version="1.0" encoding="UTF-8"?>
<!--

 Copyright (c) 2011, Oracle and/or its affiliates. All rights reserved.

-->
<!DOCTYPE auto_install SYSTEM "file:///usr/share/install/ai.dtd.@DTD_VERSION_AI@">

<auto_install>
    <ai_instance name="zone_default">
        <target>
            <logical>
                <zpool name="rpool">
                    <filesystem name="export" mountpoint="/export"/>
                    <filesystem name="export/home"/>
                    <be name="solaris">
                        <options>
                            <option name="compression" value="on"/>
                        </options>
                    </be>
                </zpool>
            </logical>
        </target>

        <software type="IPS">
            <destination>
                <image>
                    <!-- Specify locales to install -->
                    <facet set="false">facet.locale.*</facet>
                    <facet set="true">facet.locale.de</facet>
                    <facet set="true">facet.locale.de_DE</facet>
                    <facet set="true">facet.locale.en</facet>
                    <facet set="true">facet.locale.en_US</facet>
                    <facet set="true">facet.locale.es</facet>
                    <facet set="true">facet.locale.es_ES</facet>
                    <facet set="true">facet.locale.fr</facet>
                    <facet set="true">facet.locale.fr_FR</facet>
                    <facet set="true">facet.locale.it</facet>
                    <facet set="true">facet.locale.it_IT</facet>
```

EXAMPLE 12–1 Default Zone AI Manifest *(Continued)*

```
                    <facet set="true">facet.locale.ja</facet>
                    <facet set="true">facet.locale.ja_*</facet>
                    <facet set="true">facet.locale.ko</facet>
                    <facet set="true">facet.locale.ko_*</facet>
                    <facet set="true">facet.locale.pt</facet>
                    <facet set="true">facet.locale.pt_BR</facet>
                    <facet set="true">facet.locale.zh</facet>
                    <facet set="true">facet.locale.zh_CN</facet>
                    <facet set="true">facet.locale.zh_TW</facet>
                </image>
            </destination>
            <software_data action="install">
                <name>pkg:/group/system/solaris-small-server</name>
            </software_data>
        </software>
    </ai_instance>
</auto_install>
```

Non-Global Zone Configuration Profiles

You can provide a configuration profile for a zone to configure zone parameters such as language, locale, time zone, terminal, users, and the root password for the zone administrator. You can configure the time zone, but you cannot set the time. You can configure DNS.

If you specify configuration that is not allowed in a zone, those property settings are ignored.

The following file shows a sample configuration profile file for non-global zones.

```
<!DOCTYPE service_bundle SYSTEM "/usr/share/lib/xml/dtd/service_bundle.dtd.1">
<service_bundle type="profile" name="sysconfig">
  <service version="1" type="service" name="system/config-user">
    <instance enabled="true" name="default">
      <property_group type="application" name="root_account">
        <propval type="astring" name="login" value="root"/>
        <propval type="astring" name="password" value="encrypted_password"/>
        <propval type="astring" name="type" value="normal"/>
      </property_group>
    </instance>
  </service>
  <service version="1" type="service" name="system/timezone">
    <instance enabled="true" name="default">
      <property_group type="application" name="timezone">
        <propval type="astring" name="localtime" value="UTC"/>
      </property_group>
    </instance>
  </service>
  <service version="1" type="service" name="system/environment">
    <instance enabled="true" name="init">
      <property_group type="application" name="environment">
        <propval type="astring" name="LC_ALL" value="C"/>
      </property_group>
```

```
    </instance>
  </service>
  <service version="1" type="service" name="system/identity">
    <instance enabled="true" name="node">
      <property_group type="application" name="config">
        <propval type="astring" name="nodename" value="z2-test"/>
      </property_group>
    </instance>
  </service>
  <service version="1" type="service" name="system/keymap">
    <instance enabled="true" name="default">
      <property_group type="system" name="keymap">
        <propval type="astring" name="layout" value="US-English"/>
      </property_group>
    </instance>
  </service>
  <service version="1" type="service" name="system/console-login">
    <instance enabled="true" name="default">
      <property_group type="application" name="ttymon">
        <propval type="astring" name="terminal_type" value="vt100"/>
      </property_group>
    </instance>
  </service>
  <service version="1" type="service" name="network/physical">
    <instance enabled="true" name="default">
      <property_group type="application" name="netcfg"/>
    </instance>
  </service>
</service_bundle>
```

◆ ◆ ◆ CHAPTER 13

Running a Custom Script During First Boot

To perform any additional installation or configuration that cannot be done in the AI manifest or in a system configuration profile, you can create a script that is executed at first boot by a run-once SMF service.

1. Create the first-boot script.
2. Create the manifest for an SMF service that runs once at first boot and executes the script.
3. Create an IPS package that contains the service manifest and the script.
4. Add the package to an IPS package repository.
5. Install that package during the AI installation by specifying that package in the AI manifest.

The service runs and executes the script at first reboot after the AI installation.

Creating a Script To Run at First Boot

Near the top of the SMF service manifest shown in "Creating an SMF Manifest File" on page 171, the service is enabled by the following line:

```
<create_default_instance enabled='true' />
```

At the end of your first-boot script, disable the service and uninstall the package so that the first-boot script runs only one time.

```
#!/bin/sh

svcadm disable svc:/site/first-boot-script-svc:default
pkg uninstall pkg:/first-boot-script

exit $SMF_EXIT_OK
```

In this example, `first-boot-script-svc` is the SMF service created in "Creating an SMF Manifest File" on page 171, and `first-boot-script` is the IPS package created in "Creating an IPS Package For the Script and Service" on page 172.

Tip –

- Use only one first-boot script to avoid having different commands in different scripts that collide with one another.
- If you must reboot in the first-boot script, make the reboot the last action in the script.

EXAMPLE 13–1 Sample First-Boot Script

This example shows a sample first-boot script named `/opt/site/first-boot-script.sh`. This script first saves a copy of the boot environment (BE) that was just created by the AI installation. Saving a copy of the BE before the first-boot script modifies it enables you to easily recover from any problems introduced by the script by booting into the saved BE.

```
#!/bin/sh

# Load SMF shell support definitions
. /lib/svc/share/smf_include.sh

echo "Save original boot environment first."
# Obtain the active BE name from beadm: The active BE on reboot has an R in
# the third column of 'beadm list' output. Its name is in column one.
bename=`beadm list -Hd|nawk -F ';' '$3 ~ /R/ {print $1}'`
beadm create ${bename}.orig

# Add support for faster serial console
echo "Setting up support for faster serial console"
! grep console115200 >/dev/null /etc/ttydefs && \
    echo "console115200:115200 hupcl opost onlcr:115200::console115200" \
    >>/etc/ttydefs

echo "Configure ssh server for root login and X11 forwarding"
ed - << EOF
r /etc/ssh/sshd_config
/PermitRootLogin/
c
PermitRootLogin yes
.
/X11Forwarding/
c
X11Forwarding yes
.
w
q
EOF
svcadm refresh ssh

# Set up coreadm
echo "Setting core file configuration"
coreadm -G default -g /var/cores/%f.%u.%p.%t.core
coreadm -e global
coreadm -e process
coreadm -e proc-setid
coreadm -e log
```

EXAMPLE 13–1 Sample First-Boot Script *(Continued)*

```
# Disable service and uninstall package
svcadm disable svc:/site/first-boot-script-svc:default
pkg uninstall pkg:/first-boot-script

echo "Site first-boot script done. Rebooting in 5 seconds." > /dev/console

sleep 5 && reboot -p || reboot &

exit $SMF_EXIT_OK
```

Creating an SMF Manifest File

Create an SMF manifest file that defines a transient service that executes a script.

- The `duration` property near the end of the example service manifest below has the value `transient`. A transient service executes the `start` method once and does not execute it again if the method exits with `$SMF_EXIT_OK`. The `svc.startd` daemon does not try to restart the script after its first execution.
- The `start` method of the service executes the first-boot script.
- The name of the service in this example is `site/first-boot-script-svc`. After the client is booted, you can see the service in the output of the following command:

  ```
  $ svcs -a|grep first-boot-script
  STATE       STIME    FMRI
  disabled    13:51:42 svc:/site/first-boot-script-svc:default
  ```

- This example specifies the `multi-user` dependency to make sure that the first boot script executes late in the startup sequence after first boot. Depending on what your first boot script does, you might not need such a dependency. If you do not specify such a dependency, your script might run before the system is configured the way the script expects it to be.

Tip – Evaluate your script's dependencies and construct the service to run the script after its dependencies are satisfied.

EXAMPLE 13–2 Sample SMF Service Manifest

The following file is the SMF manifest file for the `first-boot-script-svc` service: `first-boot-script-svc-manifest.xml`. This service is enabled by default and does not restart. The script that the service runs in this example is `/opt/site/first-boot-script.sh`.

