Installing Oracle® Solaris 11.1 Systems

Part No: E28980–01
October 2012

ORACLE®

121203@25097

Contents

Figures

Tables

Examples

Preface

Installing Oracle Solaris 11.1 Systems provides instructions for installing and configuring the Oracle Solaris operating system (OS) using any of the following methods:

- Oracle Solaris Live Media installer
- Oracle Solaris interactive text installer
- Oracle Solaris Automated Installer (AI) feature
- Oracle Solaris SCI Tool interactive system configuration tool
- `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 11.1 OS.

How This Book Is Organized

This book contains the following parts and chapters:

Part I, "Oracle Solaris 11.1 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 Live Media"
- 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, "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 15, "Troubleshooting Automated Installations," discusses some possible failures and how to recover.

Part IV, "Performing Related Tasks," covers areas related to installation.

- Appendix A, "Working With Oracle Configuration Manager"
- Appendix B, "Using the Device Driver Utility"

Related Information

Creating a Custom Oracle Solaris 11.1 Installation Image explains how to use the Oracle Solaris Distribution Constructor tool to customize your installation image.

Creating and Administering Oracle Solaris 11.1 Boot Environments describes how to manage multiple boot environments on your Oracle Solaris system, including non-global zones.

Managing Services and Faults in Oracle Solaris 11.1 describes the Oracle Solaris Service Management Facility (SMF) feature. You can use SMF profiles to configure your system.

Adding and Updating Oracle Solaris 11.1 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 11.1 Package Repositories describes how to create a local copy of an Oracle Solaris IPS package repository, or how to create your own custom repository.

See the Oracle Solaris 11.1 System Administration documentation for more information about how to administer Oracle Solaris 11.1 systems.

For information about configuring DHCP, see the following resources:

- *Working With DHCP in Oracle Solaris 11.1*
- The `dhcpd.conf`(5) man page
- The DHCP section of the Internet Systems Consortium (ISC) web site

Transitioning From Oracle Solaris 10 JumpStart to Oracle Solaris 11.1 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	Description	Example
`AaBbCc123`	The names of commands, files, and directories, and onscreen computer output	Edit your `.login` file. Use `ls -a` to list all files. `machine_name% you have mail.`

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

aabbcc123	Placeholder: replace with a real name or value	The command to remove a file is `rm` *filename*.

TABLE P–1 Typographic Conventions *(Continued)*

Typeface	Description	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#`

P A R T I

Oracle Solaris 11.1 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. This chapter describes the 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 Live Media"	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 58	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 11.1 Installation Image*, including custom Live Media images, text installer images, and automated installation images.

Simple, Preset Installations

The GUI installer on the Live Media 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 11.1 Release Notes*.
- Both installers enable you to select, create, or modify disk partitions during an installation.

The Live Media contains a set of software that is appropriate for a desktop or laptop. The text installer installs 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 enables you to further customize the installation specifications for any particular system.

  For further information, see "Performing a Text Installation Over the Network" on page 58.
- 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 11.1 Installation Image*.
Updating an Installed Oracle Solaris System	You cannot use an installer to update an existing Oracle Solaris installed system. Instead, you need to use the `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 11.1 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 information:

- Chapter 2, "Preparing for the Installation"
- Chapter 3, "Using Live Media"
- Chapter 4, "Using the Text Installer"
- "Performing a Text Installation Over the Network" on page 58
- Chapter 5, "Automated Installations That Boot From Media"
- Chapter 6, "Unconfiguring or Reconfiguring an Oracle Solaris instance"

◆ ◆ ◆ CHAPTER 2

Preparing for the Installation

Before installing your system, review the following information:

System Requirements for Live Media and Text Installations

The following table outlines requirements for installing the Oracle Solaris 11.1 release using a Live Media installation image or a text installation image.

Requirement	Description
Memory	To check the minimum memory requirements for the current release, see *Oracle Solaris 11.1 Release Notes*. **Note** – The text installer requires less memory than the Live Media installer. The exact minimum requirement varies depending on system specifications. 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 11.1 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. In this release, Oracle Solaris for the x86 platform uses the new version of the GRand Unified Bootloader (GRUB 2). Oracle Solaris recognizes Windows and ensures that Windows partitions remain unchanged during an installation. When the installation completes, and the system reboots, the GRUB 2 menu displays both the Windows and the Oracle Solaris boot entries. For more information about GRUB 2, see "Introducing GRUB 2" in *Booting and Shutting Down Oracle Solaris 11.1 Systems*.
Solaris 10 OS	The Live Media installer on the cannot be used to install multiple instances of the Oracle Solaris operating system. 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 Live Media 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.
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 Live Media 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.

The installer uses GPT formatting when installing on to a whole disk or an unformatted disk. However, existing GPT partitions or DOS partitions are retained by default and displayed by the installer, so you can retain and install into an existing partition.

Caution – GPT formatting is currently not available on SPARC platforms.

This section also describes how to set up Solaris VTOC slices.

Guidelines for Partitioning a System Prior To Installation

When installing Oracle Solaris from the Live Media ISO image or from the text installer image, you can use the entire disk, or you can install the operating system on a partition. In addition, the text installer can be installed on a slice.

You can create a partition for installing Oracle Solaris prior to installation using commercial products or open-source tools. Or, you can create a partition during the Oracle Solaris installation. On x86 based systems, the Oracle Solaris installers use GRUB 2, which supports booting multiple operating systems on one or more drives. After partitioning and installing the various operating systems, you can deploy any of them by selecting the appropriate menu entry in the GRUB 2 menu at boot time.

For more information about GRUB 2, see "Introducing GRUB 2" in *Booting and Shutting Down Oracle Solaris 11.1 Systems*.

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.

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

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. The installer uses GPT formatting when installing on to a whole disk or an unformatted disk. However, existing GPT partitions or DOS partitions are retained by default and displayed by the installer, so you can retain and install into an existing partition. In addition, for the text installer *only*, you can select, create, or modify VTOC slices during an interactive installation.

Caution – GPT formatting is currently not available on SPARC platforms.

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

- Note the following partitioning specifications:
  - If the disk contains existing DOS partitions, up to four DOS primary partitions are displayed. If a DOS extended partition exists, its logical partitions are also displayed in the disk layout order within the extended partition. Only one Solaris partition is allowed, and that Solaris partition must be used for the installation. The Solaris partition can be a logical partition within an extended partition.
  - If the disk contains existing GPT partitions, the GPT partitions are displayed. Up to seven GPT partitions are supported. You can create one or more Solaris partitions during the installation, but you must choose one Solaris partition as the installation target. If there are multiple, existing Solaris GPT partitions, the first suitable Solaris GPT partition will be chosen by default as the installation target.
- 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 there is an existing Solaris partition and you make no modifications to any of the other existing partitions, the installation default overwrites the Solaris partition *only*. That partition can be a logical partition within an existing extended partition. Other existing partitions are not changed.
- A Solaris partition must be used for the installation.
- 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 partitions for the selected disk in the same order that they are laid out on the disk. Unused disk space is displayed for these 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.
- 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 minimum and recommended 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 the Oracle Solaris operating system on the existing Solaris partition using its current size. Select the Partition a Disk option. No other changes are required.
If no Solaris partition exists, you must create a new Solaris partition.	If there is currently no existing Solaris partition on the system, you must 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.

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

Partitioning Option	Description and User Action (if required)
Install the Oracle Solaris operating system on a different Solaris partition.	You can install the operating system on a different Solaris partition. 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. **Note –** If the system has existing DOS partitions, only one Solaris partition is allowed. You must first change the existing Solaris partition type to Unused before you create a new Solaris partition.
Create a new Solaris partition within an extended partition.	You can create a new Solaris partition within an 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. **Note –** If the system has existing DOS partitions, only one Solaris partition is allowed. You must first change the existing Solaris partition type to Unused before you create a Solaris partition within an extended partition.
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 a 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 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`.
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 the Oracle Solaris OS, you need to determine whether your system's devices are supported. Review the Hardware Compatibility Lists (HCL) at `http://www.oracle.com/webfolder/technetwork/hcl/index.html`. The HCL provide information about hardware that is certified or reported to work with the Oracle Solaris operating system.

You can, also, use the **Oracle Device Detection Tool** before or after an installation to determine whether a device driver is available. 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 Oracle Device Detection Tool, see "How to Use the Oracle Device Detection Tool" on page 36.

Note – After an installation, you can use the Device Driver Utility to perform the similar tasks. For more information about the Device Driver Utility, see Appendix B, "Using the Device Driver Utility."

▼ How to Use the Oracle Device Detection Tool

Before or after you perform an installation, you can use the Oracle Device Detection Tool as follows 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.

Using Oracle Configuration Manager

In this Oracle Solaris release, during an interactive installation, you will be prompted to configure the Oracle Configuration Manager and the Oracle Auto Service Request utilities for your installed system if those services are going to be installed on your system.

- Oracle Configuration Manager sends periodic data to the Oracle support organization describing a system's software configuration.
- Oracle Auto Service Request sends data to the Oracle support organization when a Fault Management Architecture (FMA) event occurs, indicating a hardware or software issue.

Note – All Data is transmitted in secure mode.

When performing an interactive installation, you have the following options.

- The default Support Registration installer panel provides an anonymous registration address. If you use this anonymous address or another email address with no password, anonymous system configuration will be uploaded to the Oracle support organization. But, because you did not provide your My Oracle Support login information, My Oracle Support will not receive any of your identifying customer information when information is sent about the installed system's configuration.
- You can replace the anonymous email address in the Support Configuration panel with your My Oracle Support login ID and add your My Oracle Support password. Use this option if you want to see your customer information in My Oracle Support and receive security updates. With this option, Oracle Auto Service Request will also be started.

  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 delete the anonymous email address in the Support Configuration panel and leave that field blank, Oracle Configuration Manager will be started in a disconnected mode. No data will be sent to My Oracle Support. In this mode, the Oracle Configuration Manager can still be manually activated in order to send data. For example, if you are asked by a tech support person to provide data on your system, you could manually use Oracle Configuration Manager to provide that data.

Unless Oracle Configuration Manager is in disconnected mode, during the first reboot, an Oracle Configuration Manager service runs and attempts to register the system with the registration server. If this registration succeeds, an upload of the 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*.

Regardless of whether you chose to allow the registration, you can still choose to register or re-register your system later with the Oracle Configuration Manager in order to facilitate future support.

You may choose to register or re-register in situations such as the following:

- You previously registered anonymously.
- You previously disconnected Oracle Configuration Manager.
- The My Oracle Support credentials could not be validated when they were entered because Oracle could not be contacted. For example, automatic registration was unable to complete due to a network proxy requirement.

You can register or re-register by using the `configCCR` utility (`/usr/sbin/configCCR`) in interactive mode. For example, run the following command to remove existing configuration specifications:

```
# /usr/lib/ocm/ccr/bin/configCCR -r
```

Then, run the following command to manually configure Oracle Configuration Manager:

```
# /usr/lib/ocm/ccr/bin/configCCR -a
```

After completing registration, you can enable the service as follows:

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

Once the service is enabled, the Oracle Configuration Manager client will be restarted when the system is rebooted.

For further information about Oracle Configuration Manager and Oracle Auto Service Request, see the following:

- Appendix A, "Working With Oracle Configuration Manager"
- `configCCR`(1M) man page
- *Oracle Configuration Manager Installation and Administration Guide*
- `http://www.oracle.com/support/policies.html`
- Oracle Auto Service Request documentation at `http://www.oracle.com/asr`

CHAPTER 3

Using Live Media

This chapter describes how to perform installations using a Live Media image.

Installing With the GUI installer

When installing Oracle Solaris software, consider the following information:

- See "System Requirements for Live Media and Text Installations" on page 29.
- The installer on the Live Media ISO image is for x86 platforms only.
- If you are installing Oracle Solaris on a system that will have more than one operating system installed in it, you can partition your disk during the installation process.

  Note the following:

  - The installer uses GPT formatting when installing onto a whole disk or an unformatted disk. However, existing GPT partitions or DOS partitions are retained by default and displayed by the installer, so you can retain and install into an existing partition. For more information, see "Guidelines for Partitioning a System During an Interactive Installation" on page 31.

  Caution – GPT formatting is currently not available on SPARC platforms.

  - In this release, Oracle Solaris for the x86 platform installs the new version of the GRand Unified Bootloader (GRUB 2). For information about GRUB 2, see "Introducing GRUB 2" in *Booting and Shutting Down Oracle Solaris 11.1 Systems*.

  If you prefer, you can use a third-party or open–source partitioning tool to create a new partition or make adjustments to pre–existing partitions prior to an installation. See "Guidelines for Partitioning a System Prior To Installation" on page 31.

- In this release, you can use the GUI installer to install the Oracle Solaris operating system onto an iSCSI target if the iSCSI target can act as a boot disk and if the system has the necessary support for iSCSI booting.

  If your system supports autodiscovery of iSCSI disks, the installer provides that option. Alternately, you can manually enter values to specify the iSCSI target in the installation screens.

  For further information, see the installation procedure in this chapter. Also, see the `iscsiadm`(1M) man page.
- The GUI installer cannot upgrade your operating system. However, after you have installed the Oracle Solaris operating system, 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 11.1 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 the GUI Installer

The default network and security settings used by the GUI installer on the Live Media 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 Prepare for a GUI Installation

Complete the following tasks before performing a GUI installation.

1 **If you do not have the Live Media, download the Live Media ISO image.**

To download the Oracle Solaris Live Media 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, 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.

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

a. **Verify that your system meets all of the necessary system requirements.**

See "System Requirements for Live Media and Text Installations" on page 29.

b. **Verify that you have all of the necessary device drivers.**

See "Ensuring That You Have the Proper Device Drivers" on page 35.

3 **Choose one of the following options for installing the Oracle Solaris operating system:**

- **If this installation will provide the only operating system on your system, see "Installing With the GUI installer" on page 39.**
- **If you are setting up an environment that supports the installation of multiple operating systems:**
  a. **Review the specifications in "Preparing a Boot Environment for Installing Multiple Operating Systems" on page 30.**
  b. **Back up your system.**
  c. **If you want to partition your system prior to the installation, see "Partitioning Your System" on page 30.**

▼ How to Perform a GUI Installation

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

On the Live Media, when the GRUB2 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 Live Media or your system does not have a graphics card, the system boots in console mode when you insert the Live Media. In this case, you cannot perform a GUI installation. See "What to Do If Your System Boots in Console Mode" on page 45.

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

2 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.

---

3 Install any missing drivers that are required for installation.

When you boot the Live Media, 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.

4 On the Live Media desktop, double-click the Install Oracle Solaris icon to start the GUI installer.

5 In the Welcome panel, select Next.

6 In the Disk Discovery panel, select the type of disk that you want the installer to discover.

- Local Disks – This is the default option for disks that are attached to the computer, including internal and external hard disks.
- iSCSI – If you want the installer to search for remote disks that are accessible over a network using the iSCSI standard, select this option. Additional fields display as follows:
  - Use DHCP autodiscovery – If your system supports autodiscovery of iSCSI disks, this option is enabled. Selecting this option populates the criteria fields with the values returned from autodiscovery. You can then select the "Specify search criteria" option to further refine these values.
  - Specify search criteria – You can select this option and manually provide the iSCSI search values.

Target IP	The IP address of the iSCSI target. Four numbers in the range 0-255 must be entered. The system at this IP address must be online and accessible from this system. These fields are mandatory.
LUN	The Logical Unit Number of the iSCSI device located at the provided IP address. The LUN is often a numerical value such as "0", "1", and so on. This field is optional.

Target Name	The name of the iSCSI target in iSCSI Qualified Name (IQN) format. This field is optional.
Port	The port number used in conjunction with the provided IP address for discovering the iSCSI device. The default value of "3260" is the port typically used for iSCSI. This field is optional.
Initiator Name	The initiator node name to be set for the iSCSI discovery session. For iSCSI booting, this field is hidden because the initiator node name cannot be modified. This field is optional.
Use CHAP	Select this option if you want to enter CHAP (Challenge-Handshake Authentication Protocol) authentication details.
Name	The CHAP name to be used for authentication. This field is optional.
Password	The CHAP secret value for authentication. If provided, this value must be between 12 and 16 characters long. This field is optional.

If you choose the iSCSI option, a delay might occur when you select Next while the details entered are validated. If the iSCSI LUN cannot be discovered, an error is displayed. You cannot proceed until the problem is resolved, either by entering valid criteria or by deselecting iSCSI.

7 **In the Disk Selection panel, if multiple installation targets are 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.**

The installer uses GPT formatting when installing onto a whole disk or an unformatted disk. However, existing GPT partitions or DOS partitions are retained by default and displayed by the installer, so you can retain and install into an existing partition.

Caution – GPT formatting is currently not available on SPARC platforms.

Note the following:

- If the disk contains existing DOS partitions, up to four DOS primary partitions are displayed. If a DOS extended partition exists, its logical partitions are also displayed in the disk layout order within the extended partition. Only one Solaris partition is allowed, and that Solaris partition must be used for the installation. The Solaris partition can be a logical partition within an extended partition.
- If the disk contains existing GPT partitions, the GPT partitions are displayed. Up to seven GPT partitions are supported. You can create one or more Solaris partitions during the installation, but you must choose one Solaris partition as the installation target. If there are multiple, existing Solaris GPT partitions, the first suitable Solaris GPT partition will be chosen by default as the installation target.

You have the option to modify the partition layout. For instructions, see the "Guidelines for Partitioning a System During an Interactive Installation" on page 31.

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 shows 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 **In the Support Configuration panels, you can accept the anonymous registration for OCM and ASR.**

The default Support Configuration installer panel provides an anonymous registration address. If you use this anonymous address with no password, My Oracle Support (MOS) will receive information about the installed system's configuration, but will not receive any of your customer information when the system configuration is uploaded to the Oracle support organization.

Alternately, you can register for security updates or disconnect OCM as follows:

- You can replace the anonymous email address in the Support Configuration panel with your My Oracle Support login ID and add your My Oracle Support password. Use this option if you want to see your customer information in My Oracle Support and receive security updates. With this option, ASR will also be started.
- If you delete the anonymous email address in the Support Configuration panel and leave that field blank, OCM will be started in a disconnected mode. No data will be sent to My Oracle Support. Or, if you delete the anonymous email address and replace it with another email address other than your MOS login ID, OCM will send data to Oracle support in an unauthenticated mode.

For further information, see "Using Oracle Configuration Manager" on page 36.

11 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.

12 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.

13 Review the installation logs.

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

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 Live Media 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 Live Media or your system does not have a graphics card, the system boots in console mode when you insert the Live Media. In this case, you cannot perform a GUI installation.

Your two alternatives are as follows:

- Use the text installer image instead of the Live Media 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 as described in "How to Install Oracle Solaris From the Live Media If Your System Boots in Console Mode" on page 46.

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 Live Media If Your System Boots in Console Mode

Before You Begin For this procedure, two networked systems are required: the system on which the Live Media was booted (target system) and a remote system from which the installation will be performed. Both systems must have network access. The two systems are not required to 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 Live Media, 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.**

Adding Software After Live Media Installation

To add software packages after you have installed the operating system, use the `pkg` commands as described in the `pkg`(1) man page. 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.

