Installing Oracle® Solaris 11.3 Systems

Part No: E54756
May 2019

ORACLE®

Installing Oracle Solaris 11.3 Systems

Part No: E54756

Référence: E54756

Accès aux services de support Oracle

Les clients Oracle qui ont souscrit un contrat de support ont accès au support électronique via My Oracle Support. Pour plus d'informations, visitez le site `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` ou le site `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` si vous êtes malentendant.

Contents

Figures

Tables

Examples

Using This Documentation

- **Overview** – Describes how to install the Oracle Solaris 11.3 release and configure Automated Install servers
- **Audience** – Technicians, system administrators, and authorized service providers
- **Required knowledge** – Advanced experience troubleshooting and replacing hardware

Product Documentation Library

Documentation and resources for this product and related products are available at http://www.oracle.com/pls/topic/lookup?ctx=E53394-01.

Feedback

Provide feedback about this documentation at http://www.oracle.com/goto/docfeedback.

PART I

Oracle Solaris 11.3 Installation Options

This section provides a general overview of the installation options. The following topics are covered:

♦ ♦ ♦ 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 table compares the capabilities of the various installation options.

TABLE 1 Installation Options

Installation Option	For More Information	Minimal Preparations	Requires AI Server	Installs Packages from a Package Repository
x86 only: GUI installer	Chapter 3, "Using Live Media"	Yes	No, installs from media	No
Text installer	Chapter 4, "Using the Text Installer"	Yes	No, installs from media	No
Text install with boot help from an server	"How to Start a Text Installation Over the Network" on page 55	No	Yes, retrieves install image from server	No
Automated installer with media	Chapter 5, "Automated Installations That Boot From Media"	No	Server needed if you want to customize install media, but not needed for installation	Yes
Automated installer for multiple stems	Chapter 7, "Automated Installation of Multiple Systems"	No	Yes, server required	Yes

In addition, you have the option of creating installation images, including custom Live Media images, text installer image and automated installation images. See *Creating a Custom Oracle Solaris 11.3 Installation Image*.

Simple Installations

The GUI installer on Live Media and the text installer are simple installation methods.

- You can use either method to install Oracle Solaris on the x86 platform. You can also use the text installer to install Oracle Solaris on the SPARC platform.
- Both installers can function with a minimum of memory. See *Oracle Solaris 11.3 Release Notes* for memory requirements.
- 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 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 set up 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 install client boots.

Note - The text installer installs 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 the installed system, you should install the `solaris-large-server` package set.

- Besides the ability to modify 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 Chapter 2, “Preparing for the Installation”.

Installations Requiring AI Server Setup

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

To use AI, you must first set up an AI server on your network, unless you only want to boot from AI media which is described below. When an AI client boots, the system gets installation

specifications from the AI server, retrieves software packages from an Oracle Solaris package repository, and the software is installed on the stem.

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

AI can perform “hands-free” automatic network installations on both x86 and SPARC based systems. The AI 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 “What Is an Automated Installation?” on page 73.

In addition to the “hands-free” network installations, you can 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 “How to Start a Text Installation Over the Network” on page 55.

If you have access to an AI image, even if you haven't configured an AI server, you can download the image and store it on the network or locally. Next, you can copy the image to removable media Then, you can boot the AI media or ISO image directly on each of your systems. Installations that use AI media are non-interactive. For instructions, see Chapter 5, “Automated Installations That Boot From Media”.

Additional Installation Options

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

Creating 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 11.3 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 Software in Oracle Solaris 11.3*.

What's New in Installation for Oracle Solaris 11.3

This section highlights information for existing customers about important new features related to installation in this release.

- The text installer has been enhanced to include a new configuration panel for boot pools. See "How to Perform a Text Installation" on page 48.
- The `installadm` command has been enhanced.
  - For information about the changes to the command for system configuration profiles, see "Adding System Configuration Profiles to an Install Service" on page 189.
  - For information about administering manifests using the interactive editor, see "Creating an AI Manifest Using the `installadm` Command" on page 171.
- AI manifests can now be configured to support the following actions:
  - "Specifying a Root Pool and Boot Pool in an AI Manifest" on page 177
  - "Accessing a Unified Archive Using SSL Client Authentication" on page 182
  - "Accessing a Unified Archive Using Http Authentication Tokens" on page 182
  - "Accessing a Secure IPS Repository" on page 183
- System configuration profiles can now configure an Infiniband (IB) link. See "Specifying an IB Link in a System Configuration Profile" on page 210.

PART II

Installing Using Installation Media

This section provides a overview of the installation options when using media. The following topics are covered:

- 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 2

Preparing for the Installation

Before installing your system, review the information in this chapter including system requirements for installation, suggestions for partitioning your system, and help with Oracle Configuration Manager.

System Requirements for Live Media and Text Installations

To check the minimum memory and disk space requirements for installing the Oracle Solaris 11.3 release using a Live Media installation image or a text installation image, see *Oracle Solaris 11.3 Release Notes*.

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.

Note - Non Oracle x86 systems with Intel® Virtualization Technology for Directed I/O (VT-d) must have the `Intel VT-d` parameter set to `Enabled` before you install Oracle Solaris on those systems. Refer to their respective documentation for instructions to set this parameter.

Preparing a System for Installing Multiple Operating Systems

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

TABLE 2 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 installs 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 11.3 Systems*.
Solaris 10 OS	You cannot use the Live Media installer 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. You can use the Live Media and text installers to replace releases starting with Solaris 10 1/06 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, you do not need to change the existing extended partition during an installation. You can create, resize, or delete an extended partition when you install Oracle Solaris by using either the Live Media 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 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.

Note - See SPARC: GPT Labeled Disk Support for more information about applying GPT-aware firmware on supported SPARC based systems.

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

Guidelines for Partitioning a System Prior To Installation

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

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, on a SPARC client, the text installer can install 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 11.3 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.

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 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. In addition, for the text installer *only*, you can select, create, or modify VTOC slices during an interactive installation.

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

- 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 Oracle Solaris 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 Oracle Solaris partition and you make no modifications to any of the other existing partitions, the installation default overwrites the Oracle 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 3 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 by selecting a primary partition or a logical partition and then changing its type to Solaris. During an installation, this modification erases the existing partition contents.
Increase the space that is allocated to a Solaris partition and install on that partition.	If enough disk space is available, you can increase the size that is allocated to a Solaris partition before installing the software on that partition. The available space contains any contiguous unused space before or after the selected partition. If you enlarge the partition, unused space after the partition is used first. Then, unused space before the partition is used, which changes the starting cylinder of the selected partition.
Install 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 a pool named `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 the type of 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 4 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	Change 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 `https://www.oracle.com/webfolder/technetwork/hcl/index.html`. The HCL provides information about hardware that is certified or reported to work with the Oracle Solaris operating system.

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:

- If you want to send an anonymous system configuration to My Oracle Support without any identifying customer information, use the default Support Registration installer panel anonymous registration address or another email address with no password.
- If you want to see your customer information in My Oracle Support and receive security updates, replace the anonymous email address in the Support Configuration panel with your My Oracle Support login ID and add your My Oracle Support password. 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 do not want to automatically send data to My Oracle Support, 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. 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. You can tune scheduling by using `/usr/sbin/emCCR`. See the `emCCR`(1M) man page.

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.

To register or re-register, use 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 service 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`(8) man page
- Oracle Configuration Manager Installation and Administration Guide
- `https://www.oracle.com`. From the Menu, select any of the options in the *Support and Services → Support Policies and Practices* submenu.
- Oracle Auto Service Request documentation at `https://www.oracle.com/support/premier/auto-service-request.html`

♦ ♦ ♦ 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 27.
- 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 29.
  - In this release, Oracle Solaris for the x86 platform installs the new version of the GRand Unified Bootloader (GRUB 2). For information more about GRUB 2, see "Introducing GRUB 2" in *Booting and Shutting Down Oracle Solaris 11.3 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 29.
- 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 "How to Perform a GUI Installation" on page 37 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 Software in Oracle Solaris 11.3*.
- 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 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 actions in this procedure before performing a GUI installation.

1. **If you do not have Live Media, download the Live Media ISO image.**
   To download the Oracle Solaris Live Media ISO image, go to `https://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 for a USB image or to an accessible file system for an ISO image. For USB images, you can use the `dd` or the `usbcopy` command. Here is an example of how to make a copy with the `dd` command:

```
# dd bs=16k conv=sync if=<image path> of=/dev/rdsk/c4t0d0p0
```

Note - To add the `usbcopy` utility to your system, install 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 27.

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

      See "Ensuring That You Have the Proper Device Drivers" on page 32.

3. **If you are installing multiple operating systems, set up the required environment.**

   a. **Review the specifications in "Preparing a System for Installing Multiple Operating Systems" on page 27.**

   b. **Back up your system.**

   c. **If you want to partition your system prior to the installation, see "Partitioning Your System" on page 28.**

Next Steps See "How to Perform a GUI Installation" on page 37.

▼ How to Perform a GUI Installation

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

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

- If you are prompted to log in , 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 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. Provide four numbers in the range 0-255 . The system at this IP address must be online and accessible. This field is 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 you provided 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.

   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 29.

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 and 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, you must provide a login name and password. 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 provide 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, determine how to configure 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 33.

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

You have two alternatives:

- 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 Live Media If Your System Boots in Console Mode” on page 42.

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 Live Media If Your System Boots in Console Mode

Before You Begin For this procedure, two networked systems are required: the system on which 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 do not have 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 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
```

8. **Run the GUI installer:**

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

Note - Installer graphic display might be imperfect using this method.

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

Adding Software After a 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 Software in Oracle Solaris 11.3* 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 example to install additional software:

```
# 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 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 27.
- 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.
  - 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 29.
  - 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 11.3 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 29.
- 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 Software in Oracle Solaris 11.3*.
- 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. To use iSCSI, the network interface for the system must be configured with a static IP address before starting the installation process. Note the following considerations when performing an iSCSI installation:

- An iSCSI boot on SPARC platforms is supported with OpenBoot level 4.31 or later, and does not require a specific NIC. The `boot` command in OpenBoot takes a series of keywords to identify the destination iSCSI target or uses the parameters stored in the network-boot-parameters NVRAM variable. The command uses the format `boot net:keyword=`*value*.
- On x86 platforms, the host that is being booted must use NICs that are iSCSI Boot Firmware Table (iBFT) capable or have a main board BIOS that is iBFT capable. To configure iSCSI boot properly, refer to the documentation for your specific NIC hardware.

For further information, see "How to Perform a Text Installation" on page 48. 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.

- 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 system. In particular, the text installer does not install the GNOME desktop. To install additional packages after an installation performed with the text installer, see "Adding Software After a Text Installation" on page 56.

Networking Configuration With the Text Installer

The networking panel in the text installer prompts you to select a network interface to configure. Only one wired network interface may be configured. You can also select to skip the network configuration process.

When an interface is selected, you can choose to allow DHCP to configure the interface or choose to manually configure the interface with a static IPv4 address. For static addresses, the IPv4 default route can also be provided. In both cases, IPv6 autoconfiguration is enabled on the interface.

How to Navigate Within the Text Installer

Use the function keys listed at the bottom of each panel to navigate between the panels. Use the arrow keys to move between fields in a given panel. If your keyboard does not have function keys or if the keys do not respond, press ESC to view alternate keys for navigation.

At any time during the installation, you may back up to a previous panel.

Text Installation Tasks

This section includes the following tasks:

▼ How to Prepare for a Text Installation

Complete the actions in this procedure 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 `https://www.oracle.com/technetwork/server-storage/solaris11/downloads/index.html`. This site also includes USB images that can be used on removable media.

   After you download the image, copy the image to removable media for a USB image or to an accessible file system for an ISO image. For USB images, you can use the `dd` or the `usbcopy` command. Here is an example of how to make a copy with the `dd` command:

   ```
   # dd bs=16k conv=sync if=<image path> of=/dev/rdsk/c4t0d0p0
   ```

   Note - To add the `usbcopy` utility to your system, install 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 27.

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

See “Ensuring That You Have the Proper Device Drivers” on page 32.

3. If you are installing multiple operating systems, set up the required environment.

a. Review the specifications in “Preparing a System for Installing Multiple Operating Systems” on page 27.

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 29.

Next Steps See “How to Perform a Text Installation” on page 48.

▼ How to Perform a Text Installation

1. Insert the text installation media and boot the system using the media. If requested, make any preliminary keyboard and language selections.

The keyboard and language selections are requested during the x86 installation process. These values are preset for the SPARC installation process.

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 285. After you have installed the drivers, restart the text installation and return to the installation menu.

3. (Optional) To use iSCSI disk discovery, select option 3.

At the shell prompt, follow the steps to configure a network interface. See Chapter 3, “Configuring and Administering IP Interfaces and Addresses in Oracle Solaris” in *Configuring*

and Managing Network Components in Oracle Solaris 11.3 for more information. After you have configured a network interface, exit the shell by pressing Control-D.

4. **(Optional) To boot using an iSCSI device using IPoIB, select option 3.**

   See Chapter 3, "Configuring and Administering IP Interfaces and Addresses in Oracle Solaris" in *Configuring and Managing Network Components in Oracle Solaris 11.3* for more information.

   a. **Create an IPoIB partition link.**

      In this example, the available links are displayed, and then a partition link is created using `net5`. The last command shows that the link has been created.

```
# dladm show-ib
LINK     HCAGUID         PORTGUID        PORT STATE   GWNAME     GWPORT     PKEYS
net4     212800013F2EC6 212900013F2EC7 1    down    -          --FFFF
net5     212800013F2EC6 212900013F2EC8 1    up      -          --FFFF
# dladm create-part -l net5 -P 0xFFFF ibd5
# dladm show-part
LINK         PKEY  OVER         STATE    FLAGS
ibd5         FFFF  net5         unknown  ----
```

   b. **Create an IP interface and an IP address.**

      This example uses `ibd5` for an IPoIB installation that was created in the previous step. The IP address should be static.

```
# ipadm create-ip ibd5
# ipadm create-addr -T static -a 6.6.6.53/24 ibd5/v4
```

      This example uses `net0` which is normally used for iSCSI connections.

```
# ipadm create-ip net0
# ipadm create-addr -a 192.0.2.5/27 net0
```

   c. **Exit the shell by pressing Control-D.**

5. **Initiate the installation by selecting the first option on the installation menu.**

```
Welcome to the Oracle Solaris 11.3 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]:
```

Use the "Continue" function key to move to the next panel.

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.

6. **In the Discovery Selection panel, select the discovery method for the disk that you want to install the system on.**

   - 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. Make sure to complete Step 3, if you want to use iSCSI. An additional panel prompts for the following information:

Target IP	The IP address of the iSCSI target. Provide four numbers in the range 0-255. The system at this IP address must be online and accessible. This field is required.
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 because the initiator node name cannot be modified. This field is generated for you.
CHAP Name	If using CHAP for authentication, the CHAP (Challenge-Handshake Authentication Protocol) name to be used for authentication. This field is optional.
CHAP 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 you provided 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 Disks panel, select the disk to install the OS on.**

   If more than one target disk is listed, select one of the disks or accept the default.

8. **In the Partitions panel, Choose whether to install the operating system on the whole disk or on a part of the disk.**

   The following choices are displayed:

   - Use the entire disk
   - Use a GPT partition

   Note - During a SPARC installation, the panel prompts for information about slices instead of partitions.

9. **(Optional) In the Partition Selection panel, modify the partition 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.

   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.

   The SPARC installation process will prompt for information about the disk slices.

For detailed partitioning instructions, see "Guidelines for Partitioning a System During an Interactive Installation" on page 29, or see the online help in the installer.

10. **(Optional) In the Boot Pool panel, modify the boot pool devices.**

    In this panel, the dedicated on-board devices are displayed first. By default, the Dedicated and Selected columns show Yes for these devices. You can change the boot pool devices using the F5 key.

11. **In the System Identity panel, provide a computer name to identify the system on the network.**

12. **In the Network panel, specify how to configure the wired Ethernet network connection.**

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

      The installer continues to the Time Zone panels.

    - **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. **In the Manually Configure panel, 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. **In the DNS Name Service panel, if you select to have the system use the DNS name service:**

         i **In the DNS Server Addresses panel, type at least one IP address for a DNS server.**

         ii **In the DNS Search List panel, provide at least one domain name to be searched when a DNS query is made.**

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

e. In the Domain Name panel, provide the domain where the system resides for the alternate name service, if you selected one.

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

---

f. In the LDAP Profile Panel, if you selected LDAP on the Alternate Name Service panel, provide LDAP configuration specifications as follows:

- The LDAP profile to be used to configure the LDAP name service on the system
- The IP address for the LDAP profile server
- The LDAP search base
- In the LDAP Proxy panel, specify whether LDAP proxy bind information will be provided.

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

g. In the NIS Name Server panel, if you selected NIS on the Alternate Name Service panel, 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:

- Find One

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

- Specify One - Type the name server's host name or IP address in the subpanel.

13. **In the Time Zone panels, select the region, location, and time zone.**

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

14. **Select the language and language territory in the Locale panels.**

15. **Set the date and time in the next panel.**

16. **Select the keyboard layout in the next panel.**

17. **Create accounts in 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.

18. **In the Support - Registration panel, determine whether or how you want to use Oracle Configuration Manager or start Oracle Auto Service Request.**

For further information, see "Using Oracle Configuration Manager" on page 33.

19. **In the Support - Network Configuration panel, select an access method for OCM and ASR.**

The following options are available:

- No proxy
- Proxy - the next panel prompts for the proxy hostname, port number, and username and password if using secure proxy.

- Aggregation Hubs - the next panel prompts for the OCM Hub URL and the ASR Manager URL.

20. **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.

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

    Use the restart function key to start the Oracle Solaris installation process.

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

22. **Review the installation logs.**

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

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

How to Start 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 to boot a system over the network and then starting an interactive text installation. 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.

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

   For instructions, see "Creating an Install Service" on page 97.

2. **Boot the AI client over the network.**

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

     ```
     # boot net:dhcp
     ```

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

     ```
     Welcome to the Oracle Solaris 11.3 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 system.**

   For instructions, see "How to Perform a Text Installation" on page 48.

Note - The package set installed by the text installer is the `solaris-large-server` package set. When you use the text installer after booting 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 Software in Oracle Solaris 11.3* for more information.

```
# pkg install solaris-desktop

# pkg install solaris-large-server
```

Adding Software After a 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. You can also 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 Software in Oracle Solaris 11.3* 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 example to install additional software:

```
# pkg install package–name
```

If you would rather create a new boot environment and install the package in the new boot environment, use the following command:

```
# pkg install --require-new-be --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.

♦ ♦ ♦ 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 server.
- Install a SPARC system that does not have WAN boot capability.
- Troubleshoot an ailing system. You can boot the system from the removable media and then inspect the installed system and run diagnostics.

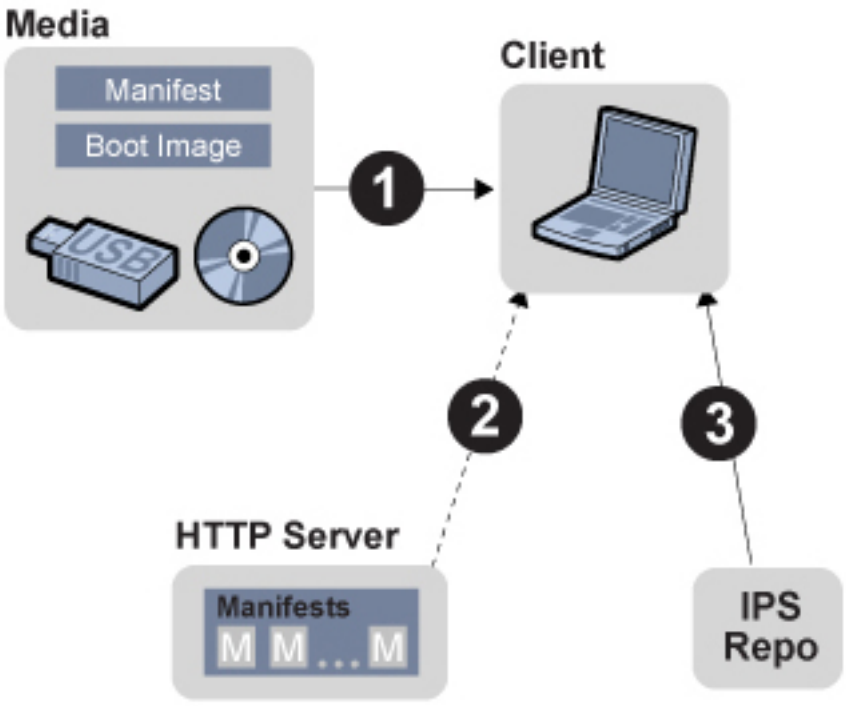
Installation using AI media has the following characteristics:

- You do not need to set up an AI 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 62.

FIGURE 1 AI Install Using Media



System Requirements for Installing Using AI Media

Both SPARC and x86 systems must meet the following requirements:

- Memory - To check the minimum memory requirement for the current release, see *Oracle Solaris 11.3 Release Notes*
- Disk Space - To check the disk space requirements for the current release, see *Oracle Solaris 11.3 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: `https://www.oracle.com/technetwork/server-storage/solaris11/downloads/index.html`.

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 62.

3. **Create bootable media.**

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

   Note - To add this utility to your system, install the `pkg:/install/distribution-constructor` package.

   Alternatively, you can use the `dd` command instead of the `usbcopy` utility: For example:

   ```
   # dd bs=16k conv=sync if=<image path> of=/dev/rdsk/c4t0d0p0
   ```

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 63 and "Booting an x86 System From AI Media" on page 64 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 "How to Use the SCI Tool" on page 68.

▼ How to Create a Persistent Device Alias for a USB Flash Drive on a SPARC System

To be able to create a persistent device alias for a USB flash drive on a SPARC system, you must use the OBP. Therefore, you must shut the system down. Once the alias is created, you will not need to re-create the alias as long as you reuse the same port.

1. **Shut down the system and leave it at the boot prompt.**

2. **Identify available disks on the system.**

   This example shows the selection of the second device.

```
{0} ok show-disks
   a) /pci@400/pci@0/pci@9/pci@0/usb@0,2/hub@2/storage@3/disk
   b) /pci@400/pci@0/pci@9/pci@0/usb@0,2/hub@2/storage@2/disk
   c) /pci@400/pci@0/pci@1/scsi@0/disk
   d) /iscsi-hba/disk
   q) NO SELECTION

 Enter Selection, q to quit: b
   /pci@400/pci@0/pci@9/pci@0/usb@0,2/hub@2/storage@2/disk has been selected.
   Type ^Y ( Control-Y ) to insert it in the command line.
   e.g. ok nvalias mydev
           for creating devalias mydev for
   /pci@400/pci@0/pci@9/pci@0/usb@0,2/hub@2/storage@2/disk
```

3. **Set an alias for the USB flash drive.**

```
{0} ok nvalias usbdrive ^Y
```

4. **Boot from the USB flash drive.**

```
{0} ok boot usbdrive
   Boot device: /pci@400/pci@0/pci@9/pci@0/usb@0,2/hub@2/storage@2/disk File
   and args:
```

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/manifest/default.xml` in the image.

```
# /usr/sbin/mount -o ro -F hsfs /tmp/sol-11_3-20-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.

To find out how to change installation specifications such as target disk or additional packages to install, see the `aimanifest(1M)` 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.

Using the Default AI Manifest to Boot a SPARC System From AI Media

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.

Using a Custom AI Manifest to Boot a SPARC System From AI Media

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.

Booting 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 1.99.5.11.0.175.2.0.0.20.0

Oracle Solaris 11.3 Automated Install custom
Oracle Solaris 11.3 Automated Install
Oracle Solaris 11.3 Automated Install custom ttya
Oracle Solaris 11.3 Automated Install custom ttyb
Oracle Solaris 11.3 Automated Install ttya
Oracle Solaris 11.3 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.
```

Using the Default AI Manifest to Boot a x86 System From AI Media

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 11.3 Automated Install
Oracle Solaris 11.3 Automated Install ttya
Oracle Solaris 11.3 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.

Using a Custom AI Manifest to Boot a x86 System From AI Media

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

```
Oracle Solaris 11.3 Automated Install custom
Oracle Solaris 11.3 Automated Install custom ttya
Oracle Solaris 11.3 Automated Install custom ttyb
```

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

```
Enter the AI manifest location [URL, /filepath, '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.

Booting an x86 Image Without Installing

You might want to boot from media but not install, for example, to troubleshoot or to examine a system.

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, if `install=true` is specified in the kernel line for the GRUB2 entry that you plan to use, edit the line to remove `install=true`. 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/log/install/install_log` after you reboot.

♦ ♦ ♦ CHAPTER 6

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 reconfigure an Oracle Solaris instance if circumstances require.

Using the `sysconfig` Command

If you need to change the configuration at a later time, you can use the `sysconfig configure` command. The same command is used to reconfigure the system's global zone or an Oracle Solaris instance in a non-global zone. This configuration can occur either non-interactively or interactively. A non-interactive reconfiguration uses an existing configuration profile whose specifications in the content are applied to the system. An interactive reconfiguration uses the System Configuration Interactive (SCI) tool to create a new configuration, and the specifications overwrite the current profile.

To use the non-interactive method, first ensure that a configuration profile already exists whose specifications will replace the current configuration. Then type:

```
$ sysconfig configure -c configuration-profile
```

Note - More examples of the use of the `sysconfig` command are available in Chapter 11, “Defining AI Client System Configuration Parameters”.

EXAMPLE 1 Reconfiguring a System Using a System Configuration Profile

The following command specifies that the system be configured using the existing system configuration profile named `myprofile.xml`.

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

How to Use the SCI Tool

The SCI Tool enables you to configure system configuration profiles interactively.