```
<?xml version="1.0"?>
<!DOCTYPE service_bundle SYSTEM "/usr/share/lib/xml/dtd/service_bundle.dtd.1">
<service_bundle type='manifest' name='first-boot-script:site-first-boot-script-svc'>

<service
    name='site/first-boot-script-svc'
```

EXAMPLE 13–2 Sample SMF Service Manifest *(Continued)*

```
    type='service'
    version='1'>
    <create_default_instance enabled='true' />
    <single_instance />

<!-- Run the script late in the startup sequence after first boot. -->
<dependency name='multi-user' grouping='require_all' restart_on='none' type='service'>
    <service_fmri value='svc:/milestone/multi-user:default' />
</dependency>

<exec_method
    type='method'
    name='start'
    exec='/opt/site/first-boot-script.sh'
    timeout_seconds='360'>
    <method_context>
        <method_credential user='root'/>
    </method_context>
</exec_method>

<exec_method
    type='method'
    name='stop'
    exec=':true'
    timeout_seconds='60'
/>

<property_group name='startd' type='framework'>
    <propval name='duration' type='astring' value='transient' />
</property_group>

</service>
</service_bundle>
```

Creating an IPS Package For the Script and Service

Create an IPS package that contains:

- The service manifest file from "Creating an SMF Manifest File" on page 171.
- The first-boot script from "Creating a Script To Run at First Boot" on page 169.
- Any files needed by the script that cannot be provided from another location such as the install server.

▼ How To Create and Publish the IPS Package

1 Create the directory hierarchy.

In this example, the service manifest is installed into `/lib/svc/manifest/site`, and the first-boot script is installed into `/opt/site`.

```
$ mkdir -p proto/lib/svc/manifest/site
$ mkdir -p proto/opt/site
$ cp first-boot-script-svc-manifest.xml proto/lib/svc/manifest/site
$ cp first-boot-script.sh proto/opt/site
```

2 Create the package manifest.

Create the following file named `first-boot-script.p5m`.

```
set name=pkg.fmri value=first-boot-script@1.0,5.11
set name=pkg.summary value="AI first boot script"
set name=pkg.description value="Script that runs at first boot after AI installation"
file path=lib/svc/manifest/site/first-boot-script-svc-manifest.xml mode=0444 \
    owner=root group=sys
dir path=opt/site mode=0755 owner=root group=sys
file path=opt/site/first-boot-script.sh mode=0555 owner=root group=sys
```

Depending on what your first-boot script does, you might need to specify dependencies. If you modify this manifest, use `pkglint` to verify the new manifest is correct. You can ignore warnings.

```
# pkglint first-boot-script.p5m
```

3 Create the repository for the package.

This example creates the repository in the local directory, with `firstboot` as the publisher.

Note – Create the repository in a directory that is accessible by the AI clients at installation time.

```
$ pkgrepo create firstbootrepo
# pkgrepo -s firstbootrepo add-publisher firstboot
```

4 Publish the package.

```
# pkgsend publish -d ./proto -s ./firstbootrepo first-boot-script.p5m
pkg://firstboot/first-boot-script@1.0,5.11:20111101T024901Z
PUBLISHED
```

Clients can install the package from the `firstbootrepo` repository. The `firstboot` publisher with `firstbootrepo` origin is defined in the AI manifest as shown in the next section.

List the package to verify that the package is available.

```
$ pkg list -g ./firstbootrepo first-boot-script
NAME (PUBLISHER)                 VERSION    IFO
first-boot-script (firstboot)    1.0        ---
```

You might want to try a test installation of the package. The `-n` option means do not actually install the package.

```
# pkg set-publisher -g ./firstbootrepo firstboot
# pkg install -nv first-boot-script
           Packages to install:         1
     Estimated space avaialble: 111.26 GB
Estimated space to be consumed:  72.42 MB
       Create boot environment:        No
Create backup boot environment:        No
          Rebuild boot archive:        No

Changed packages:
firstboot
  first-boot-script
    None -> 1.0,5.11:20111101T024901Z
```

Next Steps See *Copying and Creating Oracle Solaris 11 Package Repositories* for instructions to make the new repository accessible to client systems through either NFS sharing or HTTP.

Installing the First Boot Package on the AI Client

Create a custom AI manifest file and add the new package, publisher, and repository information.

▼ How To Install the IPS Package

1 Add the package to the AI manifest.

Add the package to the software installation section of the AI manifest. Either customize an AI manifest XML file or write a derived manifests script to add these elements. See Chapter 10, "Provisioning the Client System," for information about customizing an AI manifest.

Use the `installadm export` command to retrieve the content of one or more existing AI manifests. The following example shows the XML elements you need to add.

```
<software type="IPS">
  <source>
    <publisher name="solaris">
      <origin name="http://pkg.oracle.com/solaris/release"/>
    </publisher>
    <publisher name="firstboot">
      <origin name="file:///net/host1/export/firstbootrepo"/>
    </publisher>
  </source>
  <software_data action="install">
    <name>pkg:/first-boot-script</name>
  </software_data>
</software>
```

Make sure the origin is a URI the clients can access during AI installation. Use `zfs set sharenfs` to export the repository so that clients can access the local repository.

2 Update the modified AI manifest in the AI install service.

Use the `installadm update-manifest` command to replace the AI manifest content with the content that includes the first-boot script package. Any criteria or default status remain with the manifest or script following the update.

3 Network boot the client.

Network boot the client to use AI to install the Oracle Solaris 11 OS and your custom `first-boot-script` package. When the client is booted after installation, the service runs and executes the first-boot script.

◆ ◆ ◆ CHAPTER 14

Setting Up Oracle Configuration Manager For Use By AI Client Systems

Oracle Configuration Manager enables you to log your system configurations with My Oracle Support. When you do this, Oracle can provide more proactive and targeted support.

This chapter explains how to ensure that Oracle Configuration Manager works on your AI client installations.

Default Behavior of Oracle Configuration Manager on AI Clients

When the client first boot reaches the `network` milestone, the `system/ocm` SMF service starts.

The `system/ocm` service checks whether the `opt_out` property is specified. The `opt_out` property is not specified by default.

- If the `opt_out` property is specified, the service disables itself and exits. To specify the `opt_out` property, see "Opting Out of Oracle Configuration Manager" on page 180.
- If the `opt_out` property is not specified, the `system/ocm` service checks whether the `response_file_pkg_name` property is specified. The `response_file_pkg_name` property is not specified by default.
  - If the `response_file_pkg_name` property is not specified, Oracle Configuration Manager uses the default response file.

    Oracle Configuration Manager performs an anonymous registration and begins to collect system data and send the data to an Internet location, `https:ccr.oracle.com/`.
  - If the `response_file_pkg_name` property is specified, Oracle Configuration Manager uses the custom response file that you deliver in that package. The package name is given in this property, not the file name. To specify the `response_file_pkg_name` property, see "Providing a Custom Response File" on page 178.

You can use this custom response file to associate your support identifier with this system information and to specify a proxy or an Oracle support hub for systems that do not have Internet access.

Providing a Custom Response File

You need to provide a custom response file if you want to specify any of the following information:

- Your customer support identifier to associate this data with your company.
- A proxy or an Oracle support hub to enable data collection and sending for a system that does not have Internet access.

▼ How To Create and Install a Custom Response File Package

Perform the following steps to provide a custom IPS package with the custom response file.

1 Create the custom response file.

Create the response file using an existing Oracle Configuration Manager installation. For information about creating the custom response file, see "Creating a Response File (`http://download.oracle.com/docs/cd/E23562_01/doc.1035/e22050/admin.htm#BAJICIFE`)" in *Oracle Configuration Manager Installation and Administration Guide*. The response file must be named `ocm.rsp`.

2 Create a custom IPS package.

Create an IPS package that contains the custom response file. If you create different custom response files for different systems, put each response file in a separate package. The path and file name are the same for each custom response file. The package names can be different. Only one Oracle Configuration Manager custom response file package can be included in any AI manifest.

a. Create the directory hierarchy.

The custom response file must be installed as `/var/ocm/.rsp/ocm.rsp` on the AI client system.