Note – Installing, updating, and uninstalling packages require increased privileges. See "Installation Privileges" in *Adding and Updating Oracle Solaris 11.1 Software Packages* for more information.

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.

```
$ pfexec pkg install --be-name new–BE–name package–name
```

This sample command includes 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 operating system, consider the following information:

- See "System Requirements for Live Media and Text Installations" on page 29.
- 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.

  Note the following:

  - The installer uses GPT formatting when installing onto a whole disk or an unformatted disk. However, existing GPT partitions or DOS partitions are retained by default and displayed by the installer, so you can retain and install into an existing partition. For more information, see "Guidelines for Partitioning a System During an Interactive Installation" on page 31.

  Caution – GPT formatting is currently not available on SPARC platforms.

  - In this release, the Oracle Solaris installers use GRUB 2 for x86 systems. GRUB 2 supports booting multiple operating systems on one or more drives. For information about GRUB 2, see "Introducing GRUB 2" in *Booting and Shutting Down Oracle Solaris 11.1 Systems*

  You also have the option to use an open–source or third-party partitioning tool to create a new partition or make adjustments to pre–existing partitions prior to an installation. See "Guidelines for Partitioning a System Prior To Installation" on page 31.

- The Oracle Solaris installers cannot upgrade your operating system. However, after you have installed the Oracle Solaris operating system, 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 11.1 Software Packages*.
- In this release, you can use the text installer to install the Oracle Solaris operating system onto an iSCSI target if the iSCSI target can act as a boot disk and if the system has the necessary support for iSCSI booting.

  If your system supports autodiscovery of iSCSI disks, the installer provides that option. Alternately, you can manually enter values to specify the iSCSI target in the installation screens.

  For further information, see the installation procedure in this chapter. Also, see the `iscsiadm`(1M) man page.
- 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 Live Media contains a set of software that is appropriate for a desktop or laptop. The text installer installs 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 58.

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 Live Media 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 Prepare for a Text Installation

Complete the following tasks before you perform a text installation.

1 **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, 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.

---

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

a. **Verify that your system meets all of the necessary system requirements.**

See "System Requirements for Live Media and Text Installations" on page 29.

b. **Verify that you have all of the necessary device drivers.**

See "Ensuring That You Have the Proper Device Drivers" on page 35.

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

a. **Review the specifications in "Preparing a Boot Environment for Installing Multiple Operating Systems" on page 30.**

b. **Back up your system.**

c. **If you want 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 31.

▼ How to Perform a Text Installation

1 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.

---

2 (Optional) To 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 229. After you have installed the drivers, restart the text installation and return to the installation menu.

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

```
Welcome to the Oracle Solaris 11.1 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.

---

4 Continue past the welcome panel.

5 In the Disk Discovery panels, select the type of disk that you want the installer to discover.

- Local Disks – This is the default option for disks that are attached to the computer, including internal and external hard disks.
- iSCSI – If you want the installer to search for remote disks that are accessible over a network using the iSCSI standard, select this option. Additional fields display as follows:
  - Use DHCP autodiscovery – If your system supports autodiscovery of iSCSI disks, this option is enabled. Selecting this option populates the criteria fields with the values returned from autodiscovery. You can then select the "Specify search criteria" option to further refine these values.
  - Specify search criteria – You can select this option and manually provide the iSCSI search values.

Target IP	The IP address of the iSCSI target. Four numbers in the range 0-255 must be entered. The system at this IP address must be online and accessible from this system. These fields are mandatory.
Target LUN	The Logical Unit Number of the iSCSI device located at the provided IP address. The LUN is often a numerical value such as "0", "1", and so on. This field is optional.
Target Name	The name of the iSCSI target in iSCSI Qualified Name (IQN) format. This field is optional.
Port	The port number used in conjunction with the provided IP address for discovering the iSCSI device. The default value of "3260" is the port typically used for iSCSI. This field is optional.
Initiator Name	The initiator node name to be set for the iSCSI discovery session. For iSCSI booting, this field is hidden as the initiator node name cannot be modified. This field is optional.
Use CHAP	Select this option if you want to enter CHAP (Challenge-Handshake Authentication Protocol) authentication details.
Name	The CHAP name to be used for authentication. This field is optional.
Password	The CHAP secret value for authentication. If provided, this value must be between 12 and 16 characters long. This field is optional.

If you choose the iSCSI option, a delay might occur when you select Next while the details entered are validated. If the iSCSI LUN cannot be discovered, an error is displayed. You cannot proceed until the problem is resolved, either by entering valid criteria or by deselecting iSCSI.

6 **In the Disks Selection panel, if more than one target disk is 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.

The installer uses GPT formatting when installing onto a whole disk or an unformatted disk. However, existing GPT partitions or DOS partitions are retained by default and displayed by the installer, so you can retain and install into an existing partition.

Caution – GPT formatting is currently not available on SPARC platforms.

Note the following:

- If the disk contains existing DOS partitions, up to four DOS primary partitions are displayed. If a DOS extended partition exists, its logical partitions are also displayed in the disk layout order within the extended partition. Only one Solaris partition is allowed, and that Solaris partition must be used for the installation. The Solaris partition can be a logical partition within an extended partition.
- If the disk contains existing GPT partitions, the GPT partitions are displayed. Up to seven GPT partitions are supported. You can create one or more Solaris partitions during the installation, but you must choose one Solaris partition as the installation target. If there are multiple, existing Solaris GPT partitions, the first suitable Solaris GPT partition will be chosen by default as the installation target.

For detailed partitioning instructions, see "Guidelines for Partitioning a System During an Interactive Installation" on page 31, or see the online help in the installer.

9 **Provide 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:**

   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 find a name server only 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 **In the Support Configuration panels, you can accept the anonymous registration for OCM and ASR.**

The default Support Configuration installer panel provides an anonymous registration address. If you use this anonymous address with no password, My Oracle Support (MOS) will receive information about the installed system's configuration, but will not receive any of your customer information when the system configuration is uploaded to the Oracle support organization.

Alternately, you can register for security updates or disconnect OCM as follows:

- You can replace the anonymous email address in the Support Configuration panel with your My Oracle Support login ID and add your My Oracle Support password. Use this option if you want to see your customer information in My Oracle Support and receive security updates. With this option, ASR will also be started.
- If you delete the anonymous email address in the Support Configuration panel and leave that field blank, OCM will be started in a disconnected mode. No data will be sent to My Oracle Support. Or, if you delete the anonymous email address and replace it with another email address other than your MOS login ID, OCM will send data to Oracle support in an unauthenticated mode.

For further information, see "Using Oracle Configuration Manager" on page 36.

14 **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.

15 **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.

16 **Review the installation logs.**

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

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

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.

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.

Note – Installing, updating, and uninstalling packages require increased privileges. See "Installation Privileges" in *Adding and Updating Oracle Solaris 11.1 Software Packages* for more information.

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:

```
$ pfexec pkg install package–name
```

Replace the *package–name* 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.

```
$ pfexec pkg install  --be-name new–BE–name  package–name
```

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.**

- **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 11.1 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 52.

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 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.

Note that installing, updating, and uninstalling packages require increased privileges. See "Installation Privileges" in *Adding and Updating Oracle Solaris 11.1 Software Packages* for more information.

```
$ pfexec pkg install solaris-desktop

$ pfexec pkg install solaris-large-server
```

◆ ◆ ◆ CHAPTER 5

Automated Installations That Boot From Media

You can initiate an automated installation of the Oracle Solaris 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.

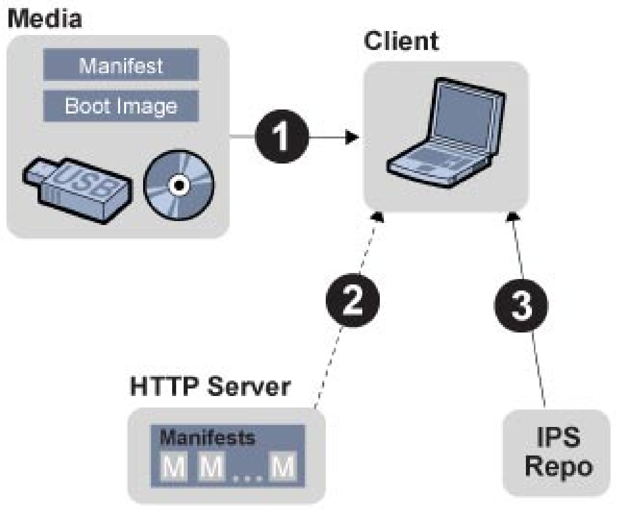
Installation using AI media has the following characteristics:

- 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 63.

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 11.1 Release Notes*.
Disk Space	To check the disk space requirements for the current release, see *Oracle Solaris 11.1 Release Notes*.
Network Access	The system to be installed must be able to access an IPS repository that contains the packages to be installed on the client system. Also, 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 63.

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 64 and "Booting an x86 System From AI Media" on page 65 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 73.

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 sample 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.

```
# /usr/sbin/mount -o ro -F hsfs /home/username/sol-11_1-20-ai-x86.iso /mnt
# cp /mnt/auto_install/manifest/default.xml /home/username/custom.xml
# umount /mnt
```

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 127 as the base to create a custom manifest.

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

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 1.99.5.11.0.175.1.0.0.20.0

Oracle Solaris 11.1 Automated Install custom
Oracle Solaris 11.1 Automated Install
Oracle Solaris 11.1 Automated Install custom ttya
Oracle Solaris 11.1 Automated Install custom ttyb
Oracle Solaris 11.1 Automated Install ttya
Oracle Solaris 11.1 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 11.1 Automated Install
Oracle Solaris 11.1 Automated Install ttya
Oracle Solaris 11.1 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 11.1 Automated Install custom
Oracle Solaris 11.1 Automated Install custom ttya
Oracle Solaris 11.1 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 for the GRUB2 entry that you use, if `install=true` is specified in the line that starts with "$multiboot", the installation automatically begins. If you want to boot the x86 system without immediately starting an automated installation, examine the GRUB2 menu entry that you plan to choose. If `install=true` is specified in the kernel line for that GRUB2 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	The "system" grouping includes all the other groupings.
`identity`	System nodename	Unknown
`kbd_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
`support`	OCM and ASR support	Default setting is anonymous registration with OCM and ASR

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 69.
- 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 69.
- 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 73.

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 shut down 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 70.

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 73.

▼ 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. **Provide 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:**

   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 find a name server only 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 **In the Support Configuration panels, you can accept the anonymous registration for OCM and ASR.**

The default Support Configuration panel provides an anonymous registration address. If you use this anonymous address with no password, My Oracle Support (MOS) will receive information about the system's configuration, but will not receive any of your customer information when the system configuration is uploaded to the Oracle support organization.

Alternately, you can register for security updates or disconnect OCM as follows:

- You can replace the anonymous email address in the Support Configuration panel with your My Oracle Support login ID and add your My Oracle Support password. Use this option if you want to see your customer information in My Oracle Support and receive security updates. With this option, ASR will also be started.
- If you delete the anonymous email address in the Support Configuration panel and leave that field blank, OCM will be started in a disconnected mode. No data will be sent to My Oracle Support. Or, if you delete the anonymous email address and replace it with another email address other than your MOS login ID, OCM will send data to Oracle support in an unauthenticated mode.

For further information, see "Using Oracle Configuration Manager" on page 36.

9 **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 to successfully use that profile for reconfiguration.

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

P A R T   I I I

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) or the Oracle Solaris 11.1 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. SPARC clients can optionally get network configuration and location of the install server from the `network-boot-arguments` variable set in the Open Boot PROM (OBP).
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 to Use the Automated Installer

To use AI to install client systems over the network, you must 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, name service server, and the location of the install server to the client machine to be installed. SPARC clients can optionally get their network configuration and install server location from the `network-boot-arguments` variable set in the OBP. See "Installing a SPARC Client" on page 208 for an example of using the `network-boot-arguments` variable. 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. A repository is a location from where software packages are retrieved. The location is specified by a Universal Resource Identifier (URI). The IPS package repository can be on the install server, on another server on the local network, or on the Internet. See "Configuring Publishers" in *Adding and Updating Oracle Solaris 11.1 Software Packages* for information about accessing a package repository.

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 SMF profile files. The AI 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 installation 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 a different network or locale setup or a unique host name and IP address, for example.

The installation begins when you boot the client. When the client boots, the client is directed 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 14, "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 69 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.

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

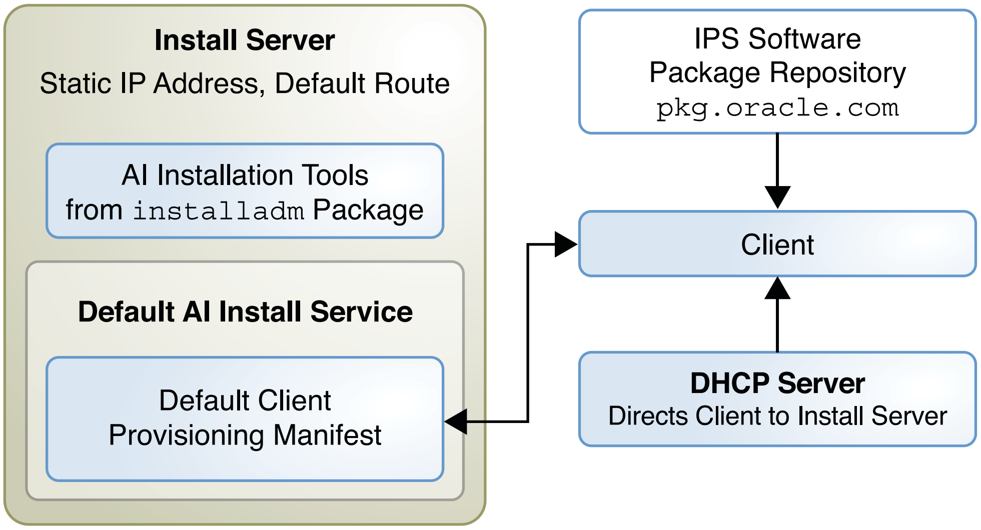
The minimum you have to do to use AI is create one install service. In this minimal scenario, all clients are the same architecture and are to be installed with the same version of the Oracle Solaris OS. The installations use the default AI manifest. For the Oracle Solaris 11.1 release, the default AI manifest specifies the most recent version of the Oracle Solaris 11.1 release available from the `http://pkg.oracle.com/solaris/release` IPS package repository.

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 and the necessary information is available in the DHCP configuration for clients to boot the service. For SPARC clients, you can optionally set the `network-boot-arguments` variable in the OBP to boot the service, as shown in “Installing a SPARC Client” on page 208.
5. 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.
6. Network boot the client.

FIGURE 7–1 Minimum Requirements for AI Use


Install Server
Static IP Address, Default Route
AI Installation Tools from installadm Package
Default AI Install Service
Default Client Provisioning Manifest
IPS Software Package Repository pkg.oracle.com
Client
DHCP Server
Directs Client to Install Server

In this scenario, when you network boot the client, the following steps are performed:

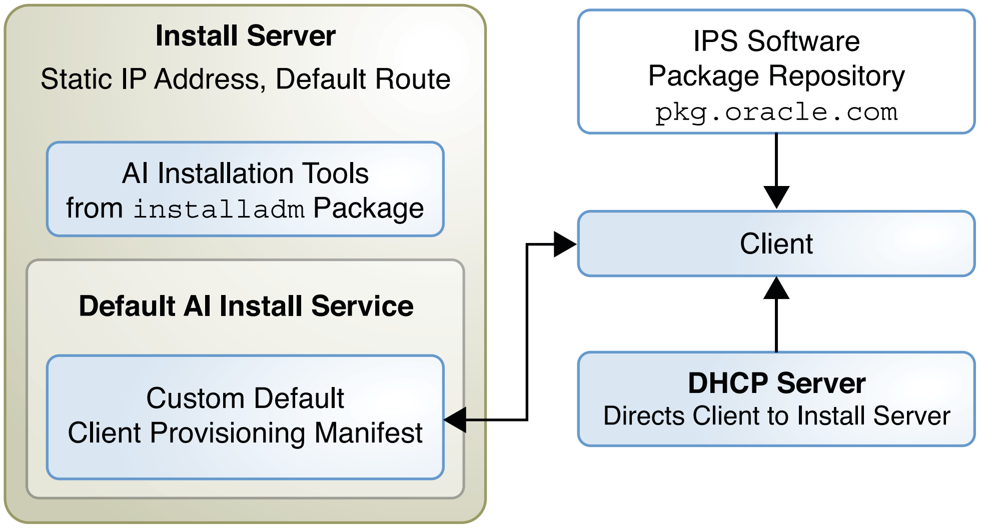
1. The client gets the install server address from the DHCP server. SPARC clients can optionally get the install server address from the `network-boot-arguments` variable in the OBP.
2. The client uses the `default-`*arch* install service if the architecture matches.
3. The client uses the default AI 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.

Customizing 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.

FIGURE 7–2 Customizing Installation Instructions


Install Server
Static IP Address, Default Route
AI Installation Tools from installadm Package
Default AI Install Service
Custom Default Client Provisioning Manifest
IPS Software Package Repository pkg.oracle.com
Client
DHCP Server
Directs Client to Install Server

In this scenario, when you network boot the client, the following steps are performed:

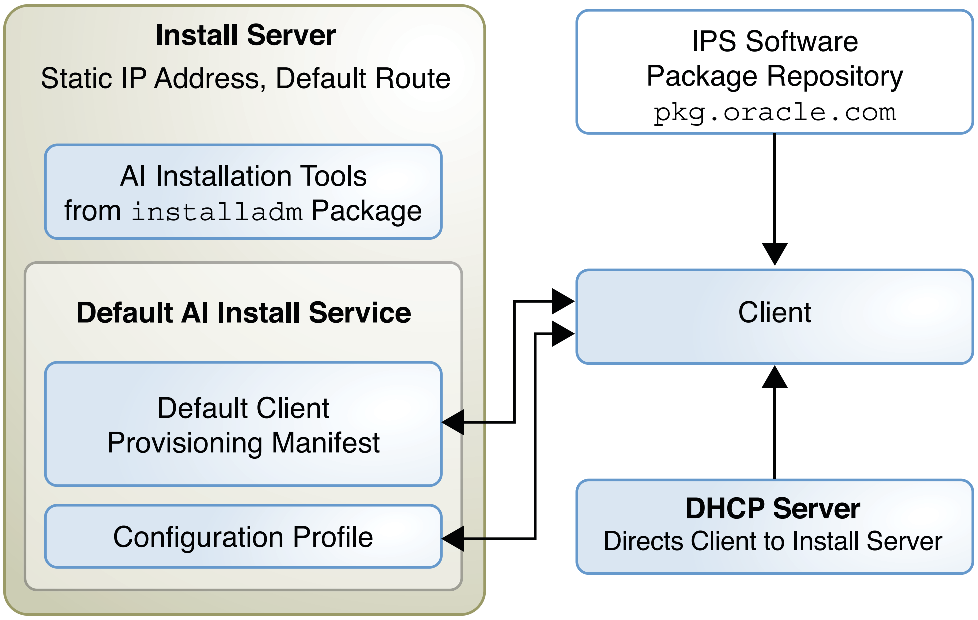
1. The client gets the install server address from the DHCP server. SPARC clients can optionally get the install server address from the `network-boot-arguments` variable in the OBP.
2. The client uses the `default-`*arch* install service if the architecture matches.
3. The client is directed to the 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.

Providing 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.

FIGURE 7–3 Providing System Configuration Instructions


Install Server
Static IP Address, Default Route
AI Installation Tools from installadm Package
Default AI Install Service
Default Client Provisioning Manifest
Configuration Profile
IPS Software Package Repository pkg.oracle.com
Client
DHCP Server
Directs Client to Install Server

In this scenario, when you network boot the client, the following steps are performed:

1. The client gets the install server address from the DHCP server. SPARC clients can optionally get the install server address from the `network-boot-arguments` variable in the OBP.
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.

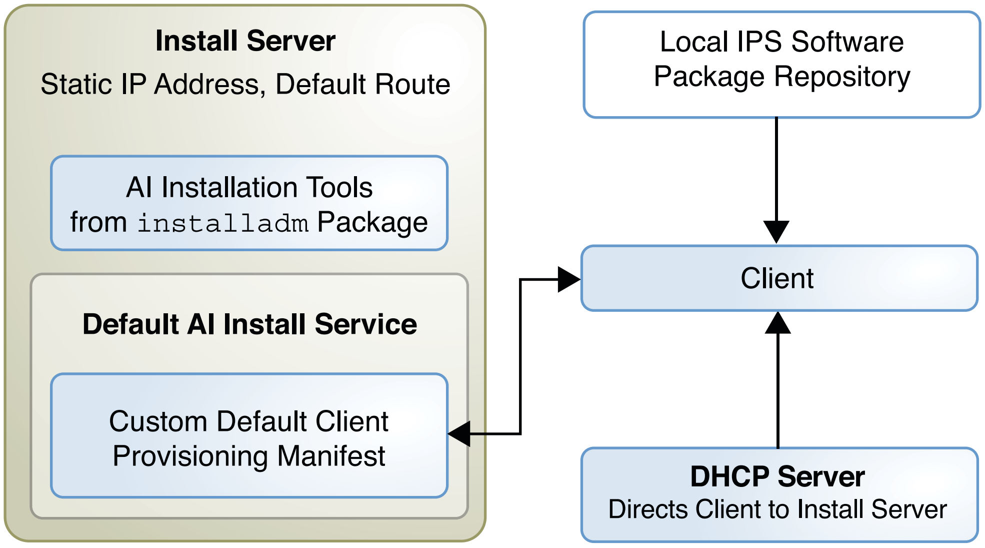
4. The client is directed to the correct system configuration profile by criteria specified to `create-profile` for the `default-`*arch* install service.
5. The client is configured according to the selected configuration profile. If no configuration profile is selected because criteria do not match, the interactive configuration tool starts.

Providing 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 11.1 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.

FIGURE 7–4 Providing a Local IPS Package Repository


Install Server
Static IP Address, Default Route
AI Installation Tools from installadm Package
Default AI Install Service
Custom Default Client Provisioning Manifest
Local IPS Software Package Repository
Client
DHCP Server
Directs Client to Install Server

In this scenario, when you network boot the client, the following steps are performed:

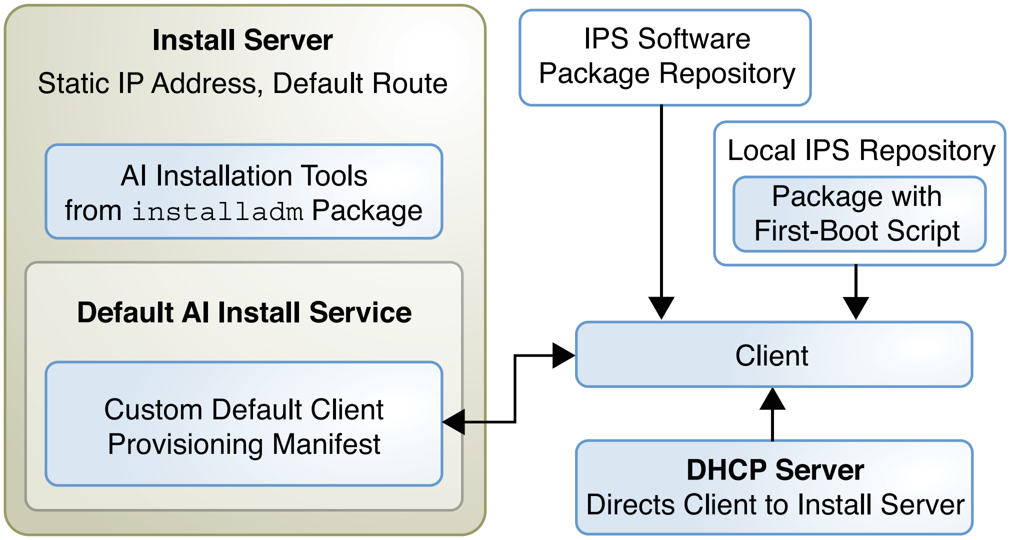
1. The client gets the install server address from the DHCP server. SPARC clients can optionally get the install server address from the `network-boot-arguments` variable in the OBP.
2. The client uses the `default-`*arch* install service if the architecture matches.
3. The client is provisioned according 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.

Providing 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.

FIGURE 7–5 Providing a Custom First-Boot Script


Install Server
Static IP Address, Default Route
AI Installation Tools from installadm Package
Default AI Install Service
Custom Default Client Provisioning Manifest
IPS Software Package Repository
Local IPS Repository
Package with First-Boot Script
Client
DHCP Server
Directs Client to Install Server

In this scenario, when you network boot the client, the following steps are performed:

1. The client gets the install server address from the DHCP server. SPARC clients can optionally get the install server address from the `network-boot-arguments` variable in the OBP.
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.

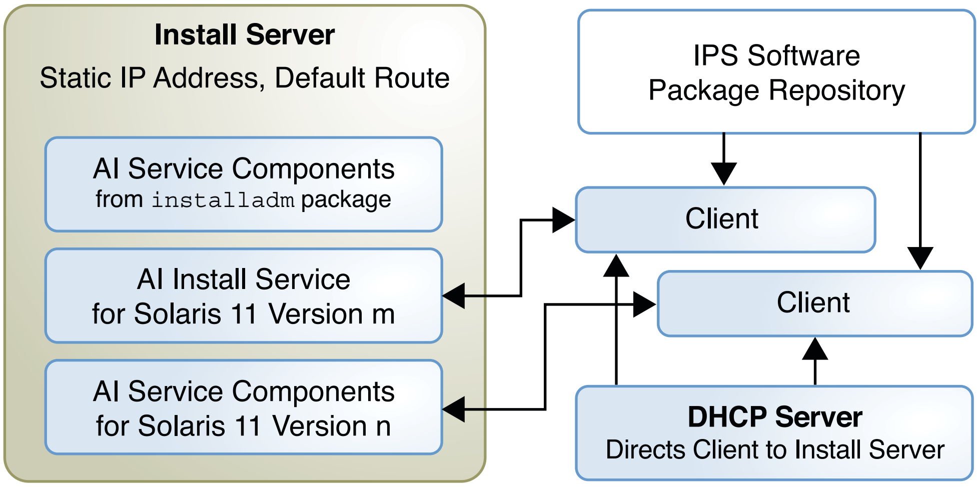
Providing Additional AI Install Services

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.

FIGURE 7–6 Providing Additional AI Install Services


Install Server
Static IP Address, Default Route
AI Service Components from installadm package
AI Install Service for Solaris 11 Version m
AI Service Components for Solaris 11 Version n
IPS Software Package Repository
Client
Client
DHCP Server
Directs Client to Install Server

In this scenario, when you network boot the client, the following steps are performed:

1. The client gets the install server address from the DHCP server. SPARC clients can optionally get the install server address from the `network-boot-arguments` variable in the OBP.
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 function as an install server. On the install server, create an AI install service to provide a net image and instructions for installing the desired Oracle Solaris 11 release 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 AI Server Setup 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 90.
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 91.
Install the AI tool set.	See "Installing the AI Installation Tools" on page 92.
Set up an install service.	See "Creating an AI Install Service" on page 96. 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 11.1 OS is already installed. If you need to install or update the Oracle Solaris 11.1 OS on your AI install server, check Chapter 4, "Using the Text Installer," and Chapter 3, "Using Live Media," 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.

Install Service Operation Privileges

Many of the commands used with automated installation require increased privilege. Use one of the following methods to gain more privilege:

Rights profiles

Use the `profiles` command to list the rights profiles that are assigned to you.

Software Installation

If you have the Software Installation rights profile, you can use the `pfexec` command to install and update packages.