1. **Become the root role.**

2. **Choose one of the following commands:**

   - **Replace the current configuration profile.**

     ```
     # sysconfig configure
     ```

     The current configuration of a system is stored in the `sc_profile.xml` file. If you use this command, the settings that you define in the SCI Tool panels will replace the current configuration.

   - **Create a new configuration profile.**

     ```
     # sysconfig create-profile -o new-profile-location
     ```

     This command creates a new `sc_profile.xml` in the location that you specified. For example, if you specified `-o /tmpdir`, then the new profile is `/tmpdir/sc_profile.xml`. Thus, the current configuration profile is not overwritten. If you prefer, you can rename the profile later after you have completed setting parameters in the SCI Tool panels.

   Either command syntax opens the SCI Tool. The tool consists of interactive panels for setting system configuration information.

3. **On each panel of the SCI Tool, provide values to the configuration parameters as prompted and as needed.**

   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.

4. **To apply the specifications, save and exit the tool.**

   If you used the command `sysconfig configure`, the settings are applied to the system. If you used the command `sysconfig create-profile`, the settings are saved to `sc_profile.xml` in the location you specified, for example, `/tmpdir/sc_profile.xml`.

   Note - For descriptions of the panels, see "How to Perform a Text Installation" on page 48, beginning with Step 6.

Functional Groupings Overview

When you unconfigure or reconfigure an Oracle Solaris instance, you can change the configuration data for the whole system or for specific subsystems. These subsystems are referred to as functional groupings. The `system` functional grouping to changes all of the functional groups on the system. Alternately you can specify one or more functional grouping to change only specific configuration components.

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

TABLE 5 Functional Groupings

Grouping	Components	Unconfigured State
`date_time`	System date and time	N/A
`identity`	System nodename	Unknown
`keyboard`	Keyboard	U.S. English
`location`	Timezone	`UTC`
	Locale	C locale
`naming_services`	DNS, NIS and LDAP clients, nsswitch	No network naming services
`network`	Network	No network
`support`	OCM and ASR support	Default setting is anonymous registration with OCM and ASR
`system`	Full system	The “system” grouping includes all the other groupings.
`users`	Root	Empty root password
	Initial user account	Remove user account

EXAMPLE 2 Creating and Using a Profile to Configure Functional Groupings

The following example creates a profile for the `network` and `naming_services` functional groups. Then the profile is used to reconfigure the functional groupings.

```
# sysconfig create-profile -g network,naming_services -o /tmp/myprofile.xml
# sysconfig configure -g network,naming_services -c /tmp/myprofile.xml
```

PART III

Installing Using an Install Server

This section describes automated installation of client systems over a network. The following topics are covered:

♦ ♦ ♦ CHAPTER 7

Automated Installation of Multiple Systems

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

What Is an Automated Installation?

AI automates the installation of the Oracle Solaris OS on SPARC and x86 clients over the network. You can customize the installation with pre-installation instructions that define the disk layout and software package selection. You can also provide custom system configuration parameters that define the host name, network configuration, user accounts and other post-installation client-specific instructions. All customizations can be made on a client-by-client basis and scaled for large environments.

Note - Non Oracle x86 systems with Intel® Virtualization Technology for Directed I/O (VT-d) must have the `Intel VT-d` parameter set to `Enabled` before you install Oracle Solaris on those systems. Refer to their respective documentation for instructions to set this parameter.

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

1. The AI client boots over the network and gets its network configuration and the location of the AI server from the DHCP server. SPARC clients can optionally get network configuration information and the location of the AI server from the `network-boot-arguments` variable set in the Open Boot PROM (OBP).
2. The install service provides a boot image to the AI client.
3. Characteristics of the AI client determine the installation instructions and system configuration instructions that are used to install the AI client.

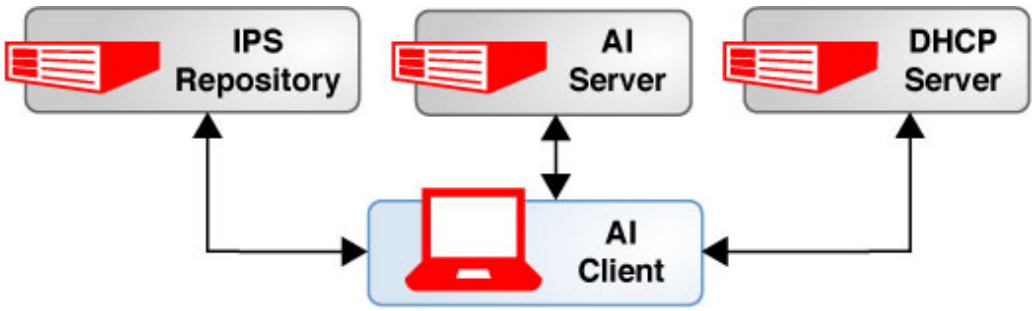
4. The Oracle Solaris OS is installed on the AI client. The installer pulls packages from the package repository or a system image from an archive specified by the installation instructions in the install service.

Components of the Automated Installer

A network using AI is composed of the following components.

- A DHCP server that provides the AI client with host information.
- One or more IPS (Image Packaging System) repositories that provide the software packages to install on the AI client. The system image for the AI client may also be created from an archive located on the network.
- An AI server that has configuration instructions for the AI clients.
- One or more AI clients.

FIGURE 2 AI Network Example



The DHCP server, IPS server, and AI server do not need to be hosted on separate systems. In particular, installing the AI server and the DHCP server on the same system eases administration steps because the `installadm` command will update the DHCP service if the DHCP service is co-located with the install service.

The AI server itself can contain the following components:

- One or more install services. Each service is configured to match the architecture and OS to be installed on the AI client.
- One or more AI manifests. An AI manifest provides installation instructions, such as the disk layout to use or the packages to add.

- Optional system configuration profiles. These profiles provide system configuration information, such as which timezone or name service to use.

In addition, you can create an IPS package to deliver a first-boot script to the AI client during the installation process. This script can perform additional installation or configuration steps that cannot be done using an AI manifest or system configuration profile, such as adding a third-party utility to an AI client.

DHCP Servers Supporting AI

The DHCP server manages network connections between the AI server and the clients. It provides necessary network information as well as the AI server location to the clients. You should co-locate the AI server and DHCP server to facilitate administration.

A SPARC client can be configured to locate the AI server without DHCP, but an x86 client cannot. For alternative ways to perform AI without DHCP, see “Installing an x86 AI Client” on page 257 and “Installing a SPARC AI Client” on page 254.

IPS Repositories Supporting AI

The client stems you want to install must be able to access an Oracle Solaris Image Packaging System (IPS) software package repository or an Oracle Solaris Unified Archive. 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 AI server, on another stem on the local network, or elsewhere on the Internet. See “Configuring Publishers” in *Adding and Updating Software in Oracle Solaris 11.3* for information about accessing a package repository. The IPS server can also provide any first-boot scripts that are needed to completely configure the AI client.

AI Server

To use AI to stall systems over the network, you must first set up an AI install service on an AI server. For the complete procedure, see Chapter 8, “Setting Up an AI Server”. Part of the procedure shows how to create a static network address for the AI server because the IP address for the server is included in the files created for each AI client. If the IP address of the AI server changes, then the configuration files for all AI clients have to be re-created.

Install Services

Each server can include one or more install services. You must create an install service for each version of the OS and for each client architecture you need to support. For example you could have an install service for SPARC clients booting Oracle Solaris 11.2, another for SPARC clients booting Oracle Solaris 11.3, and then two more to provide the same services for x86 clients. Each install service includes a SPARC or x86 boot image, one or more installation instruction files (AI manifests), and optional system configuration profiles. “Creating an Install Service” on page 97 provides instructions for creating and maintaining install services.

The boot image provided by the AI server is not a complete installation. The boot image creates a configuration on the stem in which the installation can run. client stems must access an IPS package repository or an archive to complete their installations.

AI Manifests

An AI manifest includes client provisioning or installation instructions. Each AI client uses only one AI manifest, although many AI clients can share one manifest. The AI manifest specifies one or more IPS package repositories where the AI client retrieves the packages needed to complete the installation. You can use an archive can be used in place of the IPS packages. The AI manifest also can include the names of additional packages to install, and information such as the target installation device and partition information. If you need to install two client stems with the same version of the Oracle Solaris OS but they need to be installed differently in other ways, then create two AI manifests associated with one AI install service. The different AI manifests can specify different packages to install or a different slice as the install target, for example. See Chapter 10, “Defining AI Client Installation Parameters” for information about creating and customizing AI manifests, either prior to booting the AI client or dynamically at installation time.

System Configuration Profiles

If AI clients 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. A profile that sets up the time zone may be used by several AI clients. See Chapter 11, “Defining AI Client System Configuration Parameters” for information about profiles.

If no profiles are configured for an AI client, an interactive tool will prompt for system configuration information after the AI client boots once the installation is done. You would need to manually provide the information as prompted.

First-boot Scripts

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. See Chapter 13, “Running a Custom Script During First Boot” for detailed information.

AI Clients

The installation begins when you boot the AI client. When the AI client boots, the client is directed to the AI server, and the client accesses the correct install service and the correct AI manifest and system configuration profiles associated with that service. Chapter 9, “Assigning Customizations to AI Clients” explains how an AI server identifies the correct AI manifest and system configuration profiles to use when an AI client is installed.

Securing AI

You can secure automated installations with the Transport Layer Security (TLS) protocol. You can assign private certificate and key pairs and Certificate Authority (CA) certificates to the AI server and to the AI clients. You can further secure SPARC clients with OBP HMAC and encryption keys. For more information, see “Increasing Security for Automated Installations” on page 109.

AI and Zones

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 installation.

Overview of the AI Configuration Process

This section describes the primary distinct ways to use AI. Note that some of these actions are optional depending on how much customization you want to do automatically. “Automated

Installer Use Cases" on page 81 provides more information about the actions that might be appropriate for your situation.

1. Configure the AI server to provide the necessary configuration to support AI clients. See "How to Set Up An AI Server" on page 92 for complete instructions. When the AI server is configured:
   - The correct OS is installed.
   - The network interface is configured using a static IP address.
   - The `installadm` package is installed.

   Optionally, multicast DNS may be enabled.
2. Create an install service to create a boot image for the AI clients to use. You must create an install service for each client architecture and each version of the OS that you need to be able to install. When you create the first install service for a particular architecture on an AI server, an alias 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. See "Creating an Install Service" on page 97 for more information.
3. (Optional) Associate an AI client with a service. See "Associating an AI Client With an Install service" on page 105.This step is necessary if you are supporting one client architecture with multiple versions of the OS.
4. (Optional) Create an AI manifest to define where the package repository or archive used to install the AI client is located. See Chapter 10, "Defining AI Client Installation Parameters" for more information. When creating an AI manifest, you can associate the manifest with particular AI clients by specifying criteria. For more information, see "Matching AI Clients With Installation and Configuration Instructions" on page 139. Once you have created the AI manifest, you must associate the manifest with an install service. See "Associating Client-Specific Installation Instructions With an Install Service" on page 106 for full instructions. Creating a manifest and associating that manifest with an install service is necessary if you want to customize the installation for an AI client. Here is some of the information that can be customized:
   - The disk layout
   - The location of the IPS repository or archive
   - The locales to be installed
   - The packages to be installed
5. (Optional) Create one or more system configuration profiles to provide additional configuration information for the AI client. See Chapter 11, "Defining AI Client System Configuration Parameters" for more information. When creating a system configuration profile, you can associate the profile with particular AI clients by specifying criteria. For more information, see "Matching AI Clients With Installation and Configuration Instructions" on page 139. Once you have created a system configuration profile, you must associate the profile with an install service. See "Associating Client-Specific

Configuration Instructions With Install Services" on page 108 for full instructions. Creating one or more system configuration profiles and associating them with an install service is necessary if you want to automatically configure any of the following:

- SMF services, including enabling, disabling and setting properties
- User accounts and groups
- Nodename
- Time zone and locale
- Terminal type and keyboard layout
- Network interfaces
- Name service
- Oracle Configuration Manager and Oracle Auto Service Request (ASR)

You can choose to configure any or all of these items by creating a system configuration profile. Once the AI client has been installed, you will need to configure manually any items that are not specified in a system configuration profile. If you choose not to use any profiles, you will be prompted to make the changes after the installation has completed.

6. (Optional) Create a first-boot script to provide configuration requirements that can't be added to an AI manifest or profile. See Chapter 13, "Running a Custom Script During First Boot" for more information.
7. Make sure the AI clients can access a DHCP server and an IPS server or a stem storing the archive, as needed.
8. Boot the AI client over the network using the install service.

Booting an AI Client

When you network boot an AI client, the following actions occur:

1. The AI client gets the AI server address from the DHCP server.

   SPARC clients can optionally get the AI server address from the `network-boot-arguments` variable in the OBP.
2. The AI client gets the boot program from the AI server and loads the boot program.
   - SPARC clients get the `wanboot` boot program from the AI server.
   - X86 clients get the GRUB menu (which includes the install service name) and the `pxegrub` boot program from the AI server.
3. The AI client downloads the boot archive from the AI server and loads the kernel.
4. The AI client uses HTTP to download the install program.

   SPARC clients get the install service name with the install program.

5. The AI server selects a manifest based on the selection criteria. Installation changes are made on the AI client based on the manifest selected.
6. The AI server selects profiles based on the AI client selection criteria.
7. The AI install program installs the AI client packages from the IPS repository or the image from an archive.
8. The AI client is rebooted to use the image on the local disk.
9. If no system configuration profiles were used, you are prompted for system configuration information.
10. If appropriate, system configuration changes are made on the AI client based on the information given in the selected profiles
11. If appropriate, the first-boot script is run.

Planning for an AI Server

Before you install the AI server, you have to make some configuration choices as described in this section.

Configuring Network Interfaces on an AI Server

You can configure an AI server on a stem that has multiple interfaces. By default, the AI server is configured to support AI clients on all network interfaces. If multiple network interfaces are on the stem hosting the AI server, you can disable the install service for those networks that you don't want to stem to support, as shown in Example 3, “Disabling AI Support on a Network,” on page 94.

Identifying Necessary Install Instances

Each AI server includes one or more install services. Each install service is set up to support an architecture (SPARC or x86) and a given OS release (such as Oracle Solaris 11.2 or 11.3). You might need only one or two of the possible options. For example, if you have only x86 clients but want to have the option of installing Oracle Solaris 11.2 or 11.3, you would need to create two install services, one service for each version of the OS. If you have both x86 and SPARC clients and want to only support Oracle Solaris 11.3 installations, you would create an

install service for each architecture. See “Creating an Install Service” on page 97 for more information.

Automated Installer Use Cases

The examples in this section illustrate the major steps to configure services on an AI server. The cases start from the simplest service configuration and develop to a more advanced configurations. Most of the configuration steps for establishing an install service are optional and whether you use them depends on your environment.

AI Server Supporting One Architecture and One OS

This configuration is one of the simplest to create. This configuration requires that you manually enter all of the configuration data for each installation.

FIGURE 3 AI Server Supporting One Architecture and One OS

To create this environment:

1. Configure the AI server. See “How to Set Up An AI Server” on page 92.
2. Create an install service. See “Creating an Install Service” on page 97.

After completing these steps, the AI server includes an install service that includes the default AI manifest. When the AI client is booted after the installation completes, an interactive tool prompts for system configuration information because no system configuration profile

was configured. Any additional configuration steps will need to be done manually. System configuration profiles can be added to automate the system configuration steps.

AI Server Supporting Two Architectures

In this example, an AI server is configured to support x86 and SPARC client architectures.

FIGURE 4 AI Server Supporting Two Architectures

To create this environment:

1. Configure the AI server. See "How to Set Up An AI Server" on page 92.
2. Create an install service for the x86 clients. See "Creating an Install Service" on page 97.
3. Create an install service for the SPARC clients.

After completing these steps, the AI server includes two install services, one for each client architecture. Each service includes a default manifest. When the AI client is booted after the installation completes, an interactive tool prompts for system configuration information because no system configuration profile was configured. Any additional configuration steps will need to be done manually. Profiles can be added to automate the system configuration steps.

AI Server Supporting One Architecture and Two Disk Configurations

In this example, an AI server is configured to support only one client architecture with two configurations for the AI client disk layout.

FIGURE 5 AI Server Supporting One Architecture and Two Disk Layouts

To create this environment:

1. Configure the AI server. See "How to Set Up An AI Server" on page 92.
2. Create an install service for the AI clients. See "Creating an Install Service" on page 97.
3. Create an AI manifest for the second disk layout. See Chapter 10, "Defining AI Client Installation Parameters".
4. Associate the second manifest with the install service. See "Associating Client-Specific Installation Instructions With an Install Service" on page 106.

After completing these steps, the AI server includes one install service with two manifests. The default manifest includes one disk layout. The second manifest includes a second disk layout. The second manifest is associated with criteria to identify the AI clients that should use it. All other AI clients will use the default manifest. When the AI client is booted after the installation completes, an interactive tool prompts for system configuration information because no system configuration profile was configured. Any additional configuration steps will need to be done manually. Profiles can be added to automate the system configuration steps.

AI Server Supporting One Architecture and Two Time Zones

In this example, an AI server is configured to support only one client architecture with two system configuration profiles that set the time zone.

FIGURE 6 AI Server Supporting One Architecture and Two Disk Configurations

To create this environment:

1. Configure the AI server. See "How to Set Up An AI Server" on page 92.
2. Create an install service for the AI clients. See "Creating an Install Service" on page 97.
3. Create a system configuration profile for one time zone. See Chapter 11, "Defining AI Client System Configuration Parameters".
4. Create a system configuration profile for the second time zone.
5. Associate both system configuration profiles with the install service. See "Associating Client-Specific Configuration Instructions With Install Services" on page 108.

After completing these steps, the AI server includes one install service, which includes the default manifest, and two profiles that set different values for the time zone. Each profile is

associated with criteria to identify which AI clients should use which profile. When the AI client is booted after the installation completes, all additional configuration information not included in the system configuration profile will need to be configured manually.

AI Server Supporting One Architecture and Two Releases

In this example, an AI server is configured to support only one client architecture with two versions of the Oracle Solaris OS.

FIGURE 7 AI Server Supporting One Architecture and Two Releases

To create this environment:

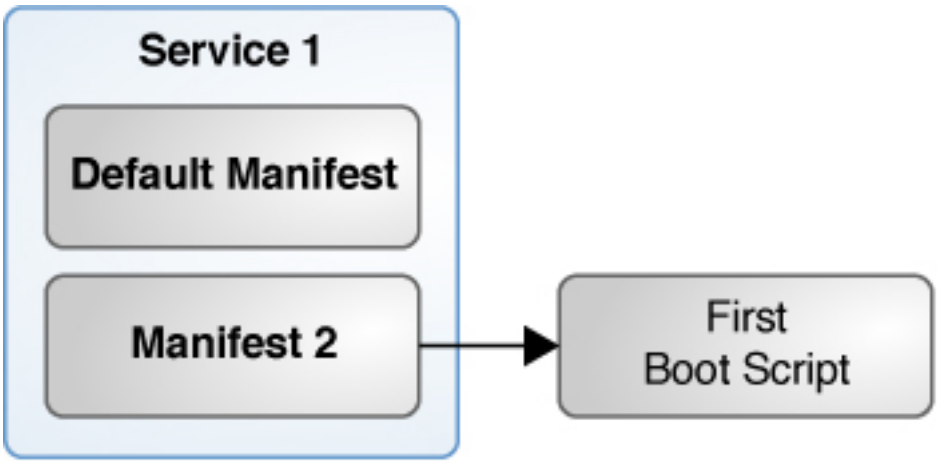
1. Configure the AI server. See "How to Set Up An AI Server" on page 92.
2. Create an install service for the first OS. See "Creating an Install Service" on page 97.
3. Create an second install service for the second OS. See "Creating an Install Service" on page 97.
4. Create client definitions to associate the AI clients with the appropriate service. See "Associating an AI Client With an Install service" on page 105.

After completing these steps, the AI server includes two install services. The client definitions determine which AI client uses which service. When the AI client is booted after the installation completes, an interactive tool prompts for system configuration information because no system configuration profile was configured. Any additional configuration steps will need to be done manually. Profiles can be added to automate the system configuration steps.

AI Server Supporting One Architecture with Additional Configuration for Some AI Clients

In this example, an AI server is configured to support only one client architecture, but one set of AI clients needs additional configuration which can not be done in a manifest or profile.

FIGURE 8 AI Server Supporting One Architecture With Additional Configuration for Some AI Clients



To create this environment, use the following steps:

1. Configure the AI server. See "How to Set Up An AI Server" on page 92.
2. Create an install service for the AI clients. See "Creating an Install Service" on page 97.
3. Create a first-boot script. See Chapter 13, "Running a Custom Script During First Boot".
4. Create a package that includes the first-boot script and add the package to a package repository. See "How to Create and Publish the IPS Package" on page 242.
5. Create a second manifest which includes the first-boot package. See "Associating Client-Specific Installation Instructions With an Install Service" on page 106.
6. Associate the second manifest with the install service. See "Associating Client-Specific Installation Instructions With an Install Service" on page 106.

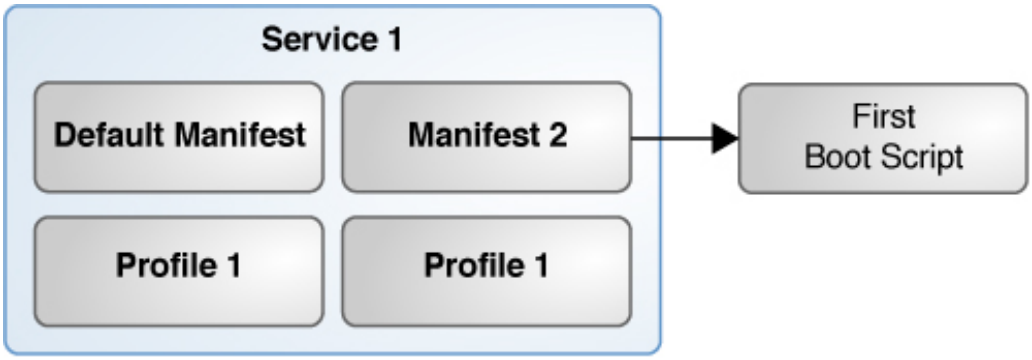
After completing these steps, the AI server includes one install service with two manifests. The second manifest includes instructions to install the first-boot service package, which runs the

first-boot script once the AI client has rebooted. The second manifest is associated with criteria to identify which AI clients should use it. All other AI clients use the default manifest. When the AI client is booted after the installation completes, the first-boot script is run. Next, an interactive tool prompts for system configuration information because no system configuration profile was configured. Any additional configuration steps will need to be done manually. Profiles can be added to automate the system configuration steps.

AI Server Supporting Many Configuration Changes

In this example, an AI server is configured to support one architecture with two configurations. The first configuration is very basic. The second configuration could be used to choose a different disk layout and to do additional configuration. Both configurations use the same system configuration profile to configure the time zone.

FIGURE 9 AI Server Supporting Many Configuration Changes



To create this environment:

1. Configure the AI server. See “How to Set Up An AI Server” on page 92.
2. Create an install service for the AI clients. See “Creating an Install Service” on page 97.
3. Create a first-boot script. See Chapter 13, “Running a Custom Script During First Boot”.
4. Create a package that includes the first-boot script and add the package to a package repository. See “How to Create and Publish the IPS Package” on page 242.
5. Create a second manifest which includes the first-boot package and define the second disk layout. See Chapter 10, “Defining AI Client Installation Parameters”.

6. Associate the second manifest with the install service. See “Associating Client-Specific Installation Instructions With an Install Service” on page 106.
7. Create a system configuration profile to set the time zone. See Chapter 11, “Defining AI Client System Configuration Parameters”.
8. Associate the system configuration profile with the install service. See “Associating Client-Specific Configuration Instructions With Install Services” on page 108.

After completing these steps, the AI server includes one install service with two manifests. The second manifest includes information about the second disk layout and instructions to install the first-boot service package which includes the first-boot script. The first-boot script is run once the AI client has rebooted. The second manifest is associated with criteria to identify the AI clients that should use it. All other AI clients use the default manifest. AI clients using either manifest may be configured to also use the profile depending on the criteria associated with the profile. When the AI client is booted after the installation completes, the first-boot script is run. All additional configuration information not included in the system configuration profile will need to be configured manually.

♦ ♦ ♦ CHAPTER 8

Setting Up an AI Server

To AI clients over the network, AI requires a separate system to function as an AI server. On the AI server, create an install service to provide a net image and instructions for installing the desired Oracle Solaris release on different AI clients. The following topics are covered:

AI Server Setup Tasks

The high-level steps to set up an AI server are:

- Check whether the server meets the minimum requirements to be an AI server. For more information, see "AI Server Requirements" on page 90.
- Decide which method to use to be able to use the AI commands. For a full description, see "Install Service Administrator Privileges" on page 90.
- Configure the AI server to use a static IP address and a default route, install the AI package, and, if needed, enable the `svc:/network/dns/multicast` SMF service. See "Configuring an AI Server" on page 91 for full instructions.

Note - For a description of the example IP addresses used in this guide, see the `IP address` entry in *Glossary of Networking Terms*.

- If needed, change additional settings on the AI server such as the networks on which to enable install services, the AI web server port number, and the default image path for all images. See "Changing the Configuration of an AI Server" on page 94

AI Server Requirements

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

Operating system Install the Oracle Solaris OS on the AI server. For more information, see "Comparing Installation Options" on page 21. To find out how to update the software on an existing AI server, see "Image Update Overview" in *Adding and Updating Software in Oracle Solaris 11.3*. Using the most recent version of the OS on an AI server enables you to use all of the new functionality while still supporting the installation of older versions.