```
$ mkdir -p proto/var/ocm/.rsp
$ cp ocm.rsp proto/var/ocm/.rsp
```

b. Create the package manifest.

Create the following file named `ocsresp.p5m`.

```
set name=pkg.fmri value=ocsresp@1.0,5.11
set name=pkg.summary value="Oracle Configuration Manager custom response file"
```

```
set name=pkg.description value="Oracle Configuration Manager custom response file"
depend fmri=pkg:/system/ocm type=require
file var/ocm/.rsp/ocm.rsp group=root mode=0444 owner=root path=var/ocm/.rsp/ocm.rsp
```

The `depend` and `file` lines are the only lines that are required. The `summary` and `description` lines are useful if users try to find or view this package in a repository.

c. Create the repository for the package.

This example creates the repository in the local directory, with `ocm` as the publisher.

Note – Create the repository in a directory that is accessible by the AI clients at installation time.

```
$ pkgrepo create customocm
# pkgrepo -s customocm add-publisher ocm
```

d. Publish the package.

```
$ pkgsend publish -d ./proto -s ./customocm ocsresp.p5m
pkg://ocm/ocsresp@1.0,5.11:20110708T174359Z
PUBLISHED
```

Clients can install the package from the `customocm` repository. The `ocm` publisher with `customocm` origin is defined in the AI manifest as shown in the next step.

See *Copying and Creating Oracle Solaris 11 Package Repositories* for instructions to make the new repository accessible to client systems through either NFS sharing or HTTP.

3 Add the package to the AI manifest.

Add the package to the software installation section of the AI manifest. Either customize an AI manifest XML file or write a derived manifests script to add these elements. See Chapter 10, "Provisioning the Client System," for information about customizing an AI manifest.

Use the `installadm export` command to retrieve the content of one or more existing AI manifests. The following example shows the XML elements you need to add. Only one Oracle Configuration Manager custom response file package can be included in any AI manifest.

```
<software type="IPS">
  <source>
    <publisher name="solaris">
      <origin name="http://pkg.oracle.com/solaris/release"/>
    </publisher>
    <publisher name="ocm">
      <origin name="file:///net/host/export/customocm"/>
    </publisher>
  </source>
  <software_data action="install">
    <name>pkg://ocm/ocsresp</name>
  </software_data>
</software>
```

Make sure the origin is a URI the clients can access during AI installation. Use `zfs set sharenfs` to export the repository so that clients can access the local repository.

4 Update the modified AI manifest in the AI install service.

Use the `installadm update-manifest` command to replace the AI manifest content with the content that includes the custom Oracle Configuration Manager response file package. Any criteria or default status remain with the manifest or script following the update.

5 Add the custom response file package property to the system configuration profile.

Since a client can use any number of system configuration profiles, you might want to put this property setting in a separate profile so that you can easily add this setting for any or all clients.

```
<service name='system/ocm' type='service' version='1' >
  <instance name='default' enabled='true' >
    <property_group name='reg' type='framework'>
      <propval name='response_file_pkg_name' type='astring' value='pkg:/ocsresp' />
    </property_group>
  </instance>
</service>
```

6 Add the modified system configuration profile to the AI install service.

Use the `create-profile` subcommand of the `installadm` command to add the new or customized system configuration profile to the AI install service.

In the following example, `filename` is the full path name of the configuration profile. The file `criteria.xml` specifies which clients should use this profile. If this profile contains only the response file package name setting, and if the same custom response file should be used by all clients, omit the `-C` or `-c` options.

```
# installadm create-profile -n svcname -f filename -C criteria.xml
```

Opting Out of Oracle Configuration Manager

If you prefer to opt out of using Oracle Configuration Manager for some clients, add the following element to a system configuration profile for those clients:

```
<service name='system/ocm' type='service' version='1' >
  <instance name='default' enabled='false' >
    <property_group name='reg' type='framework'>
      <propval name='opt_out' type='astring' value='true' />
    </property_group>
  </instance>
</service>
```

Add the new or modified configuration profile to the AI install service with criteria to identify those clients.

◆ ◆ ◆ CHAPTER 15

Installing Client Systems

This chapter gives the system requirements for AI clients and describes how to associate each client with the correct AI install service.

How a Client Is Installed

When you set up your install server, you created at least one install service for each client architecture and each version of the Oracle Solaris OS that you plan to install. When you created each install service, you created customized installation instructions and system configuration instructions for different clients as needed. To start the automated installation, you just need to boot the client.

After you network boot the client, the installation and configuration of the client are completed using a net image, installation specifications, and system configuration specifications provided by the install service.

1. The administrator network boots the client.
2. The client system contacts the DHCP server and retrieves the client IP address, the boot file, and the IP address of the install server if needed.
3. The client system loads the net image from one of the following sources:
   - The install service assigned to this client with the `installadm create-client` command.
   - The default install service for this architecture.
4. The client system completes its installation using the AI manifest determined as described in "Selecting the AI Manifest" on page 108.
5. The client system reboots if `auto_reboot` is set in the AI manifest, or the client is rebooted by the system administrator.
6. During reboot, the client system is configured in one of the following ways:

- Using system configuration profiles determined as described in "Selecting System Configuration Profiles" on page 109.
- Using the administrator's responses in the interactive system configuration tool.

When the AI client installation is finished, the message "Automated Installation succeeded" displays on the screen, a completion message displays in the `/system/volatile/install_log` log file, and the `svc:/application/auto-installer` SMF service on that client reaches the `online` state.

Client System Requirements

The client systems for automated installation must meet the following requirements. Any system that meets these requirements can be used as an automated install client, including laptops, desktops, virtual machines, and enterprise servers.

SPARC and x86 Client System Requirements

SPARC and x86 clients of AI installation over the network must meet the following requirements:

Memory 1 GB minimum

Disk space 13 GB minimum

Network access Client systems must be able to access the following resources during the installation:

- A DHCP server that provides network configuration information
- The AI install server
- An IPS repository that contains the packages to be installed on the client system

Additional SPARC Client System Requirements

Firmware The firmware on SPARC clients must be updated to include the current version of the Open Boot PROM (OBP) that contains the latest WAN boot support.

WAN boot SPARC clients of AI installation over the network must support WAN boot.

To boot over the network, AI requires WAN boot support for SPARC clients. You can check whether your client Open Boot PROM (OBP) supports WAN boot by checking whether `network-boot-arguments` is a valid variable that can be set in the `eeprom`.

If the variable `network-boot-arguments` is displayed, or if the command returns the output `network-boot-arguments: data not available`, the OBP supports WAN boot and the client can be installed over the network.