```
$ pfexec pkg install install/installadm
```

Install Service Management

If you have the Install Service Management rights profile, you can use the `pfexec` command to create install services and add system configuration profiles to an install service, for example.

```
$ pfexec installadm create-service
```

Service Management

If you have the Service Management rights profile, you can configure and enable SMF services. The Service Management rights profile does not need `pfexec`.

```
$ svcadm refresh system/install/server:default
```

sudo	Depending on the security policy at your site, you might be able to use the `sudo` command with your user password to execute a privileged command. `$ sudo pkg install install/installadm`
Roles	Use the `roles` command to list the roles that are assigned to you. If you have the root role, you can use the `su` command with the root password to assume the root role.

AI Server Software Requirements

Operating system	Install the Oracle Solaris 11.1 OS on the AI server. To install the Oracle Solaris 11.1 OS 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 *Connecting Systems Using Fixed Network Configuration in Oracle Solaris 11.1*.
Default router	Ensure that your AI server has a default route set by using the `netstat -rn` command. If your AI server does not have a default route set, use the `route add default` command to statically set a default route. See the `route`(1M) man page for information about how to use the `route` command.
Multicast DNS	(Optional) Enable multicast DNS to advertise the install services on your AI server. See "Enabling Multicast DNS" on page 93.
Software package repository	Ensure that the install server can access an IPS software package repository. AI requires the `install/installadm` package. Use the `pkg list` command as shown in "Installing the AI Installation Tools" on page 92.
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. "Creating an Install Service Including Local DHCP Setup" on page 100 shows an example of using AI to set up DHCP on the install server. For information about configuring DHCP, see the "Related Information" on page 18.

Installing 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 maintain 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 "Maintaining an Install Server" on page 103 and the `installadm`(1M) man page 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 show whether your IPS package repository contains a version of the `installadm` package that you can install in this image.

```
$ pkg list -a installadm
NAME (PUBLISHER)                                       VERSION                IFO
install/installadm                                     0.5.11-0.175.1.0.0.24.0 ---
```

If more than one publisher is defined for this image, use the `-v` option to show which publisher provides the `installadm` package. The following example shows that `solaris` is the publisher of this package:

```
$ pkg list -av installadm
FMRI                                                                          IFO
pkg://solaris/install/installadm@0.5.11,5.11-0.175.1.0.0.24.0:20120815T024057Z ---
```

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  P LOCATION
solaris                 origin  online  F file:///export/repoSolaris11/
isv.com  (non-sticky)   origin  online  F http://pkg.isv.com/
```

Use the pkg install command to install the installadm package.

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

DOWNLOAD                      PKGS       FILES    XFER (MB)   SPEED
Completed                      1/1       72/72      0.3/0.3    0B/s

PHASE                                        ITEMS
Installing new actions                     138/138
Updating package state database               Done
Updating image state                          Done
Creating fast lookup database                 Done
Reading search index                          Done
Updating search index                          1/1
```

```
$ 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: 0.175.1.0.0.24.0
Packaging Date: Mon Aug 15 02:40:57 2012
          Size: 1.21 MB
          FMRI: pkg://solaris/install/installadm@0.5.11,5.11-0.175.1.0.0.24.0:20120815T024057Z
```

Configuring 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.

Enabling Multicast DNS

Install clients discover the install service from which they boot. Enable multicast DNS (mDNS) to advertise the install services on your AI server so that they can be discovered over the network.

If you have multiple AI servers on your network with the same install service names, you should disable multicast DNS to prevent conflicts.

Use the svcs command to check the status of the svc:/network/dns/multicast service, and then use the svcadm command to enable the service if necessary:

```
 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
```

Configuring a Multihomed Install Server

A multihomed host is a system with more than one interface and acts as a host on multiple IP subnets. For more information about multihomed hosts, see "Configuring Multihomed Hosts" in *Configuring and Administering Oracle Solaris 11.1 Networks*.

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
```

Configuring 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
$ svcadm 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.

Configuring the Default Image Path

Use the all_services/default_imagepath_basedir property to change the default base directory for images created by the installadm create-service command.

Images are created in a *service_name* directory at the location specified by the value of the all_services/default_imagepath_basedir property. The default value of this property is /export/auto_install. Thus, by default, the net image for the *service_name* service is created at /export/auto_install/*service_name*.

The following commands configure the AI server to create new install services at /export/aiimages/*service_name* by default:

```
$ svccfg -s system/install/server:default \
setprop all_services/default_imagepath_basedir = /export/aiimages
$ svcadm refresh system/install/server:default
```

You can override this default directory for a specific install service by using the -d option of the installadm create-service command. You can change the image path for a specific existing install service by using the following command:

```
$ pfexec installadm set-service -o imagepath=newpath  service_name
```

Automatically Updating the ISC DHCP Configuration

By default, the local ISC DHCP configuration is automatically updated when client and service configurations are modified in the install server. If you do not want the local ISC DHCP configuration to be automatically maintained, set the all_services/manage_dhcp property value to false.

When you use the installadm create-service command to configure DHCP as shown in "Creating an Install Service Including Local DHCP Setup" on page 100, the command exits if the value of the all_services/manage_dhcp property is false The exit message instructs you to set the value of the all_services/manage_dhcp property to true and invoke the installadm create-service command again.

Use the following command to check the value of the all_services/manage_dhcp property:

```
$ svcprop -p all_services/manage_dhcp svc:/system/install/server:default
true
```

Creating 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.

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 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, but this default service shares a net image with the explicitly created service. 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 106 for more information about setting the `aliasof` property. To update the net image of the service for which the `default-`*arch* service is an alias, use the `update-service` subcommand as shown in "Updating an Install Service" on page 107.

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 the value of the `all_services/manage_dhcp` property is `true`. See "Automatically Updating the ISC DHCP Configuration" on page 95 for more information about the `all_services/manage_dhcp` property.

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://10.80.238.5:5555/solaris11_1-i386`. See "Configuring the Web Server Host Port" on page 94 to use a different port.

The `installadm` `create-service` command does not require any arguments or options. The two options described below are commonly used. For information about all options, see "Creating an Install Service" on page 104 or the `installadm`(1M) man page.

```
installadm create-service  [-s source] [-y]
```

`-s` *source* The *source* argument specifies the data source for the net image. The value of *source* can be one of the following:

- The FMRI identifier of the IPS AI net image package, which is `install-image/solaris-auto-install` in the Oracle Solaris 11.1 release.
- The full path name of an AI ISO image file.

If you do not specify *source*, the newest available version of the `install-image/solaris-auto-install` package is used. The package is retrieved from the publisher specified by the `-p` option or from the first publisher in the install server's publisher preference list that provides an instance of the 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 publisher origin.

`-y` If you do not specify the `-d` option, specify the `-y` option to suppress the prompt to confirm the use of the automatically generated image path.

The `create-service` command can set up DHCP on the AI install server as shown in "Creating an Install Service Including Local DHCP Setup" on page 100.

Creating an Install Service Without Setting Up DHCP

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 output provides the boot file required for DHCP configuration.

Creating 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.

```
$ pfexec installadm create-service \
-s /var/tmp/images/sparc/sol-11_1-ai-sparc.iso -y
```

```
Creating service from: /var/tmp/images/sparc/sol-11_1-ai-sparc.iso
Setting up the image ...

Creating sparc service: solaris11_1-sparc

Image path: /export/auto_install/solaris11_1-sparc

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

Creating default-sparc alias

Service discovery fallback  mechanism set up
Creating SPARC configuration file
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

Refreshing install services
$ installadm list
Service Name      Alias Of          Status  Arch   Image Path
------------      --------          ------  ----   ----------
default-sparc     solaris11_1-sparc on      sparc  /export/auto_install/solaris11_1-sparc
solaris11_1-sparc -                 on      sparc  /export/auto_install/solaris11_1-sparc
```

The following operations are performed as a result of executing this `installadm create-service` command:

1. The install service is automatically named `solaris11_1-sparc`.
2. The default install service net image directory, `/export/auto_install/solaris11_1-sparc`, is created. Because the `-y` option is specified, the prompt to confirm that this default destination is acceptable is suppressed.
3. The ISO file, `/var/tmp/images/sparc/sol-11_1-ai-sparc.iso`, is unpacked into the net image location, `/export/auto_install/solaris11_1-sparc`.
4. The `wanboot.conf` file for this service is generated at `/export/auto_install/solaris11_1-sparc/wanboot.conf`.
5. The AI SMF service, `system/install/server`, is refreshed to mount `/export/auto_install/solaris11_1-sparc` as `/etc/netboot/solaris11_1-sparc`.
6. Because this is the first SPARC install service created on this install server, the `default-sparc` service alias is automatically created. The image from `solaris11_1-sparc` is used by the alias, so `/export/auto_install/solaris11_1-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 assignment occurs regardless of whether the `-i` and `-c` options are used.

Creating 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. If this install server is a SPARC system, you must supply the `-a i386` option to specify that you want to create an x86 install service.

In addition to the boot file required for DHCP configuration, this command output also provides the boot server IP required for DHCP configuration.

```
$ pfexec installadm create-service -y

Creating service from: pkg:/install-image/solaris-auto-install
DOWNLOAD                                  PKGS       FILES    XFER (MB)   SPEED
Completed                                  1/1     514/514  291.8/291.8    0B/s

PHASE                                          ITEMS
Installing new actions                       661/661
Updating package state database                 Done
Updating image state                            Done
Creating fast lookup database                   Done
Reading search index                            Done
Updating search index                            1/1

Creating i386 service: solaris11_1-i386

Image path: /export/auto_install/solaris11_1-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: 192.168.1.111
Boot file(s):
    bios clients (arch 00:00):  default-i386/boot/grub/pxegrub2
    uefi clients (arch 00:07):  default-i386/boot/grub/grub2netx64.efi

Refreshing install services
$ installadm list

Service Name      Alias Of          Status  Arch   Image Path
------------      --------          ------  ----   ----------
default-i386      solaris11_1-i386  on      i386   /export/auto_install/solaris11_1-i386
default-sparc     solaris11_1-sparc on      sparc  /export/auto_install/solaris11_1-sparc
```

```
solaris11_1-i386  -                  on      i386   /export/auto_install/solaris11_1-i386
solaris11_1-sparc -                  on      sparc  /export/auto_install/solaris11_1-sparc
```

The following operations are performed as a result of executing this `installadm create-service` command:

1. The install service is automatically named `solaris11_1-i386`.
2. Because no net image source option is specified, the newest version of the `install-image/solaris-auto-install` package is retrieved from the first publisher in the install server publisher list that provides this package.
3. The default install service net image directory, `/export/auto_install/solaris11_1-i386`, is created. 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/solaris11_1-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 "Creating an Install Service" on page 104 for information about the `-a` option.
5. The GRUB menu is mounted at `/etc/netboot/solaris11_1-i386/grub.cfg`.
6. The AI SMF service, `system/install/server`, is refreshed to mount `/export/auto_install/solaris11_1-i386` as `/etc/netboot/solaris11_1-i386`.
7. Because this is the first x86 install service created on this install server, the `default-i386` service alias is automatically created. The image from `solaris11_1-i386` is used by the alias, so `/export/auto_install/solaris11_1-i386` is also mounted as `/etc/netboot/default-i386`.
8. The boot server IP required for DHCP configuration is provided. The boot files required for DHCP configuration, `default-i386/boot/grub/pxegrub2` and `default-i386/boot/grub/grub2netx64.efi`, are also provided.
9. If a local ISC DHCP server is already configured, the boot files of the new `default-i386` alias are set as the default boot files for all x86 clients. This assignment occurs regardless of whether the `-i` and `-c` options are used.