Note - Any Oracle Solaris 11 release (including updates and SRUs) can be used as the installed OS on an AI server which installs 11.3 clients.

Memory The minimum requirement is 1 GB of memory.

Disk space Additional disk space required to operate as an AI 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 OS that you plan to install on AI client. Each net image is approximately 300-400 MB.

Install Service Administrator 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.

Configuring an AI Server

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

How to Set Up An AI Server

This procedure describes how to set up a server so you can perform automatic installations of Oracle Solaris over the network.

1. **Install Oracle Solaris 11.3.**

   Although you can install any version of Oracle Solaris 11 on an AI server, the Oracle Solaris 11.3 release includes several server-side enhancements which can only be used if that version is installed on the AI server. If you have an existing Oracle Solaris AI server, see the documentation for that release for AI server configuration instructions. For suggestions on which installation method to use, see "Comparing Installation Options" on page 21.

2. **Become an administrator.**

   For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

3. **Verify that the network interface IP address is static.**

   In this example `net0` is configured using a static IP address.

```
# ipadm
NAME              CLASS/TYPE STATE        UNDER      ADDR
net0              ip         ok           --         --
   net0/v4        static     ok           --         192.0.2.5/24
lo0               loopback   ok           --         --
   lo0/v4         static     ok           --         127.0.0.1/24
   lo0/v6         static     ok           --         ::1/128
```

   If the address is not static, follow the steps below.

   a. **Create the IP interface.**

```
# ipadm create-ip net0
```

   b. **Configure a static IP on the interface.**

```
# ipadm create-addr -T static -a local=192.0.2.5/24 net0
```

   c. **Verify configuration.**

```
# ipadm
    NAME      CLASS/TYPE     STATE     UNDER     ADDR
    net0      loopback       ok        --        --
   net0/v4     static    ok        --        192.0.2.5/24
```

4. **(Optional) Establish a default route for the AI server.**

```
# route -p add default 192.0.2.1
```

5. **Install the AI package.**

   a. **Verify that the AI package is not already installed.**

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

   b. **Verify that your IPS package repository contains the AI package.**

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

   c. **Install the AI package.**

```
# 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
```

6. **Enable multicast DNS (mDNS).**

   The mDNS service enables AI clients to find replicated install services on other AI servers on the same subnet. For more information about mDNS, see "Multicast DNS and Service Discovery" in *Working With Oracle Solaris 11.3 Directory and Naming Services: DNS and NIS*.

   a. **If needed, install the mDNS package.**

```
# pkg install pkg:/service/network/dns/mdns
```

   b. **Update name service switch information.**

To be able to resolve local hosts, change the `config/host` property of the `name-service/switch` service to include `mdns` as a source. For example:

```
#  /usr/sbin/svccfg -s svc:/system/name-service/switch
svc:/system/name-service/switch> setprop config/host = astring: "files dns mdns"
svc:/system/name-service/switch> select system/name-service/switch:default
svc:/system/name-service/switch:default> refresh
svc:/system/name-service/switch> quit
```

c. Enable the mDNS service.

```
# svcadm enable svc:/network/dns/multicast:default
```

Enabling mDNS in this way ensures that your changes persist through upgrades and reboots. For more information, see the `svcadm`(1M) man page.

Changing the Configuration of an AI Server

Several configuration settings can be changed on an AI server by using the `installadm set-server` command. These configuration settings include:

- The networks to exclude from AI support
- The networks to include AI support on
- The AI web server port

You must become an administrator to use this command. For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

EXAMPLE 3 Disabling AI Support on a Network

By default, the AI server is configured to serve AI clients on all networks that the server is connected. In this example, the `192.0.2.0/27` network interface will no longer service AI requests.

```
# installadm set-server -L 192.0.2.0/27
```

EXAMPLE 4 Including Networks to Be Supported by an AI Server

In some situations, it is easier to list the networks that should support install services rather than listing the networks to be excluded. The following command shows how to allow install services on two networks.

```
# installadm set-server -l 192.0.2.0/27, 192.0.2.32/27
```

EXAMPLE 5 Configuring the AI Web Server Port Number

An AI server hosts install services using a web server. By default, the AI web server is hosted on port 5555. You can view install service files at `http://localhost:5555`. You can change the port number used for the web server. The following command configures the AI server to host install services from port 7000:

```
# installadm set-server -p 7000
```

EXAMPLE 6 Configuring the Secure AI Web Server port Number

A secure AI server hosts install services using a web server. By default, the secure AI web server is hosted on port 5556. You can securely view install service files at `http://localhost:5556`. You can change the port number used for the web server. The following command configures the AI server to host secure install services from port 7001:

```
# installadm set-server -P 7001
```

EXAMPLE 7 Configuring the Default Image Path

By default, images are created in a *service-name* directory in `/export/auto_install`. Thus, by default, the net image for the *service-name* service is created at `/export/auto_install/`*service-name*. The following command configures the AI server to create new install services at `/export/aiimages/`*service-name* by default:

```
# installadm set-server -d /export/aiimages
```

EXAMPLE 8 Defining IP Addresses and Number of AI Clients for an AI Server

The following example sets up an I server to act as a DHCP server for the network. The DHCP server will be set up to serve twenty IP addresses (`-c`), starting from `203.0.113.1` (`-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.

If the IP range requested is not on a subnet that the AI server is directly connected to and the I 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 AI server and DHCP relays are employed. In other configurations, the software can determine this automatically.

```
# installadm set-server -i 203.0.113.1 -c 20
```

EXAMPLE 9 Disabling Automatic Updates of the Local DHCP Service on an AI Server

By default, the local ISC DHCP configuration is automatically updated when AI client and service configurations are modified in the AI server. If you do not want the local ISC DHCP configuration to be automatically maintained, use the following command:

```
# installadm set-server -M
Changed Server
Disabling SMF service svc:/network/dhcp/server:ipv4
Refreshing SMF service svc:/system/install/server:default
```

EXAMPLE 10 Enabling DHCP Updates on the AI Server

If automatic updates to the ISC DHCP configuration is disabled, you can enable it using the following command:

```
# installadm set-server -m
Warning: AI server will now manage DHCP
Changed Server
Enabling SMF service svc:/network/dhcp/server:ipv4
```

Configuring the Web Server User Files Directory

The AI web server serves net images, AI manifests, and system configuration profiles that have been added using the `installadm` command. The AI web server can also serve files provided by the AI user or administrator.

You can store user files that do not need to be secured in the directory specified by the `all_services/webserver_files_dir` property of the `svc:/system/install/server:default` SMF service. This property does not have a default value. If you specify a value for this property, the value must be a directory on the local system. That directory can be viewed through the AI web server at the following URL, where *server* is the AI server's hostname or IP address and *port* is the AI web server port number discussed in Example 5, “Configuring the AI Web Server Port Number,” on page 95:

`http://`*server*`:`*port*`/files`

User files that do need to be secured can be stored in the directory specified by the `all_services/webserver_secure_files_dir` property. This property does not have a default value. If you specify a value for this property, the value must be a directory on the local system. That directory can be viewed through the AI web server at the following URL, where *server* is the AI server's hostname or IP address, and *secure-port* is the secure AI web server port number

discussed in Example 6, “Configuring the Secure AI Web Server port Number,” on page 95 above:

`https://`*server*`:`*secure-port*`/secure_files`

If the AI manifest specifies an IPS package repository that requires a certificate and key, you can store those publisher credentials, and then specify this URI in the AI manifest. Only AI clients that have security credentials assigned can access this directory.

Tip - For greatest security, files in the `webserver_secure_files_dir` directory should be owned by user `webservd` and group `webservd` and have no world access.

Working With Install Services

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

- “Creating an Install Service” on page 97
- “Associating AI Clients With Install Services” on page 105
- “Associating Client-Specific Installation Instructions With an Install Service” on page 106
- “Associating Client-Specific Configuration Instructions With Install Services” on page 108
- “How to Configure Security for Automated Installations” on page 111
- “How to Configure Kerberos Clients Using AI” on page 120
- “Showing Information About Install Services” on page 125

Creating an Install Service

An AI 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 S that you want to install.

How to Create an Install Service

Create an install service for each client architecture, SPARC or x86, and for each version of the Oracle Solaris operating system you want to be able to install.

1. **Become an administrator.**

   For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

2. **Verify DHCP service.**

   Make sure that local or remote DHCP support is set up as appropriate.

3. **Create the install service.**

   See the `installadm`(1M) man page for all of the options that you can use. For example:

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

source	Specifies the data source for the net image. The value can be the FMRI identifier of the IPS AI net image package, which by default is `install-image/solaris-auto-install`. The value can also be the full pathname of an AI ISO image file.
`-y`	Suppresses the prompt to confirm the use of the automatically generated image path

Example 11 Creating a SPARC Install Service Using an ISO File With DHCP Enabled on the AI Server

This example creates an install service for SPARC clients where the network consists of a single subnet and the AI server also acts as the DHCP server for the network. 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.

If the AI client is not on a subnet that the AI server is directly connected to and the AI 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 AI server and DHCP relays are employed. In other configurations, the software can determine this automatically.

The only difference in the output if this command was run with an x86 ISO file, would be the name of the ISO file, the name and architecture type of the created services, and the description of the boot files that are created. For x86 output, see Example 12, "Creating an X86 Install Service Using an IPS Package," on page 99.

```
# installadm create-service -s /var/tmp/images/sparc/sol-11_3-ai-sparc.iso -y
  0% : Service svc:/network/dns/multicast:default is not online.  Installation services
 will not be advertised via multicast DNS.
  0% : Creating service from: /var/tmp/images/sparc/sol-11_3-ai-sparc.iso
 36% : Transferring contents
 36% : Creating sparc service: solaris11_3-sparc
 36% : Image path: /export/auto_install/solaris11_3-sparc
 36% : Setting "solaris" publisher URL in default manifest to:
 36% :  http://pkg.oracle.com/solaris/release/
 36% : Creating default-sparc alias
 36% : Setting "solaris" publisher URL in default manifest to:
 36% :  http://pkg.oracle.com/solaris/release/
 36% : Setting the default SPARC bootfile(s) in the local DHCP configuration to:
 36% :  http://203.0.113.2:5555/cgi-bin/wanboot-cgi
100% : Created Service: 'solaris11_3-sparc'
100% : Refreshing SMF service svc:/system/install/server:default
100% : Restarting SMF service svc:/network/dhcp/server:ipv4
# installadm list
Service Name           Status Arch  Type Secure Alias Aliases Clients Profiles Manifests
------------           ------ ----  ---- ------ ----- ------- ------- -------- ---------
default-sparc          on     sparc iso  no     yes   0       0       0        1
solaris11_3-sparc      on     sparc iso  no     no    1       0       0        1
```

Example 12 Creating an X86 Install Service Using an IPS Package

This example uses an x86 AI server without a local DHCP service. it creates an install service for x86 clients using an net image from an IPS package. This command also illustrates default behavior when options are not specified. If no other information is provided when using an IPS package, the architecture for the AI client is assumed to match the architecture of the AI server. If this AI 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 address required for DHCP configuration.

The only difference in the output if this command was run to create a SPARC, would be the name and architecture type of the services, and the description of the boot files that are created. For SPARC output, see Example 11, “Creating a SPARC Install Service Using an ISO File With DHCP Enabled on the AI Server,” on page 98.

```
# installadm create-service -y
  0% : Creating service from: pkg:/install-image/solaris-auto-install
  0% : Using publisher(s):
  0% :     solaris: http://pkg.oracle.com/solaris/release/
  5% : Refreshing Publisher(s)
```

```
  7% : Startup Phase
 15% : Planning Phase
 61% : Download Phase
 90% : Actions Phase
 91% : Finalize Phase
 91% : Creating i386 service: solaris11_3-i386
 91% : Image path: /export/auto_install/solaris11_3-i386
 91% : Setting "solaris" publisher URL in default manifest to:
 91% :  http://pkg.oracle.com/solaris/release/
 91% : DHCP is not being managed by install server.
 91% : Creating default-i386 alias
 91% : Setting "solaris" publisher URL in default manifest to:
 91% :  http://pkg.oracle.com/solaris/release/
 91% : DHCP is not being managed by install server.
 91% :  No local DHCP configuration found. This service is the default
 91% :  alias for all PXE clients. If not already in place, the following should
 91% :  be added to the DHCP configuration:
 91% : Boot server IP: 203.0.113.2
 91% : Boot file(s):
 91% :     bios clients (arch 00:00):  default-i386/boot/grub/pxegrub2
 91% :     uefi clients (arch 00:07):  default-i386/boot/grub/grub2netx64.efi
 91% :
100% : Created Service: 'solaris11_3-i386'
100% : Refreshing SMF service svc:/system/install/server:default
# installadm list
Service Name           Status Arch  Type Secure Alias Aliases Clients Profiles Manifests
------------           ------ ----  ---- ------ ----- ------- ------- -------- ---------
default-i386           on     i386  pkg  no     yes   0       0       0        1
solaris11_3-i386       on     i386  pkg  no     no    1       0       0        1
```

Example 13 Creating an Install Service for a Different Architecture

By default, when an install service is created, the architecture will be the same as the AI server. If you want to create a service of another architecture, use the -a option. The following example creates an x86 service on a SPARC AI server.

```
# installadm create-service -n solaris11_3-i386 -a i386 -y
```

Example 14 Creating a Service That Automatically Installs an X86 Client

The default entry in the GRUB menu on an x86 client will not automatically start AI. To customize the GRUB menu so that the installation is started automatically, you can use the following command:

```
# installadm create-service -s /var/tmp/images/i386/sol-11_3-ai-x86.iso -y -b
 install=true
```

What Happens When an Install Service Is Created?

When an 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 AI 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 AI clients of that architecture that are not explicitly associated with a different install service with the `create-client` subcommand.

To change the service that `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 the net image that the service uses. See "Managing Install Services" on page 131 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 Existing Install Service" on page 133.

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 Example 9, "Disabling Automatic Updates of the Local DHCP Service on an AI Server," on page 96 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://203.0.113.5:5555/solaris11_3-i386`. See Example 5, "Configuring the AI Web Server Port Number," on page 95 to use a different port.

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

1. If you don't define an install service name, a name will be generated for you. You can specify the service name by including the `-n` option in the command line when you create the install service.
2. If 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 AI server publisher list that provides this package.

3. The default install service net image directory is created. The directory name includes the service name, such as `/export/auto_install/solaris11_3-sparc` or `/export/auto_install/solaris11_3-i386`. To suppress confirmation prompts, specify the `-y` option.
4. Depending on the source for the net image, one of these two operations occur:

    a. If no net image source option is specified, the `install-image/solaris-auto-install` package is installed into the net image directory.

    By default, the variant of the `install-image/solaris-auto-install` package that is installed matches the architecture of the AI server. If the AI server is an x86 system and you want to create a SPARC install service on this server, you would need to use the `-a` option. See Example 13, “Creating an Install Service for a Different Architecture,” on page 100 for information about the `-a` option.

    b. If a net image source option is specified, the image file is unpacked or installed into the net image directory.
5. Files are created according to the install service architecture.
    - For SPARC clients: The `wanboot.conf` file for this service is generated at `/etc/netboot/wanboot.conf`
    - For x86 clients: The GRUB menu is mounted at `/etc/netboot/solaris11_3-i386/grub.cfg`.
6. The AI SMF service, `system/install/server`, is refreshed to mount `/export/auto_install/`*service-name* as `/etc/netboot/`*service-name*.
7. If this is the first SPARC install service created on this AI server, the `default-sparc` service alias is automatically created. Also, the `/export/auto_install/`*service-name* is mounted as `/etc/netboot/default-sparc`.

    For the first x86 install service, the `default-i386` service alias is created and the `/etc/netboot/default-i386` mount point would be created.
8. For SPARC clients, the configuration file `/etc/netboot/wanboot.conf` is symbolically linked to `/etc/netboot/default-sparc/wanboot.conf`. Also, the configuration file `/etc/netboot/system.conf` is symbolically linked to `/etc/netboot/default-sparc/system.conf`.
9. A DHCP service is created if necessary, and IP addresses 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`.
10. For configurations that do not use a local DHCP service:
    - For SPARC clients: The boot file required for DHCP configuration, `http://203.0.113.5:5555/cgi-bin/wanboot-cgi`, is provided.

- For x86 clients: The boot server IP address 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.

11. If a local ISC DHCP server is already configured, the boot file of the new `default-sparc` or `default-i386` alias is set as the default boot file for all matching AI clients. This assignment occurs regardless of whether the `-i` and `-c` options are used.

Example DHCP Configuration Files To Support AI Clients

This section shows how `installadm` might add information to the DHCP configuration file for an ISC DHCP configuration. For more information about configuring ISC DHCP, see the Chapter 2, "Administering the ISC DHCP Service" in *Working With DHCP in Oracle Solaris 11.3*.

ISC DHCP Configuration for an Oracle Solaris 11.3 `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 11.3 `i386` install service previously created:

```
subnet 203.0.113.0 netmask 255.255.255.224 {
  range 203.0.113.2 203.0.113.20;
  option broadcast-address  203.0.113.2224;
  option routers 203.0.113.1;
  next-server 203.0.113.25;
}
```

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 11.3 `i386` install service previously created:

```
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 11.3 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       : 203.0.113.25        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 11.3 `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://203.0.113.5:5555/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 11.3 `sparc` install service:

```
class "SPARC" {
  match if not (substring(option vendor-class-identifier, 0, 9) = "PXEClient");
  filename "http://203.0.113.5:5555/cgi-bin/wanboot-cgi";
}
```

Associating AI Clients With Install Services

The `installadm create-client` command associates an AI client with a specific install service. You can also provide custom settings for x86 clients. See "Setting Up an AI Client" on page 251 for more examples and sample output.

The `installadm delete-client` command removes the association of an AI client to an install service.

Associating an AI Client With an Install service

An AI 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 AI client is associated only with the install service that was specified last. You must be an administrator to run this command.

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

EXAMPLE 15 Associating a SPARC Client With an Install Service

The following command adds the AI client with MAC address `00:14:4f:a7:65:70` to the `solaris11_3-sparc` install service:

```
# installadm create-client -e 00:14:4f:a7:65:70 -n solaris11_3-sparc
```

EXAMPLE 16 x86: Associating an X86 Client With an Install Service and Redirecting Output to a Serial Line

The following example adds an x86 client and changes the boot properties in the client-specific `/etc/netboot/grub.cfg` file. In this example, the installation output is redirected to a serial console device.

```
# installadm create-client -e c0ffeec0ffee -n solaris11_3-i386 -b console=ttya
```

EXAMPLE 17 x86: Changing Boot Properties for an X86 Client

For x86 client systems, you can use the `-G` option to specify a custom GRUB2 menu to use when booting the client. This example specifies a custom GRUB2 menu named `/etc/netboot/grub.custom.cfg`.

```
# installadm create-client -e c0ffeec0ffee -n solaris11_3-i386 -G /etc/netboot/
grub.custom.cfg
```

Note that the `-b` and `-G` options cannot both be used at the same time.

If you want an AI Installation to automatically start, use the `-b install=true` option when creating the AI client using the `create-client` subcommand. To apply this setting to all AI clients of a service, you can use this option when creating the service using the `create-service` subcommand.

Deleting an AI Client From an Install Service

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

`installadm delete-client -e` *mac-addr*

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

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

Customizing Installation Instructions

You can employ AI manifests or derived manifest scripts to provide specific installation instructions for an install service. System configuration profiles provide configuration instructions. Multiple system configuration profiles can be associated with a service or with an AI client. Also, the profiles can be shared between many services.

Associating Client-Specific Installation Instructions With an Install Service

Use the `installadm create-manifest` command to associate a custom AI manifest with a specific install service. You can also add a derived manifest script to an install service. Note that each install service may have multiple AI manifests or derived manifest scripts associated with it. Any manifest or script that is not configured as the default manifest for a given service, should have client criteria defined, so that the right instructions are used for each AI client.

Before using any of the following examples, you must have first created an AI manifest. See Chapter 10, “Defining AI Client Installation Parameters” for instructions on creating an AI manifest.

The syntax of the command is:

```
installadm create-manifest -n service -f filename
```

service — Specifies the service that the manifest or derived manifest script is to be associated with.

filename — Identifies the path of the manifest or derived manifest script to associate with the service.

EXAMPLE 18 Associating Client Criteria With a Manifest

This example adds the `manifest-sparc-ent.xml` manifest to the `solaris11_3-sparc` install service. The `-c` option specifies that any AI clients that are using this install service and identify themselves as M5000 or M4000 servers are assigned the `manifest-sparc-ent.xml` installation instructions. The `-m` option sets the AI instance name of the manifest to `sparc-ent`.

```
# installadm create-manifest -n solaris11_3-sparc -f ./manifest-sparc-ent.xml \
-m sparc-ent -c platform="SUNW,SPARC-Enterprise"
```

EXAMPLE 19 Associating Client Criteria With a Script

This example adds the `manifest-sparc-ent.xml` manifest to the `solaris11_3-sparc` install service. The client criteria are defined in the `criteria-sparc-ent.xml` file.

```
# installadm create-manifest -n solaris11_3-sparc -f ./manifest-sparc-ent.xml \
-m sparc-ent -C ./criteria-sparc-ent.xml
```

The content of the `criteria-sparc-ent.xml` file is as follows:

```
<ai_criteria_manifest>
    <ai_criteria name="platform">
        <value>SUNW,SPARC-Enterprise</value>
    </ai_criteria>
</ai_criteria_manifest>
```

EXAMPLE 20 Creating a Default Manifest for an Install Service

This example uses the `-d` option to specify that the named manifest or script is the new default for this service. Any client criteria associated with the manifest are stored but are ignored while this manifest or script is the default.

```
# installadm create-manifest -n solaris11_3-sparc -f ./manifest-sparc-ent.xml -d
```

Associating Client-Specific Configuration Instructions With Install Services

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

You must have first created a system configuration profile. See Chapter 11, “Defining AI Client System Configuration Parameters” for instructions.

The syntax of the command is:

```
# installadm create-profile -n service -f filename
```

service Specifies the service that the profile is to be associated with.

filename Identifies the pathname of the profile to associate with the service.

EXAMPLE 21 Associating Client Criteria With A System Configuration Profile

The following command adds the `profile-sparc-ent.xml` profile to the `solaris11_3-sparc` install service. The `-c` option specifies that any AI clients that are using this install service and identify themselves as M4000 or M5000 servers are assigned the `profile-sparc-ent.xml` system configuration information. The `-p` option sets the name of the profile to `sparc-ent`.

```
# installadm create-profile -n solaris11_3-sparc -f ./profile-sparc-ent.xml \
-p sparc-ent -c platform="SUNW,SPARC-Enterprise"
```

Administering the AI SMF Service

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

EXAMPLE 22 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:

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

EXAMPLE 23 Disabling the AI SMF Service

To disable the AI SMF service:

```
$ svcadm disable svc:/system/install/server:default
```

Increasing Security for Automated Installations

You can secure automated installations with the Transport Layer Security (TLS) protocol. To be authenticated with TLS, you must assign the AI server and each AI client a private certificate and key pair. In addition, you must provide the Certificate Authority (CA) certificate used to generate and sign certificates. To enable security for SPARC clients, you must generate an OBP HMAC key and encryption key for each client . These keys also secure the download of the initial network boot files.

You may enable security for x86 clients as well, but note that x86 clients use PXEBoot, so the initial network boot phase is not secured. To enable security for x86 clients, you must create the x86 install service from a custom AI image that includes the CA certificates and the client certificate and key files. See Chapter 3, “Building an Image” in *Creating a Custom Oracle Solaris 11.3 Installation Image* on how to build custom AI media which includes security certificates. After creating the install service from this image, security must be set on the install service using the same security certificates that were used during the construction of that AI image.

You can secure an automated installation in the following ways:

- Server authentication: The identity of the AI server can be verified.
- Client authentication: The identity of the AI client can be verified.
- Controlling access to automated installations.
- Controlling access to server data.
- Protecting client data for all AI clients or separately for specified AI clients.
- Encrypting data so that it cannot be read over the network.
- Accessing secured IPS package repositories.

- Having the web server securely publish a user-specified directory. Client authentication is required to access this directory.

In addition to securing the AI process, you can increase security within your network by using AI to provision Kerberos in the AI clients. For instructions, see "How to Configure Kerberos Clients Using AI" on page 120.

Configuring Security Credentials

Use the `installadm` command to configure security credentials for the AI server, for a specified AI client, for AI clients of a specified install service, and for any AI client that does not already have credentials. Configure AI server authentication before client authentication because server credentials are required by the web server for TLS.

You can use the `installadm` command to accomplish the following tasks:

- **Automatically generate credentials.** If you are not using user-supplied credentials, you can simply use the `-g` option to automatically generate a private X.509 certificate and key pair, an X.509 CA certificate, and OBP keys. For more information see "Configuring AI Server Credentials" on page 113
- **Input user-supplied credentials.** If you are using user-supplied credentials, use the `-C`, `-K`, and `-A` options to specify these user-supplied credentials.

  You can specify just the CA certificate (the `-A` option) and specify the private certificate and key separately (the `-C` and `-K` options), or you can specify all three options in one command. If you specify just the `-C` and `-K` options, the associated CA certificate (the `-A` option) must have been previously specified. The `-C` and `-K` options must be specified as a pair; you cannot specify just one of them.

  The argument of the `-C` option is the path to a PEM-encoded X.509 certificate file.

  The argument of the `-K` option is the path to a PEM-encoded X.509 private key file. This key file must have any passphrase removed.

  The argument of the `-A` option is the path to a PEM-encoded X.509 Certificate Authority (CA) certificate file. CA certificates must have unique subject lines. You only need to specify each CA chain of trust one time. If the CA chain includes more than one CA certificate file, use separate `-A` options in one `installadm` command.

  OBP keys are generated if they do not already exist. If OBP keys are generated, the OBP commands to set these keys are displayed.
- **Generate OBP keys.** OBP keys are automatically generated if they do not already exist when you use the `-g`, `-C`, `-K`, or `-A` options. See "OBP Security Keys for SPARC Clients" on page 118 for information about using the `-E` and `-H` options.

- **Display credentials.** At any time you can use the `installadm list` command to display the current set of credentials for the AI server, install service or a specific AI client.

Order of Precedence for Security

When determining which security settings are in effect and which credentials are used, the order of precedence is a follows:

1. AI server has no credentials. This is the default state. There is no added security and all firmware keys for any AI client must be cleared.
2. Security is disabled server-wide. There is no added security and all firmware keys for any AI client must be cleared.
3. AI client service policy is set to `disable`. There is no added security and all firmware keys for any AI client must be cleared.
4. Use custom client credentials created with the `set-client` subcommand
5. Use install service credentials created with `set-service` subcommand
6. Use default client credentials created with the `set-server -D` subcommand
7. If there are no client credentials for the AI client and the service policy is `require-server-auth`, then the default client OBP keys are used

▼ How to Configure Security for Automated Installations

1. Become an administrator.

For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

2. Generate security credentials for the AI server.

The following command automatically generates an X.509 root CA certificate and signing CA certificate, a server certificate and private key, and OBP keys for AI server authentication. The CA certificate and the OBP keys are generated only if they do not already exist. If OBP keys are generated, the OBP commands to set these keys are displayed.