```
# eeprom | grep network-boot-arguments
network-boot-arguments: data not available
```

If the command results in no output, then WAN Boot is not supported and the client cannot be installed over the network. See Chapter 5, “Automated Installations That Boot From Media .”

```
# eeprom | grep network-boot-arguments
```

Setting Up an Install Client

On the install server, use the `installadm create-client` command to associate a particular client with a particular install service.

The `installadm create-client` command requires the following information:

- MAC address for the client
- Name of the install service for the client to use for installation

For x86 clients, you can optionally specify boot properties.

Setting Up an x86 Client

The following example associates the x86 client with MAC address `0:e0:81:5d:bf:e0` with the `s11-x86` install service. The DHCP configuration output by this command must be added to the DHCP server. If this DHCP configuration is not done, the client cannot boot the `s11-x86` install service.

```
# installadm create-client -n s11-x86 -e 0:e0:81:5d:bf:e0
No local DHCP configuration found. If not already configured, the
following should be added to the DHCP configuration:
    Boot server IP      : 10.80.239.5
    Boot file           : 0100E0815DBFE0
```

You can also view the results of the `installadm create-client` command in the `/etc/netboot` directory. This listing shows install service files for this client only. Lines in the `menu.lst`file are broken for readability.

```
# cd /etc/netboot
# ls -l
lrwxrwxrwx  Aug 26 08:27 0100E0815DBFE0 -> ./s11-x86/boot/grub/pxegrub
-rw-r--r--  Aug 26 08:27 menu.lst.0100E0815DBFE0
drwxr-xr-x  Aug 26 08:26 s11-x86
```

```
# cat menu.lst.0100E0815DBFE0
default=0
timeout=30
min_mem64=0
title Oracle Solaris 11 11/11 Text Installer and command line
        kernel$ /s11-x86/platform/i86pc/kernel/$ISADIR/unix -B
install_media=http://$serverIP:5555//export/auto_install/s11-x86,install_service=s11-x86,
install_svc_address=$serverIP:5555
        module$ /s11-x86/platform/i86pc/$ISADIR/boot_archive

title Oracle Solaris 11 11/11 Automated Install
        kernel$ /s11-x86/platform/i86pc/kernel/$ISADIR/unix -B install=true,
install_media=http://$serverIP:5555//export/auto_install/s11-x86,install_service=s11-x86,
install_svc_address=$serverIP:5555,livemode=text
        module$ /s11-x86/platform/i86pc/$ISADIR/boot_archive
```

Setting Up a SPARC Client

The following example associates the SPARC client with MAC address `00:14:4f:a7:65:70` with the `s11-sparc` install service.

```
# installadm create-client -n s11-sparc -e 00:14:4f:a7:65:70
```

The DHCP server does not require configuration since the SPARC `wanboot-cgi` boot file has already been configured by `create-service`. See "Create an AI Install Service" on page 86.

Deleting a Client From a Service

Use the `installadm delete-client` command to delete a client from an install service.

```
# installadm delete-client macaddr
```

You do not need to specify the service name since a client can be associated with only one install service.

Installing Clients

Boot the client to start the installation. This section shows you exactly how to boot a SPARC or x86 client. This section also describes how you can monitor installation progress remotely.

Using Secure Shell to Remotely Monitor Installations

You can enable network access to an automated install client by using `ssh`. You can use this access to remotely observe an installation in progress.

Enable remote access by setting the option `livessh` to `enable` in the installation configuration file. When this access is enabled, you can log in to the AI client by using the username `jack` and password `jack`.

Monitoring x86 Client Installations

For x86 systems, the `menu.lst` configuration file is created in the following way:

- If you used the `installadm create-client` command, the filename is `/etc/netboot/menu.lst.01`*MAC_address*, where *MAC_address* is the MAC address that was specified in the `installadm create-client` command.
- If you did not use the `installadm create-client` command, the filename is `/etc/netboot/`*service_name*`/menu.lst`, where *service_name* is the name of the install service that was created by the `installadm create-service` command.

In this file, options are provided as kernel parameters. In the following example, the `livessh` and `install_debug` options are set to `enable`.

```
kernel$ ... -B install_media=...,livessh=enable,install_debug=enable
```

Monitoring SPARC Client Installations

For SPARC systems, the `system.conf` file can be accessed through the service's net image directory mounted under the `/etc/netboot` directory: `/etc/netboot/`*svcname*`/system.conf`.

In the `system.conf` file, the options are defined as name-value pairs. In the following example, the `livessh` option is set to `enable`.

```
$ cat /etc/netboot/svc1/system.conf
...
livessh=enable
...
```

Installing a SPARC Client

Use the following command to boot SPARC clients from the OBP prompt:

```
ok boot net:dhcp - install
```

SPARC Client Network Boot Sequence

The following events occur during AI boot of a SPARC client:

1. The client boots and gets an IP address and the location of the `wanboot-cgi` file from the DHCP server.
2. The `wanboot-cgi` program reads `wanboot.conf` and sends the location of the WAN boot binary to the client.
3. The WAN boot binary is downloaded using HTTP, and the client boots the WAN boot program.
4. WAN boot gets the `boot_archive` file, and the Oracle Solaris OS is booted.
5. Image archives, `solaris.zlib` and `solarismisc.zlib`, are downloaded using HTTP.
6. The AI manifest and system configuration profiles are downloaded from an AI install service specified either from the mDNS lookup or from the `system.conf` file.
7. The AI install program is invoked with the AI manifest to perform the installation of the Oracle Solaris OS to the client.

Sample SPARC Network Boot Output

The following output from the boot sequence is displayed:

```
{0} ok boot net:dhcp - install

SPARC Enterprise T5120, No Keyboard
Copyright 2008 Sun Microsystems, Inc.  All rights reserved.
OpenBoot 4.29.1, 16256 MB memory available, Serial #81036844.
Ethernet address 0:14:4f:d4:86:2c, Host ID: 84d4862c.

Boot device: /pci@0/pci@0/pci@1/pci@0/pci@2/network@0:dhcp  File and args: - install
1000 Mbps full duplex  Link up
Timed out waiting for BOOTP/DHCP reply
<time unavailable> wanboot info: WAN boot messages->console
<time unavailable> wanboot info: configuring /pci@0/pci@0/pci@1/pci@0/pci@2/network@0:dhcp

1000 Mbps full duplex  Link up
<time unavailable> wanboot info: Starting DHCP configuration
<time unavailable> wanboot info: DHCP configuration succeeded
<time unavailable> wanboot progress: wanbootfs: Read 368 of 368 kB (100%)
<time unavailable> wanboot info: wanbootfs: Download complete
Fri Aug 26 14:34:53 wanboot progress: miniroot: Read 221327 of 221327 kB (100%)
Fri Aug 26 14:34:53 wanboot info: miniroot: Download complete
SunOS Release 5.11 Version snv_175 64-bit
Copyright (c) 1983, 2011, Oracle and/or its affiliates. All rights reserved.
Remounting root read/write
Probing for device nodes ...
Preparing network image for use
Downloading solaris.zlib
--2011-08-26 13:47:31--  http://host1:5555/install/images/sparc_snv175//solaris.zlib
idn_decode failed (9): 'System iconv failed'
Resolving host1... 10.80.238.5
```

```
idn_decode failed (9): ‘System iconv failed’
Connecting to host1|10.80.238.5|:5555... connected.
HTTP request sent, awaiting response... 200 OK
Length: 135977984 (130M) [text/plain]
Saving to: ‘/tmp/solaris.zlib’