Creating 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. Make sure the value of the `all_services/manage_dhcp` property is `true`. See "Automatically Updating the ISC DHCP Configuration" on page 95 for more information about the `all_services/manage_dhcp` property.

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. 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 is directly connected to and the install server is multihomed, use the `-B` option to provide the address of the boot file server (usually an IP address on this system). This option should only be necessary when multiple IP addresses are configured on the install server and DHCP relays are employed. In other configurations, the software can determine this automatically.

```
$ pfexec installadm create-service \
-s /var/tmp/images/i386/sol-11_1-ai-x86.iso \
-i 10.80.239.150 -c 20 -y

Creating service from: /var/tmp/images/i386/sol-11_1-ai-x86.iso
Setting up the image ...

Creating i386 service: solaris11_1-i386

Image path: /export/auto_install/solaris11_1-i386

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

Refreshing install services

Creating default-i386 alias.

Setting the default PXE bootfile(s) in the local DHCP configuration to:
bios clients (arch 00:00):  default-i386/boot/grub/pxegrub2
uefi clients (arch 00:07):  default-i386/boot/grub/grub2netx64.efi

Refreshing install services
```

The following operations are performed as a result of executing this `installadm create-service` command:

1. The install service is automatically named `solaris11_1-i386`.
2. The default install service net image directory, `/export/auto_install/solaris11_1-i386`, is created. Because the `-y` option is specified, the prompt to confirm that this default destination is acceptable is suppressed.

3. The ISO file, /var/tmp/images/i386/sol-11_1-ai-x86.iso, is unpacked into the net image location, /export/auto_install/solaris11_1-i386.
4. The GRUB menu is mounted at /etc/netboot/solaris11_1-i386/grub.cfg.
5. The AI SMF service, system/install/server, is refreshed to mount /export/auto_install/solaris11_1-i386 as /etc/netboot/solaris11_1-i386.
6. Because this is the first x86 install service created on this install server, the default-i386 service alias is automatically created. The image from solaris11_1-i386 is used by the alias, so /export/auto_install/solaris11_1-i386 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. The svc:/network/dhcp/server service is online.
8. The boot files default-i386/boot/grub/pxegrub2 and default-i386/boot/grub/grub2netx64.efi are added to the local DHCP configuration as the default boot files for PXE clients.

The following sections show how installadm might add information to the DHCP configuration file for an ISC DHCP configuration. For more information about configuring ISC DHCP, see the "Related Information" on page 18.

ISC DHCP Configuration for an Oracle Solaris 11.1 i386 Install Service

The following example shows how installadm might add the IP addresses specified using the -i and -c options to the /etc/inet/dhcpd4.conf file for an ISC DHCP configuration for the Oracle Solaris 11.1 i386 install service created above:

```
subnet 10.80.239.0 netmask 25.255.255.0 {
  range 10.80.239.150 10.80.239.169;
  option broadcast-address  10.80.239.255;
  option routers 10.80.239.1;
  next-server 10.80.239.170;
}
```

The following example shows how installadm might set the default PXE boot files in the /etc/inet/dhcpd4.conf file for an ISC DHCP configuration for the default-i386 Oracle Solaris 11.1 i386 install service created above:

```
class "PXEBoot" {
  match if (substring(option vendor-class-identifier, 0, 9) = "PXEClient");
  if option arch = 00:00 {
    filename "default-i386/boot/grub/pxegrub2";
  } else if option arch = 00:07 {
    filename "default-i386/boot/grub/grub2netx64.efi";
  }
}
```

ISC DHCP Configuration for an Oracle Solaris 11 i386 Install Service

If you created an Oracle Solaris 11 `i386` install service instead of an Oracle Solaris 11.1 service, you would see output similar to the following example:

```
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
```

The following example shows how `installadm` might set the default PXE boot file in the `/etc/inet/dhcpd4.conf` file for an ISC DHCP configuration for an Oracle Solaris 11 `i386` install service:

```
class "PXEBoot" {
  match if (substring(option vendor-class-identifier, 0, 9) = "PXEClient");
  if option arch = 00:00 {
    filename "default-i386/boot/grub/pxegrub";
  }
}
```

ISC DHCP Configuration for an Oracle Solaris 11.1 sparc Install Service

If you created a `sparc` install service instead of an `i386` service, you would see output similar to the following example:

```
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
```

The following example shows how `installadm` might set the default boot file in the `/etc/inet/dhcpd4.conf` file for an ISC DHCP configuration for an Oracle Solaris 11.1 `sparc` install service:

```
class "SPARC" {
  match if not (substring(option vendor-class-identifier, 0, 9) = "PXEClient");
  filename "http://10.80.238.5:5555/cgi-bin/wanboot-cgi";
}
```

Maintaining an Install Server

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

- "Adding, Modifying, or Deleting an Install Service" on page 104
- "Associating Clients With Install Services" on page 108
- "Associating Client-Specific Installation Instructions With Install Services" on page 110
- "Associating Client-Specific Configuration Instructions With Install Services" on page 112
- "Exporting an AI Manifest or a System Configuration Profile" on page 115
- "Modifying Criteria for an AI Manifest or a System Configuration Profile" on page 115

- "Listing All Install Services on the Install Server" on page 116
- "Listing Clients Associated With Install Services" on page 117
- "Listing All AI Manifests and System Configuration Profiles" on page 118

Adding, Modifying, or Deleting 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.

Creating an Install Service

Use the following command to create an install service. See "Creating an AI Install Service" on page 96 for examples.

```
installadm create-service [-n svcname] [-s source]
    [-p publisher=origin] [-a architecture]
    [-d imagepath] [-y] [-t aliasof]
    [-i start] [-c count]
    [-b property=value,...] [-B server]
```

`-n` *svcname*
: The value of the *svcname* argument can consist of alphanumeric characters, underscores (_), and hyphens (`-`). The first character of *svcname* cannot be a hyphen. The length of the value of the *svcname* argument cannot exceed 63 characters. If you do not provide a name for the install service, a default name is assigned. The default name indicates the architecture and the OS version.

`-s` *source*
: The *source* argument specifies the data source for the net image. The value of *source* can be one of the following:

1. The FMRI identifier of the IPS AI net image package, which is `install-image/solaris-auto-install` in the Oracle Solaris 11.1 release.
2. The full path name of an AI ISO net image file.

If you do not specify *source*, the newest available version of the `install-image/solaris-auto-install` package is used. The package is retrieved from the publisher specified by the `-p` option or from the first publisher in the install server's publisher preference list that provides an instance of the 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 publisher origin.

`-p` *publisher=origin*
This option is used only when the net image source is an IPS package. This option specifies the IPS package repository from where you want to retrieve the `install-image/solaris-auto-install` package. The value of the *publisher* argument is the publisher name, and the *origin* is the URI, as in `solaris=http://pkg.oracle.com/solaris/release/`.

If the `-p` option is not specified, the publisher used is the first publisher in the install server's publisher preference list that provides an instance of the package.

`-a` *architecture*
This option is used only when the net image source is an IPS package. The *architecture* argument specifies the architecture of the clients to be installed with this service. You can specify either `i386` or `sparc`. The default value is the architecture of the install server.

`-d` *imagepath*
The value of the *imagepath* argument is the path at which to create the net image. The `install-image/solaris-auto-install` package is installed to this location, or the specified ISO file is unpacked at this location. If you do not specify *imagepath*, the image is created in a *svcname* directory at the location defined by the value of the `all_services/default_imagepath_basedir` property. For the default value of this property, or to change the value of this property, see "Configuring the Default Image Path" on page 95. If you do not specify *imagepath*, 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 the use of the automatically generated image path.

`-t` *aliasof*
This option designates the new service as an alias. The new service shares the net image of the *aliasof* service but has its own manifests, profiles, and clients.

`-i` *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 if the value of the `all_services/manage_dhcp` property is `true`. See "Automatically Updating the ISC DHCP Configuration" on page 95 for more information about the `all_services/manage_dhcp` property.

`-c` *count*
Sets up a total number of IP addresses in the DHCP configuration equal to the value of the *count*. The first IP address is the value of *start* that is provided by the `-i` option.

`-b` *property=value,...*
For x86 services only. This option sets a property value in the service-specific `grub.cfg` 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 *property=value* pairs.

`-B` *server*
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 property=value svcname
```

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

`aliasof=`*aliasof*
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 *aliasof* 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 "Listing All Install Services on the Install Server" on page 116 to confirm that *svcname* is an alias.

Manifests, profiles, and client bindings that were added to either *svcname* or *aliasof* 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 DTDs 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*
Designates a particular manifest or derived manifests script that is already registered with the specified 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
```

`imagepath=`*newpath*
Relocates the image for a service after that service has been created:

```
$ pfexec installadm set-service -o imagepath=/export/aiimages/solaris11_1-i386 solaris11_1-i386
```

Updating an Install Service

Use the following command to update the image associated with an alias of a service that was created using an IPS AI net image package.

```
installadm update-service [-p|--publisher publisher=origin]
    [-s|--source FMRI] svcname
```

This command updates the image associated with *svcname*, where *svcname* is an alias of a service that was created using an IPS AI net image package. A new service is created with the updated image, and *svcname* is aliased to the new service.

`-p|--publisher` *publisher=origin*
: This option specifies the IPS package repository from which to update the *svcname* image. An example value is `solaris=http://pkg.oracle.com/solaris/release/`.

  If the `-p` option is not specified, the publisher used is the publisher that was used to create the image of the service for which *svcname* is an alias. The following `pkg publisher` command shows how to display the *svcname* publisher:

```
$ installadm list
Service Name     Alias Of          Status  Arch  Image Path
------------     --------          ------  ----  ----------
default-i386     solaris11_1-i386  on      i386  /export/auto_install/solaris11_1-i386
solaris11_1-i386 -                 on      i386  /export/auto_install/solaris11_1-i386
$ pkg -R /export/auto_install/solaris11_1-i386 publisher
PUBLISHER        TYPE     STATUS   URI
solaris          origin   online   http://pkg.oracle.com/solaris/release/
```

`-s|--source` *FMRI*
: This option specifies the FMRI of the net image package for the update.

  If the `-s` option is not specified, the newest available version of the `install-image/solaris-auto-install` package is used from the publisher specified in the description of the `-p` option.

Renaming an Install Service

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

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

The value of the *newsvcname* argument can consist of alphanumeric characters, underscores (_), and hyphens (`-`). The first character of *newsvcname* cannot be a hyphen. The length of the value of the *newsvcname* argument cannot exceed 63 characters.

Enabling or Disabling 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
```

When you disable an install service, any clients associated with that install service remain associated. See "Associating Clients With Install Services" on page 108. This means that when you re-enable an install service, you do not need to re-associate the clients of that install service.

If you disable the default install service for an architecture or the service that the default service is aliased to, any clients of that architecture that are not associated with some other service will not boot.

Deleting 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 if the value of the `all_services/manage_dhcp` property is `true`. See "Automatically Updating the ISC DHCP Configuration" on page 95 for more information about the `all_services/manage_dhcp` property.

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.

If you delete the default install service for an architecture or the service that the default service is aliased to, any clients of that architecture that are not associated with some other service will not boot. You will be prompted to confirm the operation.

Associating 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 205 for more examples and sample output.

Adding a Client To an Install Service

Use the `installadm create-client` command to associate the *macaddr* client with the *svcname* install service and to provide custom client settings for x86 clients.

```
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 if the value of the `all_services/manage_dhcp` property is `true`. See "Automatically Updating the ISC DHCP Configuration" on page 95 for more information about the `all_services/manage_dhcp` property.

To find the MAC address of a system, use the `dladm` command as shown in the following example. See the `dladm`(1M) man page for more information.

```
$ dladm show-linkprop -p mac-address
LINK   PROPERTY       PERM VALUE             DEFAULT POSSIBLE
net0   mac-address    rw   8:0:20:0:0:1      8:0:20:0:0:1      --
net1   mac-address    rw   0:14:4f:45:c:2d   0:14:4f:45:c:2d   --
```

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

The following command adds the client with MAC address `00:14:4f:a7:65:70` to the `solaris11_1-sparc` install service:

```
$ pfexec installadm create-client -e 00:14:4f:a7:65:70 -n solaris11_1-sparc
```

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

```
$ pfexec installadm create-client -e c0ffeec0ffee -n solaris11_1-i386 -b console=ttya
```

Associating 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.

Deleting 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 if the value of the `all_services/manage_dhcp` property is `true`. See "Automatically Updating the ISC DHCP Configuration" on page 95 for more information about the `all_services/manage_dhcp` property.

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

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

Associating 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.

Adding an AI Manifest to an Install Service

Use the `installadm create-manifest` command to add the *manifest* custom AI manifest or derived manifests script to the *svcname* install service.

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

The value of the *manifest* argument 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 "Adding a Derived Manifests Script to an Install Service" on page 149.

The value of the *manifest* argument is the name displayed by the `installadm list` command. See "Listing All AI Manifests and System Configuration Profiles" on page 118. If the `-m` option is not provided, the name of the manifest is the value of the `name` attribute of the `ai_instance` element, if present, or the base name of the *filename* value.

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 the criteria specified by the `-c` and `-C` options 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 solaris11_1-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.

```
$ pfexec installadm create-manifest -f ./mymanifests/manifest_t200.xml \
-m t200 -n solaris11_1-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.

```
$ pfexec installadm create-manifest -f ./mymanifests/manifest_t200.xml \
-m t200 -n solaris11_1-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>
```

Updating an AI Manifest

Use the installadm update-manifest command to replace the contents of the *manifest* AI manifest or derived manifests script file with the contents of the *filename* manifest or script file for the *svcname* install service. Criteria, default status, and *manifest* name are not changed as a result of the update.

```
installadm update-manifest -n svcname
    -f filename [-m manifest]
```

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 "Adding a Derived Manifests Script to an Install Service" on page 149.

The *manifest* manifest must already exist in the *svcname* service. Use the installadm list command to confirm. See "Listing All AI Manifests and System Configuration Profiles" on page 118.

If *manifest* 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 *filename* manifest, if this attribute is specified and if the value of this attribute matches the *manifest* name of an existing manifest for this install service.
- The base name of the *filename* value if this name matches the *manifest* name of an existing manifest for this install service.

The following command updates the content of the `t200` manifest in the `solaris11_1-sparc` service with the content of `./mymanifests/manifest_newt200.xml`. The name of the manifest in `installadm list` is still `t200`.

```
$ pfexec installadm update-manifest -n solaris11_1-sparc \
-f ./mymanifests/manifest_newt200.xml -m t200
```

Deleting an AI Manifest

Use the `installadm delete-manifest` command to remove the *manifest* AI manifest or derived manifests script from the *svcname* install service. The value of the *manifest* argument is the manifest name that the `installadm list` command returns. See "Listing All AI Manifests and System Configuration Profiles" on page 118.

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

You cannot delete the default AI manifest.

The following command removes the `t200` AI manifest from the `solaris11_1-sparc` install service:

```
$ pfexec installadm delete-manifest -m t200 -n solaris11_1-sparc
```

Associating 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.

Adding a System Configuration Profile to an Install Service

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

```
installadm create-profile -n svcname
    -f filename... [-p profile]
    [-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 *filename* file can contain variables that are replaced with values from the client's installation environment during the installation process. See "Using System Configuration Profile Templates" on page 166 for more information.

The value of the *profile* argument is the profile name displayed by the `installadm list` command once the profile has been added. See "Listing All AI Manifests and System Configuration Profiles" on page 118. If the `-p` option is not provided, the name of the profile is the base name of the *filename* file. The `-p` option is not valid when more than one *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 `solaris11_1-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.

```
$ pfexec installadm create-profile -f ./myprofiles/profile_t200.xml \
-p t200 -n solaris11_1-sparc -c platform="SUNW,Sun-Fire-T200"
```

Updating a System Configuration Profile

Use the `installadm update-profile` command to replace the specified profile from the *svcname* install service with the contents of *filename*. Any criteria remain with the profile following the update.

```
installadm update-profile -n|--service svcname
    -f|--file filename
    [-p|--profile profile]
```

The profile to be updated is the *profile* profile from the *svcname* install service, if specified. If *profile* is not specified from the *svcname* install service, the name of the profile to be updated is the profile with the base name of *profile*.

The following command updates the content of the `t200` profile in the `solaris11_1-sparc` service with the content of `./myprofiles/profile_newt200.xml`.

```
$ pfexec installadm update-profile -n solaris11_1-sparc \
-f ./myprofiles/profile_newt200.xml -p t200
```

Validating a System Configuration Profile

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

```
installadm validate -n svcname -P filename... | -p profile...
```

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

Use the `-p` option to validate profiles that have already been added to the *svcname* install service using the `create-profile` subcommand. Use the `installadm list` command, as shown in "Listing All AI Manifests and System Configuration Profiles" on page 118, to display possible values for *profile*. 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 *filename* and *profile* 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`.

Deleting a System Configuration Profile

Use the `installadm delete-profile` command to remove the *profile* system configuration profile from the *svcname* install service. The value of the *profile* argument is the profile name that the `installadm list` command returns. See "Listing All AI Manifests and System Configuration Profiles" on page 118.

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

The following command removes the `t200` system configuration profile from the `solaris11_1-sparc` install service.

```
$ pfexec installadm delete-profile -p t200 -n solaris11_1-sparc
```

Exporting 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... -p profile...
    [-o pathname]
```

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

The *manifest* can be the name of 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 for the following tasks:

- 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.

Modifying 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 with system configuration profiles that you already added to the *svcname* install service using `create-manifest` or `create-profile`.

```
installadm set-criteria -m manifest -p profile... -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* and *profile* names are the names that the `installadm list` command returns. See "Listing All AI Manifests and System Configuration Profiles" on page 118.

If the `-m` option specifies the default manifest for the service, the criteria are added or changed, but the criteria are ignored when installing clients. See "Listing All AI Manifests and System Configuration Profiles" on page 118.