```
# installadm set-server -g
The root CA certificate has been generated.
The CA signing certificate request has been generated.
The signing CA certificate has been generated.
A new certificate key has been generated.
```

```
A new certificate has been generated.
Generating new encryption key...
To set the OBP encryption key for server authentication only, enter
    this OBP command:
  set-security-key wanboot-aes 8d210964e95f2a333c5e749790633273
Generating new hashing key (HMAC)...
To set the OBP hashing (HMAC) key for server authentication only,
    enter this OBP command:
  set-security-key wanboot-hmac-sha1 4088861239fa3f3bed22f8eb885bfa476952fab4
Configuring web server security.
Changed Server
```

For more information about configuring AI server credentials, see "Configuring AI Server Credentials" on page 113.

3. (Optional) Set the install service security policy.

The following example specifies a security setting that requires client authentication to use an install service. To protect all AI clients and all data associated with a specific install service, use the `require-client-auth` install service security setting to require all AI clients to be secured with both server and client authentication. In this example, an AI client must have X.509 credentials to access any *svcname* install service data.

```
# installadm set-service -p require-client-auth -n svcname
```

For more information about configuring install service security policies, see "Configuring Secure Install Services" on page 114.

4. Generate credentials for an AI client.

The following example automatically generates a private X.509 certificate and key pair and an X.509 CA certificate for authentication of the specified AI client, where `02:00:00:00:00:00` is the MAC address of the client. Client credentials that are assigned by specifying the MAC address are unique for each AI client. The CA certificate is generated only if it does not already exist. If the AI client is a SPARC system, OBP keys are also generated if they do not already exist, and the OBP commands to set these keys are displayed.

```
# installadm set-client -e 02:00:00:00:00:00 -g
Generating credentials for client 02:00:00:00:00:00...
A new certificate key has been generated.
A new certificate has been generated.
Generating new encryption key...
To set the OBP encryption key, enter this OBP command:
  set-security-key wanboot-aes 030fd11c98afb3e434576e886a094c1c
Generating new hashing key (HMAC)...
To set the OBP hashing (HMAC) key, enter this OBP command:
  set-security-key wanboot-hmac-sha1 e729a742ae4ba977254a2cf89c2060491e7d86eb
Changed Client: '02:00:00:00:00:00'
```

For more information about configuring client credentials, see “Configuring Client Credentials” on page 115.

5. **Set OBP keys for SPARC clients.**

   For SPARC clients that have security credentials assigned, you must set OBP security keys (hashing key and encryption key) when you boot the AI client start an AI installation. The following example sets the OBP AES encryption key on a SPARC client console.

   ```
   ok set-security-key wanboot-aes 030fd11c98afb3e434576e886a094c1c
   ```

   The following example sets the OBP hashing (HMAC) key on a SPARC client console.

   ```
   ok set-security-key wanboot-hmac-sha1 e729a742ae4ba977254a2cf89c2060491e7d86eb
   ```

   See “Installing a SPARC AI Client Using Secure Download” on page 254 for more information and examples.

6. **Modify the AI manifest to install from a secure IPS repository.**

   If an AI manifest specifies a publisher that has a secure origin, specify the key and certificates in the `credentials` sub-element of the `publisher` element. See the Software section of the `ai_manifest(4)` man page for details. You can specify an SSL key and certificate in attributes of the `image` element, but this key and certificate apply only to the first publisher specified in the manifest. If keys and certificates are specified both in an `image` element and in a `credentials` element, the credentials specified in the `credentials` element are used. Consider locating key and certificate files in a user-specified directory on the AI web server. See “Configuring the Web Server User Files Directory” on page 96 for information.

Configuring AI Server Credentials

AI server security provides the following benefits:

- AI clients can verify the identity of the AI server.
- AI clients receive data that is automatically encrypted through TLS so that it cannot be read by monitoring network traffic.

See instructions about using the `-C`, `-K` and `-A` options in “Configuring Security Credentials” on page 110.

EXAMPLE 24 Generating AI Server Credentials Using User-Supplied Credentials

The following example specifies user-supplied credentials. OBP keys are generated if they do not already exist. If OBP keys are generated, the OBP commands to set these keys are displayed.

```
# installadm set-server -C server.crt -K server.key -A cacert.pem
```

If the CA certificate is not specified, the CA certificate used to generate these client credentials must have been previously assigned.

The default client OBP keys are used when the install service of the AI client has the policy `require-server-auth` and no client or service credentials are assigned. In addition, default client credentials must be in use.

Configuring Secure Install Services

Use the `installadm create-service` command to configure an install service when creating it. For more information, see "How to Create an Install Service" on page 98. Use the `installadm set-service` command to reconfigure an existing install service. This section describes how to set a security policy for your install service.

Each install service may have one security policy set. The available choices are:

`require-client-auth`

Confirms the identity of the AI client. Requires client and server authentication for all AI clients of the specified service. This option also requires encryption.

Requires all AI clients of the service to authenticate with client authentication. All AI clients of the specified service must be assigned credentials, and all SPARC clients of this service must have their OBP keys defined. Any AI clients of the service that are not configured for client authentication will not be able to use this install service.

`require-server-auth`

Confirms the identify of the AI server. Requires all AI clients of the specified service to perform server authentication. This option also requires encryption.

Requires at least AI server authentication for access to the specified install service. Client authentication is optional, but you must provide any assigned or attributed client credentials. You must also define OBP keys for all SPARC clients of this service.

`optional`

Allows both authenticated and unauthenticated AI clients to access the install service. The option also requires encryption if the AI server has credentials. This is the default behavior.

You must provide any assigned client credentials. AI clients without assigned or attributed credentials do not use OBP keys or server authentication. Server authentication is provided only for AI clients configured for client authentication.

`encr-only`

For x86 clients only: Enables SSL/TLS end-to-end encryption without requiring authentication. Without authentication, the identities of the AI client and AI server are not guaranteed. Data in transit is not readable over the network by third parties.

`disable`

Disables all security for all AI clients of the specified service.

Clients of this service are not authenticated. No credentials are issued. Clients of this service cannot access the `webserver_secure_files_dir` directory described in "Configuring the Web Server User Files Directory" on page 96. Use this setting with caution: Any install service files that were previously protected by authentication are no longer protected. Client data is not secured from unwanted access. To re-enable authentication, specify the `set-service` subcommand again with a different security policy value.

EXAMPLE 25 Requiring AI Server Authentication During Installation

This example specifies a security setting that requires server authentication to use an install service. Use the `require-server-auth` install service security setting to require AI clients of the specified service to at least authenticate the AI server.

```
# installadm set-service -p require-server-auth -n install-service
```

EXAMPLE 26 x86: Requiring Encryption During Installation

This example specifies a security setting that uses encryption but does not require authentication. On x86 clients, to protect data transfers for a specific install service but not require client or server authentication, use the `encr-only` security setting. You still need a server certificate. The data will be protected from snooping over the network, but the AI server will provide the data to any AI client that issues the proper request to the server.

```
# installadm set-service -p encr-only -n install-service
```

Configuring Client Credentials

I client security provides the following benefits:

- The AI server can verify the identity of the AI clients.

- Data is encrypted over the network.
- For AI clients with custom credentials, any published files specific to a client are not readable by any other AI client.
- Only authenticated clients can access the user-specified secure directory described in "Configuring the Web Server User Files Directory" on page 96

You can generate or specify credentials for a particular AI client, for AI clients of a particular install service, or for any AI client that does not already have credentials. OBP keys that are generated are for bidirectional (client and server) authentication. If you assign security credentials to a SPARC client, you must provide the OBP keys when you boot the AI client start an installation. See "Installing a SPARC AI Client Using Secure Download" on page 254.

Note - You can use the `create-client` subcommand to move an AI client from one install service to another. The subcommand used on existing AI clients with security credentials will have no effect on that client's credentials.

EXAMPLE 27 Using User-Supplied Credentials for Specific AI Clients

This example specifies user-supplied credentials. If the AI client is a SPARC system, OBP keys are generated if they do not already exist. If OBP keys are generated, the OBP commands to set these keys are displayed.

```
# installadm set-client -e 02:00:00:00:00:00 -C client.crt -K client.key -A cacert.pem
```

See the comments about using the `-C`, `-K`, and `-A` options in "Configuring Security Credentials" on page 110. If the CA certificate is not specified, the CA certificate used to generate these client credentials must have been previously assigned.

See "OBP Security Keys for SPARC Clients" on page 118 for information about using the `-E` and `-H` options.

EXAMPLE 28 Credentials for AI Clients of a Specific Install Service

This example provides credentials for any AI client that is assigned to the `solaris11_3-sparc` install service and that does not already have credentials assigned.

```
# installadm set-service -g -n solaris11_3-sparc
Generating credentials for service solaris11_3-sparc...
A new certificate key has been generated.
A new certificate has been generated.
Generating new encryption key...
To set the OBP encryption key, enter this OBP command:
  set-security-key wanboot-aes 34bc980ccc8dfee478f89b5acbdf51b4
```

```
Generating new hashing key (HMAC)...
To set the OBP hashing (HMAC) key, enter this OBP command:
  set-security-key wanboot-hmac-sha1 b8a9f0b3472e8c3b29443daf7c9d448faad14fee
```

Because this install service is a SPARC install service, OBP keys are also generated, and the OBP commands to set these keys are displayed.

AI clients that are subsequently assigned to the `solaris11_3-sparc` install service also use these credentials if those clients are not assigned credentials by specifying their MAC address.

This option is useful when you want a uniform set of applications across multiple AI clients. However, all clients of this install service that are not assigned credentials by specifying their MAC addresses have identical credentials and can view each other's installation data.

See the comments about using the `-C`, `-K`, and `-A` options in “Configuring Security Credentials” on page 110.

See “OBP Security Keys for SPARC Clients” on page 118 for information about using the `-E` and `-H` options.

EXAMPLE 29 Default AI Client Credentials

This example provides a default set of credentials for any AI client with no credentials assigned.

```
# installadm set-server -D -g
Generating default client credentials...
A new certificate key has been generated.
A new certificate has been generated.
Generating new encryption key...
To set the OBP encryption key, enter this OBP command:
  set-security-key wanboot-aes 7cdbda5b8fc4b10ffbd29fa19d13af77
Generating new hashing key (HMAC)...
To set the OBP hashing (HMAC) key, enter this OBP command:
  set-security-key wanboot-hmac-sha1 14effe2c515da4940ef1db165791e92790163004
```

Because some AI clients might be SPARC clients, OBP keys are also generated, and the OBP commands to set these keys are displayed.

Once default client credentials are assigned, all AI clients will be expected to do client and server authentication, and firmware keys will be required for all SPARC clients of the AI server. Also, because multiple AI clients will have identical credentials, they will be able to view each other's installation data.

See the comments about using the `-C`, `-K`, and `-A` options in “Configuring Security Credentials” on page 110.

See "OBP Security Keys for SPARC Clients" on page 118 for information about using the -E and -H options.

OBP Security Keys for SPARC Clients

For SPARC clients to benefit from increased security features, you must set OBP security keys when you boot the AI client start an installation.

A hashing (HMAC) key and an encryption key are automatically generated and displayed if they do not already exist when you use the `installadm` command with `set-server`, `set-service`, or `set-client` subcommands to generate or specify TLS credentials. These firmware keys are not automatically regenerated when the same command is repeated.

You can use the -E and -H options to regenerate the OBP keys. Do not specifying the -E or -H option before OBP keys exist. The encryption key or HMAC that already exists is invalidated and replaced. Use the -E option to regenerate the encryption key. Use the -H option to regenerate the hashing key. You can specify both the -E and -H options, just the -E option, or just the -H option. When you run the command, the OBP keys that already exist are invalidated and replaced with the newly generated values. The OBP commands to set these keys are displayed.

To display the OBP command to set the OBP security keys at a later time, use the -v option with the `list` subcommand, as in the following example:

```
# installadm list -v -e mac-addr
```

This command shows the correct OBP keys for this AI client, whether the TLS credentials were specified using the client MAC address using the install service name, or are default client credentials. The output from the `list` subcommand shows whether the OBP keys are defined for this particular AI client, for a specified install service, or for the default AI client, as shown in Example 43, "Listing AI Client Security Information," on page 130.

Disabling and Enabling Security

This section describes the options you can use to disable security requirements without deleting the security configuration, and then re-enable security requirements using the previously configured server and client authentication settings.

Security is enabled by default. While security is disabled, no credentials are issued to AI clients and no credentials are required from AI clients. While security is disabled, no HTTPS network protection is provided for any of the AI files served to an AI client. User-specified secure files served by the AI web server (as described in "Configuring the Web Server User Files Directory" on page 96) are not accessible while security is disabled.

While security is disabled, you can continue to configure security. Any changes are effective when security is re-enabled.

Use the following command to disable security enforcement server-wide:

```
# installadm set-server -S
Refreshing web server.
Automated Installer security has been disabled.
```

Use caution when disabling security for systems that already have install services configured. The secured install service data will not require authentication to access, and non-authenticated clients will be able to install Oracle Solaris through AI.

Use the following command to re-enable security enforcement after security enforcement has been disabled with `set-security --disable`:

```
# installadm set-security -s
Configuring web server security.
Refreshing web server.
Warning: client 02:00:00:00:00:00 of service solaris11_3-i386
is required to have credentials but has none.
Automated Installer security has been enabled.
```

Deleting Credentials

Use the `installadm` command to delete security credentials. The `set-server`, `set-service`, and `set-client` subcommands can be used to delete security credentials.

Security credentials are also removed when you run the `delete-client` or `delete-service` subcommands. The `delete-client` command removes all client-specific credentials. The `delete-service` subcommand removes all service-specific credentials as well as any client-specific credentials for all AI clients of that service and any alias services.

Caution - Deleted credentials cannot be recovered, and the TLS security protocol cannot function without server credentials. AI security will be disabled prior to deleting the server credentials.

EXAMPLE 30 Deleting Credentials for One AI Client

This example deletes the private key and certificate, any CA certificate, and any OBP keys that were assigned to the AI client by using a MAC address. If OBP keys are set in

the client firmware, unset them as described in “Deleting the Hash Key and Encryption Key” on page 255.

```
# installadm set-client -e mac-addr -x
```

EXAMPLE 31 Deleting a CA Certificate

This example deletes the specified CA certificate for all AI clients that use that CA certificate. The value of the --hash option argument is the hash value of the certificate's X.509 subject, as displayed by the `list` subcommand and shown in Example 43, “Listing AI Client Security Information,” on page 130. Any AI clients that are using the specified CA certificate are counted and displayed along with a prompt to confirm you want to continue.

```
$ installadm set-client -x --hash b99588cf
  Identifier hash: b99588cf
  Subject: /C=CZ/O=Oracle Czech s.r.o./OU=install/CN=genca
  Issuer:  /C=CZ/O=Oracle Czech s.r.o./OU=install/CN=genca
  Valid from Apr 27 13:12:27 2012 GMT to Apr 27 13:12:27 2015 GMT
This CA has the following uses:
        WARNING: this is the server CA certificate
Deleting this Certificate Authority certificate can prevent
    credentials from validating.
Do you want to delete this Certificate Authority certificate [y|N]: y
Deleting all references to Certificate Authority with hash value b99588cf
```

Caution -In this example, all instances of this CA certificate are deleted for all AI clients that use it; the affected clients can no longer be authenticated. Once the specified CA certificate is used to generate certificates, the `installadm` command can no longer generate certificates.

EXAMPLE 32 Deleting AI Server Security Credentials

This example deletes the AI server's private key and certificate, any CA certificate, and the OBP keys for server authentication only:

```
# installadm set-server -x
```

How to Configure Kerberos Clients Using AI

In this procedure, the keytab file for the Kerberos client has already been created and stored on the AI server. In the examples use auto-registration to configure Kerberos clients by using pre-

existing credentials or using new principals. The auto-registration process is simpler because you do not have to create and encode keytab files for individual Kerberos clients.

1. **Become an administrator.**

   For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

2. **Create an install service, if needed.**

```
$ installadm create-service -n krb-sparc \
    -d /export/auto_install/krb-sparc \
    -s /export/auto_install/iso/sol-11_3-ai-sparc.iso
Creating service from:
/export/auto_install/iso/sol-11_3-ai-sparc.iso
Setting up the image ...
Creating sparc service: krb-sparc
Image path: /export/auto_install/krb-sparc
Refreshing install services
```

3. **Associate Kerberos clients with a service.**

   Repeat this step for all AI clients that need to be installed running Kerberos. In this example the AI client using the address of `11:11:11:11:11:11` is associated with the `krb-sparc` install service.

```
$ installadm create-client -n krb-sparc -e 11:11:11:11:11:11
Adding host entry for 11:11:11:11:11:11 to local DHCP configuration.
```

4. **Create credentials for the Kerberos clients.**

```
$ installadm set-client -c 11:11:11:11:11:11 -g
Generating credentials for client 11:11:11:11:11:11...
A new certificate key has been generated.
A new certificate has been generated.
```

5. **Create a system configuration profile that defines the contents of the Kerberos configuration file.**

   This example creates a profile by running the `kclient` command interactively. Alternatively, you could invoke the command using command-line options or using an input profile. For more information see the `kclient`(1M) man page.

   In this example, the KDC is running on an MIT server. To view sample output for a Solaris KDC, see Example 33, "Downloading Existing Keys While Deploying Kerberos Clients," on

page 123. To view sample output for an AD client, see Example 35, “Automatically Joining an Kerberos Client to a MS AD Domain,” on page 124.

```
$ kclient -x /root/krb-sc.xml
Starting client setup
-------------------------------------------------
Is this a client of a non-Solaris KDC ? [y/n]: y
Which type of KDC is the server:
       ms_ad: Microsoft Active Directory
       mit: MIT KDC server
       heimdal: Heimdal KDC server
       shishi: Shishi KDC server
Enter required KDC type: mit
Do you want to use DNS for kerberos lookups ? [y/n]: n
       No action performed.
Enter the Kerberos realm: EXAMPLE.COM
Specify the master KDCs for the above realm using a comma-separated list: kdc.example.
com
Do you have any slave KDC(s) ? [y/n]: y
Enter a comma-separated list of slave KDC host names: kdc2.example.com
Do you have multiple domains/hosts to map to a realm ? [y/n]: n
       No action performed.
Setting up /root/krb-sc.xml.
```

6. **(Optional) Convert a Kerberos client's binary keytab file into an XML profile.**

   This step is not needed if the keys can be obtained through auto-registration or if the Kerberos client is keyless. The Kerberos client needs to have a keytab file created, which is often done by the KDC administrator when a client is first configured.

   ```
   $ kclient-kt2prof -k ./host1.keytab -p /root/host1.xml
   ```

7. **Create client profiles to configure the rest of the Kerberos client.**

   Because a profile must be used in this procedure, configure as much of the Kerberos client as possible using system configuration profiles.

8. **(Optional) Set the security policy for profiles.**

   If the client profiles include a keytab, you should assign the `require-client-auth` security policy to the service so that only authenticated AI clients can download their keytab file.

   ```
   $ installadm set-service -p require-client-auth -n krb-sparc
   ```

9. **Associate the client profiles with the stall service.**

   Associate the profiles for the Kerberos configuration file, the client keytab file, and any other profiles that you have created to the install service.

```
$ installadm create-profile -n krb-sparc -f /root/krb-sc.xml
Profile krb-sc.xml added to database.
$ installadm create-profile -n krb-sparc -f /root/host1.xml -c mac="11:11:11:11:11:11"
Profile host1.xml added to database.
```

10. **Boot the Kerberos client to start the AI process.**

Example 33 Downloading Existing Keys While Deploying Kerberos Clients

Note that using auto-registration only works if the KDC is either Solaris KDC or MS AD. If the KDC is MIT, Heimdal or Shishi, only pre-generated keytab transfer is possible.

In order to use auto-registration to download existing keys, you must first have created a admin principal on the KDC with `c` and `i` administration privileges. In this example, the name of the principal is `download/admin`.

In this example, the KDC is running Oracle Solaris. Also, the keys for the Kerberos client have already been created.

This example shows how to add the `download/admin` principal when you are creating the system configuration profile for the Kerberos configuration file. The `download/admin` principal is a special admin principal that is used to transfer existing keys from the KDC server when the Kerberos client is deployed.

```
$ kclient -x /root/krb-sc.xml
Starting client setup
-------------------------------------------------
Is this a client of a non-Solaris KDC ? [y/n]: n
        No action performed.
Do you want to use DNS for kerberos lookups ? [y/n]: n
        No action performed.
Enter the Kerberos realm: EXAMPLE.COM
Specify the master KDCs for the above realm using a comma-separated
list: kdc.example.com
Do you have any slave KDC(s) ? [y/n]: y
Enter a comma-separated list of slave KDC host names: kdc2.example.com
Do you have multiple domains/hosts to map to realm  ? EXAMPLE.COM [y/n]: n
        No action performed.
Should the client automatically join the realm ? [y/n]: y
Enter the krb5 administrative principal to be used: download/admin
Password for download/admin: xxxxxxxx
Do you plan on doing Kerberized nfs ? [y/n]: n
        No action performed.
Is this client a member of a cluster that uses a logical host name ? [y/n]: n
        No action performed.
```

```
Do you have multiple DNS domains spanning the Kerberos realm EXAMPLE.COM ? [y/n]: n
        No action performed.
Setting up /root/krb-sc.xml.
```

Example 34 Creating New Keys While Deploying Kerberos Clients

Note that using auto-registration only works if the KDC is either Solaris KDC or MS AD. If the KDC is MIT, Heimdal or Shishi, only pre-generated keytab transfer is possible.

In order to use auto-registration to download new keys, you must first have created an admin principal on the KDC with `a`, `c` and `i` administration privileges. In this example, the name of the principal is `create/admin`.

In this example, the KDC is running Oracle Solaris. This example adds the `create/admin` principal when you are creating the system configuration profile for the Kerberos configuration file. The `create/admin` principal is a special admin principal that is used to transfer new keys from the KDC server when the Kerberos client is deployed. This command includes more options so fewer questions are asked.

```
$ kclient -x /root/krb-sc.xml -R EXAMPLE.COM -a create/admin -d none -m kdc.example.com
Starting client setup
-------------------------------------------------
Do you have multiple domains/hosts to map to realm  ? EXAMPLE.COM [y/n]: n
        No action performed.
Should the client automatically join the realm ? [y/n]: y
Password for create/admin: xxxxxxxx
Setting up /root/krb-sc.xml.
```

Example 35 Automatically Joining an Kerberos Client to a MS AD Domain

In this example, the Kerberos client is joining an AD domain. Use the following command to add the `Administrator` principal when you are creating the system configuration profile for the Kerberos configuration file.

```
$ kclient -x /root/krb-sc.xml
Starting client setup
-------------------------------------------------
Is this a client of a non-Solaris KDC ? [y/n]: y
Which type of KDC is the server:
        ms_ad: Microsoft Active Directory
        mit: MIT KDC server
        heimdal: Heimdal KDC server
        shishi: Shishi KDC server
Enter required KDC type: ms_ad
Should the client automatically join AD domain ? [y/n]: y
Enter the Kerberos realm: EXAMPLE.COM
```

```
Enter the krb5 administrative principal to be used: Administrator
Password for Administrator: xxxxxxxx
Setting up /root/krb-sc.xml.
```

Showing Information About Install Services

Use the `installadm list` command to show information about install services, as well as the AI clients, AI manifests and system configuration profiles that are associated with the services. This section includes:

- Example 36, “Listing All Install Services on the AI Server,” on page 125 shows how to list all of the install services on an AI server
- Example 37, “Showing Information for a Specified Install Service,” on page 126 shows how to list information about a specific install service
- Example 38, “Listing AI Clients Associated With Install Services,” on page 126 shows how to list the AI clients associated with install services
- Example 39, “Listing AI Clients Associated With a Specific Install Service,” on page 126 shows how to list the AI clients associated with a specific install service
- Example 40, “Listing All AI Manifests and System Configuration Profiles,” on page 127 shows how to list all AI manifests and system configuration profiles
- Example 41, “Listing Manifests and Profiles Associated With a Specified Install Service,” on page 128 shows how to list AI manifests and system configuration profiles associated with a specific install service
- Example 43, “Listing AI Client Security Information,” on page 130 shows how to list AI client security information
- Example 42, “Listing AI Server Security Information,” on page 128 shows how to list AI server security information

EXAMPLE 36 Listing All Install Services on the AI Server

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

```
$ /usr/sbin/installadm list

Service Name      Status Arch  Type Alias Aliases Clients Profiles Manifests
------------      ------ ----  ---- ----- ------- ------- -------- ---------
default-i386      on     i386  iso  yes   0       0       0        1
default-sparc     on     sparc iso  yes   0       0       0        1
solaris11_3-i386  on     i386  iso  no    1       0       1        1
solaris11_3-sparc on     sparc iso  no    1       0       2        1
```

The `default-i386` service was created automatically when the first `i386` service was created on this AI server. The `default-i386` service is used by any x86 client that has not been associated with the `solaris11_3-i386` service by using the `create-client` subcommand. The `default-i386` and `solaris11_3-i386` services share an installation 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 AI server. The `default-sparc` service is used by any SPARC client that has not been associated with the `solaris11_3-sparc` service by using the `create-client` subcommand. The `default-sparc` and `solaris11_3-sparc` services share an installation image but they have different AI manifests and system configuration profiles.

EXAMPLE 37 Showing Information for a Specified Install Service

This example displays information about the install service specified by the `-n` option.