100%[======================================>] 135,977,984 22.3M/s   in 6.0s

2011-08-26 13:47:37 (21.7 MB/s) - ‘/tmp/solaris.zlib’ saved [135977984/135977984]

Downloading solarismisc.zlib
--2011-08-26 13:47:37--  http://host1:5555/install/images/sparc_snv175//solarismisc.zlib
idn_decode failed (9): ‘System iconv failed’
Resolving host1... 10.80.238.5
idn_decode failed (9): ‘System iconv failed’
Connecting to host1|10.80.238.5|:5555... connected.
HTTP request sent, awaiting response... 200 OK
Length: 18826752 (18M) [text/plain]
Saving to: ‘/tmp/solarismisc.zlib’

100%[======================================>] 18,826,752  21.7M/s   in 0.8s

2011-08-26 13:47:37 (21.7 MB/s) - ‘/tmp/solarismisc.zlib’ saved [18826752/18826752]

Downloading .image_info
--2011-08-26 13:47:37--  http://host1:5555/install/images/sparc_snv175//.image_info
idn_decode failed (9): ‘System iconv failed’
Resolving host1... 10.80.238.5
idn_decode failed (9): ‘System iconv failed’
Connecting to host1|10.80.238.5|:5555... connected.
HTTP request sent, awaiting response... 200 OK
Length: 82 [text/plain]
Saving to: ‘/tmp/.image_info’

100%[======================================>] 82          --.-K/s   in 0s

2011-08-26 13:47:37 (2.22 MB/s) - ‘/tmp/.image_info’ saved [82/82]

Done mounting image
Configuring devices.
Hostname: solaris
Service discovery phase initiated
Service name to look up: default-sparc
Service discovery finished successfully
Process of obtaining install manifest initiated
Using the install manifest obtained via service discovery

Automated Installation started
The progress of the Automated Installation will be output to the console
Detailed logging is in the logfile at /system/volatile/install_log
Press RETURN to get a login prompt at any time.

solaris console login: 13:48:35    Install Log: /system/volatile/install_log
13:48:35    Using XML Manifest: /system/volatile/ai.xml
13:48:35    Using profile specification: /system/volatile/profile
13:48:35    Using service list file: /var/run/service_list
13:48:35    Starting installation.
13:48:35    0% Preparing for Installation
```

```
13:48:36    100% manifest-parser completed.
13:48:36    0% Preparing for Installation
13:48:36    1% Preparing for Installation
13:48:37    2% Preparing for Installation
13:48:37    3% Preparing for Installation
13:48:37    4% Preparing for Installation
13:48:47    7% target-discovery completed.
13:48:47    === Executing Target Selection Checkpoint ==
13:48:48    Selected Disk(s) : c3t0d0
13:48:48    13% target-selection completed.
13:48:48    17% ai-configuration completed.
13:49:01    19% target-instantiation completed.
13:49:02    19% Beginning IPS transfer
13:49:02    Creating IPS image
13:50:58    Installing packages from:
13:50:58        solaris
13:50:58            origin:  http://pkg.example.com/solaris/
14:48:40    21% generated-transfer-1491-1 completed.
14:48:41    23% initialize-smf completed.
14:48:43    Installing SPARC bootblk to root pool devices: ['/dev/rdsk/c3t0d0s0']
14:48:43    Setting openprom boot-device
14:48:44    33% boot-configuration completed.
14:48:44    35% update-dump-adm completed.
14:48:45    37% setup-swap completed.
14:48:45    40% set-flush-ips-content-cache completed.
14:48:47    42% device-config completed.
14:48:49    44% apply-sysconfig completed.
14:48:49    46% transfer-zpool-cache completed.
14:49:03    87% boot-archive completed.
14:49:04    89% transfer-ai-files completed.
14:49:04    99% create-snapshot completed.
14:49:05    Automated Installation succeeded.
14:49:05    You may wish to reboot the system at this time.
Automated Installation finished successfully
The system can be rebooted now
Please refer to the /system/volatile/install_log file for details
After reboot it will be located at /var/sadm/system/logs/install_log
```

Installing an x86 Client

Initiate the x86 client installation by using one of the following methods to boot from the network:

- Press the appropriate function key. For example, some systems use F12 to boot from the network
- Change the boot order in the BIOS.

When the client boots, select the network device to boot from.

x86 Client Network Boot Sequence

The following events occur during AI boot of an x86 client:

1. The client boots and gets an IP address, and the boot file, `pxegrub`, is downloaded from the location provided by the DHCP server.
2. The `pxegrub` boot file is loaded and reads a `menu.lst` file.
3. The `pxegrub` boot file gets the `boot_archive` file and the Oracle Solaris OS is booted using TFTP.
4. The net image archives, `solaris.zlib` and `solarismisc.zlib`, are downloaded using HTTP as provided by the GRUB menu.
5. The AI manifest and system configuration profiles are downloaded from an AI install service specified from an mDNS lookup or from the GRUB menu entry that was booted.
6. The AI install program is invoked with the AI manifest to perform the installation.

Sample x86 Network Boot Output

When the system has successfully PXE booted, the following message is briefly displayed before the GRUB menu is displayed:

```
Intel(R) Boot Agent PXE Base Code (PXE-2.1 build 0.86)
Copyright(C) 1997-2007, Intel Corporation

CLIENT MAC ADDR 00 14 4F 29 04 12 GUID FF2000008 FFFF FFFF FFFF 7BDA264F1400
CLIENT IP: 10.6.68.29   MASK: 255.255.255.0    DHCP IP:  10.6.68.49
GATEWAY: 10.6.68.1
```

The GRUB menu appears with two menu entries. Select the second entry to start an automated installation:

```
Oracle Solaris 11 11/11 Text Installer and command line
Oracle Solaris 11 11/11 Automated Install
```

The default GRUB menu entry, "Text Installer and command line," boots the image without starting a hands-free automated installation. Select the second entry in the GRUB menu, "Automated Install," to initiate an automated installation. If you select the first menu entry, then when the client is booted, a menu displays as shown in "Start Installation After Booting Without Initiating an Installation" on page 202. Use this menu to examine or install the system.

Once the image is selected, the following messages are displayed:

```
Remounting root read/write snv_175 64-bit
Probing for device nodes ...acle and/or its affiliates. All rights reserved.
Preparing network image for use
Downloading solaris.zlib
--2011-08-26 07:35:13--  http://10.80.238.5:5555//install/images/i386_snv175/solaris.zlib
Connecting to 10.80.238.5:5555... connected.
```

```
HTTP request sent, awaiting response... 200 OK
Length: 130032128 (124M) [text/plain]
Saving to: ‘/tmp/solaris.zlib’

100%[======================================>] 130,032,128 16.4M/s   in 8.3s

2011-08-26 07:35:22 (14.9 MB/s) - ‘/tmp/solaris.zlib’ saved [130032128/130032128]

Downloading solarismisc.zlib
--2011-08-26 07:35:22--  http://10.80.238.5:5555//install/images/i386_snv175/solarismisc.zlib
Connecting to 10.80.238.5:5555... connected.
HTTP request sent, awaiting response... 200 OK
Length: 18758144 (18M) [text/plain]
Saving to: ‘/tmp/solarismisc.zlib’

100%[======================================>] 18,758,144  21.7M/s   in 0.8s

2011-08-26 07:35:23 (21.7 MB/s) - ‘/tmp/solarismisc.zlib’ saved [18758144/18758144]

Downloading .image_info
--2011-08-26 07:35:23--  http://10.80.238.5:5555//install/images/i386_snv175/.image_info
Connecting to 10.80.238.5:5555... connected.
HTTP request sent, awaiting response... 200 OK
Length: 241 [text/plain]
Saving to: ‘/tmp/.image_info’