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:

```
$ pfexec installadm set-criteria -m t200 -n solaris11_1-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>
```

Showing Information About Install Services

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

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

Listing All Install Services on the Install Server

The following command displays all of the install services on this server. In this example, four enabled install services are found. Disabled services have a Status value of `off`.

```
$ installadm list

Service Name      Alias Of          Status  Arch   Image Path
------------      --------          ------  ----   ----------
default-i386      solaris11_1-i386  on      i386   /export/auto_install/solaris11_1-i386
default-sparc     solaris11_1-sparc on      sparc  /export/auto_install/solaris11_1-sparc
solaris11_1-i386  -                 on      i386   /export/auto_install/solaris11_1-i386
solaris11_1-sparc -                 on      sparc  /export/auto_install/solaris11_1-sparc
```

The default-i386 service was created automatically when the first i386 service was created on this server. The default-i386 service is used by any x86 client that has not been associated with the solaris11_1-i386 service by using the create-client subcommand. The default-i386 and solaris11_1-i386 services share a net image, but they have different AI manifests and system configuration profiles.

The default-sparc service was created automatically when the first sparc service was created on this server. The default-sparc service is used by any SPARC client that has not been associated with the solaris11_1-sparc service by using the create-client subcommand. The default-sparc and solaris11_1-sparc services share a net image, but they have different AI manifests and system configuration profiles.

Showing Information for a Specified Install Service

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

```
$ installadm list -n solaris11_1-sparc

Service Name      Alias Of Status  Arch   Image Path
------------      -------- ------  ----   ----------
solaris11_1-sparc -        on      sparc  /export/auto_install/solaris11_1-sparc
```

Listing 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 "Adding a Client To an Install Service" on page 108.

```
$ installadm list -c

Service Name      Client Address    Arch  Image Path
------------      --------------    ----  ----------
solaris11_1-sparc 00:14:4F:A7:65:70 sparc /export/auto_install/solaris11_1-sparc
solaris11_1-i386  08:00:27:8B:BD:71 i386  /export/auto_install/solaris11_1-i386
                  01:C2:52:E6:4B:E0 i386  /export/auto_install/solaris11_1-i386
```

Listing 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 solaris11_1-sparc install service.

```
$ installadm list -c -n solaris11_1-sparc

Service Name      Client Address    Arch  Image Path
------------      --------------    ----  ----------
solaris11_1-sparc 00:14:4f:a7:65:70 sparc /export/auto_install/solaris11_1-sparc
```

Showing 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.

Listing 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 Service/Manifest Name and Service/Profile Name columns display the internal names of the manifests, scripts, or profiles. The Status column identifies the default manifest for each service and any inactive manifests. A manifest is inactive if it does not have any associated criteria and also is not the default. 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 the `mem1` 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
Service/Manifest Name  Status    Criteria
---------------------  ------    --------
default-i386
   orig_default        Default   None

default-sparc
   orig_default        Default   None

solaris11_1-i386
   ipv4                          ipv4     = 10.6.68.1 - 10.6.68.200
   mem1                Default   (Ignored: mem      = 2048 MB - 4095 MB)
   orig_default        Inactive  None

solaris11_1-sparc
   t200                          mem      = 4096 MB - unbounded
                                 platform = SUNWSun-Fire-T200
   mem1                Default   (Ignored: mem      = 2048 MB - 4095 MB)
   orig_default        Inactive  None

Service/Profile Name  Criteria
--------------------  --------
solaris11_1-i386
   mac2               mac      = 08:00:27:8B:BD:71
                      hostname = server2
   mac3               mac      = 01:C2:52:E6:4B:E0
```

```
                      hostname = server3
   ipv4               ipv4     = 10.0.2.100 - 10.0.2.199
   mem1               mem      = 2048 MB - 4095 MB

solaris11_1-sparc
   mac1               mac      = 01:C2:52:E6:4B:E0
                      hostname = server1
                      ipv4     = 192.168.168.251
   t200               platform = SUNWSun-Fire-T200
                      mem      = 4096-unbounded
```

Listing 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 `solaris11_1-sparc`.

```
$ installadm list -m -p -n solaris11_1-sparc
Service/Manifest Name  Status   Criteria
---------------------  ------   --------
solaris11_1-sparc
   t200                         mem      = 4096 MB - unbounded
                                platform = SUNWSun-Fire-T200
   mem1                 Default  (Ignored: mem      = 2048 MB - 4095 MB)
   orig_default         Inactive None

Service/Profile Name  Criteria
--------------------  --------
solaris11_1-sparc
   mac1               mac  = 01:C2:52:E6:4B:E0
                      hostname = server1
                      ipv4 = 192.168.168.251
   t200               platform = SUNWSun-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 8–1 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 8–2 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 "Listing All Install Services on the Install Server" on page 116 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 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 over the network, it uses an install service from the install server.

The client uses the default install service for that client architecture or 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 14, "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, the client is directed to the install server and 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 "Associating Client-Specific Installation Instructions With Install Services" on page 110.

For information about how to create custom AI manifests, see Chapter 10, "Provisioning the Client System." For information about how 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 127 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. For more information, see "Adding an AI Manifest to an Install Service" on page 110.

  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. 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

Table 9–1 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`.

The `ipv4`, `mac`, `mem`, and `network` specifications can be expressed as ranged values separated by a hyphen (`-`). To specify no limit to one end of a range, use `unbounded`. See the `mem` example below.

The `arch`, `cpu`, `hostname`, `platform`, and `zonename` specifications can be expressed as a quoted list of values separated by white space. See the `zonename` example below.

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 mem="2048-unbounded"
-c zonename="zone1 zone2"
```

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>`

TABLE 9–1 Criteria Keywords and Criteria Hierarchy *(Continued)*

Criteria Name	Description	Command Line and XML File Examples

`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>`

`platform`	Platform name returned by `uname -i` for x86 systems and `prtconf -b` for SPARC systems
	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>`

TABLE 9–1 Criteria Keywords and Criteria Hierarchy *(Continued)*

Criteria Name	Description	Command Line and XML File Examples
`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>`
`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>`

TABLE 9–1 Criteria Keywords and Criteria Hierarchy *(Continued)*

Criteria Name	Description	Command Line and XML File Examples

`zonename`	Name or list of names of 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>`

`hostname`	Client host name or list of client host names.	**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 in the below example. 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 the locales that are 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 11.1 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 11.1 Software Packages*.

```
<?xml version="1.0" encoding="UTF-8"?>
<!--

 Copyright (c) 2008, 2012, Oracle and/or its affiliates. All rights reserved.

-->
<!DOCTYPE auto_install SYSTEM "file:///usr/share/install/ai.dtd.1">
<auto_install>
  <ai_instance name="default">
    <target>
      <logical>
        <zpool name="rpool" is_root="true">
          <!--
            Subsequent <filesystem> entries instruct an installer to create
            following ZFS datasets:

                <root_pool>/export         (mounted on /export)
                <root_pool>/export/home    (mounted on /export/home)

            Those datasets are part of standard environment and should be
            always created.

            In rare cases, if there is a need to deploy an installed system
            without these datasets, either comment out or remove <filesystem>
            entries. In such scenario, it has to be also assured that
            in case of non-interactive post-install configuration, creation
```

```
              of initial user account is disabled in related system
              configuration profile. Otherwise the installed system would fail
              to boot.
            -->
            <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>
        <!--
          The version specified by the "entire" package below, is
          installed from the specified IPS repository.  If another build
          is required, the build number should 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@0.5.11-0.175.1</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 AI manifest a particular client uses depends on the criteria specified when the manifest is added to the install service. If the client does not match any criteria to use a custom AI manifest, the default manifest is used. Any AI manifest in a service can be designated to be the default for that service.

Customizing an XML AI Manifest File

Use the following procedure to create and apply a custom XML AI manifest file:

▼ How to Customize an XML AI Manifest File

1 Copy an existing AI manifest.

When you create an AI install service, that install service has a default AI manifest. See Chapter 8, "Setting Up an Install Server," for information about creating an install service.

a. List existing manifests.

Use the installadm list command to see what AI manifests you already have associated with a particular install service.

```
$ installadm list -m -n solaris11_1-i386
Service/Manifest Name  Status   Criteria
---------------------  ------   --------
solaris11_1-i386
   orig_default        Default  None
```

b. Retrieve a copy of a specific manifest.

Use the installadm export command to extract the contents of this default manifest or any other AI manifest that has been added to this service.

```
$ pfexec installadm export -n solaris11_1-i386 -m orig_default -o mem1.xml
```

A copy of orig_default is now in the file mem1.xml.

2 Modify the manifest copy.

Modify mem1.xml, adding tags and values according to the information in the ai_manifest(4) man page.

3 Add the new manifest to the install service.

Add the new AI manifest to the appropriate AI install service, specifying criteria that define which clients should use these installation instructions.

```
$ pfexec installadm create-manifest -n solaris11_1-i386 -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 -m -n solaris11_1-i386
Service/Manifest Name  Status   Criteria
---------------------  ------   --------
solaris11_1-i386
   orig_default        Default  None
```

```
mem1                         mem  = 2048 MB - unbounded
```

- **Make the new manifest the default.**

  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.

  ```
  $ pfexec installadm set-service -o default-manifest=mem1 solaris11_1-i386
  $ installadm list -m -n solaris11_1-i386
  Service/Manifest Name  Status   Criteria
  ---------------------  ------   --------
  solaris11_1-i386
     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.

- **Add the new manifest as the default.**

  If you want to add a new default manifest or script for this service, use the `-d` option with `create-manifest`. Any criteria specified are stored and ignored until another manifest is made the default.

  ```
  $ pfexec installadm create-manifest -n solaris11_1-i386 -d \
  -f ./region1.xml -m region1
  $ installadm list -m -n solaris11_1-i386
  Service/Manifest Name  Status   Criteria
  ---------------------  ------   --------
  solaris11_1-i386
     orig_default        Inactive None
     mem1                         mem  = 2048 MB - unbounded
     region1             Default  None
  ```

- **Customize an existing manifest.**

  Use the `installadm update-manifest` command to change the content of an existing manifest or script without adding a new manifest or script. Criteria, default status, and the manifest name or the script name are not changed as a result of the update.

  ```
  $ pfexec installadm update-manifest -n solaris11_1-i386
      -f ./newregion1.xml -m region1
  ```

4 **Validate the customized manifest.**

The `create-manifest` and `update-manifest` subcommands syntactically validate the XML manifest files before adding them to the install service. AI semantically 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 also "Maintaining an Install Server" on page 103 for more information about the `installadm` `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.

▼ How to Create and Apply a Derived Manifests Script

1. **Select a manifest to modify.**

   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 modify the manifest.**

   Write a script to dynamically modify the base manifest at installation time based on attributes of the client being installed.

3. **Add the script to the install service.**

   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. If you do not want to specify client selection criteria, you can add this script as the default AI manifest for the service.

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/log/install/derived/manifest.xml` on the client, and the script used to derive the manifest is copied to `/var/log/install/derived/manifest_script`.

Creating 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.

Retrieving 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 III, "Roles, Rights Profiles, and Privileges," in *Oracle Solaris 11.1 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`.

TABLE 10–1 Client Attribute Environment Variables *(Continued)*

Environment Variable Name	Description
`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 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_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*).
`SI_PLATFORM (or SI_MODEL)`	The platform of the client. Equivalent to the output of `uname -i` for x86 systems and `prtconf -b` for SPARC systems.

Customizing the AI Manifest

To add or modify XML elements in an AI manifest, use the `/usr/bin/aimanifest` command.

A file to be modified by `aimanifest` must contain at least 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:///imagepath/auto_install/ai.dtd.#">
<auto_install/>
```

The # is an integer such as 1. The value of the *imagepath* argument is the path returned by the following command, where *svcname* is the name of the install service where this derived manifests script will be added:

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

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 *imagepath*`/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. The man page and the example scripts that follow provide examples of using the `aimanifest` command.

Note – If a value specified in an `aimanifest` command contains one of the following characters, then that value must be enclosed in single or double quotation marks to prevent the character from being interpreted as part of the XML path name:

```
/'"@[]=
```

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
```

Examples of 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 Specifying Disk Partitioning Based on Disk Size

This example customizes the AI manifest to use only half of the target disk on an Oracle Solaris `fdisk` partition if the size of the disk is greater than 1 TB. Try setting `SI_DISKSIZE_1` to less

EXAMPLE 10–1 Specifying Disk Partitioning Based on Disk Size *(Continued)*

than 1 TB and then greater than 1 TB for different runs of this script. Also set `SI_NUMDISKS` and `SI_DISKNAME_1` before you run the script. Note that this script is only for use with x86 clients because the specified partitioning only applies to x86 clients.

```
#!/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

# Check that there is only one disk on the system.
if [[ $SI_NUMDISKS -gt "1" ]] ; then
    print -u2 "System has too many disks for this script."
    exit $SCRIPT_FAILURE
fi

/usr/bin/aimanifest add \
    /auto_install/ai_instance/target/disk/disk_name@name $SI_DISKNAME_1

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[disk_name@name=\"$SI_DISKNAME_1\"]/partition@name 1
    /usr/bin/aimanifest add \
        /auto_install/ai_instance/target/disk/partition[@name=1]/size@val \
        ${PARTN_SIZE}mb
else
    /usr/bin/aimanifest add \
        /auto_install/ai_instance/target/disk[disk_name@name=\"$SI_DISKNAME_1\"]/partition@action \
        use_existing_solaris2
fi
exit $SCRIPT_SUCCESS
```

For clients where the value of `SI_DISKSIZE_1` is less than or equal to 1048576, the following elements are added to `$AIM_MANIFEST`:

```
<target>
  <disk>
    <disk_name name="/dev/dsk/c0t0d0s0"/>
    <partition action="use_existing_solaris2"/>
  </disk>
  <!-- <logical> section -->
</target>
```

EXAMPLE 10–1 Specifying Disk Partitioning Based on Disk Size *(Continued)*

For clients where the value of `SI_DISKSIZE_1` is greater than 1048576, elements similar to the following are added to `$AIM_MANIFEST`, depending on the value of `SI_DISKSIZE_1`:

```
<target>
  <disk>
    <disk_name name="/dev/dsk/c0t0d0s0"/>
    <partition name="1">
      <size val="524288mb"/>
    </partition>
  </disk>
  <!-- <logical> section -->
</target>
```

The `disk_name` is specified in the command to add the partition to avoid creating a separate disk specification for the partition. The script in this example specifies that the partition is on the `$SI_DISKNAME_1` disk, not on a different disk. If the appropriate lines in this example are replaced by the following lines, you do not get the result you intend:

```
    /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
```

Instead of the output shown above, this script would give you the following incorrect output:

```
<target>
  <disk>
    <disk_name name="c0t0d0s0"/>
  </disk>
  <disk>
    <partition name="1">
      <size val="524288mb"/>
    </partition>
  </disk>
</target>
```

EXAMPLE 10–2 Specifying 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.

EXAMPLE 10–2 Specifying the Root Pool Layout Based on the Existence of Additional Disks *(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

# 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)
        /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 element:

```
<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" is_root="true">
      <vdev name="mirror_vdev" redundancy="mirror"/>
      <filesystem name="export" mountpoint="/export"/>
      <filesystem name="export/home"/>
      <be name="solaris"/>
    </zpool>
  </logical>
</target>
```

EXAMPLE 10–3 Specifying 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 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
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
```

EXAMPLE 10–3 Specifying a Mirrored Configuration If at Least Two Disks of a Specified Size Are Present *(Continued)*

```
    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 Korn shell version.

```
#!/usr/bin/python2.6

import os
import sys

from subprocess import check_call, CalledProcessError

SCRIPT_SUCCESS = 0
SCRIPT_FAILURE = 1

def main():

    # Find the disks first.
    found_1 = ""
    found_2 = ""

    si_numdisks = int(os.environ["SI_NUMDISKS"])
    for disk_num in range(1, si_numdisks + 1):
        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:
        check_call(["/usr/bin/aimanifest", "load",
            "/usr/share/auto_install/manifest/default.xml"])
    except CalledProcessError as err:
        sys.exit(err.returncode)
```

EXAMPLE 10–3 Specifying a Mirrored Configuration If at Least Two Disks of a Specified Size Are Present *(Continued)*

```
    if len(found_2):
        try:
            check_call(["/usr/bin/aimanifest", "add",
               "target/logical/zpool[@name=rpool]/vdev@redundancy", "mirror"])
            check_call(["/usr/bin/aimanifest", "set",
               "target/logical/zpool/vdev[@redundancy='mirror']@name", "mirror_vdev"])

            check_call(["/usr/bin/aimanifest", "add",
                "target/disk/disk_name@name", found_1])
            check_call(["/usr/bin/aimanifest", "set",
                "target/disk/disk_name[@name='" + found_1 + "']" + "@name_type", "ctd"])
            check_call(["/usr/bin/aimanifest", "set",
                "target/disk[disk_name@name='" + found_1 + "']" + "@in_vdev", "mirror_vdev"])

            check_call(["/usr/bin/aimanifest", "add",
                "target/disk/disk_name@name", found_2])
            check_call(["/usr/bin/aimanifest", "set",
                "target/disk/disk_name[@name='" + found_2 + "']" + "@name_type", "ctd"])
            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 Specifying 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.

```
#!/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
```

EXAMPLE 10–4 Specifying Packages to Install Based on IP Address *(Continued)*

```
# 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 Specifying that the Target Disk Must Be At Least a Certain Size

This example customizes the AI manifest to install only 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

/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
```

EXAMPLE 10–5 Specifying that the Target Disk Must Be At Least a Certain Size *(Continued)*

```
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
```

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 the package to be modified 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. Although 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
```

EXAMPLE 10–6 Script With Incorrect Manifest Specifications *(Continued)*

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:///imagepath/auto_install/ai.dtd.#">
   <auto_install/>
   ```

   The # is an integer such as 1. The value of the *imagepath* argument is the path returned by the following command, where *svcname* is the name of the install service where this derived manifests script will be added:

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

   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 addresses some of these differences.

▼ How to Test the Derived Manifests Script

This procedure describes how to test the derived manifests script on one of the intended client systems.

1. **Boot an AI image on that client system.**

   Boot an AI image on that client system in "Text Installer and command line" mode.

2. **Select Shell from the installer initial menu.**

3. **Copy your script from the AI install server.**

   Use `wget` or `sftp` to copy your script from the AI install server.

4. **Debug the script.**

   Use one of the following methods to debug the script:

   - **Run the script manually.**

- **Run AI in a test mode.**

  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 that the script validates:

```
Derived Manifest Module: XML validation completed successfully
```

5 **Copy the script back to the install server.**

Copy the script back to the install server, if changes have been made.