```
$ /usr/sbin/installadm list -n solaris11_3-sparc
Service Name                    Status Arch  Type Alias Aliases Clients Profiles
 Manifests
------------                    ------ ----  ---- ----- ------- ------- --------
 ---------
solaris11_3-sparc               on     sparc iso  no    1       0       2        1
```

EXAMPLE 38 Listing AI Clients Associated With Install Services

This example lists all the AI clients that are associated with the install services on this AI server. The AI clients were associated with the install services by using the `installadm create-client` command. See “Associating an AI Client With an Install service” on page 105.

```
$ /usr/sbin/installadm list -c

Service Name      Client Address    Arch  Secure Custom Args Custom Grub
------------      --------------    ----  ------ ----------- -----------
solaris11_3-sparc 00:14:4F:A7:65:70 sparc no     no          no
solaris11_3-i386  08:00:27:8B:BD:71 i386  no     no          no
                  01:C2:52:E6:4B:E0 i386  no     no          no
```

EXAMPLE 39 Listing AI Clients Associated With a Specific Install Service

This example lists all the AI clients that have been added to the specified install service. In the following example, one AI client is associated with the `solaris11_3-sparc` install service.

```
$ /usr/sbin/installadm list -c -n solaris11_3-sparc
```

```
Service Name      Client Address    Arch  Secure Custom Args Custom Grub
------------      --------------    ----  ------ ----------- -----------
solaris11_3-sparc 00:14:4f:a7:65:70 sparc no     no          no
```

EXAMPLE 40 Listing All AI Manifests and System Configuration Profiles

This example lists all AI manifests, derived manifest scripts, and system configuration profiles for all install services on this AI server. The Service and Manifest Name and 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 designated as 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 AI clients to use the `mem1` manifest. The original default manifest is inactive because it has no associated criteria to determine which AI clients should use it. Only the default manifest can have no associated criteria. An AI client that does not match the criteria any other manifest associated with the service uses the default manifest, in this case, `mem1`. See Chapter 9, “Assigning Customizations to AI Clients” for more information about selecting an AI manifest.

```
$ installadm list -m -p
Service Name                    Manifest Name      Type      Status    Criteria
------------                    -------------      ----      ------    --------
default-i386                    orig_default       derived   default   one
default-sparc                   orig_default       derived   default   none
solaris1132-i386                ipv4               xml       active    ipv4 =
 198.51.100.1 - 198.51.100.200
                                mem1               derived   default   (Ignored: mem =
 2048 MB - 4095 MB)
                                orig_default       derived   inactive  none
solaris11_3-sparc               sparc-ent          xml       active    mem = 4096 MB -
 unbounded
                                                                       platform =
 SUNWSPARC-Enterprise
                                mem1               derived   default   (Ignored: mem =
 2048 MB - 4095 MB)
                                orig_default       derived   inactive  none

Service Name                    Profile Name  Environment Criteria
------------                    ------------  ----------- --------
solaris11_3-i386                mac2          system      mac = 08:00:27:8B:BD:71
```

```
                                                hostname = server2
                              mac3          system      mac = 01:C2:52:E6:4B:E0
                                                        hostname = server3
                              ipv4          system      ipv4 =  203.0.113.100 -
 203.0.113.199
                              mem1          system      mem = 2048 MB - 4095 MB
solaris11_3-sparc             mac1          system      mac = 01:C2:52:E6:4B:E0
                                                        hostname = server1
                                                        ipv4 = 192.0.2.251
                              sparc-ent     system      platform = SUNWSPARC-
Enterprise
                                                        mem = 4096-unbounded
```

If you run this command with the rights profile, an additional column in the list of manifests identifies the type of the manifest, either `xml` or `derived`.

EXAMPLE 41 Listing Manifests and Profiles Associated With a Specified Install Service

This example shows all AI manifests, derived manifest scripts, and system configuration profiles associated with the install service `solaris11_3-sparc`.

```
$ installadm list -m -p -n solaris11_3-sparc
Service Name                  Manifest Name Type    Status   Criteria
------------                  ------------- ------- ------   --------
solaris11_3-sparc             sparc-ent     xml     active   mem = 4096 MB -
 unbounded
                                                             platform = SUNWSPARC-
Enterprise
                              mem1          derived default  Ignored:
                                                             mem = 2048 MB - 4095 MB)
                              orig_default  derived inactive  none

Service Name                  Profile Name  Environment Criteria
------------                  ------------  --------
solaris11_3-sparc             mac1          system      mac = 01:C2:52:E6:4B:E0
                                                        hostname = server1
                                                        ipv4 = 192.0.2.251
                              sparc-ent     system      platform = SUNWSPARC-
Enterprise
                                                        mem = 4096-unbounded
```

EXAMPLE 42 Listing AI Server Security Information

The `list` subcommand with the `-v` and `-s` options shows information about the AI server including:

- Current state of security: enabled or disabled

- Server certificate X.509 subject and issuer strings
- Dates the AI server certificate is valid
- Results of validating the AI server certificate
- Server CA certificate hash value, X.509 subject, and issuer
- Client CA certificates for client authentication
- The default client certificate

```
# installadm list -v -s
AI Server Parameter  Value
-------------------  -----
Hostname ........... install-svr
Architecture ....... i386
Active Networks .... 203.0.113.0
Http Port .............. 5555
Secure Port ............ 5556
Image Path Base Dir .... /export/auto_install
Multi-Homed? ........... no
Managing DHCP? ......... yes
DHCP IP Range .......... 192.0.2.2 - 192.0.2.19
Boot Server ............ 192.0.2.25
Web UI Enabled? ........ yes
Wizard Saves to Server?  no
Security Enabled? ...... yes
Security Key? .......... yes
Security Cert:
               Subject: /C=US/O=Oracle/OU=Solaris Deployment/CN=osol-inst
               Issuer : /C=US/O=Oracle/OU=Solaris Deployment/CN=Signing CA
               Source : Server Certificate
               Valid from:  Jan 24 22:53:00 2015 GMT
                       to:  Jan 24 22:53:00 2025 GMT
               Validates?: yes
CA Certificates:
      d09051e4 Subject: /C=US/O=Oracle/OU=Solaris Deployment/CN=Root CA
               Issuer : /C=US/O=Oracle/OU=Solaris Deployment/CN=Root CA
               Source : Server CA Certificate
               Valid from:  Jan 24 22:53:00 2015 GMT
                       to:  Jan 24 22:53:00 2025 GMT
      f9d73b41 Subject: /C=US/O=Oracle/OU=Solaris Deployment/CN=Signing CA
               Issuer : /C=US/O=Oracle/OU=Solaris Deployment/CN=Root CA
               Source : Server CA Certificate
               Valid from:  Jan 24 22:53:00 2015 GMT
                       to:  Jan 24 22:53:00 2025 GMT
Def Client Sec Key? .... yes
Def Client Sec Cert:
               Subject: /C=US/O=Oracle/OU=Solaris Deployment/CN=Client default
               Issuer : /C=US/O=Oracle/OU=Solaris Deployment/CN=Signing CA
               Source : Default Client Certificate
```

```
              Valid from: Jul 15 19:33:00 2013 GMT
                      to: Jul 13 19:33:00 2023 GMT
Def Client CA Certs .... none
Def Client FW Encr Key . adcc858c58ecae04c02282e7245c235c
Def Client FW HMAC Key . cb7bc6213512c8fa3dc7d7283a9e056dc2791f98
Number of Services ..... 102
Number of Clients ...... 37
Number of Manifests .... 108
Number of Profiles ..... 92
```

EXAMPLE 43 Listing AI Client Security Information

The `list` subcommand with the -v and -e options show the following AI client security information:

- The credentials that are used for the AI client
- The source of the client's credentials
- The validity of the client's certificate

```
# installadm list -v -e 00:14:4F:83:3F:4A
Service Name         Client Address    Arch  Secure Custom Args Custom Grub
------------         --------------    ----  ------ ----------- -----------
solaris11_3-sparc    00:14:4F:A7:65:70 sparc yes    no          no

   Client Credentials? yes
   Security Key? ..... yes
   Security Cert:
                Subject: /C=US/O=Oracle/OU=Solaris Deployment/CN=CID 01020000000000
                Issuer : /C=US/O=Oracle/OU=Solaris Deployment/CN=Signing CA
                Valid from: Jan 24 10:20:00 2015 GMT
                        to: Jan 24 10:20:00 2025 GMT

   CA Certificates:
         d09051e4 Subject: /C=US/O=Oracle/OU=Solaris Deployment/CN=Root CA
                  Issuer : /C=US/O=Oracle/OU=Solaris Deployment/CN=Root CA
                  Source : Default CA Certificate
                  Valid from: Jan 24 22:53:00 2015 GMT
                          to: Jan 24 22:53:00 2025 GMT

   FW Encr Key (AES) . 23780bc444636f124ba3ff61bdac32d1
   FW HMAC Key (SHA1) 1093562559ec45a5bb5235b27c1d0545ff259d63
   Boot Args ......... none
```

The `export` subcommand shows the TLS credentials attributed to an AI client. Adding `-C` displays the x.509 TLS certificate.

```
# installadm export -e 00:00:5E:00:53-00 -C
```

```
------ certificate: client_00:00:5E:00:53-00_cert_de22916b ------
-----BEGIN CERTIFICATE-----
MIICFDCCAX+gAwIBAgIBGTALBgkqhkiG9w0BAQswUDELMAkGA1UEBhMCVVMxDzAN
 ....
UiZDA6GOdvE=
-----END CERTIFICATE-----
```

The -K option shows the X.509 private key:

```
# installadm export -e 00:00:5E:00:53-00 -K
--------------- key: client_00:00:5E:00:53-00_key ---------------
-----BEGIN RSA PRIVATE KEY-----
MIICXQIBAAKBgQDCCJbC5Bd0uMQ0AOk4lLlQqWiQwqkx9lpIhHl31tF1/WxHi74A
 ...
SYoBeKAOPSo7Evund+bHAROl0H4QnbSJgl1UDuZr3T3h
-----END RSA PRIVATE KEY-----
```

Managing Install Services

Use the `installadm set-service` command to reconfigure an existing install service. The examples in this section show how to set install service aliases, the default AI manifest, or the image path for an install service, as well as how to update an install service.

Setting Install Service Aliases

You can use install service aliases to minimize the amount of reconfiguration that needs to be done when a new service is created. For instance, the `default-`*arch* install services are aliases. When creating a service, you can create an alias by using the `-t` option of the `create-service` subcommand. The `-t` option to the `set-service` subcommand changes the specified service to be an alias of the another service. When you use the `set-service` subcommand, the specified service must already be an alias.

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

Manifests and profiles that were added to the service prior to setting the alias are revalidated when the alias is reset because the AI DTDs and SMF DTDs associated with the new installation image could be different. This validation is the same validation that is performed by the `create-manifest` and `create-profile` commands.

EXAMPLE 44 Creating an Install Service Alias

This example creates the new `install-sparc` service as an alias to the existing `solaris11_3-sparc` install service.

```
# installadm create-service -t solaris11_3-sparc -n install-sparc
```

EXAMPLE 45 Modifying an Install Service Alias

In this example, both the `solaris11_3-i386` install service and the `install-i386` install service alias must have been created previously. The following example sets the `install-i386` install service as an alias to the `solaris11_3-i386` install service.

```
# installadm set-service -t solaris11_3-i386 -n install-i386
```

Setting the Default AI Manifest for an Install Service

These examples show how to designate a particular manifest or a derived manifest script as the default manifest for both a new install service and an existing install service.

EXAMPLE 46 Setting a Default AI Manifest When Creating an Install Service

This example sets the `mem1` manifest as the default manifest for the new `install-sparc` service. All AI clients associated with this service that do not match other client criteria will use this manifest by default.

```
# installadm create-service -M /tmp/mem1 -n install-sparc
```

EXAMPLE 47 Setting a Default AI Manifest by Modifying an Existing Install Service

This example sets the already registered `mem1` manifest as the default manifest for the existing `install-i386` service. All AI clients associated with this service that do not match other client criteria will use this manifest by default.

```
# installadm set-service -M mem1 -n install-i386
```

Setting the Image Path for an Install Service

These examples show how to define or reset the path to an installation image for a given service.

EXAMPLE 48 Setting the Image Path for an New Installation Image

This example defines the path to the installation image for the `solaris11_3-i386` service while creating the service and the net image.

```
# installadm set-service -d /export/ai-images/solaris11_3.i386 -n solaris11_3-i386
```

EXAMPLE 49 Setting the Image Path for an Existing Installation Image

This example relocates the path to the installation image for the `solaris11_3-i386` service.

```
# installadm set-service -d /export/ai-images/solaris11_3.i386 -n solaris11_3-i386
```

Updating an Existing Install Service

Use the `update-service` subcommand to update the image associated with 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 the alias is changed to use the new service.

To use a different repository when updating a service, add the `-p` option to the `update-service` subcommand. 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 *svc-name* is an alias.

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

EXAMPLE 50 Updating an Install Service

This example creates a new service, and changes the `default-i386` alias to use this new service.

```
# installadm update-service -n default-i386
```

EXAMPLE 51 Using a Different Repository When Updating an Install Service

This example shows how to identify the publisher associated with the `solaris11_3-i386` service. First determine the image path for the service using the `installadm list` subcommand. Then you can use the image path to determine the publisher that is being used.

```
$ installadm list -v -n solaris11_3-i386
Service Name      Status Arch  Type Alias Aliases Clients Profiles Manifests
------------      ------ ----  ---- ----- ------- ------- -------- ---------
solaris11_3-i386  on     i386  iso  no    1       0       1        1

   Image Path ........ /export/auto_install/solaris11_3-i386
    ....

$ pkg -R /export/auto_install/solaris11_3-i386 publisher
PUBLISHER        TYPE     STATUS   URI
solaris          origin   online   http://pkg.oracle.com/solaris/release/
```

This example specifies using the publisher at `example.com/solaris/mybuild` when updating an install service.

```
# installadm update-service -n default-i386 -p solaris=http://example.com/solaris/
mybuild
```

EXAMPLE 52 Using a Different Net Image Package When Updating an Install Service

This example specifies a specific net image package.

```
# installadm update-service -n default-i386 -s FMRI
```

Managing AI Manifests

This section shows how to update, delete, validate, or export an AI manifest.

Updating an AI Manifest

Use the `installadm update-manifest` command to replace the contents of the specified AI manifest or derived manifest script file with the contents of the manifest or script file for the specified install service. The criteria, default status, and manifest name are not changed as a result of the update.

The `update-manifest` subcommand validates XML manifest files before adding them to the install service.

The manifest must already exist in the specified service. Use the `installadm list` command to confirm, as shown in Example 40, “Listing All AI Manifests and System Configuration Profiles,” on page 127.

If no manifest is 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 specified manifest, if this attribute is specified and if the value of this attribute matches the name of an existing manifest for this install service.
- The base name of the specified file name if this name matches the name of an existing manifest for this install service.

This example updates the content of the `sparc-ent` manifest in the `solaris11_3-sparc` service with the content of `./mymanifests/manifest-new-sparc-ent.xml`. The name of the manifest in `installadm list` is still `sparc-ent`.

```
# installadm update-manifest -n solaris11_3-sparc \
-f ./mymanifests/manifest-new-sparc-ent.xml -m sparc-ent
```

Validating an AI Manifest

Use the `installadm validate` command to validate AI manifests for syntactic correctness.

Use the `-M` option to validate manifests that have not been added to the install service. The value of the `-M` argument is the pathname to the manifest.

Use the `-m` option to validate manifests that have already been added to the specified install service. Use the `installadm list` command, as shown in Example 40, “Listing All AI Manifests and System Configuration Profiles,” on page 127, to display possible values for the manifest name. The `create-manifest` subcommand validates AI manifests before adding them to the install service. The `validate -m` subcommand verifies that the manifest has not become corrupted since it was added.

You must specify a service name for manifests that have been added to an install service and manifests that have not been added yet. The service name is required for manifests that have not yet been added to an install service because the 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 AI server is running. The manifest must be validated against the DTD that will be in use on the AI client being installed. For more information see the `ai_manifest(4)` man page.

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

Deleting an AI Manifest

Use the `installadm delete-manifest` command to remove the specified AI manifest or derived manifest script from the specified install service. The manifest name is the same name that the `installadm list` command returns, as shown in Example 40, “Listing All AI Manifests and System Configuration Profiles,” on page 127.

You cannot delete the default AI manifest.

The following command removes the `sparc-ent` AI manifest from the `solaris11_3-sparc` install service:

```
# installadm delete-manifest -m sparc-ent -n solaris11_3-sparc
```

Managing System Configuration Profiles

This section provides instructions for updating, deleting, validating and exporting a system configuration profile.

Updating a System Configuration Profile

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

The name of the profile and the install service that includes the service may be specified. If the profile is not specified with the install service, the name of the profile to be updated is the base name of the file.

The following command updates the content of the `sparc-ent` profile in the `solaris11_3-sparc` service with the content of `./myprofiles/profile-new-sparc-ent.xml`.

```
# installadm update-profile -n solaris11_3-sparc \
```

```
-f ./myprofiles/profile-new-sparc-ent.xml -p sparc-ent
```

Validating a System Configuration Profile

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

Use the `-P` option to validate profiles that have not been added to the install service. The value of the `-P` argument is the is the pathname to the profile.

Use the `-p` option to validate profiles that have already been added to the specified install service. Use the `installadm list` command, as shown in Example 40, “Listing All AI Manifests and System Configuration Profiles,” on page 127, to display possible values for the profile name. 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.

You must specify a service name for profiles that have been added to an install service and profiles that have not been added yet. The service name is required for profiles that have not yet been added to an install service because the 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 AI server is running. The profile must be validated against the DTD that will be in use on the AI client being installed. For more information see the `service_bundle(4)` man page.

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 Example 40, “Listing All AI Manifests and System Configuration Profiles,” on page 127.

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

The following command removes the `sparc-ent` system configuration profile from the `solaris11_3-sparc` install service.

```
# installadm delete-profile -p sparc-ent -n solaris11_3-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 specified install service to the named file or directory.

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 specified manifest can be the name of an XML AI manifest or a derived manifest script. See Chapter 10, “Defining AI Client Installation Parameters” for information about creating manifests and derived manifest 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.

♦ ♦ ♦ CHAPTER 9

Assigning Customizations to AI Clients

To customize an installation, first customize the installation instructions in an AI manifest and the system configuration instructions in a system configuration profile. Then specify client criteria to match the customized installation and configuration instructions with AI clients identified by that criteria.

An AI install service includes one or more AI manifests with installation instructions and zero or more system configuration profiles with configuration instructions. Each AI client uses one and only one AI manifest. Each AI client can use any number of system configuration profiles. If an AI client does not use any profiles, then an interactive tool opens on that client at first boot after the installation to complete the configuration of that AI client.

Matching AI Clients With Installation and Configuration Instructions

When you use AI, you first set up an AI server. When an AI client boots over the network, it uses an install service from the AI server.

The AI 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 AI 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 an AI client to a specific install service, add that AI client to the install service (see Chapter 14, “Installing AI Clients Using an AI Server”). Specify the MAC address of the AI client and the name of the install service for this client to use. When the AI client with this MAC address boots, the client is directed to the AI server and uses the specified install service. To find the MAC address of a system, use the `dladm` command as described in the `dladm(1M)` man page.

To define more than one type of installation for one install service, create additional AI manifests and system configuration profiles. Add the new AI manifests and profiles to the AI install service. Specify criteria that define which AI clients should use which AI manifest and which system configuration profiles. See "Associating Client-Specific Installation Instructions With an Install Service" on page 106.

For information about how to create custom AI manifests, see Chapter 10, "Defining AI Client Installation Parameters". For information about how to create system configuration profiles, see Chapter 11, "Defining AI Client System Configuration Parameters".

Selecting the AI Manifest

Each AI client uses one and only one AI manifest to complete its installation. The AI manifest is selected for an AI 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 185 for an example of a default AI manifest.
- If custom AI manifests are defined for this install service but the AI client does not match criteria for any custom AI manifest, then the AI client uses the default AI manifest.
- If the AI client matches criteria that have been specified for a custom AI manifest, the AI 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 6, "Criteria Keywords and Criteria Hierarchy," on page 142 to select the manifest for the installation. The `installadm` tool verifies that criteria of the same type do not overlap. For more information, see "Associating Client-Specific Installation Instructions With an Install Service" on page 106.

  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 53 Matching AI 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 `sparc-ent.xml` AI manifest was added to the service with the following criteria file that specifies AI client platform:

```
<ai_criteria_manifest>
    <ai_criteria name="platform">
```

```
        <value>SUNW,SPARC-Enterprise</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:

```
<ai_criteria_manifest>
    <ai_criteria name="mac">
        <value>00:14:4f:a7:65:70</value>
    </ai_criteria>
</ai_criteria_manifest>
```

If an AI client with MAC address `00:14:4f:a7:65:70` is being installed, it is assigned `manifest_mac1.xml`.

If the AI client is a M4000 or M5000, it is assigned `sparc-ent.xml`.

If the AI client does not match the criteria for either AI manifest, then the default manifest for the install service is assigned to the client.

Selecting System Configuration Profiles

The same criteria keywords are used for selecting system configuration profiles for an AI client as are used for selecting an AI manifest. See Table 6, “Criteria Keywords and Criteria Hierarchy,” on page 142.

More than one system configuration profile can be selected for any particular AI client. No algorithm is needed to narrow the selection to one profile.

If client characteristics match criteria for multiple system configuration profiles, all matching 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 profiles are used to configure that AI client.

Selection Criteria

Table 6, “Criteria Keywords and Criteria Hierarchy,” on page 142 shows the criteria keywords that can be used to indicate which AI 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 6 Criteria Keywords and Criteria Hierarchy

Criteria Keyword	Description	Command Line and XML File Examples

`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>`

`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:

Criteria Keyword	Description	Command Line and XML File Examples

		`<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>`

`ipv4`	IP version 4 network address, or range of IP addresses	**CLI, single IP address:**
		
		`-c ipv4="203.0.113.127"`
		
		CLI, range of IP addresses:
		
		`-c ipv4="203.0.113.1-203.0.113.200"`
		
		XML, single IP address:
		
		`<ai_criteria name="ipv4">`
		`  <value>203.0.113.127</value>`
		`</ai_criteria>`
		
		XML, range of IP addresses:
		
		`<ai_criteria name="ipv4">`
		`  <range>`
		`  203.0.113.1`
		`  203.0.113.200`
		`  </range>`
		`</ai_criteria>`

`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>`

`mem`	Memory size in megabytes returned by `prtconf`, or a range of memory size	**CLI, one memory size:** `-c mem="4096"`

Criteria Keyword	Description	Command Line and XML File Examples

	The `unbounded` keyword indicates no upper limit in a range.	**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>`

`network`	IP version 4 network number, or a range of network numbers	**CLI, single IP address:**
		`-c network="203.0.113.0"`
		CLI, range of IP addresses:
		`-c network="203.0.113.0-203.0.113.200"`
		XML, single IP address:
		`<ai_criteria name="network">`
		`<value>203.0.113.0</value>`
		`</ai_criteria>`
		XML, range of IP addresses:
		`<ai_criteria name="network">`
		`<range>`
		`203.0.113.0`
		`203.0.113.200`
		`</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` for M4000 and M5000 servers
	`ORCL, SPARC-T4-2` for T4 servers
		CLI:
		`-c platform="SUNW,SPARC-Enterprise"`
		XML:
		`<ai_criteria name="platform">`
		`<value>SUNW,SPARC-Enterprise</value>`
		`</ai_criteria>`

`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:**

Criteria Keyword	Description	Command Line and XML File Examples

		`-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>`

♦ ♦ ♦ CHAPTER 10

Defining AI Client Installation Parameters

When you create an install service, you get a default AI manifest that specifies how to provision the AI clients. The default AI manifest is a derived manifest 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 for information about the XML elements in an AI manifest.

This chapter explains how you can create custom AI manifests for particular AI 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. See "How to Customize an XML AI Manifest File" on page 148.
- Write a script that dynamically creates an AI manifest for each AI client at installation time. Use this method to create a custom installation for each AI client, based on characteristics discovered at installation time. See "How to Create and Apply a Derived Manifest Script" on page 151.
- Use the AI manifest wizard to create a manifest without having to edit XML files. See "How to Create an AI Manifest Using the AI Manifest Wizard" on page 170.
- When running the Oracle Solaris 11.3 release, use the `installadm` command to create a new manifest or edit an existing manifest without having to edit XML files. Use this method to make changes from the command line. See "Creating an AI Manifest Using the `installadm` Command" on page 171.

Any particular install service can include both XML manifest files and scripts for generating manifest files. Any particular AI client uses only one AI manifest, either static or generated by a script. Which AI manifest a particular AI client uses depends on the criteria specified when the manifest is added to the install service. If the AI 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. **Become an administrator.**
   For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

2. **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 AI 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_3-i386
      Service Name           Manifest Name Type    Status  Criteria
      ------------           ------------- ------- ------  --------
      solaris11_3-i386       orig_default  derived 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.

      ```
      # installadm export -n solaris11_3-i386 -m orig_default -o mem1
      ```

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

3. **Modify the manifest copy.**
   Modify `mem1`, adding tags and values according to the information in the `ai_manifest(4)` man page.

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

   ```
   # installadm create-manifest -n solaris11_3-i386 -f ./mem1 -m mem1 \
   ```

```
-c mem="2048-unbounded"
```

You can specify multiple `-c` options. Alternately use `-C` to use a file that includes many client criteria. See Chapter 9, “Assigning Customizations to AI Clients” and the `set-criteria` subcommand for information about specifying client criteria.

After this command has been run the `list` subcommand shows:

```
# installadm list -m -n solaris11_3-i386
Service Name                    Manifest Name      Type    Status    Criteria
------------                    -------------      ----    ------    --------
solaris11_3-i386                mem1               derived active    mem  = 2048 MB -
 unbounded
                                orig_default       derived default   none
```

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

  You can designate any manifest file or derived manifest 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 `-M` option with the `set-service` subcommand.