100%[======================================>] 241         --.-K/s   in 0s

2011-08-26 07:35:23 (29.3 MB/s) - ‘/tmp/.image_info’ saved [241/241]

Done mounting image
Configuring devices.
Hostname: solaris
Service discovery phase initiated
Service name to look up: default-i386
Service discovery finished successfully
Process of obtaining install manifest initiated
Using the install manifest obtained via service discovery

Automated Installation started
The progress of the Automated Installation will be output to the console
Detailed logging is in the logfile at /system/volatile/install_log
Press RETURN to get a login prompt at any time.

solaris console login: 07:35:35    Install Log: /system/volatile/install_log
07:35:35    Using XML Manifest: /system/volatile/ai.xml
07:35:35    Using profile specification: /system/volatile/profile
07:35:35    Using service list file: /var/run/service_list
07:35:36    Starting installation.
07:35:36    0% Preparing for Installation
07:35:36    100% manifest-parser completed.
07:35:36    0% Preparing for Installation
07:35:36    1% Preparing for Installation
07:35:36    2% Preparing for Installation
07:35:36    3% Preparing for Installation
07:35:37    4% Preparing for Installation
07:35:51    7% target-discovery completed.
07:35:51    === Executing Target Selection Checkpoint ==
```

```
07:35:51    Selected Disk(s) : c7t0d0
07:35:51    13% target-selection completed.
07:35:51    17% ai-configuration completed.
07:36:23    19% target-instantiation completed.
07:36:23    19% Beginning IPS transfer
07:36:23    Creating IPS image
07:36:41    Installing packages from:
07:36:41        solaris
07:36:41            origin:  http://pkg.example.com/solaris/
07:53:29    21% generated-transfer-1006-1 completed.
07:53:29    23% initialize-smf completed.
07:53:30    Setting console boot device property to ttya
07:53:30    Disabling graphical console in boot loader
07:53:30    Creating Legacy GRUB config directory:
       /rpool/boot/grub
07:53:30    Installing boot loader to devices: ['/dev/rdsk/c7t0d0s0']
07:53:31    33% boot-configuration completed.
07:53:31    35% update-dump-adm completed.
07:53:31    37% setup-swap completed.
07:53:31    40% set-flush-ips-content-cache completed.
07:53:32    42% device-config completed.
07:53:46    44% apply-sysconfig completed.
07:53:46    46% transfer-zpool-cache completed.
07:54:08    87% boot-archive completed.
07:54:08    89% transfer-ai-files completed.
07:54:09    99% create-snapshot completed.
07:54:09    Automated Installation succeeded.
07:54:09    You may wish to reboot the system at this time.
Automated Installation finished successfully
The system can be rebooted now
Please refer to the /system/volatile/install_log file for details
After reboot it will be located at /var/sadm/system/logs/install_log
```

Client Installation Messages

The following messages are common to both SPARC and x86 installations.

Automated Installation Started Message

If the client is able to successfully boot and download the install files, then the following message is displayed:

```
Automated Installation started
The progress of the Automated Installation will be output to the console
Detailed logging is in the logfile at /system/volatile/install_log
Press RETURN to get a login prompt at any time.
```

You can login as `root` with the password `solaris` to monitor the installation messages in `/system/volatile/install_log`. Once the installation of packages from IPS has started, you might not see updates to this log file for an extended period.

Automated Installation Succeeded Message

If you see the following message, the installation is successful:

```
Automated Installation finished successfully
The system can be rebooted now
Please refer to the /system/volatile/install_log file for details
After reboot it will be located at /var/sadm/system/logs/install_log
```

If you have set up automatic reboot in the AI manifest, the system reboots at this time. To specify automatic reboot after successful installation, set the `auto_reboot` attribute of the `<ai_instance>` tag to `true`. The default value is `false`: The client does not automatically reboot after successful installation.

◆ ◆ ◆ CHAPTER 16

Troubleshooting Automated Installations

This chapter discusses some possible failures and how to recover.

Client Installation Fails

This section suggests actions to take if a client installation fails.

Check the Installation Logs and Instructions

If an installation to a client system failed, you can find the log at `/system/volatile/install_log`.

The AI manifest that was used for this client is in `/system/volatile/ai.xml`. The system configuration profiles that were used for this client are in `/system/volatile/profile/*`.

Check DNS

Check whether DNS is configured on your client by verifying that a non-empty `/etc/resolv.conf` file exists.

If `/etc/resolv.conf` does not exist or is empty, check that your DHCP server is providing DNS server information to the client:

```
# /sbin/dhcpinfo DNSserv
```

If this command returns nothing, the DHCP server is not set up to provide DNS server information to the client. Contact your DHCP administrator to correct this problem.

If an `/etc/resolv.conf` file exists and is properly configured, check for the following possible problems and contact your system administrator for resolution:

- The DNS server might not be resolving your IPS repository server name.
- No default route to reach the DNS server exists.

Check Client Boot Errors

Review the following additional information about errors that occur when the client system is booting.

- "SPARC Network Booting Errors and Possible Causes" on page 194
- "x86 Network Booting Errors and Possible Causes" on page 197
- "SPARC and x86 Error Messages" on page 199

SPARC Network Booting Errors and Possible Causes

This section describes errors or problems you might see when booting a SPARC client over the network and possible causes.

- "Timed out Waiting for BOOTP/DHCP Reply" on page 194
- "Boot Load Failed" on page 194
- "Internal Server Error or WAN Boot Alert" on page 195
- "Error Message 403: Forbidden or 404 Not Found" on page 195
- "Automated Installer Disabled" on page 196

Timed out Waiting for BOOTP/DHCP Reply

If a DHCP server is not responding to a SPARC client's request, the following messages display:

```
...
OpenBoot 4.23.4, 8184 MB memory available, Serial #69329298.
Ethernet address 0:14:4f:21:e1:92, Host ID: 8421e192.
Rebooting with command: boot net:dhcp - install
Boot device: /pci@7c0/pci@0/network@4:dhcp  File and args:
1000 Mbps FDX Link up
Timed out waiting for BOOTP/DHCP reply
Timed out waiting for BOOTP/DHCP reply
Timed out waiting for BOOTP/DHCP reply
Timed out waiting for BOOTP/DHCP reply
```

The timeout message indicates that the client is sending a DHCP request, and no response has been made to that request. This error is probably caused by a DHCP configuration problem. Check whether your client is configured correctly in the DHCP server.

Boot Load Failed

If the AI client starts downloading the `boot_archive`, but then fails with the error, "Boot load failed," that indicates that the client DHCP information is configured incorrectly.

```
Rebooting with command: boot net:dhcp - install
   Boot device: /pci@7c0/pci@0/network@4:dhcp  File and args:
   1000 Mbps FDX Link up
   HTTP: Bad Response: 500 Internal Server Error
   Evaluating:

   Boot load failed
```

This error could happen if another DHCP server is responding to the client. Check the DHCP configuration for this client. If the configuration appears to be correct, determine whether there is another DHCP server in the subnet.

Internal Server Error or WAN Boot Alert

After the AI client has obtained the IP address and initial parameters to start downloading the boot archive, the client might be unable to find or download the `boot_archive`.

- If the client cannot find the `boot_archive`, the following error is displayed.

```
Rebooting with command: boot net:dhcp - install
      Boot device: /pci@7c0/pci@0/network@4:dhcp  File and args:
      1000 Mbps FDX Link up
      <time unavailable> wanboot info: WAN boot messages->console
      <time unavailable> wanboot info: Starting DHCP configuration
      <time unavailable> wanboot info: DHCP configuration succeeded
      <time unavailable> wanboot progress: wanbootfs: Read 366 of 366 kB (100%)
      <time unavailable> wanboot info: wanbootfs: Download complete
      Tue Aug  5 20:46:43 wanboot alert: miniinfo: Request returned code 500
      Tue Aug  5 20:46:44 wanboot alert: Internal Server Error \
(root filesystem image missing)
```

- If the AI client finds the `boot_archive` file but cannot access the file, then the following error is displayed.