Adding a Derived Manifests Script to an Install Service

Add a script to an AI install service in the same way that 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. If you do not want to specify client selection criteria, you can use the `-d` option to add this script as the default AI manifest for the service.

```
$ pfexec installadm create-manifest -n solaris11_1-i386 -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 "Maintaining an Install Server" on page 103 for information about the `installadm list`, `export`, `create-manifest`, `set-criteria`, `update-manifest`, and `set-service` subcommands.

Example AI Manifests

The examples in this section show the XML elements that the finished AI manifest must have to achieve the stated result. These manifests can be created either by editing the XML directly or by using a derived manifests script.

All manifests shown in this section are based on the `default.xml` manifest, with the necessary modifications made. The `destination` element in the `software` element is omitted for brevity.

Specifying an iSCSI Target Device

In this example, the target for the installation is an iSCSI device. Use the `iscsi` element in the `disk` element in the `target` element. The `whole_disk` attribute of the `disk` element is set to `true`, which is typical for iSCSI disks. See the `ai_manifest(4)` man page for descriptions of the `target_name`, `target_lun`, and `target_ip` attributes.

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
 Copyright (c) 2008, 2012, Oracle and/or its affiliates. All rights reserved.
-->
<!DOCTYPE auto_install SYSTEM "file:///usr/share/install/ai.dtd.1">
<auto_install>
  <ai_instance name="default">
    <target>
      <disk whole_disk="true">
        <iscsi target_name="iqn.1986-03.com.sun:02:1234567890abcdef" \
               target_lun="1" target_ip="129.158.144.200"/>
      </disk>
      <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@0.5.11-0.175.1</name>
        <name>pkg:/group/system/solaris-large-server</name>
      </software_data>
    </software>
  </ai_instance>
</auto_install>
```

Specifying a RAID Configuration

This example specifies a RAID configuration using the two disks c0t0d0 and c0t1d0. This manifest is similar to the manifest for a mirrored configuration as shown in Example 10–3. One difference between the two manifests is that the value of the redundancy attribute is raidz instead of mirror. See the zpool(1M) man page for information about redundancy types. Another difference is that the ZFS pool is not named rpool, because rpool implies the root pool. By default, the value of the is_root attribute of the zpool element is false, so that assignment could be omitted in this example. Because no root pool is specified, do not configure an initial user for this installation.

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
 Copyright (c) 2008, 2012, Oracle and/or its affiliates. All rights reserved.
-->
<!DOCTYPE auto_install SYSTEM "file:///usr/share/install/ai.dtd.1">
<auto_install>
  <ai_instance name="default">
    <target>
      <disk in_vdev="raid_vdev" in_zpool="raidpool" whole_disk="true">
        <disk_name name="c0t0d0" name_type="ctd"/>
      </disk>
      <disk in_vdev="raid_vdev" in_zpool="raidpool" whole_disk="true">
        <disk_name name="c0t1d0" name_type="ctd"/>
      </disk>
      <logical>
        <zpool name="raidpool" is_root="false">
          <vdev name="raid_vdev" redundancy="raidz"/>
        </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@0.5.11-0.175.1</name>
        <name>pkg:/group/system/solaris-large-server</name>
      </software_data>
    </software>
  </ai_instance>
</auto_install>
```

Installing an SVR4 Package

This example demonstrates how to install a SVR4 package. SVR4 packages must be named in a software element of type SVR4. The value of the name attribute of the origin of the publisher is a directory that contains SVR4 package subdirectories or a SVR4 package datastream file. This origin name for SVR4 package subdirectories can be a full file directory path or a file URI. This origin name for a SVR4 package datastream file can be a full file directory path, a file URI, or an HTTP URI.

Tip – Do not install packages that require user input as part of their installation.

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
 Copyright (c) 2008, 2012, Oracle and/or its affiliates. All rights reserved.

-->
<!DOCTYPE auto_install SYSTEM "file:///usr/share/install/ai.dtd.1">
<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">
      <source>
        <publisher name="solaris">
          <origin name="http://pkg.oracle.com/solaris/release"/>
        </publisher>
      </source>
      <software_data action="install">
        <name>pkg:/entire@0.5.11-0.175.1</name>
        <name>pkg:/group/system/solaris-large-server</name>
      </software_data>
    </software>
    <software type="SVR4">
      <source>
        <publisher>
          <origin name="/net/host2/usr/dist"/>
        </publisher>
      </source>
      <software_data>
        <name>SUNWpackage</name>
      </software_data>
    </software>
  </ai_instance>
</auto_install>
```

◆ ◆ ◆ 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 by using `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 69 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 73 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 155 and "Example System Configuration Profiles" on page 167.

  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 used to install a client system. See "Using System Configuration Profile Templates" on page 166.

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 "Validating a System Configuration Profile" on page 114 for more information about the `validate` subcommand.

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.

Use the `installadm update-profile` command to replace the contents of a profile that has already been added to an install service.

See "Maintaining an Install Server" on page 103 and the `installadm(1M)` man page for more information about the `create-profile`, `update-profile`, `list`, `set-criteria`, and `export` subcommands.

Specifying Configuration in a System Configuration Profile

You can specify configuration of any system characteristic that is configurable by using `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 name service (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 which 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 the `smf_template(5)` man page), 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
```

Configuring Root and User Accounts

Enter the following `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 properties listed in the following table.

TABLE 11–1 root_account Property Group Properties

Property	Type	Required	Description
`password`	`astring`	required	Encrypted root password. If you do not provide a root password, the root password is empty.
`type`	`astring`	optional	Account type: `normal` or `role`. The default is `normal`.
`expire`	`string`	optional	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

This section includes the following information:

- "Creating a User Account Without Depending on the Automounter" on page 158
- "User Account Properties" on page 158
- "Configuring Multiple Initial Users" on page 159

Creating a User Account Without Depending on the Automounter

By default, when initial user accounts are created, the home directories are managed by the automounter and accessed under `/home/`*login* directories. To create initial user accounts without depending on the automounter, set the `user_account/autohome` property to the empty string ("") in the configuration profile.

Setting the `user_account/autohome` property to the empty string has the following effects:

- The home directory entry in the `/etc/passwd` file is set to the mount point of the home ZFS dataset, not to `/home/`*login*. The default mount point of the home ZFS dataset is `/export/home/`*login*.
- No mapping entry is added to the `/etc/auto_home` file.

User Account Properties

The `user_account` property group contains the properties listed in the following table.

TABLE 11–2 user_account Property Group Properties

Property	Type	Required	Description
`login`	`astring`	`required`	User's login.
`password`	`astring`	`required`	Encrypted user password.
`description`	`astring`	`optional`	Usually the user's full name.
`shell`	`astring`	`optional`	Full path name of the program used as the user's shell on login.
`uid`	`count`	`optional`	UID of the new user. The default UID is 101.
`gid`	`count`	`optional`	User's primary group membership. The default GID is 10.
`type`	`astring`	`optional`	Account type: `normal` or `role`. The default is `normal`.
`profiles`	`astring`	`optional`	One or more comma-separated execution profiles defined in the `prof_attr`(4) man page.
`roles`	`astring`	`optional`	One or more comma-separated roles defined in the `user_attr`(4) man page.
`sudoers`	`astring`	`optional`	Entry added to the `sudoers` file along with the `login`.

TABLE 11–2 user_account Property Group Properties *(Continued)*

Property	Type	Required	Description
`expire`	`astring`	`optional`	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`	`optional`	User's home directory ZFS dataset. The default is *root_pool*`/export/home/`*login*.
`home_mountpoint`	`astring`	`optional`	User's home directory mount point. The default is `/export/home/`*login*.
`autohome`	`astring`	`optional`	User's auto home directory mount point. The value is entered in the `/etc/auto_home` file for the configured user. The default value is `localhost:/export/home/`*login*. If the `autohome` property is set to the empty string (`""`), the user account is created without depending on the automounter.

Configuring Multiple Initial Users

To configure multiple users on the newly-installed system, specify the users by using the `useradd`(1M) command in a script. Then use a run-once SMF service to run the script at first boot. See Chapter 13, “Running a Custom Script During First Boot,” for instructions.

Setting the 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 properties listed in the following table.

TABLE 11–3 config Property Group Properties

Property	Type	Required	Description
`nodename`	`astring`	optional	System host name. The default is `solaris`.
`enable_mapping`	`boolean`	optional	Value used to disable node name mapping. The default is `true`.
`loopback`	`astring`	optional	Host name mapped to loopback. The default is `solaris`.

EXAMPLE 11–2 Configuring the Host Name

This example sets the system host name to `solaris`.

EXAMPLE 11–2 Configuring the Host Name *(Continued)*

```
<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>
```

EXAMPLE 11–3 Disabling Node Name Mapping

When you install the Oracle Solaris 11 OS or an Oracle Solaris 11 update release, by default the system node name is mapped to the loopback or to the IP address of the interface configured as part of installation. You can disable this default mapping by setting the `enable_mapping` property to `false`, as shown in the following example.

```
<service name="system/identity" version="1" type="service">
    <instance name="node" enabled="true">
        <property_group name="config" type="application">
            <propval name="nodename" value="solaris"/>
            <propval name="enable_mapping" value="false"/>
        </property_group>
    </instance>
</service>
```

Setting the 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 properties listed in the following table.

TABLE 11–4 timezone Property Group Properties

Property	Type	Required	Description
`localtime`	`astring`	optional	System time zone. The default is `UTC`.

EXAMPLE 11–4 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'>
```

EXAMPLE 11–4 Configuring the Time Zone *(Continued)*

```
      <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 the 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`	optional	`C`
`LC_NUMERIC`	`astring`	optional	`C`
`LC_TIME`	`astring`	optional	`C`
`LC_COLLATE`	`astring`	optional	`C`
`LC_MONETARY`	`astring`	optional	`C`
`LC_MESSAGES`	`astring`	optional	`C`
`LC_ALL`	`astring`	optional	`C`
`LANG`	`astring`	optional	`C`

EXAMPLE 11–5 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>
```

Setting the Terminal Type and Keyboard Layout

EXAMPLE 11–6 Configuring Terminal Type

The `svc:/system/console-login` SMF service configures the terminal type. See the `ttymon`(1M) man page for definition of related SMF properties.

This example sets the terminal type to `vt100`.

EXAMPLE 11–6 Configuring Terminal Type *(Continued)*

```
<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 11–7 Configuring Keyboard Layout

Use the `sysconfig create-profile` command with the `kbd_layout` grouping to generate a valid profile that configures the keyboard layout.

```
# sysconfig create-profile -g kbd_layout -o sc_kdb.xml
```

The `svc:/system/keymap` SMF service configures the 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>
```

Configuring a Static Network

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.

Note – If the installation target is an iSCSI device, do not configure that network interface in any system configuration profile for that installation. For iSCSI boot, the network interface for the iSCSI device is configured early in the client boot process. If you configure that same interface again, the `network/install` service for the interface goes into maintenance state.

To configure multiple network interfaces, specify the configuration in a script, and use a run-once SMF service to run the script at first boot. See Chapter 13, "Running a Custom Script During First Boot," for instructions and a sample script.

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 169.

The `install_ipv4_interface` property group contains the properties listed in the following table.

TABLE 11–6 install_ipv4_interface Property Group Properties

Property	Type	Required	Description
`name`	`astring`	`required`	Name of the network interface.
`address_type`	`astring`	`required`	Value used to construct the `-T` option for the `ipadm`(1M) `create-addr` subcommand. Valid values are `static` or `dhcp`.
`static_address`	`net_address_v4`	`optional`	Only required with an `address_type` of `static`. Used to construct the local address for the `ipadm`(1M) `create-addr` subcommand.
`dhcp_wait`	`astring`	`optional`	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`	`optional`	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 properties listed in the following table.

TABLE 11–7 install_ipv6_interface Property Group Properties

Property	Type	Required	Description
`name`	`astring`	`required`	Name of the network interface.
`address_type`	`astring`	`required`	Value used to construct the `-T` option for the `ipadm`(1M) `create-addr` subcommand. Valid values are `static` or `addrconf`.

TABLE 11–7 install_ipv6_interface Property Group Properties *(Continued)*

Property	Type	Required	Description
`static_address`	`net_address_v6`	`optional`	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`	`optional`	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`	`optional`	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`	`optional`	Only applies with an `address_type` of `addrconf`. Used to construct the `-p stateful=yes|no` portion of the `ipadm`(1M) `create-addr` subcommand.
`default_route`	`net_address_v6`	`optional`	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 properties listed in the following table.

TABLE 11–8 config Property Group Properties

Property	Type	Required	Description
`domain`	`astring`	`optional`	Local domain name. Used to construct the `domain` directive in `resolv.conf`(4).
`nameserver`	`net_address_list`	`required`	List of IPv4 and IPv6 addresses. Used to construct the `nameserver` directives in `resolv.conf`(4).
`search`	`astring_list`	`optional`	List of domain values for the search list for host name lookup. Used to construct the `search` directive in `resolv.conf`(4).

Configuring Name Service

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 171.

Setting Up Oracle Configuration Manager and Oracle Auto Service Request

Oracle Configuration Manager enables you to log your system configurations with My Oracle Support, and Oracle Auto Service Request can automatically generate service requests for specific hardware faults.

Use the `sysconfig` `create-profile` command with the `support` grouping to generate a valid profile that configures Oracle Configuration Manager and Oracle Auto Service Request.

```
# sysconfig create-profile -g support -o sc_support.xml
```

The output profile sets up the first phase of registration, which is the same for all clients that match the following criteria:

- The systems use the same My Oracle Support credentials to register. All client systems that use this profile register with My Oracle Support in the same way. The data from all of these clients will be associated with the same My Oracle Support account.
- The systems access My Oracle Support through the same network configuration. All client systems that use this profile access My Oracle Support through the same proxy servers and aggregation hubs, for example.

If you need to create additional profiles for different groups of AI client systems, you should rerun the `sysconfig` `create-profile` command, rather than copy and edit an existing profile. If your proxy server has a user name and password, then you must rerun `sysconfig create-profile` since the passwords are encrypted.

Using System Configuration Profile Templates

Profiles can contain variables that are replaced with values from the client's installation environment during the installation process. In this way, a single profile file can set different configuration parameters on different clients. See Table 11–9 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, and `AI_IPV4` is a placeholder for the client system's IP address.

```
<?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="10.0.0.1"/>
      </property_group>
    </instance>
  </service>
</service_bundle>
```

The following command creates a system configuration profile in the install service that will be customized for each installation client without changing the input `hostandIP.xml` file.

```
$ pfexec installadm create-profile -n solaris11_1-i386 -f /export/hostIPnet.xml
```

While the `hostandIP.xml` file remains unchanged, the profiles that are applied to a client are customized. For example, the `hostandIP.xml` profile might have the following content when a client with host name `server1` is installed:

```
<?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="10.0.0.2/8"/>
        <propval name="default_route" type="net_address_v4" value="10.0.0.1"/>
      </property_group>
    </instance>
  </service>
</service_bundle>
```

The following table shows the variables that can be used as placeholders in template profiles.

TABLE 11–9 Variables for System Configuration Template Profiles

Variable Name	Description
`AI_ARCH`	Kernel architecture from `uname -m`
`AI_CPU`	Processor type from `uname -p`
`AI_HOSTNAME`	Client DNS name
`AI_IPV4`	IP version 4 network address
`AI_MAC`	Hexadecimal MAC address with colon (:) separators
`AI_MEM`	Memory size in megabytes returned by `prtconf`
`AI_NETWORK`	IP version 4 network identifier
`AI_SERVICE`	Install service name
`AI_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, 2012, 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
* Time zone 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="uid" value="101"/>
          <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="init" enabled="true">
        <property_group name="environment">
```

```
            <propval name="LANG" 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

A version of this sample profile is available at `/usr/share/auto_install/sc_profiles/static_network.xml`. The version of this profile that is shown below is modified to configure 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` and `example2.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

```
<?xml version="1.0"?>
<!--
Copyright (c) 2011, 2012, Oracle and/or its affiliates. All rights reserved.
-->

<!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="$5$dnRfcZse$Hx4aBQ161Uvn9ZxJFKMdRiy8tCf4gMT2s2rtkFba2y4"/>
            <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="init" enabled="true">
        <property_group name="environment">
          <propval name="LANG" 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="bge0/v4"/>
                <propval name="address_type" type="astring" value="static"/>
                <propval name="static_address" type="net_address_v4" value="10.0.0.10/8"/>
                <propval name="default_route" type="net_address_v4" value="10.0.0.1"/>
            </property_group>

            <property_group name="install_ipv6_interface" type="application">
                <propval name="name" type="astring" value="bge1/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="8.8.8.8"/>
                </net_address_list>
            </property>
            <property name="search">
                <astring_list>
                    <value_node value="example1.com example2.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 73 and the `sysconfig`(1M) man page for more information about using `sysconfig` to create a system configuration profile.

Configuring Name Service NIS

EXAMPLE 11–8 Enabling 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>
    <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 11–9 Configuring NIS and Disabling 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">
```

EXAMPLE 11–9 Configuring NIS and Disabling DNS *(Continued)*

```
  <!-- 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">
    <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 11–10 Configuring 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>
```

EXAMPLE 11–10 Configuring NIS *(Continued)*

```
  <!-- 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 -->
  <service version="1" type="service" name="network/nis/client">
    <instance enabled="true" name="default"/>
  </service>
</service_bundle>
```

EXAMPLE 11–11 Enabling 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.example.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.example.com example.com example.com' />
            </astring_list>
        </property>
    </property_group>
```

EXAMPLE 11–11 Enabling NIS and DNS For a Specified Domain *(Continued)*

```
    <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'/>
        <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 11–12 Configuring DNS With a 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"/>
```

EXAMPLE 11–12 Configuring DNS With a Search List *(Continued)*

```
  </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>
    <instance enabled="true" name="default"/>
  </service>
</service_bundle>
```

Configuring Name Service LDAP

EXAMPLE 11–13 Configuring 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 (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">
```

EXAMPLE 11–13 Configuring LDAP and LDAP Search Base *(Continued)*

```
        <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 11–14 Configuring 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 (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"/>
```

EXAMPLE 11–14 Configuring LDAP With a Secure LDAP Server *(Continued)*

```
      <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"/>
  </service>
</service_bundle>
```

Using DNS With LDAP

The DNS name service can be used in conjunction with the 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 "Specifying Non-Global Zones in the Global Zone AI Manifest" on page 182 for information about the `configuration` element.
2. When the system first boots after the global zone installation, the zone's 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 183 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

Specifying Non-Global Zones 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 184 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 187 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 69 for information about using the interactive configuration tool.

The following example adds the `/tmp/zmanifest.xml` AI manifest to the `solaris11_1-sparc` install service and specifies that `zone1` and `zone2` should use this manifest.

```
$ pfexec installadm create-manifest -n solaris11_1-sparc -f /tmp/zmanifest.xml \
-m zmanifest -c zonename="zone1 zone2"
```

The following example adds the `/tmp/z1profile.xml` configuration profile to the `solaris11_1-sparc` install service and specifies that `zone1` and `zone2` should use this profile.

```
$ pfexec installadm create-profile -n solaris11_1-sparc -f /tmp/z1profile.xml \
-p z1profile -c zonename="zone1 zone2"
```

The following example adds the `/tmp/z2profile.xml` configuration profile to the `solaris11_1-sparc` install service and specifies that `zone2` should use this profile.

```
$ pfexec installadm create-profile -n solaris11_1-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 `solaris11_1-sparc` install service.

```
$ installadm list -n solaris11_1-sparc -m -p
Service/Manifest Name  Status   Criteria
---------------------  ------   --------
solaris11_1-sparc
   orig_default        Default  None
   line1-netra2000              mac       = 00:14:4F:2D:7A:DC
   zmanifest                    zonename  = zone1 zone2

Service/Profile Name  Criteria
--------------------  --------
solaris11_1-sparc
   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`.