  ```
  # installadm set-service -M mem1 -n solaris11_3-i386
  # installadm list -m -n solaris11_3-i386
  Service Name                    Manifest Name      Type    Status
   Criteria
  ------------                    -------------      ----    ------
   --------
  solaris11_3-i386                mem1               derived default / active   mem
   = 2048 MB - unbounded
                                  orig_default       derived inactive           none
  ```

  In this example, the original default is now inactive because it has no criteria to specify which AI 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.

  ```
  # installadm create-manifest -n solaris11_3-i386 -d \
  -f ./region1.xml -m region1
  # installadm list -m -n solaris11_3-i386
  Service Name                    Manifest Name      Type    Status   Criteria
  ------------                    -------------      ----    ------   --------
  solaris11_3-i386                mem1               derived active   mem  = 2048 MB
   - unbounded
  ```

```
region1            xml     default  none
orig_default       derived inactive 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.

  ```
  # installadm update-manifest -n solaris11_3-i386
      -f ./newregion1.xml -m region1
  ```

5. **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 installation time.

Note - If an invalid manifest is provided to an AI client, the automated installation aborts. To investigate the cause of the validation failure, see the `/system/volatile/install_log` on the AI client.

See also "Working With Install Services" on page 97 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 installation is to write a script that dynamically creates an AI manifest for each AI client at installation time. The script can query environment variables and other client configuration information to create a custom AI manifest for each AI client. Because the manifest is based on attributes of each AI 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 manifest script that adds the parameters that are different for each AI client to the common manifest when each stem is installed. For example, a derived manifest script can detect the number and size of disks attached to each system and modify the AI manifest at installation time to specify a custom disk layout for each AI client.

▼ How to Create and Apply a Derived Manifest 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 AI client that uses this derived manifest 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 AI client being installed.

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

   Add the derived manifest script to the appropriate AI install service, specifying criteria that define which AI 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 installation time to produce an instance of an AI manifest. AI syntactically validates the resulting manifest.

   Note - If a manifest is not created or the derived manifest does not validate, the installation aborts. To investigate the cause of the validation failure, see the `/system/volatile/install_log` on the AI client.

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

Creating a Derived Manifest Script

In general, a derived manifest 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 AI client. A derived manifest script can also combine multiple partial AI manifests. The final derived manifest must be complete and must pass validation.

A derived manifest 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 manifest 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 *Securing Users and Processes in Oracle Solaris 11.3*.

In addition to using commands to read system attributes, attributes of the AI client are available through the environment variables shown in the following table.

TABLE 7 AI Client Attribute Environment Variables

Environment Variable Name	Description
`SI_ARCH`	The architecture of the AI client to be installed. Equivalent to the output of `uname -p`.
`SI_CONFIG_PROFILE_DIR`	The directory where user supplied system configuration profiles may be stored and used by the install service.
`SI_CPU`	The ISA or processor type of the AI client to be installed. Equivalent to the output of `uname -p`.
`SI_DISKNAME_#`	A flat set of variables representing the names of the disks found on the AI 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 AI 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 AI client as set in the install environment.
`SI_HOSTNAME`	The host name of the AI client as set in the install environment.
`SI_STALL_PROFILE_DIR`	The directory where user supplied system configuration profiles for the install environment may be stored and used by the install service.
`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_KARCH`	The kernel architecture of the AI client. Equivalent to the output of `uname -m`.
`SI_MEMSIZE`	The amount of physical memory on the AI client. The size is an integer number of megabytes.

Environment Variable Name	Description
SI_NATISA	The native instruction set architecture of the AI client. Equivalent to the output of isainfo -n.
SI_NETWORK	The network number of the AI client. The network number is (IP_ADDR & *netmask*).
SI_NUMDISKS	The number of disks on the AI client.
SI_PLATFORM (or SI_MODEL)	The platform of the AI client. Equivalent to the output of uname -i for x86 systems and prtconf -b for SPARC systems.
SI_SYSPKG	The release of the Oracle Solaris incorporation package on the AI client (currently named entire). If the AI client's entire package is pkg://solaris/entire@0.5.11,5.11-0.175.0.0.0.2.0:20111020T143822Z, the value of SI_SYSPKG would be pkg:/entire@0.5.11-0.175.0. For an update or SRU release, if the AI client's entire pkg is pkg://solaris/entire@0.5.11,5.11-0.175.1.19.0.6.0:20140508T221351Z, the value of SI_SYSPKG would be pkg:/entire@0.5.11-0.175.1.

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 manifest script will be added:

```
<!DOCTYPE auto_install SYSTEM "file:///imagepath/auto_install/ai.dtd.1">
<auto_install/>
```

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 manifest script will be added:

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

Note - Change the *imagepath* back to ///usr/share before using the script to install an AI client.

Use the load subcommand of the aimanifest command to load a base manifest before any other aimanifest call in the derived manifest script. Any files that you load must be accessible by the AI client at 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 pathname:

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

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 manifest 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 Manifest Scripts

This section shows how to write derived manifest 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`, `delete`, 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_LOGFILE` 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 AI 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 7, “AI Client Attribute Environment Variables,” on page 152 have values and are available for a derived manifest script to use.

EXAMPLE 54 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 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 AI 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>
```

For AI 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 55 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 manifest scripts during AI installations.

This example demonstrates using the `aimanifest` return path (`-r` option). See the `aimanifest(1M)` man page for more information about the return path.

```
#!/bin/ksh93

SCRIPT_SUCCESS=0
SCRIPT_FAILURE=1

function handler
{
    exit $SCRIPT_FAILURE
}

trap handler ERR

/usr/bin/aimanifest load /usr/share/auto_install/manifest/default.xml

# Use the default if there is only one disk.
if [[ $SI_NUMDISKS -ge 2 ]] ; then
    typeset -i disk_num

    # Turn on mirroring. Assumes a root zpool is already set up.
    vdev=$(/usr/bin/aimanifest add -r \
        target/logical/zpool[@name=rpool]/vdev@name mirror_vdev)
    /usr/bin/aimanifest set ${vdev}@redundancy mirror

    for ((disk_num = 1; disk_num <= $SI_NUMDISKS; disk_num++)) ; do
        eval curr_disk="$"SI_DISKNAME_${disk_num}
        disk=$(/usr/bin/aimanifest add -r target/disk@in_vdev mirror_vdev)
        /usr/bin/aimanifest set ${disk}@in_zpool rpool
        /usr/bin/aimanifest set ${disk}@whole_disk true
        disk_name=$(/usr/bin/aimanifest add -r \
            ${disk}/disk_name@name $curr_disk)
        /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 56 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 manifest 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

    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)

    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 57 Specifying Packages to Install Based on IP Address

This example customizes the AI manifest to install one package if the IP address of the AI client is in a specified range, and install a different package if the IP address of the AI 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 manifest 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

# 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 58 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 manifest 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
fi
```

```
exit $SCRIPT_SUCCESS
```

EXAMPLE 59 Adding a System Configuration Profile

Sometimes a system configuration change is needed for each AI client. Rather than having to create an individual system configuration profile on the AI server for each AI client, you could configure a derived manifest script to create the profile for you. The profile must be stored in `/system/volatile/profile` in order for the install service to be able to use it. In this example the settings for the local default router are used when the AI client is reconfigured.

```
ROUTER-CONFIG=/system/volatile/profile/router-config.xml
ROUTER=`netstat -rn | grep "^default" | awk '{print $2}'`

cat<<EOF>${ROUTER-CONFIG}
<!DOCTYPE service_bundle SYSTEM "/usr/share/lib/xml/dtd/service_bundle.dtd.1">
 <service_bundle type="profile" name="router">
   <service name="network/install" version="1" type="service">
     <instance name="default" enabled="true">
       <property_group name="install_ipv4_interface" type="application">
         <propval name="default_route" type="net_address_v4" value="${ROUTER}"/>
       </property_group>
     </instance>
   </service>
 </service_bundle>
EOF
```

EXAMPLE 60 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
```

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 Manifest Scripts

To test your derived manifest 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 manifest script will be added:

```
<!DOCTYPE auto_install SYSTEM "file:///imagepath/auto_install/ai.dtd.1">
<auto_install/>
```

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 manifest script will be added:

Note - Make sure to reset the *imagepath* to the default path, `///usr/share`, before trying to use the script on an AI client.

```
$ installadm list -v -n svcname| grep Image
```

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. Set environment variables in the test environment with values that represent the systems that will be installed using this derived manifest 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 7, “AI Client Attribute Environment Variables,” on page 152 have values and are available for a derived manifest script to use.
5. 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 manifest 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.

6. Use the `validate` subcommand on the resulting manifest.

```
$ /usr/bin/aimanifest validate
```

Messages are displayed only if the validation fails.

The intended system might be very different from the AI server or other system where you might test the derived manifest 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 Manifest Script in an Install Environment

This procedure describes how to test the derived manifest script on one of the intended systems without running the full installation process.

1. **Boot an AI image on a system.**
   Boot an AI image on the system in “Text Installer and command line” mode.
2. **Select Shell from the installer initial menu.**
3. **Copy your script from the AI server.**
   Use `wget` or `sftp` to copy your script from the AI 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 AI server.**

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

Adding a Derived Manifest Script to an Install Service

Add a derived manifest 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 AI clients will use this script to create a manifest for their installation. Also, 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 and XML manifests are both shown when you list manifests associated with a service. A script is listed with the manifest type of `derived`.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 derived manifest script to an install service, the script is not validated.

Run the script in an environment similar to the intended system. See "Testing Derived Manifest Scripts" on page 165 for full instructions.

Add the script to the appropriate AI install service, specifying criteria that define which AI 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.

```
# installadm create-manifest -n solaris11_3-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, "Assigning Customizations to AI Clients" for information about specifying client criteria.

See also "Working With Install Services" on page 97 for more information about the `installadm list`, `export`, `create-manifest`, `set-criteria`, `update-manifest`, and `set-service` subcommands.

Creating an AI Manifest Using the AI Manifest Wizard

The AI manifest wizard is a Browser User Interface (BUI) web application that can be used to create an AI manifest without having to manually edit XML files. The application is reached using the URL for the AI server. The wizard includes eight main screens that allows the user to configure many of the sections of an AI manifest.

Configuring an AI Server for the AI Manifest Wizard

The following examples show how to disable the AI manifest wizard and how to allow users to save manifest files on the AI server.

EXAMPLE 61 Disabling the AI Manifest Wizard

By default the AI manifest wizard is enabled when the AI server is enabled. To disable the AI manifest wizard, use the following command:

```
# installadm set-server -U
```

EXAMPLE 62 Allowing Manifest Files To Be Saved on the AI server

By default, any AI manifests that are created by the AI manifest wizard can not be saved on the AI server. The files must be saved to the users desktop. To allow for the files to be saved on the AI server, use the following command:

```
# installadm set-server -z
```

Once the manifest is saved, the file is stored in `/var/ai/wizard-manifest/`. Then an `installadm` command can be run to associate this manifest with an install service.

▼ How to Create an AI Manifest Using the AI Manifest Wizard

This wizard can also be started by running `/usr/bin/ai-wizard` on the AI server.

Before You Begin In order to make it easier to add a manifest that was created using the AI manifest wizard, you may want to enable the ability to save the generated manifest to a temporary location on the AI server. For more information, see Example 62, “Allowing Manifest Files To Be Saved on the AI server,” on page 169.

1. **Start the AI manifest wizard.**

   The application is reached using the URL for the AI server. By default, the URL for an AI server named `ai-server` would be: `http://ai-server.domain:5555`.

2. **In the Welcome screen: identify the service to associate the manifest with.**

   This screen lists all of the install services that are configured, as well as the status and client architecture for each install service. The first item on the list relates to the AI server itself, and is always displayed, so if no services are configured you can still create a manifest by selecting this item. Select the install service you want to associate the AI manifest with, then click `Start`.

3. **In the Introduction screen: select a manifest name and the target.**

   Type in the name for the AI manifest or choose to use `default`. Also choose whether the AI manifest is for a global or a non-global zone. This last value is called the target. Click `Next` to continue.

4. **In the Root Pool screen: enter information about the root pool.**

   In this screen, you can set the root pool name, the boot environment name, select whether the root pool is mirrored, and define swap and dump device configuration parameters. Click `Next` to continue.

5. **In the Data Pools screen: enter information about additional ZFS storage pools.**

   You can specify up to 5 data pools. For each data pool you must specify a pool name and a mountpoint. For each data pool you can choose a redundancy level of `None`, `Mirror`, `Raid-Z`, `Raid-Z1`, `Raid-Z2` or `Raid-Z3`. Click `Next` to continue.

6. **In the Disks screen: allocate disks for the root and storage pools.**

   This screen lists all of the pools that you have defined in the previous screens. For each pool, you can configure one of the following:

   - Disk keyword, `Auto` or `Boot Disk`

- Disk property, such as Device Size or Device Type
- Disk name, such as CTD Name or Volume Id

If you need to configure more than one of these properties at a time. Select one of them to add when using the wizard, then after the manifest is created add additional properties, by editing the file or by using the aimanifest command. Click Next to continue.

7. **In the Repositories screen: define the IPS repositories.**

   The default Oracle Solaris support and release repositories are automatically defined for you. Also you can enter information about no more than five additional repositories that include packages that you want to add to the AI client. Each repository will need a repository name and an origin URI.

   For each repository after clicking Add Details, you can also indicate the SSL certificate file, the SSL key file, as well as any backup origin URIs for the repository. Click Next to continue.

8. **In the Software screen: select the software packages to be installed.**

   You can choose to select to install the large server, small server, and desktop group packages as needed. In addition you can add package FMRIs for any additional or custom packages that need to be added when the AI client is created. Click Next to continue.

9. **In the Zones screen: define zone names and define zone configuration file URI.**

   For each zone that you want to be added when the AI client is created, enter the zone name and the URI for the zone configuration file. Click Next to continue.

10. **In the Review screen: check the information that you have entered, then create the manifest by clicking Save.**

    If the AI server has been configured to allow server side saving, your manifest will be saved to the AI server and you will have the option to save the file locally as well. Otherwise you will only be able to save the manifest locally.

Creating an AI Manifest Using the installadm Command

For the Oracle Solaris 11.3 release, the installadm create-manifest and installadm update-manifest commands have been enhanced to allow you to edit the manifest being created or updated. Additionally, it is now possible to create a new manifest from an existing manifest in the install service. For example:

create-manifest -n *servicename* -m *new-manifest* brings up a manifest in the installadm interactive mode that is based on reasonable defaults.

`create-manifest -n` *servicename* `-f` *file* creates a manifest using the contents of the named `file`.

`create-manifest -n` *servicename* `-f` *file* `-e` allows you to edit a manifest using the contents of the named `file`. If the file is a derived manifest script, then you will be placed in an editor. The editor selected is either defined by the `VISUAL` or `EDITOR` environmental variables. If neither variable is defined, then the `vi` editor is used. If the file is not a derived manifest script, then you will edit the manifest in the `installadm` interactive mode.

`create-manifest -n` *servicename* `-M` *manifest* `-m` *new-manifest* creates a new manifest using the contents of the existing `manifest`.

`create-manifest -n` *servicename* `-M` *manifest* `-e` -m *new-manifest* creates a new manifest using the contents of the existing `manifest`. If the existing manifest is a derived manifest script, then you will be placed in an editor. The editor selected is either defined by the `VISUAL` or `EDITOR` environmental variables. If neither variable is defined, then the `vi` editor is used. If the existing manifest is not a derived manifest script, then you will edit the manifest in the `installadm` interactive mode.

`update-manifest -n` *servicename* `-m` *new-manifest* updates a manifest. If the manifest is a derived manifest script, then you will be placed in an editor. The editor selected is either defined by the `VISUAL` or `EDITOR` environmental variables. If neither variable is defined, then the `vi` editor is used. If the manifest is not a derived manifest script, then you will edit the manifest in the `installadm` interactive mode.

`update-manifest -n` *servicename* `-f` *file* `-m` *manifest* updates a manifest using the contents of the named `file`

`update-manifest -n` *servicename* `-f` *file* `-m` *manifest* `-e` allows you to edit a manifest that has been updated using the contents of the `file`. If the file is a derived manifest script, then you will be placed in an editor. The editor selected is either defined by the `VISUAL` or `EDITOR` environmental variables. If neither variable is defined, then the `vi` editor is used. If the file is an xml file, then you will edit the manifest in the `installadm` interactive mode.

`installadm` Interactive Editor Commands

You can get into the interactive editor mode either by using the `-e` option on the command line. Once in the interactive edit mode, you can use the following commands:

add	Adds an object or property. If you add the `-w` option the command will prompt for objects and properties. See the description of `walk` below.
cancel	Discards any changes made on the current level and navigates up one level.

commit	At the top level, validates changes, saves the manifest and continues editing.
delete	Deletes an object or property.
end	Validates changes made on the current level and if no errors occur navigates up one level.
exit	Prompts whether to save the manifest and exit after the changes are validated, exit without saving uncommitted changes or continue editing.
info	Displays all objects and properties up to one level down.
move	Changes the order of multiple objects, such as software publishers.
select	Selects an object and navigates to that level.
set	Sets the value of a property.
walk	Prompts for each settable property and any settable subobjects or subproperties for the select object.

Examples of Using the installadm Interactive Edit Mode

Several example showing the interactive mode are given in the installadm(1M) man page. The following sections provide additional examples.

EXAMPLE 63 Creating a Manifest Entry with installadm in the Walk Mode

In this example a second publisher is added to the manifest named test.

```
# installadm create-manifest -n default-sparc -m test
installadm:test> select software
installadm:test:software> info
   type: IPS
   name: <not specified>
   facet[1]:
     ...
   facet[20]:
      name: facet.locale.zh_TW
      value: true
   publisher:
      name: solaris
```

```
      key: <not specified>
      cert: <not specified>
      ca-cert: <not specified>
      origin: http://pkg.oracle.com/solaris/release
      mirror: <not specified>
      cmd-options: <not specified>
   pkg-list:
      action: install
      name: pkg:/entire@0.5.11-0.175.3
      name: pkg:/group/system/solaris-large-server
      reject: <not specified>
installadm:test:software> add -w publisher
   *** To terminate walk, use Ctrl-D ***
   name [<not specified>]: firstboot
   key [<not specified>]:
   cert [<not specified>]:
   ca-cert [<not specified>]:
   origin [<not specified>]: file:///net/host1/export/firstbootrepo
   origin [<not specified>]:
   mirror [<not specified>]:
   cmd-options [<not specified>]:
installadm:test:software:publisher> info
   name: firstboot
   key: <not specified>
   cert: <not specified>
   ca-cert: <not specified>
   origin: file:///net/host1/export/firstbootrepo
   mirror: <not specified>
   cmd-options: <not specified>
installadm:test:software:publisher> end
installadm:test:software> exit
1. Save manifest and exit
2. Exit without saving uncommitted changes
3. Continue editing
Please select choice: 1
Created Manifest: 'test'
```

EXAMPLE 64 Changing the Order of an Object with `installadm`

In this example, the order of the publishers is switched.

```
# installadm update-manifest -n default-sparc -m test
installadm:test> select software
installadm:test:software> info
   type: IPS
   name: <not specified>
   facet[1]:
     ...
```

```
   facet[20]:
      name: facet.locale.zh_TW
      value: true
   publisher[1]:
      name: solaris
      key: <not specified>
      cert: <not specified>
      ca-cert: <not specified>
      origin: http://pkg.oracle.com/solaris/release
      mirror: <not specified>
      cmd-options: <not specified>
   publisher[2]:
      name: firstboot
      key: <not specified>
      cert: <not specified>
      ca-cert: <not specified>
      origin: http://example.com/solaris/mybuild
      mirror: <not specified>
      cmd-options: <not specified>
   pkg-list:
      action: install
      name: pkg:/entire@0.5.11-0.175.3
      name: pkg:/group/system/solaris-large-server
      reject: <not specified>
installadm:test:software> move publisher 2 1
installadm:test:software> info
   type: IPS
   name: <not specified>
     ...
  publisher[1]:
      name: firstboot
      key: <not specified>
      cert: <not specified>
      ca-cert: <not specified>
      origin: http://example.com/solaris/mybuild
      mirror: <not specified>
      cmd-options: <not specified>
   publisher[2]:
      name: solaris
      key: <not specified>
      cert: <not specified>
      ca-cert: <not specified>
      origin: http://pkg.oracle.com/solaris/release
      mirror: <not specified>
      cmd-options: <not specified>
   pkg-list:
      action: install
      name: pkg:/entire@0.5.11-0.175.3
```

```
      name: pkg:/group/system/solaris-large-server
      reject: <not specified>
installadm:test:software> exit
1. Save manifest and exit
2. Exit without saving uncommitted changes
3. Continue editing
Please select choice: 1
Updated Manifest: 'test'
```

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 manifest script.

All manifests shown in this section are based on the sample default XML file located at */image-path*`/auto_install/manifest/default.xml`, with the necessary modifications made. The `destination` element in the `software` element is omitted for brevity.

Caution - The `entire` package constrains system package versions to the same build, so it must be included in each manifest. Proper system updates and correct package selection depend on the presence of this package. Do not remove the installation of this package from your AI manifest, and do not uninstall this package after installation. Removing this package will result in an unsupported system.

The following examples are included:

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.

```
<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="192.0.2.200"/>
      </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@0.5.11-0.175.2</name>
        <name>pkg:/group/system/solaris-large-server</name>
      </software_data>
    </software>
  </ai_instance>
</auto_install>
```

Specifying a Root Pool and Boot Pool in an AI Manifest

The following sample AI manifest defines an iSCSI device as the root pool, and configures the boot pool to be named `mybpool`. The default name for a boot pool is `bpool`. Because no disk

devices are included with the boot pool description, the dedicated eUSB disks will be used by default.

```
<auto_install>
  <ai_instance name="default">
    <target>
      <disk in_zpool="rpool" whole_disk="true">
        <iscsi target_port="326" target_ip="6.6.6.53"\
               target_lun="1">
      </disk>
      <logical>
        <zpool name="rpool" is_root="true">
          <filesystem name="export" mountpoint="/export"/>
          <filesystem name="export/home"/>
          <be name="solaris"/>
        </zpool>
        <zpool name="mybpool" is_boot="true">
      </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@0.5.11-0.175.2</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 56, “Specifying a Mirrored Configuration If at Least Two Disks of a Specified Size Are Present,” on page 159. 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.

```
<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@0.5.11-0.175.2</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.

```
<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@0.5.11-0.175.2</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>
```

Installing Multiple SVR4 Packages

To install multiple SVR4 packages, you will need to specify the software tag for each package as shown below.

```
<software type="SVR4">
  <source>
    <publisher>
      <origin name="/net/192.0.2.2/svr4/app1.pkg"/>
```

```
    </publisher>
  </source>
  <software_data>
    <name>application1</name>
  </software_data>
</software>
<software type="SVR4">
  <source>
    <publisher>
      <origin name="/net/192.0.2.2/svr4/app2.pkg"/>
    </publisher>
  </source>
  <software_data>
    <name>application2</name>
  </software_data>
</software>
```

Reusing Existing Disk Slices or Partitions

This sample shows how to specify to use existing disk slices for a SPARC client. For disk slices, the dimensions (start_sector and size) from an existing slice are reused. The configuration process will not search the slices to see if there is already a version of Solaris installed.

```
    <disk>
      <disk_name name="c1t0d0" name_type="ctd"/>
      <slice name="0" action="use_existing" force="true" in_zpool="rpool">
    </disk>
```

The following example shows how to specify for an x86 client that an existing partition on a disk should be reused during the AI process. For partitions, the existing dimensions for the named slice should be reused. In this case which partition is reused is automatically determined during the configuration process.

```
    <partition action="use_existing_solaris2">
        <slice action="use_existing" name="0" force="true"/>
    </partition>
```

Accessing a Unified Archive Using SSL Client Authentication

To access a unified archive using SSL, include a `credentials` element, to provide key, certificate, and CA certificate information in the AI manifest. In this example, the unified archive is called `sslarchive.ua`.

```
<software type="ARCHIVE">
   <source>
       <file uri="https://example.com/sslarchive.ua"/>
           <credentials>
               <key src="file://root/key.pem"/>
               <cert src="file://root/cert.pem"/>
               <cacert src="file://root/cacert.pem"/>
           </credentials>
       </file>
   </source>
</software>
```

See "Increasing Security for Automated Installations" on page 109 for information about setting up SSL on your AI server.

Accessing a Unified Archive Using Http Authentication Tokens

To use an HTTP authentication token when accessing a unified archive, add the token to a `credentials` element. Also, add a line defining the unified archive.