```
Rebooting with command: boot net:dhcp - install
      Boot device: /pci@7c0/pci@0/network@4:dhcp  File and args:
      1000 Mbps FDX Link up
      <time unavailable> wanboot info: WAN boot messages->console
      <time unavailable> wanboot info: Starting DHCP configuration
      <time unavailable> wanboot info: DHCP configuration succeeded
      <time unavailable> wanboot progress: wanbootfs: Read 366 of 366 kB (100%)
      <time unavailable> wanboot info: wanbootfs: Download complete
      Tue Aug  5 20:53:02 wanboot alert: miniroot: Request returned code 403
      Tue Aug  5 20:53:03 wanboot alert: Forbidden
```

For both of these problems, fix the `boot_archive` file configured for this client. Check the path name and permissions of the `boot_archive` at `$IMAGE/boot/boot_archive`.

Error Message 403: Forbidden or 404 Not Found

These messages, “ERROR 403: Forbidden” and “ERROR 404: Not Found” are displayed if the AI client successfully downloads the `boot_archive` and boots the Oracle Solaris kernel, but fails to get one of the image archives. An error message is displayed indicating which file is causing the problem. For example, in the following output on a SPARC client, the `solaris.zlib` file does not exist or is not accessible at the specified location.

```
<time unavailable> wanboot info: Starting DHCP configuration
<time unavailable> wanboot info: DHCP configuration succeeded
<time unavailable> wanboot progress: wanbootfs: Read 368 of 368 kB (100%)
<time unavailable> wanboot info: wanbootfs: Download complete
Fri Aug 26 16:26:52 wanboot progress: miniroot: Read 221327 of 221327 kB (100%)
Fri Aug 26 16:26:53 wanboot info: miniroot: Download complete
Warning: 'SUNW,UltraSPARC-IIIi' support will be removed in a future release of Solaris.
SunOS Release 5.11 Version snv_175 64-bit
Copyright (c) 1983, 2011, Oracle and/or its affiliates. All rights reserved.
WARNING: i2c_0 failed to add interrupt.
WARNING: i2c_0 operating in POLL MODE only

Hardware watchdog enabled
Remounting root read/write
Probing for device nodes ...
Preparing network image for use
Downloading solaris.zlib
--2011-08-26 23:19:57--  http://10.134.125.136:5555/export/auto_install/175s//solaris.zlib
Connecting to 10.134.125.136:5555... connected.
HTTP request sent, awaiting response... 404 Not Found
2011-08-26 23:19:57 ERROR 404: Not Found.

Could not obtain http://10.134.125.136:5555/export/auto_install/175s//solaris.zlib from install server
Please verify that the install server is correctly configured and reachable from the client
Requesting System Maintenance Mode
```

This problem can be caused by one of the following conditions.

- The image path configured in WAN boot is not correct.
- The image path does not exist or is incomplete.
- Access is denied due to permission issues.

Check your DHCP configuration or the contents of the target directory you specified when you ran `installadm create-service`. Check your WAN boot configuration.

Automated Installer Disabled

When installing the Oracle Solaris OS on your client system, you need to include the `install` argument when you boot, as follows, in order to initiate an installation.

```
ok boot net:dhcp - install
```

If you booted without the `install` boot argument, the SPARC client boots into the automated installer boot image, but the installation does not start. The following message is displayed.

```
Auto-installer disabled. Enable the auto-installer service
by running the following command:
svcadm enable svc:/application/auto-installer:default
```

To start an automated installation, you can log in and enable the install service as shown in the message, or you can reboot your system using the command shown above with the `install` argument.

x86 Network Booting Errors and Possible Causes

This section describes errors or problems you might see when booting an x86 client over the network and possible causes.

No DHCP or Proxy DHCP Offers Were Received

If a DHCP server is not responding to an x86 client's request, you see the following messages:

```
Intel(R) Boot Agent PXE Base Code (PXE-2.1 build 0.86)
   Copyright(C) 1997-2007, Intel Corporation

   CLIENT MAC ADDR 00 14 4F 29 04 12 GUID FF2000008 FFFF FFFF FFFF 7BDA264F1400
   DHCP......... No DHCP or ProxyDHCP offers were received
   PXE-MOF: Exiting Intel Boot Agent
```

The timeout message indicates that the client is sending a DHCP request and not getting a response. This issue is probably due to an error in the DHCP configuration. Check to see if your client is configured correctly in the DHCP server.

TFTP Error or System Hangs After GATEWAY Message

The DHCP server provides an IP address and a location of the initial boot program as part of the DHCP response.

- If the boot program does not exist, then the AI client boot cannot proceed. The following message is displayed:

```
Intel(R) Boot Agent PXE Base Code (PXE-2.1 build 0.86)
     Copyright(C) 1997-2007, Intel Corporation

     CLIENT MAC ADDR 00 14 4F 29 04 12 GUID FF2000008 FFFF FFFF FFFF 7BDA264F1400
     CLIENT IP: 10.6.68.29   MASK: 255.255.255.0    DHCP IP:  10.6.68.49
     GATEWAY: 10.6.68.1
     TFTP.
     PXE-T02:    Access Violation
     PXE-E3C: TFTP Error - Access violation
     PXE-MOF: Exiting Intel Boot Agent
```

- If the boot program exists, but it is an incorrect program, the AI client hangs after displaying this message:

```
Intel(R) Boot Agent PXE Base Code (PXE-2.1 build 0.86)
     Copyright(C) 1997-2007, Intel Corporation

     CLIENT MAC ADDR 00 14 4F 29 04 12 GUID FF2000008 FFFF FFFF FFFF 7BDA264F1400
     CLIENT IP: 10.6.68.29   MASK: 255.255.255.0    DHCP IP:  10.6.68.49
     GATEWAY: 10.6.68.1
```

System Hangs After GRUB Menu Entry is Selected

If the client is able to do the initial boot, but the kernel cannot be booted, the system hangs after you select the entry from the GRUB menu.

On the install server, check whether the `menu.lst` file for this client is pointing to a valid boot archive. The boot directory of the image on the server should be loopback mounted under the `/etc/netboot` directory as shown in this sample excerpt from `df -k`:

```
/install/images/x86_snv175 \
28046887     8432439    19614448    31%    /etc/netboot/x86_snv175
```

If you know the name of the target directory that you used in the `installadm create-service` command, then you can use that information to determine whether that target directory is mounted.

HTTP Request Sent Results in 403 Forbidden or 404 Not Found

On the install server, if one of the install programs is inaccessible or does not exist in the location specified in the `menu.lst` file under `/etc/netboot`, then the client is able to boot, but is not able to download that file. An error message is displayed indicating which file is causing the problem. For example, in the following output on an x86 client, the `solaris.zlib` file does not exist at the specified location.

```
SunOS Release 5.11 Version snv_175 64-bit
Copyright (c) 1983, 2011, Oracle and/or its affiliates. All rights reserved.
Remounting root read/write
Probing for device nodes ...
Preparing network image for use
Downloading solaris.zlib
--2011-08-18 20:02:26--  http://10.134.125.136:5555//export/auto_install/s11-x86/solaris.zlib
Connecting to 10.134.125.136:5555... connected.
HTTP request sent, awaiting response... 404 Not Found
2011-08-18 20:02:26 ERROR 404: Not Found.