The `target` section defines a ZFS file system for the zone. The `destination` section specifies which locales to install. The `software_data` section specifies installing the `solaris-small-server` package. The `solaris-small-server` package is a group package of tools and device drivers that you might want in most non-global zones that you install. For a complete list of packages that are included in the `solaris-small-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 11.1 Software Packages*.

Notice that no package source is specified. See `pkg.sysrepo`(1M) for information about the system repository.

```
<?xml version="1.0" encoding="UTF-8"?>
<!--

 Copyright (c) 2011, 2012, Oracle and/or its affiliates. All rights reserved.

-->
<!DOCTYPE auto_install SYSTEM "file:///usr/share/install/ai.dtd.1">

<auto_install>
    <ai_instance name="zone_default">
        <target>
            <logical>
                <zpool name="rpool">
                    <!--
                      Subsequent <filesystem> entries instruct an installer
                      to create following ZFS datasets:

                          <root_pool>/export         (mounted on /export)
                          <root_pool>/export/home    (mounted on /export/home)

                      Those datasets are part of standard environment
                      and should be always created.

                      In rare cases, if there is a need to deploy a zone
                      without these datasets, either comment out or remove
```

EXAMPLE 12–1 Default Zone AI Manifest *(Continued)*

```
                      <filesystem> entries. In such scenario, it has to be also
                      assured that in case of non-interactive post-install
                      configuration, creation of initial user account is
                      disabled in related system configuration profile.
                      Otherwise the installed zone would fail to boot.
                    -->
                    <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>
                    <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 name services.

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.

Implementing Run Once at First Boot Controls

The following procedure shows how to ensure that the script runs only at the first boot of the newly installed system, and that the script runs only one time.

▼ How to Ensure One Run at First Boot

1 Create a service to run the script.

These easiest way to create this simple service is to use the `svcbundle` command as shown in "Using the Manifest Creation Tool" on page 193.

2 Set a script completion flag before the script runs.

Define a Boolean completion property in the service manifest, and set its value to `false`. See the `completed` property in the manifest in Example 13–3.

3 **Set the script completion flag at the end of the script.**

Use the svccfg command to set the completed property to true at the end of the script. Use the svcadm command to refresh the service with the new property value. See the end of the sample script in Example 13–1.

4 **Disable the service if the script completed.**

In the service manifest, the default service instance is created and enabled. The service is disabled in the script. When you exit your first-boot script, use the SMF_EXIT_TEMP_DISABLE exit code to exit the start method of the service and temporarily disable the service. The service is disabled, and the stop method of the service does not run.

Temporarily disabling the service is preferable to permanently disabling the service so that the service can be more easily re-enabled. In some situations, the script (and therefore the service) must be re-run to update configuration work that was done, such as zone cloning or migration. If the service is permanently disabled, the svcadm enable command must be run to re-enable the service.

Temporarily disabling the service is also preferable to leaving the service online. A service that is online might appear to be doing work on every reboot. In this example, the name of the service is site/first-boot-script-svc. After the client is booted, you can see the service is in the disabled state:

```
$ svcs first-boot-script-svc
STATE          STIME    FMRI
disabled        8:24:16 svc:/site/first-boot-script-svc:default
```

Creating a Script to Run at First Boot

To know what source you can use for your script, you need to know what tools are installed on the client system at first boot. The solaris-large-server package is installed by default. If you installed that group package, you have Python, bash, ksh, and other tools available to you at first boot. 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 11.1 Software Packages*. If you want to use a source for your script that is not available in the solaris-large-server package, identify the package you need and specify it in the AI manifest. For information about how to find the names of other packages that you might want to install, see *Adding and Updating Oracle Solaris 11.1 Software Packages*.

Tip –

- Use only one first-boot script to avoid having different commands in different scripts that collide with one another.
- Do not reboot in the first-boot script.

EXAMPLE 13–1 Template First-Boot Script

This example shows operations that should be done in any first-boot script.

- A first-boot script must load `/lib/svc/share/smf_include.sh` in order to use definitions such as SMF method exit codes.
- The script should test whether it already ran in a prior boot. If the `completed` property is already set to `true`, then exit the `start` method and temporarily disable the service.

  The following line in the script gets the value of the `completed` property in the `config` property group in the `site/first-boot-script-svc:default` service instance and assigns that value to the local `completed` variable.

  ```
  completed=‘svcprop -p config/completed site/first-boot-script-svc:default‘
  ```

  The following line in the script sends the `SMF_EXIT_TEMP_DISABLE` exit code to the service `start` method, with `method_completed` as the short reason for the exit and "`Configuration completed`" as the longer description of the reason for the exit.

  ```
  smf_method_exit $SMF_EXIT_TEMP_DISABLE script_completed "Configuration completed"
  ```
- A first-boot script should save 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.
- When the script has finished its work, the script must set the value of the `completed` property to `true`, refresh the service with the new property value, and exit the `start` method and temporarily disable the service. Use the `svccfg` command to set the `completed` property to `true`, and use the `svcadm` command to refresh the service.

Remember that by default, `sh` is `ksh93`.

```
#!/bin/sh

# Load SMF shell support definitions
. /lib/svc/share/smf_include.sh

# If nothing to do, exit with temporary disable
completed=‘svcprop -p config/completed site/first-boot-script-svc:default‘
[ "${completed}" = "true" ] && \
    smf_method_exit $SMF_EXIT_TEMP_DISABLE completed "Configuration completed"

# 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.
```

EXAMPLE 13–1 Template First-Boot Script *(Continued)*

```
bename=`beadm list -Hd|nawk -F ';' '$3 ~ /R/ {print $1}'`
beadm create ${bename}.orig
echo "Original boot environment saved as ${bename}.orig"

# Place your one-time configuration tasks here

# Record that this script's work is done
svccfg -s site/first-boot-script-svc:default setprop config/completed = true
svcadm refresh site/first-boot-script-svc:default

smf_method_exit $SMF_EXIT_TEMP_DISABLE method_completed "Configuration completed"
```

EXAMPLE 13–2 First-Boot Script that Configures Multiple IP Interfaces

This example shows a first-boot script named `first-boot-script.sh` that configures addresses on two IP interfaces and adds a default route.

```
#!/bin/sh

# Load SMF shell support definitions
. /lib/svc/share/smf_include.sh

# If nothing to do, exit with temporary disable
completed=`svcprop -p config/completed site/first-boot-script-svc:default`
[ "${completed}" = "true" ] && \
    smf_method_exit $SMF_EXIT_TEMP_DISABLE completed "Configuration completed"

# 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
echo "Original boot environment saved as ${bename}.orig"

# Create and configure addresses on two IP interfaces
/usr/sbin/ipadm create-ip net0
/usr/sbin/ipadm create-ip net1
/usr/sbin/ipadm create-addr -a 10.153.125.222/24 net0
/usr/sbin/ipadm create-addr -a 169.254.182.77/24 net1

# Add a default route with net0 as the gateway
/usr/sbin/route add default 10.153.125.1 -ifp net0

# Record that this script's work is done
svccfg -s site/first-boot-script-svc:default setprop config/completed = true
svcadm refresh site/first-boot-script-svc:default

smf_method_exit $SMF_EXIT_TEMP_DISABLE method_completed "Configuration completed"
```

Creating an SMF Manifest File

Create an SMF manifest file that defines a service that executes a script.

- The `start` method of the service executes the first-boot script.
- 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 adequately configured.

  Tip – Evaluate your script's dependencies and construct the service to run the script after its dependencies are satisfied.

- The `completed` property is defined with a value of `false`.

Using the Manifest Creation Tool

You can use the `svcbundle` command to generate a valid service manifest. In the following example, notice that by default, a manifest generated by the `svcbundle` command specifies a transient service and specifies the `multi-user` dependency.

EXAMPLE 13–3 Generated SMF Service Manifest

In the following command, the name of the script shown in "Creating a Script to Run at First Boot" on page 190 is specified as the value of `start-method`. The name of the script is specified as `/opt/site/first-boot-script.sh` because the package created in "Creating an IPS Package for the Script and Service" on page 196 installs the `first-boot-script.sh` script into `/opt/site/first-boot-script.sh`.

In the following command, the `completed` property is specified by a colon-separated list of property group name, property name, property type, and initial property value.

```
$ svcbundle -s service-name=site/first-boot-script-svc \
-s start-method=/opt/site/first-boot-script.sh \
-s instance-property=config:completed:boolean:false \
> first-boot-script-svc-manifest.xml
```

In the generated service manifest shown below, the first-boot script, `/opt/site/first-boot-script.sh`, is the value of the `exec` attribute of the `start` method. The `completed` property is specified in the `instance` element that defines the default instance of this service, `first-boot-script-svc:default`.

```
<?xml version="1.0" ?>
<!DOCTYPE service_bundle
  SYSTEM '/usr/share/lib/xml/dtd/service_bundle.dtd.1'>
```

EXAMPLE 13–3 Generated SMF Service Manifest *(Continued)*

```
<!--
    Manifest created by svcbundle (2012-Jul-13 16:39:30-0700)
-->
<service_bundle type="manifest" name="site/first-boot-script-svc">
    <service version="1" type="service" name="site/first-boot-script-svc">
        <!--
            The following dependency keeps us from starting until the
            multi-user milestone is reached.
        -->
        <dependency restart_on="none" type="service"
            name="multi_user_dependency" grouping="require_all">
            <service_fmri value="svc:/milestone/multi-user"/>
        </dependency>
        <exec_method timeout_seconds="60" type="method" name="start"
            exec="/opt/site/first-boot-script.sh"/>
        <!--
            The exec attribute below can be changed to a command that SMF
            should execute to stop the service.  See smf_method(5) for more
            details.
        -->
        <exec_method timeout_seconds="60" type="method" name="stop"
            exec=":true"/>
        <!--
            The exec attribute below can be changed to a command that SMF
            should execute when the service is refreshed.  Services are
            typically refreshed when their properties are changed in the
            SMF repository.  See smf_method(5) for more details.  It is
            common to retain the value of :true which means that SMF will
            take no action when the service is refreshed.  Alternatively,
            you may wish to provide a method to reread the SMF repository
            and act on any configuration changes.
        -->
        <exec_method timeout_seconds="60" type="method" name="refresh"
            exec=":true"/>
        <property_group type="framework" name="startd">
            <propval type="astring" name="duration" value="transient"/>
        </property_group>
        <instance enabled="true" name="default">
            <property_group type="application" name="config">
                <propval type="boolean" name="completed" value="false"/>
            </property_group>
        </instance>
        <template>
            <common_name>
                <loctext xml:lang="C">
                    <!--
                        Replace this comment with a short name for the
                        service.
                    -->
                </loctext>
            </common_name>
            <description>
                <loctext xml:lang="C">
                    <!--
                        Replace this comment with a brief description of
                        the service
```

EXAMPLE 13–3 Generated SMF Service Manifest *(Continued)*

```
                -->
            </loctext>
        </description>
    </template>
  </service>
</service_bundle>
```

Customizing the Generated Manifest

The service manifest generated with the `svcbundle` command might meet your needs with no modification necessary. The following example shows a modification of the service manifest.

If you modify a service manifest, use the `svccfg validate` command to ensure the manifest is still valid.

EXAMPLE 13–4 Customized SMF Service Manifest

In the following copy of the generated service manifest, the default `exec_method` timeout of 60 seconds has been increased for the `start` method. Make sure the `start` method has adequate time to run the first-boot script.

```
<?xml version="1.0" ?>
<!DOCTYPE service_bundle
  SYSTEM '/usr/share/lib/xml/dtd/service_bundle.dtd.1'>
<!--
    Manifest created by svcbundle (2012-Jul-13 16:39:30-0700)
-->
<service_bundle type="manifest" name="site/first-boot-script-svc">
    <service version="1" type="service" name="site/first-boot-script-svc">
        <!--
            The following dependency keeps us from starting until the
            multi-user milestone is reached.
        -->
        <dependency restart_on="none" type="service"
            name="multi_user_dependency" grouping="require_all">
            <service_fmri value="svc:/milestone/multi-user"/>
        </dependency>
        <!--
            Make sure the start method has adequate time to run the script.
        -->
        <exec_method timeout_seconds="360" type="method" name="start"
            exec="/opt/site/first-boot-script.sh"/>
        <!--
            The exec attribute below can be changed to a command that SMF
            should execute to stop the service.  See smf_method(5) for more
            details.
        -->
        <exec_method timeout_seconds="60" type="method" name="stop"
            exec=":true"/>
        <!--
            The exec attribute below can be changed to a command that SMF
```

EXAMPLE 13–4 Customized SMF Service Manifest *(Continued)*

```
                should execute when the service is refreshed.  Services are
                typically refreshed when their properties are changed in the
                SMF repository.  See smf_method(5) for more details.  It is
                common to retain the value of :true which means that SMF will
                take no action when the service is refreshed.  Alternatively,
                you may wish to provide a method to reread the SMF repository
                and act on any configuration changes.
            -->
            <exec_method timeout_seconds="60" type="method" name="refresh"
                exec=":true"/>
            <property_group type="framework" name="startd">
                <propval type="astring" name="duration" value="transient"/>
            </property_group>
            <instance enabled="true" name="default">
                <property_group type="application" name="config">
                    <propval type="boolean" name="completed" value="false"/>
                </property_group>
            </instance>
            <template>
                <common_name>
                    <loctext xml:lang="C">
                        <!--
                            Replace this comment with a short name for the
                            service.
                        -->
                    </loctext>
                </common_name>
                <description>
                    <loctext xml:lang="C">
                        <!--
                            Replace this comment with a brief description of
                            the service
                        -->
                    </loctext>
                </description>
            </template>
        </service>
</service_bundle>

$ svccfg validate first-boot-script-svc-manifest.xml
```

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 193.
- The first-boot script from "Creating a Script to Run at First Boot" on page 190.
- 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@1.0,5.11-0
set name=pkg.summary value="AI first-boot script"
set name=pkg.description value="Script that runs at first boot after AI installation"
set name=info.classification value=\
    "org.opensolaris.category.2008:System/Administration and Configuration"
file lib/svc/manifest/site/first-boot-script-svc-manifest.xml \
    path=lib/svc/manifest/site/first-boot-script-svc-manifest.xml owner=root \
    group=sys mode=0444
dir  path=opt/site owner=root group=sys mode=0755
file opt/site/first-boot-script.sh path=opt/site/first-boot-script.sh \
    owner=root group=sys mode=0555
```

Depending on what your first-boot script does, you might need to specify dependencies. If you modify this manifest, verify the new manifest is correct. You can ignore warnings. See Chapter 2, "Packaging Software With IPS," in *Packaging and Delivering Software With the Image Packaging System in Oracle Solaris 11.1* for information about how to create a package, including information about the `pkgdepend`, `pkgmogrify`, and `pkglint` commands.

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@1.0,5.11-0:20120716T022508Z
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.

5 **Verify that the package is available.**

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)     1.0-0      ---
```

6 **(Optional) Test installation of the package.**

The -n option indicates not to install the package.

```
$ pfexec pkg set-publisher -g ./firstbootrepo firstboot
$ pkg publisher
PUBLISHER  TYPE    STATUS P LOCATION
solaris    origin  online F http://http://pkg.oracle.com/solaris/release/
firstboot  origin  online F file:///home/user1/firstboot/firstbootrepo/
$ pkg list -af first-boot-script
NAME (PUBLISHER)                  VERSION    IFO
first-boot-script (firstboot)     1.0-0      ---
$ pfexec pkg install -nv first-boot-script
           Packages to install:        1
     Estimated space available: 50.68 GB
Estimated space to be consumed: 64.66 MB
       Create boot environment:       No
Create backup boot environment:       No
          Rebuild boot archive:       No

Changed packages:
firstboot
  first-boot-script
    None -> 1.0,5.11-0:20120716T022508Z
Planning linked: 0/2 done; 1 working: zone:z2
Linked image 'zone:z2' output:
|      Estimated space available: 50.68 GB
| Estimated space to be consumed: 62.07 MB
|           Rebuild boot archive:       No
`
Planning linked: 1/2 done; 1 working: zone:z1
Linked image 'zone:z1' output:
|      Estimated space available: 50.67 GB
| Estimated space to be consumed: 62.07 MB
|           Rebuild boot archive:       No
```

Next Steps See *Copying and Creating Oracle Solaris 11.1 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.

Testing the First-Boot Service

To test the service before you test an AI installation, you can simply install the package on a test system and reboot that test system.

```
$ pfexec pkg install first-boot-script
           Packages to install:  1
       Create boot environment: No
Create backup boot environment: No

DOWNLOAD                                  PKGS       FILES    XFER (MB)   SPEED
```

```
Completed                                  1/1         2/2      0.0/0.0    0B/s

PHASE                                          ITEMS
Installing new actions                           7/7
Updating package state database                 Done
Updating image state                            Done
Creating fast lookup database                   Done
Reading search index                            Done
$ pkg list first-boot-script
NAME (PUBLISHER)                                VERSION      IFO
first-boot-script (firstboot)                   1.0-0        i--
$ pkg info first-boot-script
          Name: first-boot-script
       Summary: AI first-boot script
   Description: Script that runs at first boot after AI installation
      Category: System/Administration and Configuration
         State: Installed
     Publisher: firstboot
       Version: 1.0
 Build Release: 5.11
        Branch: 0
Packaging Date: July 23, 2012 02:50:31 PM
          Size: 3.89 kB
          FMRI: pkg://firstboot/first-boot-script@1.0,5.11-0:20120723T145031Z
```

Reboot the test system. If the script created a new boot environment as shown above, be sure to boot into that new boot environment.

Check that the script is in the `/opt/site` directory and the effects of the script are correct.

Check the state of the service. If the script finished and exited correctly, the service should be in the disabled state.

```
$ svcs first-boot-script-svc
STATE          STIME    FMRI
disabled        8:24:16 svc:/site/first-boot-script-svc:default
```

Use one of the following commands to check the value of the `completed` property:

```
$ svcprop first-boot-script-svc:default
config/completed boolean true
$ svcprop -p config/completed first-boot-script-svc:default
true
```

If you want to review the service log file, use the following command to find the location of the log file:

```
$ svcs -x first-boot-script-svc
svc:/site/first-boot-script-svc:default (?)
 State: disabled since July 23, 2012 08:24:16 AM PDT
Reason: Temporarily disabled by service method: "Configuration completed."
   See: http://support.oracle.com/msg/SMF-8000-1S
   See: /var/svc/log/site-first-boot-script-svc:default.log
Impact: This service is not running.
```

The log file contains the following information:

```
[ Jul 23 08:22:57 Enabled. ]
[ Jul 23 08:24:14 Executing start method ("/opt/site/first-boot-script.sh"). ]
[ Jul 23 08:24:16 Method "start" exited with status 101. ]
[ Jul 23 08:24:16 "start" method requested temporary disable: "Configuration completed"
 ]
[ Jul 23 08:24:16 Rereading configuration. ]
```

▼ How to Update the Script or Service

If you change the script or the service manifest, use this procedure to install the update.

1 **Copy the updated files to your prototype directory.**

```
$ cp first-boot-script-svc-manifest.xml proto/lib/svc/manifest/site
$ cp first-boot-script.sh proto/opt/site
```

2 **Increment the package version.**

In the package manifest, change the value of the `pkg.fmri` attribute to the following, for example:

```
first-boot-script@1.0,5.11-0.1
```

3 **Publish the new version.**

Publish the new version of the package to the repository.