```
<software type="ARCHIVE">
    <source>
        <file uri="http://example.com/httparchive.ua"/>
            <credentials>
                <http_auth_token>my-specifically-granted-auth-token</http_auth_token>
            </credentials>
        </file>
    </source>
</software>
```

Accessing a Secure IPS Repository

In order to access a secure IPS repository, you must specify the key and certificate for the repository.

```
<publisher name="solaris">
    <origin name="https://pkg.oracle.com/solaris/support/"/>
    <credentials>
        <key src="/var/pkg/ssl/Oracle_Solaris_11_Support.key.pem" />
        <cert src="/var/pkg/ssl/Oracle_Solaris_11_Support.certificate.pem" />
    </credentials>
</publisher>
```

Changing the Locale When Installing the `solaris-minimal-server` Package

If you install `solaris-minimal-server` group, only the C/POSIX locale is installed on the system. If you use any of the UTF-8 locales, you need to install the `system/locale` package and customize IPS `facet.locale` attribute setting as needed. If you do not set `facet.locale.*` to false all of the available UTF-8 locales will be installed. For example, if your system's default locale is set to `en_US.UTF-8` in the system configuration profile, you should add these lines to the manifest to add the `en_US` locale:

```
<software type="IPS">
  <destination>
    <image>
      <!-- Specify locales to install -->
      <facet set="false">facet.locale.*</facet>
      <facet set="true">facet.locale.en</facet>
      <facet set="true">facet.locale.en_US</facet>
    </image>
  </destination>
...
    <software_data action="install">
       <name>pkg:/entire@5.11-5.11.0</name>
       <name>pkg:/group/system/solaris-minimal-server</name>
       <name>pkg:/system/locale</name>
    </software_data>
  </software>
```

Not Installing the Man Page Package with the `solaris-minimal-server` Package

Even though the `solaris-minimal-server` group does not include the `man` command, by default it includes man pages. To prevent the installation of the man pages, you need to set the `doc.man` facet to `false` as shown:

```
  <software type="IPS">
    <destination>
      <image>
        <!-- Specify that man pages should not be installed -->
        <facet set="false">doc.man</facet>
      </image>
    </destination>
...
    <software_data action="install">
       <name>pkg:/entire@5.11-5.11.0</name>
       <name>pkg:/group/system/solaris-minimal-server</name>
       <name>pkg:/system/locale</name>
    </software_data>
  </software>
```

Adding a Specific Package when Installing a Package Group

Even though the `solaris-minimal-server` group does not include the `man` command, by default it includes man pages. To add the `man` command to the system during installation, you need to include the `/text/doctools` package as show below:

```
  <software type="IPS">
...
    <software_data action="install">
       <name>pkg:/entire@5.11-5.11.0</name>
       <name>pkg:/group/system/solaris-minimal-server</name>
       <name>pkg:/text/doctools</name>
    </software_data>
  </software>
```

Default AI Manifest

The default AI manifest for an install service is a derived manifest. When you create an install service, a default manifest called `orig_default` is created for the service. The default manifest that is created for you might be slightly different for different install images. The `ai_manifest(4)` man page provides more information about the XML contents of this file.

♦ ♦ ♦ CHAPTER 11

Defining AI Client System Configuration Parameters

This chapter describes how to specify information needed to configure the AI client after installation. You can specify configuration of anything that is configurable by using Service Management Facility (SMF) properties.

Providing Configuration Profiles

System configuration profiles specify system configuration as a set of configuration parameters in the form of an 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 AI client can use any number of system configuration profiles. For example, a AI client might be assigned one profile that provides just the host name and IP address for that client. The same AI 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 AI client, the interactive configuration tool opens on that system upon the first boot after installation. You would need to manually provide the information as prompted.

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 called `sc_profile.xml` from responses you enter interactively:

  ```
  # sysconfig create-profile [-o directory] [other-options]
  ```

  For other options you can use, see the `sysconfig(1M)` man page.

  By default, the profile is created in the `/system/volatile/profile/` directory. To create `sc_profile.xml` elsewhere, specify the `-o` *directory* option. Before issuing the command, ensure first that *directory* exists. The new profile overwrites any `sc_profile.xml` existing in *directory*.
- Create the system configuration profile manually, using the property specifications shown in "Specifying Configuration in a System Configuration Profile" on page 190 and "Example System Configuration Profiles" on page 204.

  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 derived manifest script can create a system configuration profile when the script is run. See Example 59, "Adding a System Configuration Profile," on page 164.

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 system

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 137 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 AI clients select that system configuration profile. If no criteria are specified, all AI clients use this profile.

Specify the environment for which the profile will be applied to the AI client with the `-e` option. The environment can be set to one of the following: `install` for a profile used in the installation environment, `system` for a profile to be used after installing or `all` for a profile that can be used in either environment.

A single AI client can match and use more than one system configuration profile. Make sure that no AI client uses a set of profiles such that a particular property is specified more than one time. If an AI 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 AI client does not match any criteria specified for any system configuration profile in the install service, the interactive configuration tool opens on that system on first boot after installation.

Use the `installadm list` command to list profiles that have been added to a given install service and list the criteria and environment 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 set-profile` command to rename a profile or set the environment value of a profile so that the profile will be used in the install environment.

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 “Working With Install Services” on page 97 and the `installadm(1M)` man page for more information about these subcommands.

Specifying Configuration in a System Configuration Profile

You can specify configuration of any system characteristic that is configurable by using SMF 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] [`*property-group*`/`*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(5).
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
(5).
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(5).
  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 directory]
```

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 - Generate an encrypted password, or password hash, by using the `pwhash` command. You can provide the resulting hash for any account.

Configuring the Root Account

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

TABLE 8 `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`.
`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 65 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 193
- “User Account Properties” on page 193

- “Configuring Multiple Initial Users” on page 194

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 system 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 9 `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 pathname of the program used as the user's shell on login.
`uid`	`count`	`optional`	UID of the new user. y default the UID is set to the next number above the highest number currently assigned.
`gid`	`count`	`optional`	User's primary group membership. The default GID is 10, which is the `staff` group.
`type`	`astring`	`optional`	Account type: `normal` or `role`.
`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 `/etc/sudoers.d/svc-system-config-user` file along with the `login`.
`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.

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

EXAMPLE 66 Configuring an Account That Does Not Use a Password

In this example, the password is set to NP so that the account can only be accessed using the `su` command as a privileged user or the `ssh` command with key-based authentication.

```
<service name="system/config-user" version="1" type="service">
    <instance name="default" enabled="true">
        <property_group name="user_account">
          <propval name="login" value="jack"/>
          <propval name="password" value="NP"/>
          <propval name="description" value="default_user"/>
          <propval name="shell" value="/usr/bin/bash"/>
          <propval name="gid" value="10"/>
          <propval name="uid" value="1001"/>
          <propval name="type" value="normal"/>
          <propval name="roles" value="root"/>
          <propval name="profiles" value="System Administrator"/>
        </property_group>
    </instance>
</service>
```

Configuring Multiple Initial Users

To configure multiple users on the newly-installed system, specify the users by using the `useradd` 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.

Configuring SSH Keys

The `ssh_public_keys` property group holds pre-generated ssh keys. The keys will be written to the users `$HOME/ .ssh/authorized_keys` file when the system is configured.

EXAMPLE 67 Configuring SSH Keys

```
 <property_group name="user_account" type="application">
   <...>
   <property type="astring" name="ssh_public_keys">
       <astring_list>
           <value_node value='[<options>] <key-type> <base64-encoding-key>
[<comment>] />'
           <value_node value='[<options>] <key-type> <base64-encoding-key>
[<comment>] />'
       </astring_list>
    </property>
  </property_group>
```

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 directory]
```

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 10 `config` Property Group Properties

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

EXAMPLE 68 Configuring the Host Name

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

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

```
        </property_group>
    </instance>
</service>
```

EXAMPLE 69 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 directory]
```

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 `timezone` Property Group Properties

Property	Type	Required	Description
`localtime`	`astring`	optional	System time zone.

EXAMPLE 70 Configuring the Time Zone

This example sets the time zone to Central European Time/Prague, CZ.

```
<service name='system/timezone' version='1'>
```

```
  <instance name='default' enabled='true'>
    <property_group name='timezone'>
      <propval name='localtime' value='Europe/Prague'/>
    </property_group>
  </instance>
</service>
```

The `svc:/system/environment:init` SMF service sets the locale for the system.

The `environment` property group can define the following environment variables. See the `environ(5)` man page for information about environment variables.

TABLE 12 `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 71 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

The following examples show how to set the terminal type and keyboard layout for the console in a system configuration profile.

EXAMPLE 72 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`.

```
<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 73 Configuring Keyboard Layout

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

```
# sysconfig create-profile -g keyboard [-o directory]
```

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 Network Interfaces

Use the `sysconfig create-profile` command with the `network` grouping to generate a valid profile that configures the network. This command will start the SCI Tool, which will prompt you for the information needed to configure an interface.

```
# sysconfig create-profile -g network [-o directory]
```

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 multiple IPv4 and IPv6 interfaces and, optionally, a default route reachable by these interfaces. The service allows you to configure IPv4 and IPv6 interfaces. The service uses its properties and the `ipadm` command to configure the network interfaces. Similarly, the service uses its properties and the `route` command to define a default route.

See the examples in “Specifying Static Network Configuration” on page 207.

The `install_ipv4_interface` property group only allows one interface to be configured, but the `ipv4_interface` property group allows for multiple interfaces to be configured. Both IPv4 property groups contain the properties listed in the following table.

TABLE 13 Property Group Properties for an IPv4 Network Interface

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 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 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 create-addr` subcommand.
`default_route`	`net_address_v4`	`optional`	Used to define a default route using the `route`command. `#` **`/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 only allows one interface to be configured, but the `ipv6_interface` property group allows for multiple interfaces to be configured. The property groups for an IPv6 interface contains the properties listed in the following table.

TABLE 14 Property Group Properties for an IPv6 Network Interface

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 create-addr` subcommand. Valid values are `static` or `addrconf`.
`static_address`	`net_address_v6`	`optional`	Only required with an `address_type` of `static`. Used to construct the local address for the `ipadm 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 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 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 create-addr` subcommand.

`default_route`	`net_address_v6`	`optional`	Used to define a default route using the `route` command.
			`# /usr/sbin/route \`
			`-p add default` *default-route* `\`
			`-ifp` *ifname*
			The value of *ifname* is the interface name portion of the `name` property.

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 the name service switch.

```
# sysconfig create-profile -g naming_services [-o directory]
```

The `naming_services` grouping includes two SMF services.

- The `svc:/system/name-service/switch` service manages the naming service.
- The `svc:/network/dns/client` service manages the DNS service.

The `svc:/system/name-service/switch` SMF service configures the name service switch. 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 213. The `config` property group of the `svc:/system/name-service/switch` service includes the properties listed in the following table. For a complete listing of all of the properties see the `nsswitch.conf(4)` man page.

TABLE 15 config Properties of the `svc:/system/name-service/switch` Property Group

Property	Type	Required	Description
default	astring	optional	Sets the default source configuration for all name service switch databases
bootparam	astring	optional	Overrides the default source configuration for the bootparams database
ether	astring	optional	Overrides the default source configuration for the ethers database
group	astring	optional	Overrides the default source configuration for the group database
host	astring	optional	Overrides the default source configuration for the host database
netmask	astring	optional	Overrides the default source configuration for the netmask database
network	astring	optional	Overrides the default source configuration for the network database
password	astring	optional	Overrides the default source configuration for the passwd database
protocol	astring	optional	Overrides the default source configuration for the protocol database
rpc	astring	optional	Overrides the default source configuration for the rpc database

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 configuration information for the DNS service. See the examples in "Specifying Name Service Configuration" on page 213.

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

TABLE 16 `config` Property Group Properties

Property	Type	Required	Description
`domain`	`astring`	`optional`	Local domain name. Used to construct the `domain` directive in `resolv.conf`.
`nameserver`	`net_address_list`	`required`	List of IPv4 and IPv6 addresses. Used to construct the `nameserver` directives in `resolv.conf`.
`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`.

Configuring Kerberos

A system configuration profile that includes Kerberos configuration information for an AI client should be created by the `kclient` command. Although the profile can be viewed, editing the file by hand is not suggested. For more information, see "How to Configure Kerberos Clients Using AI" on page 120.

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 directory]
```

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

- The systems use the same My Oracle Support credentials to register. All AI clients hat use this profile register with My Oracle Support in the same way. The data from all of these AI clients will be associated with the same My Oracle Support account.
- The systems access My Oracle Support through the same network configuration. All AI client 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, 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 AI client's installation environment during the installation process. In this way, a single profile file can set different

configuration parameters on different AI clients. See Table 17, “Variables for System Configuration Template Profiles,” on page 204 for a list of variables you can use.

In the following example profile named `hostIPnet.xml`, `AI_HOSTNAME` is a placeholder for the AI clients host name, and `AI_IPV4` is a placeholder for the AI clients 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>
    </instance>
  </service>
  <service version="1" type="service" name="network/install">
    <instance enabled="true" name="default">
      <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="203.0.113.1"/>
      </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 AI client without changing the input `hostandIP.xml` file.

```
# installadm create-profile -n solaris11_3-i386 -f /export/hostIPnet.xml
```

While the `hostandIP.xml` file remains unchanged, the profiles that are applied to an AI client are customized. For example, the `hostandIP.xml` profile might have the following content when a AI 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>
    </instance>
  </service>
  <service version="1" type="service" name="network/install">
```

```
    <instance enabled="true" name="default">
      <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="203.0.113.2/27"/>
        <propval name="default_route" type="net_address_v4" value="203.0.113.1"/>
      </property_group>
    </instance>
  </service>
</service_bundle>
```

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

Note - Profile template variables are not supported in zones profiles.

TABLE 17 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	I client DNS name
AI_IPV4	IP version 4 network address
AI_IPV4_PREFIXLEN	Prefix length of the IPv4 network address
AI_MAC	Hexadecimal MAC address with colon (:) separators
AI_MEM	Memory size in megabytes returned by `prtconf`
AI_NETLINK_DEVICE	Name of network interface physical device
AI_NETLINK_VANITY	Default vanity name of network interface
AI_NETWORK	IP version 4 network identifier
AI_ROUTER	IP version 4 network address of the AI client's default router
AI_ZONENAME	AI client zone name

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. Portions of these samples can be pieced together as needed. The following sample system configuration profiles are included in `/usr/share/auto_install/sc_profiles`:

- `enable_sci.xml` - starts the system configuration interactive tool during the boot process.

- `sc_sample.xml` - shows how to configure a user account, root role, console terminal type, keyboard, locale, timezone, and a single IPv4 network interface. See
- `static_network.xml` - same as `sc_sample.xml`, except the network configuration is using profile template variables to use the network configuration from the netbooted AI client. Also this profile includes an example of configuring a naming service.
- `unconfig.xml` - unconfigures the AI client during the boot process.
- `install_env/dns.xml` - partial profile that includes configuration information to configure a DNS client.
- `install_env/network.xml` - partial profile that includes configuration information for a static IP address and an InfiniBand datalink with static IP.

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 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 the installadm(8) man page 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/nstall" version="1">
      <instance name="default" enabled="true">
          <property_group name="install_ipv4_interface" type="application">
            <propval name="name" type="astring" value="{{AI_NETLINK_VANITY}}/v4"/>
            <propval name="address_type" type="astring" value="dhcp"/>
          </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 `203.0.113.10` and netmask `255.255.255.224`
- `203.0.113.1` 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) 2010, 2011, 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="install_ipv4_interface" type="application">
            <propval name="name" type="astring" value="{{AI_NETLINK_VANITY}}/v4"/>
            <propval name="address_type" type="astring" value="static"/>
            <propval name="static_address" type="net_address_v4" value="{{AI_IPV4}}/
{{AI_IPV4_PREFIXLEN}}"/>
            <propval name="default_route" type="net_address_v4" value="{{AI_ROUTER}}"/>
          </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="203.0.113.10
/27"/>
                <propval name="default_route" type="net_address_v4" value="203.0.113.1"/
>
            </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 an IB Link in a System Configuration Profile

The following example shows how to define an Infiniband (IB) link in a system configuration profile.

```
<!DOCTYPE service_bundle SYSTEM "/usr/share/lib/xml/dtd/service_bundle.dtd.1">
<service_bundle type="profile" name="system configuration">

    <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="net5_partlink" type="ib_partlink">
               <propval name="name" type="astring" value="FFFF.net5"/>
               <propval name="link" type="astring" value="net5"/>
               <propval name="pkey" type="astring" value="FFFF"/>
            </property_group>

            <property_group name="FFFF_net5_ipv4" type="ipv4_interface">
                <propval name="name" type="astring" value="FFFF.net5/v4"/>
                <propval name="address_type" type="astring" value="static"/>
                <propval name="static_address" type="net_address_v4" value="203.0.113.49
/27"/>

            <property_group name="net0_ipv4" type="ipv4_interface">
                <propval name="name" type="astring" value="net0/v4"/>
                <propval name="address_type" type="astring" value="static"/>
                <propval name="static_address" type="net_address_v4" value="203.0.113.49
/27"/>
                <propval name="default_route" type="net_address_v4" value="203.0.113.32"
/>
            </property_group>

        </instance>
    </service>

    <service name="network/dns/client" version="1">
        <property_group name="config">
            <property name="nameserver">
                <net_address_list>
                    <value_node value="192.0.2.35"/>
                    <value_node value="198.51.100.0/27"/>
                    <value_node value="198.51.100.32/27"/>
                </net_address_list>
            </property>
            <property name="search">
                <astring_list>
                    <value_node value="example.com"/>
                </astring_list>
            </property>
        </property_group>
        <instance name="default" enabled="true"/>
    </service>

    <service version="1" name="system/name-service/switch">
        <instance enabled="true" name="default"/>
        <property_group name="config">
```

```
            <propval name="default" value="files"/>
            <propval name="host" value="files dns mdns"/>
        </property_group>
    </service>

    <service version="1" name="system/name-service/cache">
        <instance enabled="true" name="default"/>
    </service>
```

Configuring Multiple IPv4 Interfaces

This example uses the `ipv4_interface` property group type which allows for multiple interfaces to be configured. You can also use the `ipv6_interface` property group type.

```
<?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="network/install">
            <instance enabled="true" name="default">
                <property_group name="install_ipv4_interface_0" type="ipv4_interface">
                    <propval type="net_address_v4" name="static_address" value="
203.0.113.10/27"/>
                    <propval type="astring" name="name" value="net0/v4"/>
                    <propval type="astring" name="address_type" value="static"/>
                </property_group>
                <property_group name="install_ipv4_interface_1" type="ipv4_interface">
                    <propval type="net_address_v4" name="static_address" value="
203.0.113.11/27"/>
                    <propval type="astring" name="name" value="net1/v4"/>
                    <propval type="astring" name="address_type" value="static"/>
                </property_group>
            </instance>
         </service>
     </service_bundle>
      <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="203.0.113.10
/27"/>
                <propval name="default_route" type="net_address_v4" value="203.0.113.1"/
>
            </property_group>
```

Adding User SSH Keys

This example shows how to use the `ssh_public_keys` to add ssh keys for a user during an automated install session. Each key will be added to the `$HOME/.ssh/authorized_keys` for the named user.

```
    <property_group type="application" name="user_account">
            ...
      <property type="astring" name="ssh_public_keys">
        <astring_list>
          <value_node value='[<options>] <key-type> <base64-encoding-key> [<comment>]' /
>
          <value_node value='[<options>] <key-type> <base64-encoding-key> [<comment>]' /
>
        </astring_list>
      </property>
    </property_group>
```

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 "How to Use the SCI Tool" on page 68 and the `sysconfig`(1M) man page for more information about using `sysconfig` to create a system configuration profile.

Configuring Name Service NIS

EXAMPLE 74 Enabling NIS For a Specified Domain

This example profile performs the following configuration:

- Enables NIS for `example.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='example.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 75 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 `example.domain.com`
- Enables the name service cache service, which is required
- Disables the DNS name service

```
<?xml version='1.0'?>
<!DOCTYPE service_bundle SYSTEM "/usr/share/lib/xml/dtd/service_bundle.dtd.1">
<service_bundle type="profile" name="sysconfig">
  <!-- service name-service/switch below for NIS only - (see nsswitch.conf(5)) -->
  <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="example.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 76 Configuring NIS

The following profile configures the NIS name service with a NIS server IP address of `203.0.113.10` and domain of`mydomain.com`. The NIS server is not required to be on the same subnet when the NIS 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(5)) -->
  <service version="1" type="service" name="system/name-service/switch">
    <property_group type="application" name="config">
      <propval type="astring" name="default" value="files nis"/>
      <propval type="astring" name="printer" value="user files nis"/>
      <propval type="astring" name="netgroup" value="nis"/>
    </property_group>
    <instance enabled="true" name="default"/>
  </service>
  <!-- name-service/cache must be present along with name-service/switch -->
  <service version="1" type="service" name="system/name-service/cache">
```

```
    <instance enabled="true" name="default"/>
  </service>
  <!-- if no DNS, must be explicitly disabled to avoid error msgs -->
  <service version="1" type="service" name="network/dns/client">
    <instance enabled="false" name="default"/>
  </service>
  <service version="1" type="service" name="network/nis/domain">
    <property_group type="application" name="config">
      <propval type="hostname" name="domainname" value="mydomain.com"/>
      <!-- Note: use property with net_address_list and value_node as below -->
      <property type="net_address" name="ypservers">
        <net_address_list>
          <value_node value="203.0.113.10"/>
        </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 77 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.oracle.com'/>
        <property name='nameserver' type='net_address'>
          <net_address_list>
            <value_node value='192.0.2.52/27' />
            <value_node value='192.0.2.41/27' />
```

```
            <value_node value='192.0.2.15/27' />
          </net_address_list>
       </property>
      <property name='search' type='astring'>
          <astring_list>
            <value_node value='us.oracle.com oracle.com oraclecorp.com' />
          </astring_list>
      </property>
      </property_group>
    <instance name='default' enabled='true' />
    </service>
    <service name='network/nis/domain' type='service' version='1'>
      <property_group name='config' type='application'>
          <propval name='domainname' type='hostname' value='mydomain.com'/>
      </property_group>
    <instance name='default' enabled='true' />
  /service>
    <service name='network/nis/client' type='service' version='1'>
      <property_group name='config' type='application'>
          <propval name='use_broadcast' type='boolean' value='true'/>
      </property_group>
      <instance name='default' enabled='true' />
    </service>
    <service name='system/name-service/switch' type='service' version='1'>
      <property_group name='config' type='application'>
          <propval name='default' type='astring' value='files nis'/>
           <propval name='host' type='astring' value='files dns'/>
          <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 the DNS Name Service

EXAMPLE 78 Configuring DNS With a Search List

The following example profile configures the following parameters:

- Name service: DNS
- Server IP addresses: `192.0.2.1` and `198.51.100.1`

- Domain: `example.domain.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(5)) -->
  <service version="1" type="service" name="system/name-service/switch">
    <property_group type="application" name="config">
      <propval type="astring" name="default" value="files"/>
      <propval type="astring" name="host" value="files dns"/>
      <propval type="astring" name="printer" value="user files"/>
    </property_group>
    <instance enabled="true" name="default"/>
  </service>
  <!-- name-service/cache must be present along with name-service/switch -->
  <service version="1" type="service" name="system/name-service/cache">
    <instance enabled="true" name="default"/>
  </service>
  <service version="1" type="service" name="network/dns/client">
    <property_group type="application" name="config">
      <!-- Note: use property with net_address_list and value_node as below -->
      <property type="net_address" name="nameserver">
        <net_address_list>
          <value_node value="192.0.2.1"/>
          <value_node value="198.51.100.1"/>
        </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="my.domin.com domain.com"/>
        </astring_list>
      </property>
    </property_group>
    <instance enabled="true" name="default"/>
  </service>
</service_bundle>
```

Configuring Name Service LDAP

EXAMPLE 79 Configuring LDAP and LDAP Search Base

This example profile configures the following parameters:

- DAP me server IP address: `203.0.113.10`
- Domain `example.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">
        <host_list>
          <value_node value="203.0.113.10"/>
        </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="example.domain.com"/>
    </property_group>
    <instance enabled="true" name="default"/>
  </service>
</service_bundle>
```

EXAMPLE 80 Configuring LDAP With a Secure LDAP Server

This example profile configures the following parameters:

- AP server IP address: `203.0.113.10`
- Domain `example.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"/>
      <property type="host" name="server_list">
        <host_list>
          <value_node value="203.0.113.10"/>
        </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="example.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="203.0.113.10"/>
        </net_address_list>
      </property>
      <propval type="astring" name="domain" value="example.domain.com"/>
      <property type="astring" name="search">
        <astring_list>
          <value_node value="example.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.example.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="example.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="203.0.113.10"/>
        </net_address_list>
      </property>
      <propval type="astring" name="domain" value="example.domain.com"/>
      <property type="astring" name="search">
        <astring_list>
          <value_node value="example.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="example.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 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 226 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 227 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.

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 AI client o 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 AI 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 AI client 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 228 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 81, "Default Zone AI Manifest," on page 229.
System configuration profile	Optional: You can provide zero or more system configuration profiles for a non-global zone. These profiles are similar to the profiles for configuring the global zone. See Chapter 11, "Defining AI Client System Configuration Parameters" for information about system configuration profile files. You might want to provide profiles to specify zone configuration such as users and the root password for the zone administrator. See "Non-Global Zone System Configuration Profiles" on page 231 for an example profile for a non-global zone. To provide system configuration profiles for a zone, add the 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 profile. If you do not provide any system configuration profile files, the system configuration interactive tool runs and queries for required data on first boot of the zone. You would need to manually provide the information as prompted.