Could not obtain http://10.134.125.136:5555//export/auto_install/s11-x86/solaris.zlib from install server
Please verify that the install server is correctly configured and reachable from the client

Requesting System Maintenance Mode
(See /lib/svc/share/README for more information.)
Console login service(s) cannot run
```

Check the contents of the target directory that you specified when you ran the `installadm create-service` command.

Automated Installer Disabled

When installing the Oracle Solaris OS on x86 client systems for installations that boot over the network, you must select the second entry in the GRUB boot menu to initiate an automated installation. Typically, the menu entries display as follows:

```
Oracle Solaris 11 11/11 Text Installer and command line
Oracle Solaris 11 11/11 Automated Install
```

If you selected the first GRUB menu entry, or allowed the prompt to time out, the system boots into the automated install boot image, but the installation does not start. The following message is displayed:

```
Auto-installer disabled. Enable the auto-installer service
by running the following command:
svcadm enable svc:/application/auto-installer:default
```

To start an automated installation, you can log in and enable the install service as shown in the message, or you can reboot your system and select the second menu entry.

SPARC and x86 Error Messages

The following errors are common to both SPARC and x86 installations.

- "Automated Installation Failed Message" on page 199
- "Unable To Contact Valid Package Server" on page 199
- "Package Not Found" on page 201

Automated Installation Failed Message

If there is a failure during installation, then the following message is displayed:

```
Automated Installation failed. Please refer to /system/volatile/install_log file
   for details
   Apr  9 14:28:09 solaris svc.startd[7]: application/auto-installer:default
   failed fatally: transitioned to maintenance (see 'svcs -xv' for details)
```

Unable To Contact Valid Package Server

The installation client needs to reach the IPS package repository defined in the AI manifest in order to install the Oracle Solaris OS. If the client cannot access the package repository, the installation fails and the `application/auto-installer` service transitions to maintenance. The following output is an example of what is displayed on the console:

```
15:54:46    Creating IPS image
15:54:46    Error occurred during execution of 'generated-transfer-1341-1' checkpoint.
15:54:47    Failed Checkpoints:
15:54:47
15:54:47        generated-transfer-1341-1
15:54:47
15:54:47    Checkpoint execution error:
15:54:47
15:54:47        Framework error: code: 6 reason: Couldn't resolve host 'pkg.example.com'
15:54:47        URL: 'http://pkg.example.com/solaris/release/versions/0/'.
15:54:47
15:54:47    Automated Installation Failed.  See install log at /system/volatile/install_log
Automated Installation failed
```

```
Please refer to the /system/volatile/install_log file for details
Aug 31 15:54:47 line2-v445 svc.startd[8]: application/auto-installer:default failed fatally:
transitioned to maintenance (see 'svcs -xv' for details)
...
SUNW-MSG-ID: SMF-8000-YX, TYPE: defect, VER: 1, SEVERITY: major
EVENT-TIME: Wed Aug 31 15:54:47 UTC 2011
PLATFORM: SUNW,Sun-Fire-V445, CSN: -, HOSTNAME: line2-v445
SOURCE: software-diagnosis, REV: 0.1
EVENT-ID: c8a5b809-ece4-4399-9646-d8c64d78aac7
DESC: A service failed - a start, stop or refresh method failed.
AUTO-RESPONSE: The service has been placed into the maintenance state.
IMPACT: svc:/application/auto-installer:default is unavailable.
REC-ACTION: Run 'svcs -xv svc:/application/auto-installer:default' to determine the generic reason
why the service failed, the location of any logfiles, and a list of other services impacted. Please
refer to the associated reference document at http://sun.com/msg/SMF-8000-YX for the latest service
procedures and policies regarding this diagnosis.
```

Check the `/system/volatile/install_log` file for messages similar to the following messages:

```
TransportFailures: Framework error: code: 6 reason: Couldn't resolve host
'pkg.example.com'
URL: 'http://pkg.example.com/solaris/versions/0/'

TransportFailures: Framework error: code: 7 reason: Failed connect to
pkg.example.com:80; Connection refused
URL: 'http://pkg.example.com/solaris/versions/0/'

TransportFailures: http protocol error: code: 404 reason: Not Found
URL: 'http://pkg.oracle.com/mysolaris/versions/0/'
```

Depending on which messages you see, try the following possible remedies:

- Try to reach the package server from the failed client system, for example by using `ping`(1M).
- If you are using DNS, check whether DNS is correctly configured on the AI client. See "Check DNS" on page 193.
- If you are using a local repository, check whether you have made the repository accessible to all clients. See Chapter 3, "Providing Access To Your Repository," in *Copying and Creating Oracle Solaris 11 Package Repositories*.
- Make sure the URI in the AI manifest does not have a typographical error.
- Use a command such as the following command to check whether the package repository is valid:

  ```
  $ pkg list -g http://pkg.example.com/solaris/ entire
  ```

  You might need to refresh the catalog or rebuild the index.

Package Not Found

If one of the packages specified in the AI manifest cannot be located in the IPS repositories, then the installer fails before installing any packages on the disk. In the following example, the installer could not find the package `entirity` in the IPS repository. The following output is an example of what is displayed on the console:

```
14:04:02    Failed Checkpoints:
14:04:02
14:04:02        ips
14:04:02
14:04:02    Checkpoint execution error:
14:04:02
14:04:02        The following pattern(s) did not match any allowable packages.  Try
14:04:02        using a different matching pattern, or refreshing publisher information:
14:04:02
14:04:02                pkg:/entirity
14:04:02
14:04:02    Automated Installation Failed.  See install log at /system/volatile/install_log
```

The following output is an example of a portion of the `/system/volatile/install_log` log file:

```
PlanCreationException: The following pattern(s) did not match any allowable packages.
Try using a different matching pattern, or refreshing publisher information:

pkg:/entirity
```

Check whether the package in question is a valid package. If this package is available from a different IPS repository, add that IPS repository in the AI manifest by adding another `publisher` element to the `source` element.

Boot the Installation Environment Without Starting an Installation

Use one of the following methods to boot the installation environment without starting an automated installation. When the client is booted, a menu displays as shown in "Start Installation After Booting Without Initiating an Installation" on page 202. Use this menu to examine or install the system.

SPARC client booting over the network
: Use the following command to boot a SPARC client over the network without starting an automated installation:

  ```
  ok boot net: dhcp
  ```

  Do not specify the `install` flag as a boot argument.

SPARC client booting from media
: Use the following command to boot a SPARC client from media without starting an installation:

```
ok boot cdrom
```

  Do not specify the `install` flag as a boot argument.

x86 client booting over the network
: For x86 installations that boot over the network, the following GRUB menu displays:

```
Oracle Solaris 11 11/11 Text Installer and command line
Oracle Solaris 11 11/11 Automated Install
```

  The default entry, "Text Installer and command line," boots the image without starting a hands-free automated installation.

  Make sure the entry does not have the `install=true` boot property specified in its kernel line.

x86 client booting from media
: If you boot an x86 system from media and do not want to start an installation, edit the GRUB menu and remove the `install=true` boot property from the kernel line of the entry you want to boot.

In general for x86 installations, if the `install=true` boot property is specified in the kernel line of the GRUB entry you are booting from, the installation automatically starts. If you want to boot your x86 based system without initiating an automated installation, check that the GRUB boot entry does not specify the `install=true` boot property. If the property is specified, edit the kernel line of boot entry and remove the property.

Start Installation After Booting Without Initiating an Installation

If you selected a boot option that does not initiate an installation, then the following menu displays:

```
1  Install Oracle Solaris
2  Install Additional Drivers
3  Shell
4  Terminal type (currently sun)
5  Reboot
```

This menu does not have a default selection.

Select option 3 to open a shell.

Use the following commands to start an automated installation:

```
# svcadm enable manifest-locator:default
# svcadm enable svc:/application/auto-installer:default
```