```
$ pkgsend publish -d ./proto -s ./firstbootrepo first-boot-script.p5m
pkg://firstboot/first-boot-script@1.0,5.11-0.1:20120723T231948Z
PUBLISHED
```

4 **Update the package.**

Use the `pkg list -af` command to make sure you can access the new version. You might need to use the `pkg refresh firstboot` command to update the package list. Use the `pkg update` command to update the package.

5 **Reboot the test system.**

◆ ◆ ◆ CHAPTER 14

Installing Client Systems

This chapter provides 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's network configuration and the location of the install server. SPARC clients can optionally use the `network-boot-arguments` variable set in the OBP to get this information.
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 122.
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 123
- 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.

SPARC and x86 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 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

SPARC client systems must meet the following additional requirements for AI installation over the network:

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 on the `installadm create-client` command by using the `-b` option. For SPARC clients, you can use the `network-boot-arguments` variable in the OBP to set boot properties.

Setting Up a SPARC Client

The following example associates the SPARC client with MAC address `00:14:4f:a7:65:70` with the `solaris11_1-sparc` install service.

```
$ pfexec installadm create-client -n solaris11_1-sparc -e 00:14:4f:a7:65:70
```

The DHCP server does not require configuration because the SPARC `wanboot-cgi` boot file has already been configured by `create-service`. See "Creating an AI Install Service" on page 96 for more information.

The following results of this `installadm create-client` command appear in the `/etc/netboot` directory:

```
lrwxrwxrwx  1 root staff 33 2012-05-09 08:53 0100144FA76570 -> /etc/netboot/solaris11_1-sparc
```

Setting Up an x86 Client

The following example associates the x86 client with MAC address `0:e0:81:5d:bf:e0` with the `solaris11_1-i386` 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 `solaris11_1-i386` install service.

```
$ pfexec installadm create-client -n solaris11_1-i386 -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(s)        :
        bios clients (arch 00:00):  0100E0815DBFE0.bios
        uefi clients (arch 00:07):  0100E0815DBFE0.uefi
```

The following example shows how `installadm` might set the default PXE boot files for this client in the `/etc/inet/dhcpd4.conf` file for an ISC DHCP configuration for an Oracle Solaris 11.1 `i386` install service:

```
host 00E0815DBFE0 {
  hardware ethernet 00:E0:81:5D:BF:E0;
  if option arch = 00:00 {
    filename "0100E0815DBFE0.bios";
  } else if option arch = 00:07 {
    filename "0100E0815DBFE0.uefi";
  }
}
```

The following results of this `installadm create-client` command appear in the `/etc/netboot` directory:

```
lrwxrwxrwx  1 root staff   47 2012-05-08 17:49 0100E0815DBFE0.uefi -> ./solaris11_1-i386/boot/grub/grub2netx64.efi
lrwxrwxrwx  1 root staff   21 2012-05-08 17:49 0100E0815DBFE0 -> ./0100E0815DBFE0.bios
lrwxrwxrwx  1 root staff   40 2012-05-08 17:49 0100E0815DBFE0.bios -> ./solaris11_1-i386/boot/grub/pxegrub2
-rw-r--r--  1 root root  1744 2012-05-08 17:49 grub.cfg.0100E0815DBFE0
-rw-r--r--  1 root root  1212 2012-05-08 17:49 menu.conf.0100E0815DBFE0
```

Deleting a Client From a Service

Use the `installadm delete-client` command to delete a client from an install service.

```
$ pfexec 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 describes 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 by monitoring progress in the `/system/volatile/install_log` installation log file.

To enable remote access for all clients of a particular install service, set 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`.

Individual clients can also set this option on the boot command line.

Monitoring x86 Client Installations

For x86 systems, use the `-b` option with the `create-service` subcommand to set boot properties for all clients that use that service, as shown in the following example:

```
$ pfexec installadm create-service -a i386 -b livessh=enable
```

The following excerpt shows how the property appears in the `/etc/netboot/`*svcname*`/grub.cfg` file:

```
$multiboot $kern /platform/i86pc/kernel/amd64/unix -B livessh=enable,...
```

You can enable `ssh` for a single x86 client by specifying `livessh` on the boot command line. For instructions, see "Adding Kernel Arguments by Editing the GRUB Menu at Boot Time" in *Booting and Shutting Down Oracle Solaris 11.1 Systems*.

Monitoring SPARC Client Installations

For SPARC systems, access the `system.conf` file 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/solaris11_1-sparc/system.conf
...install_service=solaris11_1-sparc
install_svc_address=$serverIP:5555
```

```
livessh=enable
...
```

You can enable `ssh` for a single SPARC client by specifying `livessh` on the boot command line. The following examples show two different ways to specify this argument:

```
ok boot net:dhcp - livessh
ok boot net:dhcp - livessh=enable
```

The `livessh` specification on the boot command line overrides any setting specified in the service's `system.conf` file. For example, if the `system.conf` file specifies `livesssh=enable`, you can disable `livessh` on a particular client by specifying `livessh=disable` on the boot command line:

```
ok boot net:dhcp - livessh=disable
```

Installing a SPARC Client

Network boot SPARC clients from the OBP prompt.

- If you are using DHCP, use the following network boot command:

  ```
  ok boot net:dhcp - install
  ```
- If you are not using DHCP, use the following command to set the `network-boot-arguments` variable in the OBP. This variable is set persistently in the OBP:

  ```
  ok setenv network-boot-arguments host-ip=client-ip,
  router-ip=router-ip,subnet-mask=subnet-mask,hostname=hostname,
  file=wanboot-cgi-file
  ```

  Then use the following command to network boot the client:

  ```
  ok boot net - install
  ```

Note – When you use the `network-boot-arguments` variable, the SPARC client does not have DNS configuration information. Ensure that the AI manifest used with this client specifies an IP address instead of a host name for the location of the IPS package repository, and for any other URI in the manifest.

The following events occur during AI boot of a SPARC client:

1. The client boots and gets its network configuration and the location of the `wanboot-cgi` file from the DHCP server or from the `network-boot-arguments` variable set in its OBP.
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.

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.

The following events occur during AI boot of an x86 client:

1. The client boots and gets an IP address, and the boot file is downloaded from the location provided by the DHCP server.
2. The boot file is loaded and reads a GRUB menu file.
3. The user selects the second option, "Oracle Solaris 11.1 Automated Install," from the GRUB menu.
4. The boot file gets the boot archive file, and the Oracle Solaris OS is booted using TFTP.
5. The net image archives, `solaris.zlib` and `solarismisc.zlib`, are downloaded using HTTP as provided by the GRUB menu.
6. 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.
7. The AI install program is invoked with the AI manifest to perform the installation.

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 11.1 Text Installer and command line
Oracle Solaris 11.1 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 "Starting Installation After Booting Without Initiating an Installation" on page 220. Use this menu to examine or install the system.

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 log in as `root` with the password `solaris` to monitor the installation messages in `/system/volatile/install_log`.

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/log/install/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 15

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 212
- "x86 Network Booting Errors and Possible Causes" on page 214
- "SPARC and x86 Error Messages" on page 217

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 212
- "Boot Load Failed" on page 212
- "Internal Server Error or WAN Boot Alert" on page 213
- "Error Message 403: Forbidden or 404 Not Found" on page 213
- "Automated Installer Disabled" on page 214

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 another DHCP server is 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

The 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
Thu Jul  5 18:57:36 wanboot progress: miniroot: Read 235737 of 235737 kB (100%)
Thu Jul  5 18:57:36 wanboot info: miniroot: Download complete
SunOS Release 5.11 Version 11.1 64-bit
Copyright (c) 1983, 2012, Oracle and/or its affiliates. All rights reserved.
Remounting root read/write
Probing for device nodes ...
Preparing network image for use
Downloading solaris.zlib
--2012-07-05 18:52:30--  http://10.134.125.136:5555/export/auto_install/11_1_sparc/solaris.zlib
Connecting to 10.134.125.136:5555... connected.
HTTP request sent, awaiting response... 404 Not Found
2012-07-05 18:52:30 ERROR 404: Not Found.

Could not obtain http://10.134.125.136:5555/export/auto_install/11_1_sparc/solaris.zlib from install server
Please verify that the install server is correctly configured and reachable from the client
```

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 net image 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 in order to initiate an installation:

```
ok boot net:dhcp - install
```

If you boot without the `install` boot argument, the SPARC client boots into the automated installer boot image but the installation does not start. See "Starting Installation After Booting Without Initiating an Installation" on page 220 for instructions about how to start an automated installation from this point.

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 ProxyDHCP Offers Were Received" on page 215
- "TFTP Error or System Hangs After GATEWAY Message" on page 215
- "System Hangs After GRUB Menu Entry is Selected" on page 215
- "HTTP Request Sent Results in 403 Forbidden or 404 Not Found" on page 216
- "Automated Installer Disabled" on page 216

No DHCP or ProxyDHCP 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 whether 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 `grub.cfg` file or 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` for the image path shown by `installadm list`:

```
Filesystem      1K-blocks      Used Available Use% Mounted on
/export/auto_install/solaris11_1-i386
                 92052473  36629085  55423388  40% /etc/netboot/default-i386
/export/auto_install/solaris11_1-i386
                 92052473  36629085  55423388  40% /etc/netboot/solaris11_1-i386
```

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 `grub.cfg` file or 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 11.1 64-bit
Copyright (c) 1983, 2012, Oracle and/or its affiliates. All rights reserved.
Remounting root read/write
Probing for device nodes ...
Preparing network image for use
Downloading solaris.zlib
--2012-07-18 20:02:26--  http://10.134.125.136:5555/export/auto_install/solaris11_1-i386/solaris.zlib
Connecting to 10.134.125.136:5555... connected.
HTTP request sent, awaiting response... 404 Not Found
2012-07-18 20:02:26 ERROR 404: Not Found.

Could not obtain http://10.134.125.136:5555/export/auto_install/solaris11_1-i386/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 11.1 Text Installer and command line
Oracle Solaris 11.1 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. See "Starting Installation After Booting Without Initiating an Installation" on page 220 for instructions about how to start an automated installation from this point.

SPARC and x86 Error Messages

The following errors are common to both SPARC and x86 installations:

- "Automated Installation Failed Message" on page 217
- "Unable To Contact Valid Package Server" on page 217
- "Package Not Found" on page 218

Automated Installation Failed Message

If a failure occurs during installation, then the following message is displayed:

```
21:43:34    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
Jul  6 21:43:34 solaris svc.startd[9]: 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://support.oracle.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:

```
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`.
- If you are using DNS, check whether DNS is correctly configured on the AI client. See "Check DNS" on page 211.
- 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 11.1 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 `mypkg` 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        generated-transfer-1230-1
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:/mypkg
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:/mypkg
```

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.

Booting 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 "Starting Installation After Booting Without Initiating an Installation" on page 220. 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 11.1 Text Installer and command line
Oracle Solaris 11.1 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 boot entry as described in "Adding Kernel Arguments by Editing the GRUB Menu at Boot Time" in *Booting and Shutting Down Oracle Solaris 11.1 Systems*, and remove the property.

Starting 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

Please enter a number [1]:
```

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
```

P A R T I V

Performing Related Tasks

◆ ◆ ◆ APPENDIX A

Working With Oracle Configuration Manager

This chapter provides an overview of Oracle Configuration Manager, as well as instructions for using the service on a system running an Oracle Solaris release. The following is a list of information that is in this chapter:

Introduction to Oracle Configuration Manager

Oracle Configuration Manager is used to collect configuration information for a system and upload it to the Oracle repository. The collector of this information can be configured as a central collector, which gathers information for all products on the server, or to gather information in separate collection sites. See "About the Oracle Configuration Manager Central Collector" on page 224 for more information.

Customer support representatives can use this information to provide better service. Some of the benefits of using Oracle Configuration Manager are as follows:

- Reduces time for the resolution of support issues
- Provides proactive problem avoidance
- Improves access to best practices and the Oracle knowledge base
- Improves understanding of customer business needs and provides consistent responses and services

Oracle Configuration Manager can be run in one of two modes: connected or disconnected. The disconnected mode is needed only if your system does not have a connection to the Internet, and you cannot configure an Oracle Support Hub. In this mode, you can manually collect configuration information and upload the information to Oracle by way of a service request.

In the connected mode, Oracle Configuration Manager can be run in several network configurations as follows:

- Systems can be directly connected to the Internet.
- Systems can be connected to the Internet through a proxy server.
- Systems do not have direct access to the Internet, but they do have access to an intranet proxy server, which in turn has an Internet connection through an Oracle Support Hub.
- Systems do not have direct access to the Internet, but they do have access to an Oracle Support Hub, which in turn is connected to the Internet through a proxy server.

For more information about setting up and configuring Oracle Configuration Manager, see the Oracle Configuration Manager Installation and Administration Guide. The rest of this document focuses on the Oracle Solaris tasks that are associated with Oracle Configuration Manager.

Note – To configure Oracle Configuration Manager to use a proxy or an Oracle Support Hub, you must run the `configCCR` command in interactive mode. See Oracle Support Hub for more information.

During an Oracle Solaris 11 installation, the software attempts to set up an anonymous connection to the Oracle repository. If successful, this connection allows the installation process to proceed without prompting for any information. Ideally, you should change the registration or the network configuration after the system is fully installed. Data loaded anonymously is not tied to any organization. If the software could not connect to the Oracle repository, you can register your system manually, then enable the Oracle Configuration Manager service.

About the Oracle Configuration Manager Central Collector

The Oracle Configuration Manager collector installed as part of the Oracle Solaris operating system is configured and designated as a central collector. To reap the benefits of an Oracle Configuration Manager collector, such as a personalized support experience, quicker resolution of support issues, and proactive problem avoidance, the configuration data for each Oracle installation must be collected and uploaded. This is normally the task of the collector installed in the Oracle home. However, sometimes the collector in Oracle homes might not have been configured or is left disconnected. The purpose of the central collector is to collect those Oracle homes and upload them under its own My Oracle Support (MOS) credentials. Here are the characteristics of a central collector:

- A central collector collects:
  - The Oracle home in which it resides
  - Oracle homes on the host that do not have a configured collector

  - Oracle homes where the collector is in disconnect mode
  - Oracle homes where the collector has authenticated registration

  If a collector in an Oracle home is configured using ORACLE_CONFIG_HOME designation, the central collector will not collect that home.

- Using the `root` role, you can designate a collector installation to be a central collector by specifying the `-c` option in the `setupCCR` and `configCCR` commands. Subsequent `configCCR` commands without the `-c` option relinquish the central collector designation from the collector. Running the `setupCCR` and `configCCR` commands with the `-c` option designates the collector as a central collector. The collector installed as part of the Oracle Solaris operating system is installed using `root` permissions, hence it operates as the central collector for the host.
- The Oracle Universal Installer central inventory is the source from which the central collector obtains the set of candidate Oracle homes to be collected. The central inventory is searched by the installer as described in the documentation. The default location for the installer central inventory pointer for the Oracle Solaris operating system is `/var/opt/oracle/oraInst.loc`. If you choose to place an Oracle installation inventory in a different location, then the central inventory can not find and collect it.
- In this release, beyond the configuration information from the Oracle Solaris OS, only Oracle database and Oracle Fusion Middleware based products that use Oracle WebLogic are collected by the central collector.
- All configuration data collected by the central collector from Oracle homes is uploaded using the My Oracle Support credentials of the central collector.

Administering Oracle Configuration Manager (Tasks)

The following task map includes several procedures that are associated with using Oracle Configuration Manager on an Oracle Solaris system.

Task	Description	For Instructions
Enable the Oracle Configuration Manager service.	Enables the Oracle Configuration Manager service, after you have made configuration changes.	"How to Enable the Oracle Configuration Manager Service" on page 226
Disable the Oracle Configuration Manager service.	Disables the Oracle Configuration Manager service, before you make any significant configuration changes.	"How to Disable the Oracle Configuration Manager Service" on page 226
Manually register your system with the Oracle repository.	Changes your registration credentials.	"How to Manually Register Your System With the Oracle Repository" on page 226

Task	Description	For Instructions
Change data collection time.	Resets the data collection frequency and time.	"How to Change the Time or Frequency of Data Collection for Oracle Configuration Manager" on page 227

▼ How to Enable the Oracle Configuration Manager Service

1 **Become an administrator.**

For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

2 **Enable the Oracle Configuration Manager service.**

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

▼ How to Disable the Oracle Configuration Manager Service

1 **Become an administrator.**

For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

2 **Disable the Oracle Configuration Manager service.**

```
# svcadm disable system/ocm
```

Caution – Do not run the `emCCR stop` command on an Oracle Solaris system. Any changes to the service must be made using the Service Management Facility (SMF) feature of Oracle Solaris.

▼ How to Manually Register Your System With the Oracle Repository

1 **Become an administrator.**

For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

2 **Change your user registration.**

```
# configCCR
```

The Oracle Configuration Manager software prompts for an email account and password. Preferably, use an email account associated with your My Oracle Support identity.

If the system can communicate directly with the registration server, it does so. If not, you are prompted for the URL of an Oracle Support Hub. If a URL is usable at your site, specify it here. If you do not specify the URL of an Oracle Support Hub, or you are still unable to communicate with the registration server, then you are prompted for a network proxy.

After registration is complete, data collection begins.

See Also For more information about the `configCCR` command, see the `configCCR`(1M) man page or the Oracle Configuration Manager Installation and Administration Guide. For complete examples of an interactive session using the `configCCR` command, see the configCCR page.

▼ How to Change the Time or Frequency of Data Collection for Oracle Configuration Manager

1 **Become an administrator.**

For more information, see "How to Use Your Assigned Administrative Rights" in *Oracle Solaris 11.1 Administration: Security Services*.

2 **Reset the frequency of data collection.**

This example resets the data collection time to occur weekly on Monday mornings at 6:00 a.m.

```
# emCCR set collection_interval=FREQ=WEEKLY\; BYDAY=MON\; BYHOUR=6
```

See Also For more information about the `emCCR` command, see the `emCCR`(1M) man page or the Oracle Configuration Manager Installation and Administration Guide.

◆ ◆ ◆ APPENDIX B

Using the Device Driver Utility

The Oracle Device Driver Utility (DDU) reports whether the current release supports the devices that have been detected on your installed system.

Device Driver Utility Overview

The Device Driver Utility provides information about the devices on your installed system and the drivers that manage those devices. The DDU 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 229
- "How to Install Missing Drivers" on page 230
- "How to List Your System in the HCL" on page 232

▼ How to Start the Device Driver Utility

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

- **Start the Device Driver Utility by using one of the following methods:**
  - **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 230.

▼ 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 pop–up 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:**
  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 232.

▼ 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.

1. **Start the Device Driver Utility.**

   See "How to Start the Device Driver Utility" on page 229.

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`.**

Index