The following example adds the `/tmp/zmanifest.xml` AI manifest to the `solaris11_3-sparc` install service and specifies that `zone1` and `zone2` should use this manifest.

```
# installadm create-manifest -n solaris11_3-sparc -f /tmp/zmanifest.xml \
-m zmanifest -c zonename="zone1 zone2"
```

The following example adds the `/tmp/z1profile.xml` profile to the `solaris11_3-sparc` install service and specifies that `zone1` and `zone2` should use this profile.

```
# installadm create-profile -n solaris11_3-sparc -f /tmp/z1profile.xml \
-p z1profile -c zonename="zone1 zone2"
```

The following example adds the `/tmp/z2profile.xml` profile to the `solaris11_3-sparc` install service and specifies that `zone2` should use this profile.

```
# installadm create-profile -n solaris11_3-sparc -f /tmp/z2profile.xml \
-p z2profile -c zonename=zone2
```

The following example shows the AI manifests and system configuration profiles that have been added to the `solaris11_3-sparc` install service.

```
$ installadm list -n solaris11_3-sparc -m -p

Service Name      Manifest Name   Type    Status  Criteria
------------      -------------   ----    ------  --------
solaris11_3-sparc line1-netra2000 xml     active  mac = 00:14:4F:2D:7A:DC
                  zmanifest       xml     active  zonename = zone1,zone2
                  orig_default    derived default none

Service Name      Profile Name Environment Criteria
------------      ------------ ----------- --------
solaris11_3-sparc z1profile    system      zonename = zone1,zone2
                  z2profile    system      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. Only one zpool child element can be specified in the `logical` element.

In the `zpool` 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 81 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 Software in Oracle Solaris 11.3*.

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, 2013, 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
                  <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 System Configuration Profiles

You can provide a system 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 system 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 238.

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 84, "Generated SMF Service Manifest," on page 238.

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 82, “Template First-Boot Script,” on page 235.

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 Software in Oracle Solaris 11.3*. 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 Software in Oracle Solaris 11.3*.

- 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 82 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.
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"
```

Tip - Use the -n option to check for syntax errors in your script:

```
$ ksh -n first-boot-script.sh
```

EXAMPLE 83 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 203.0.113.5/27 net0
/usr/sbin/ipadm create-addr -a 203.0.113.35/27 net1

# Add a default route with net0 as the gateway
/usr/sbin/route add default 203.0.113.1 -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"
```

Another good use of a first-boot script is to use the `useradd` command to configure multiple initial users on the system.

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 84 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 234 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 242 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'>
<!--
    Manifest created by svcbundle (2014-Jan-14 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
                    -->
                </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 85 Customized Service Manifest : Increase the Time Allowed for the Script to Run

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 (2014-Jan-14 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
            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
```

EXAMPLE 86 Customized Service Manifest: Ensure the Script Runs After Non-Global Zones Are Installed

In the following service manifest excerpt, the dependency on `svc:/milestone/multi-user` is changed to a dependency on `svc:/system/zones-install` to ensure that the first-boot script runs after all non-global zones are installed.

```
<!--
    The following dependency keeps us from starting until all
    non-global zones are installed.
-->
<dependency restart_on="none" type="service"
    name="ngz_dependency" grouping="require_all">
    <service_fmri value="svc:/system/zones-install"/>
</dependency>
```

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 237.
- The first-boot script from "Creating a Script to Run at First Boot" on page 234.
- Any files needed by the script that cannot be provided from another location such as the AI 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 11.3* 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@1.0,5.11-0:20140114T022508Z
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.

```
# 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      ---
```

```
# 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 -> 1.0,5.11-0:20140114T022508Z
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 Package Repositories in Oracle Solaris 11.3* 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 manifest script to add these elements. See Chapter 10, “Defining AI Client Installation Parameters” 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.

```
# 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: Dec 23, 2013 02:50:31 PM
          Size: 3.89 kB
          FMRI: pkg://firstboot/first-boot-script@1.0,5.11-0:20131223T145031Z
```

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 Dec 23, 2013 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@1.0,5.11-0.1:2013123T231948Z
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 AI Clients Using an AI Server

This chapter provides the system requirements for AI clients and describes how to associate each client with the correct AI install service.

How an AI Client Is Installed

When you set up your AI 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 AI clients as needed. To start the automated installation, you just need to boot the AI client.

After you network boot the client, the installation and configuration of the AI client are completed using a net image, installation specifications, and system configuration specifications provided by the install service.

1. The administrator network boots the AI client.
2. The AI client contacts the DHCP server and retrieves the client's network configuration and the location of the AI server. SPARC clients can optionally use the `network-boot-arguments` variable set in the OBP to get this information.
3. The AI client loads the net image from one of the following sources:
   - The install service assigned to this AI client with the `installadm create-client` command
   - The default install service for this architecture
4. The AI client completes its installation using the AI manifest determined as described in "Selecting the AI Manifest" on page 140.
5. The AI client reboots if `auto_reboot` is set in the AI manifest, or the AI client is rebooted by the system administrator.
6. During reboot, the AI client is configured in one of the following ways:

- Using system configuration profiles determined as described in “Selecting System Configuration Profiles” on page 141
- Using the administrator's responses in the interactive system configuration tool

7. After reboot, any first-boot scripts that were set up for the AI client are run.

When the installation is finished, the message `Automated Installation succeeded` displays on the screen, a completion message displays in the `/system/volatile/install_log` file, and the `svc:/application/auto-installer` SMF service on that AI client reaches the `online` state.

SPARC and x86 AI Client System Requirements

The AI client for automated installation must meet the following requirements. Any system that meets these requirements can be used as an automated AI 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 AI client must be able to access the following resources during the installation:

- A DHCP server that provides network configuration information
- The AI server
- An IPS repository that contains the packages to be installed on the AI client

On SPARC systems, the firmware must be updated to include the current version of the Open Boot PROM (OBP) that contains the latest WAN boot support.

To boot over the network, AI requires WAN boot support for SPARC clients. You can check whether your AI 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 AI 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 AI client cannot be installed over the network. See Chapter 5, “Automated Installations That Boot From Media”.

```
# eeprom | grep network-boot-arguments
```

Setting Up an AI Client

On the AI server, use the `installadm create-client` command to associate a particular AI client with a particular install service.

The `installadm create-client` command requires the following information:

- MAC address for the AI client
- Name of the install service for the AI 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, include the boot arguments you want the AI client to boot with in the boot command that you type when you boot it.

Setting Up a SPARC AI Client

The following example associates the SPARC client with MAC address `00:14:4f:a7:65:70` with the `solaris11_3-sparc` install service.

```
# installadm create-client -n solaris11_3-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 Install Service” on page 97 for more information.

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

```
dr-xr-x---   2 webservd webservd     4 Apr  9 08:53 0100144FA76570
```

Setting Up an x86 AI Client

The following example associates the x86 client with MAC address `0:e0:81:5d:bf:e0` with the `solaris11_3-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 AI client cannot boot the `solaris11_3-i386` install service.

```
# installadm create-client -n solaris11_3-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      : 203.0.113.5
    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 PXE boot entry for this AI client:

```
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 root   21 May  6 10:32 0100E0815DBFE0 -> ./0100E0815DBFE0.bios
lrwxrwxrwx  1 root root   44 May  6 10:32 0100E0815DBFE0.bios -> ./solaris11_3-i386/
boot/grub/pxegrub2
lrwxrwxrwx  1 root root   51 May  6 10:32 17:49 0100E0815DBFE0.uefi -> ./solaris11_3-
i386/boot/grub/grub2netx64.efi
-rw-r--r--  1 root root  1744 May  6 10:32 17:49 grub.cfg.0100E0815DBFE0
-rw-r--r--  1 root root  1204 May  6 10:32 17:49 menu.conf.0100E0815DBFE0
```

Deleting an AI Client From a Service

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

```
$ installadm delete-client -e macaddr
```

You do not need to specify the service name because an AI client can be associated with only one install service.

Installing AI Clients

Boot the AI client to start the installation. This section describes how to boot a SPARC or x86 client. You can monitor the progress of an installation on the console of the AI client. Any errors that may occur during the installation will also be displayed on the client console. 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 AI 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 AI 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 AI clients can also set this option on the boot command line.

Monitoring x86 AI Client Installations

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

```
# 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 11.3 Systems*.

Monitoring SPARC AI 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_3-sparc/system.conf
...install_service=solaris11_3-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 AI client by specifying `livessh=disable` on the boot command line:

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

Installing a SPARC AI Client

Network boot SPARC AI clients from the OBP prompt. Decide whether you are using secure download and whether you are using DHCP.

Installing a SPARC AI Client Using Secure Download

For SPARC AI client that are secured with credentials, the net boot file and the boot file system can be securely downloaded over the network through SPARC OBP firmware configured with security keys. Firmware keys must be specified in OBP to validate the downloaded boot file and file system.

The hashing digest (HMAC) is computed with the SHA1 algorithm, and AES is the encryption method employed.

Setting the Hashing Key and Encryption Key

You can set the HMAC and encryption key at the OBP command prompt.

The following example sets the OBP HMAC on a SPARC client console with the AI-generated SHA1 value:

```
ok set-security-key wanboot-hmac-sha1 767280bd72bca8cef3d679815dfca54638691ec5
```

The following example sets the OBP AES encryption key on a SPARC client console:

```
ok set-security-key wanboot-aes 38114ef74dc409a161099775f437e030
```

Resetting the Hashing Key and Encryption Key

If the OBP keys for an AI client are regenerated in the AI server's configuration, the keys must be updated on the affected SPARC clients to perform authenticated AI installations. To invalidate existing OBP keys and generate new OBP keys, use the `-H` and `-E` options with the `installadm` command. See "OBP Security Keys for SPARC Clients" on page 118 for information about generating OBP keys for server authentication only, for a specific AI client, for a specific install service, and for the default AI client.

Deleting the Hash Key and Encryption Key

When you delete the HMAC key and encryption key, that AI client will no longer require or attempt authentication. You will not be able to use AI to install the client using any install service whose `sec` property is set to either `require-client-auth` or `require-server-auth`.

To delete the HMAC key and encryption key at the OBP command prompt, use the same command that you use to set the keys, but do not provide any values:

```
ok set-security-key wanboot-hmac-sha1
ok set-security-key wanboot-aes
```

Installing a SPARC AI Client Using DHCP

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

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

Installing a SPARC AI Client Without Using DHCP

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 AI 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 AI 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.

SPARC AI Client Network Boot Sequence

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

1. The AI 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 AI client.
3. The WAN boot binary is downloaded using HTTP, and the AI 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 AI client.

▼ How To Set the Boot Disk From OBP

Normally, when you install an AI client using an AI server, you would have selected a disk to install onto in an AI manifest. If there is no definition for a disk to install onto, then the `boot-device` OBP parameter is checked. If the parameter is not set, then the first disk that is big enough is used. To prevent AI from placing the OS on the wrong disk when a disk has not been selected in the manifest, set the `boot-device` OBP parameter.

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

```
# init 0
```

2. **List the devices on the system.**

```
ok devalias
...
disk1                    /pci@306/pci@1/SUNW,qlc@0/fp@0,0/disk@w202400a0b836a3b9,3
disk0                    /pci@306/pci@1/SUNW,qlc@0/fp@0,0/disk@w202400a0b836a3b9,1
disk                     /pci@304/pci@2/usb@0/storage@1/disk@0,0
```

3. **Set the `boot-device` parameter to the appropriate disk.**

```
ok setenv boot-device /pci@306/pci@1/SUNW,qlc@0/fp@0,0/disk@x202400a0b836a3b39,1
```

4. **(Optional) Verify the boot device.**

```
ok printenv boot-device
boot-device = /pci@306/pci@1/SUNW,qlc@0/fp@0,0/disk@x202400a0b836a3b39,1
```

5. **Boot the system to start the AI installation.**

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

Installing an x86 AI Client

Initiate the x86installation 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 AI client boots, select the network device to boot from.

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

1. The AI 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 11.3 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: 203.0.113.29   MASK: 255.255.255.224    DHCP IP:  203.0.113.49
GATEWAY: 203.0.113.1
```

The GRUB menu appears with two menu entries. Select the second entry to start an automated installation:

```
Oracle Solaris 11.3 Text Installer and command line
Oracle Solaris 11.3 Automated Install
```

The default 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 AI client is booted, a menu displays as shown in “Starting an Automated Installation from the Command Line” on page 275. Use this menu to examine or install the system.

Install Options for the `boot` Command

From the boot prompt, you can use the following options when booting an AI client:

install	Starts an automated installation session.
livessh	Enables `ssh` when the miniroot is booted. See “Using Secure Shell to Remotely Monitor Installations” on page 253 for examples.
aimanifest	Specifies a URI to the AI client manifest.
install-env-profile	Specifies a URI to a profile to be used on the miniroot, to set up networking or the name service for example.
profile	Specifies a URI to the AI client profile.

AI Client Installation Messages

The following messages are common to both SPARC and x86 installations.

Automated Installation Started Message

If the AI 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

Reboot to start the installed system
```

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 AI client does not automatically reboot after successful installation.

♦ ♦ ♦ CHAPTER 15

Troubleshooting Automated Installations

This chapter discusses some possible failures and how to recover.

AI Client Installation Fails

This section suggests actions to take if an installation fails.

Check the Installation Logs and Instructions

If an installation to an AI client failed, you can find the log at `/system/volatile/install_log`.

The AI manifest that was used for this AI client is in `/system/volatile/ai.xml`. The system configuration profiles that were used for this AI client are in `/system/volatile/profile/*`.

You can also check the AI server's webserver log in `/var/ai/image-server/logs/*`. This log is useful if the AI client is receiving an unexpected or incorrect AI manifest or system configuration profile, or when the AI client can not download files after the initial boot phase.

Check DNS

Check whether DNS is configured on your AI 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 AI client:

```
# /sbin/dhcpinfo DNSserv
```

If this command returns nothing, the DHCP server is not set up to provide DNS server information to the AI 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 AI Client Boot Errors

Review the following additional information about errors that occur when the AI client is booting.

- "SPARC Network Booting Errors and Possible Causes" on page 263
- "x86 Network Booting Errors and Possible Causes" on page 267
- "SPARC and x86 Error Messages" on page 270

Boot Disk Not Found

If the boot disk is not found during an automated installation, verify the book disk and modify the AI manifest.

1. Select the boot device explicitly in SPARC OBP or in the x86 BIOS.
2. Reboot the system.
3. Log in to the system being installed.
4. Identify the device to be used during the installation. The device is can be identified by the SYS/HDD* receptacle name or the CTD disk name as displayed in the `format` command.
5. Modify the `/system/volatile/ai.xml` manifest and replace the "boot_disk" value. For example:

```
<disk_keyword key="SYS/HDD1" name_type="receptacle"/>


<disk_keyword key="c0t5000CCA012B2A254d0" name_type="ctd"/>
```

6. Refresh the installation service.

```
# svcadm clear auto-installer
```

Related to this issue, when you are installing in UEFI mode, the boot disk cannot be located even if a disk is specified in the AI manifest's target `boot_disk` section. This limitation in UEFI installation means that if nothing is installed on the system yet, you cannot use the `boot_disk` section in the manifest to identify a boot disk. As a workaround, use a derived manifest specify an alternative installation target disk.

X86 System Will Not Boot After Install

The UEFI specification forbids EFI protective MBR partition from being marked as active, but your BIOS implementation requires that at least one hard disk have at least one MBR partition that's marked as bootable/active to boot in BIOS mode. Use the fdisk utility to set the partition's status to be "ACTIVE".

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

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 AI 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 AI 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 AI 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 AI client. Check the DHCP configuration for this AI 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 AI 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
        Mon Aug  5 20:46:43 wanboot alert: miniinfo: Request returned code 500
        Mon 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
      Mon Aug  5 20:53:02 wanboot alert: miniroot: Request returned code 403
      Mon Aug  5 20:53:03 wanboot alert: Forbidden
```

For both of these problems, fix the `boot_archive` file configured for this AI client. Check the pathname and permissions of the `boot_archive` at `$IMAGE/boot/boot_archive`.

`ERROR 403: Forbidden` Or `ERROR 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
Mon May  5 18:57:36 wanboot progress: miniroot: Read 235737 of 235737 kB (100%)
Mon May  5 18:57:36 wanboot info: miniroot: Download complete
SunOS Release 5.11 Version 11.3 64-bit
Copyright (c) 1983, 2015, Oracle and/or its affiliates. All rights reserved.
Remounting root read/write
Probing for device nodes ...
Preparing network image for use
Downloading solaris.zlib
--2015-05-05 18:52:30--  http://203.0.113.67:5555/export/auto_install/11_3_sparc/
solaris.zlib
Connecting to 203.0.113.67:5555... connected.
HTTP request sent, awaiting response... 404 Not Found
2015-05-05 18:52:30 ERROR 404: Not Found.

Could not obtain http://203.0.113.67:5555/export/auto_install/11_3_sparc/solaris.zlib
 from install server
Please verify that the install server is correctly configured and reachable from the AI
 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 Not Started

When installing the Oracle Solaris OS on your AI client 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 an Automated Installation from the Command Line” on page 275 for instructions about how to start an automated installation from this point.

Invalid HMAC Value

If you see the message `Invalid HMAC value` on the SPARC console shortly after booting a SPARC AI client, and the system returns to the `ok` prompt, one of the following conditions caused the problem:

- The AI client is secured by authentication, but you have not set the OBP keys. The solution is to set the OBP keys in the client firmware. For information about authentication, see “Increasing Security for Automated Installations” on page 109. For information about setting OBP keys, see “Installing a SPARC AI Client Using Secure Download” on page 254.
- The AI client is not secured but does have OBP keys set. The solution is to unset the OBP keys in the client firmware. See “Resetting the Hashing Key and Encryption Key” on page 255.
- The AI client's install service has a policy requiring client authentication, but no credentials applicable to the client have been assigned. Be sure that there are credentials available for all AI clients of services with policy `require-client-auth`.

The following steps show how to identify the problem.

1. Verify that security hasn't been disabled for the AI server. Use `installadm list -sv` to see if security is enabled.

2. Verify that security hasn't been disabled for the AI client's install service. Use `installadm list -vn` *svcname* to see if security not disabled.
3. If the AI client is using custom credentials, use `installadm list -ve` *macaddr* to obtain the firmware key values.
4. If the AI client is not a custom client, use `installadm list -vn default-sparc` to see if there are any firmware keys defined for the `default-sparc` service.
5. Check the policy of the AI client's service with `installadm list -vn` *svcname*.
6. If there are no credentials for the `default-sparc` service, look for default client credentials using the `installadm list -sv` command. If there are default client credentials, then use the firmware keys listed for the default AI client.
7. If there are no default client credentials, use `installadm list -vn default-sparc` to see if the service policy is set to `require-server-auth`. If so, use the firmware keys listed for the default AI client in `installadm list -sv`.

x86 Network Booting Errors and Possible Causes

This section describes errors or problems you might see when booting an x86 client over the network and possible causes:

- "No DHCP or Proxy DHCP Offers Were Received" on page 267
- "TFTP Error or System Hangs After GATEWAY Message" on page 268
- "System Hangs After GRUB Menu Entry is Selected" on page 268
- "HTTP Request Sent Results in 403 Forbidden or 404 Not Found" on page 269
- "Automated Installer Not Started" on page 269

No DHCP or Proxy DHCP Offers Were Received

If a DHCP server is not responding to an x86 client's request, you see the following messages:

```
Intel(R) Boot Agent PXE Base Code (PXE-2.1 build 0.86)
   Copyright(C) 1997-2007, Intel Corporation

   CLIENT MAC ADDR 00 14 4F 29 04 12 GUID FF2000008 FFFF FFFF FFFF 7BDA264F1400
   DHCP......... No DHCP or Proxy DHCP offers were received
   PXE-MOF: Exiting Intel Boot Agent
```

The timeout message indicates that the AI 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 AI 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: 203.0.113.67   MASK: 255.255.255.224    DHCP IP:  203.0.113.77
     GATEWAY: 203.0.113.65
     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: 203.0.113.67   MASK: 255.255.255.224    DHCP IP:  203.0.113.77
     GATEWAY: 203.0.113.65
```

System Hangs After GRUB Menu Entry is Selected

If the AI 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 AI server, check whether the `grub.cfg` file or the `menu.lst` file for this AI client is pointing to a valid boot archive. The boot directory of the image on the AI 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_3-i386
                 92052473  36629085  55423388  40% /etc/netboot/default-i386
/export/auto_install/solaris11_3-i386
                 92052473  36629085  55423388  40% /etc/netboot/solaris11_3-i386
```

HTTP Request Sent Results in 403 Forbidden or 404 Not Found

On the AI 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 AI 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 11.3 64-bit
Copyright (c) 1983, 2015, Oracle and/or its affiliates. All rights reserved.
Remounting root read/write
Probing for device nodes ...
Preparing network image for use
Downloading solaris.zlib
--2015-05-05 20:02:26--  http://203.0.113.66:5555/export/auto_install/solaris11_3-i386/
solaris.zlib
Connecting to 203.0.113.66:5555... connected.
HTTP request sent, awaiting response... 404 Not Found
2015-05-05 20:02:26 ERROR 404: Not Found.

Could not obtain http://203.0.113.66:5555/export/auto_install/solaris11_3-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 Not Started

When installing the Oracle Solaris OS on x86systems 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 11.3 Text Installer and command line
Oracle Solaris 11.3 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 an

Automated Installation from the Command Line" on page 275 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

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)
```

IPS Server Not Available

The client needs to reach the IPS package repository defined in the AI manifest in order to install the Oracle Solaris OS. If the AI 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 21 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 21 15:54:47 UTC 2013
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 AI client 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 261.
- If you are using a local repository, check whether you have made the repository accessible to all AI clients. See Chapter 3, "Providing Access To Your Repository" in *Copying and Creating Package Repositories in Oracle Solaris 11.3*.
- 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.

Boot Errors on Secured AI Client

A message similar to the following message when you boot the AI client means the TLS certificate is not yet valid:

```
SSL3_GET_RECORD:wrong version number - secure HTTPS GET REQUEST to unsecured HTTP port
```

The cause of this problem could be that the system time on the AI client precedes the time the certificate was generated. Check the system time on the AI client. See “Increasing Security for Automated Installations” on page 109 for information about how to generate and assign security credentials.

Security-related AI Failures

If you have secured your AI server and AI clients as described in “Increasing Security for Automated Installations” on page 109, and you are experiencing problems booting or installing those AI clients, try the following steps to check for authentication errors:

- Check the Apache `access_log` and `error_log` in `/var/ai/image-server/logs/` on the AI client.
- Log onto the console of the AI client. Examine the `/tmp/install_log` file and the SMF service logs in `/system/volatile/`.
- If authentication fails after the boot archive loads in the AI client, when attempting to get image files, AI manifests, or system configuration profiles, you could have a transient networking interruption. Check that the AI server is functioning correctly, and restart the installation.
- Try using the `openssl s_client` command to test the connection:

  ```
  $ openssl s_client -key client-key -cert client-certificate \
  -CAcert server-CA-certificate -connect AI-server-address:port
  ```

- Use the `installadm list -s -v` command to show the enabled or disabled state of security on the AI server. See Example 42, “Listing AI Server Security Information,” on page 128.
- Check the AI client's service policy with the `installadm list -v -n` *svcname*`S` command.
- Check assigned credentials against the CA certificates. Use the `-K` and `-C` options with the `installadm list` subcommand to list the assigned keys and certificates. Compare those keys and certificates with the expected keys and certificates using a character comparison utility such as `diff`.
- Make sure the passphrase was removed from `/var/ai/ai-webserver/tls.key/server.key` on the AI client. X.509 private key files must have any passphrase removed.
- Try using the `wget` command to fetch a file from an AI image, using the appropriate key, certificate, and CA certification, as shown in the following example:

  ```
  $ wget --private-key=client-key --certificate=client-certificate \
  --ca-certificate=server-CA-certificate \
  http://AI-server-address:5555/path-to-file-in-image
  ```

Booting the Installation Environment Without Starting an Installation

Use one of the following methods to boot the installation environment without starting an automated installation.

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 11.3 Text Installer and command line
Oracle Solaris 11.3 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 11.3 Systems*, and remove the property.

When the AI client is booted, a menu displays, as shown in "Starting an Automated Installation from the Command Line" on page 275. Use this menu to examine or install the system.

Starting an Automated Installation from the Command Line

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 xterm)
5  Reboot

Please enter a number [1]:
```

Select option 3 to open a shell.

Once a shell is running, use the following commands to start an automated installation:

```
$ svcadm enable svc:/application/manifest-locator:default
$ svcadm enable svc:/application/auto-installer:default
```

PART IV

Performing Related Tasks

This section provides information about topics related to the installation process. The following topics are covered:

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

♦ ♦ ♦ 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.

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 OCM server, or to gather information in separate collection sites. See “About the Oracle Configuration Manager Central Collector” on page 280 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 the configCCR section 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 system 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 system.

- 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

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 281
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 282
Manually register your system with the Oracle repository.	Changes your registration credentials.	"How to Manually Register Your System With the Oracle Repository" on page 282
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 283

▼ How to Enable the Oracle Configuration Manager Service

1. **Become an administrator.**

For more information, see "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

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 "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

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 "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

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 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 "Using Your Assigned Administrative Rights" in *Securing Users and Processes in Oracle Solaris 11.3*.

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 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 `https://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 285
- "How to Install Missing Drivers" on page 286
- "How to List Your System in the HCL" on page 288

▼ 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 286.

▼ 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 288.

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 285.

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

B

C

D

E

F

G

H

I

K

L

M

N

O

P

R

S

T

U

V

W

X

Z