Transitioning From Oracle® Solaris 10 JumpStart to Oracle Solaris 11.2 Automated Installer

Part No: E36806
July 2014

Contents

Tables

Using This Documentation

- **Overview** – Describes how to migrate from Oracle Solaris 10 JumpStart to Oracle Solaris 11 Automated Installer
- **Audience** – System administrators
- **Required knowledge** – Some Oracle Solaris experience is useful

Product Documentation Library

Late-breaking information and known issues for this product are included in the documentation library at http://www.oracle.com/pls/topic/lookup?ctx=E36784.

Access to Oracle Support

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

Feedback

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

♦ ♦ ♦ CHAPTER 1

Comparing JumpStart and Automated Installer

This chapter provides the following information:

- Similarities and differences between JumpStart and Automated Installer (AI)
- Information references that can help you migrate from JumpStart to AI

Similarities and Differences Between JumpStart and AI

Both JumpStart and AI provide hands-free installation of multiple systems on a network. Clients are network booted, and once the client is booted, the installer takes over.

JumpStart installs the Oracle Solaris 10 OS and earlier versions of the Oracle Solaris OS. AI installs the Oracle Solaris 11 OS and update releases.

JumpStart and AI share the following characteristics:

- Provide hands-free network installation of multiple clients by storing system configurations on an install server
- Provide for different kinds of installations on different clients in one automated installation
- Install both x86 and SPARC clients

TABLE 1-1 Comparing JumpStart and AI Tasks

Step	JumpStart	AI
Set up an install server.	Use the `setup_install_server` command.	Use the `installadm create-service` command.
Add clients to the installation.	Use the `add_install_client` command.	Use the `installadm create-client` command.
Create derived profiles	Use a begin script.	Use the derived manifests mechanism.
Specify installation instructions.	Use profile files.	Use AI manifest files.
Provision the client.	Use DVDs.	Use a package repository on the Internet or on a local network.

Step	JumpStart	AI
Specify client customizations.	Use rules files to associate clients with profile files.	Use the `create-manifest`, `create-profile`, or `set-criteria` subcommands of the `installadm` command to associate clients with AI manifests and system configuration profiles.
Specify client configuration.	Use finish scripts and `sysidcfg` files.	Use SMF (Oracle Solaris Service Management Facility) system configuration profile files. Use scripts executed by an SMF service that runs once at first boot.

JumpStart to AI Migration Information

The following items describe some of the strategies that you could use when migrating to using the Automated Installer.

- **Convert JumpStart rules, profiles, and configuration files to AI criteria, AI manifests, and SMF system configuration profiles.**

  Get the `js2ai` command:

  ```
  # pkg install install/js2ai
  ```

  Follow the instructions on these pages to convert JumpStart data to AI data:

  - `js2ai`(1M) man page
  - Chapter 2, "Converting JumpStart Rules and Profile Files"
  - Chapter 3, "Converting `sysidcfg` Configuration Files"
- **Use one server as both a JumpStart install server and an AI install server.**

  This configuration allows you to support JumpStart capable systems from a Oracle Solaris 11 server. For more information see Chapter 4, "Installing Oracle Solaris 10 by Using JumpStart on an Oracle Solaris 11 Server".
- **Dynamically derive an AI client provisioning manifest.**

  See "Creating an AI Manifest at Client Installation Time" in "Installing Oracle Solaris 11.2 Systems ". JumpStart begin scripts provide the ability to dynamically manipulate installation parameters that are passed to the installer. AI provides the ability to query client attributes at client installation time and dynamically derive a provisioning manifest customized for that client. Environment variables specify hardware attributes of the client, and most of these attributes are the same as environment variables used with JumpStart begin scripts.
- **Access a software package repository for AI installations.**

  Use an Oracle Solaris 11 package repository on the Internet, such as pkg.oracle.com.

Make a local copy of a package repository: “Copying and Creating Package Repositories in Oracle Solaris 11.2 ”.

- **Provide system configuration instructions.**

  See Chapter 11, “Configuring the Client System,” in “Installing Oracle Solaris 11.2 Systems ” for information about creating SMF profiles.

  See Chapter 12, “Installing and Configuring Zones,” in “Installing Oracle Solaris 11.2 Systems ” for information about creating an AI manifest and SMF profiles to install non-global zones as part of an AI client installation.
- **Create an SMF service that runs once at first boot and executes a user-defined script.**

  See Chapter 13, “Running a Custom Script During First Boot,” in “Installing Oracle Solaris 11.2 Systems ”.
- **Alternatively, create the installation you want by creating a custom ISO image.**

  See “Creating a Custom Oracle Solaris 11.2 Installation Image ” for information about creating a custom installation image.

♦ ♦ ♦ CHAPTER 2

Converting JumpStart Rules and Profile Files

This chapter describes how to use the js2ai command to convert JumpStart rules and profile files to AI criteria files and AI manifests.

About AI Client Criteria and AI Manifests

AI manifests are XML files that specify system provisioning such as disk layout and software packages to install. See Chapter 10, “Provisioning the Client System,” in “Installing Oracle Solaris 11.2 Systems ” and the ai_manifest(4) man page for more information about AI manifests.

AI uses client criteria to specify which client systems should use which AI manifest files to complete their installation. When an AI manifest is added to an AI install service, criteria can be specified on the command line or in a file. See Chapter 9, “Customizing Installations,” in “Installing Oracle Solaris 11.2 Systems ” for more information about specifying AI client criteria.

Comparing Rules Keywords and Criteria Directives

The following table compares JumpStart rules keywords with AI criteria directives. AI uses these criteria to apply the correct AI manifest or system configuration profile to a particular client.

TABLE 2-1 Comparing JumpStart Rules File Keywords and AI Criteria Directives

JumpStart Rules File Keyword	AI Criteria File Directive	Command-Line Example	Criteria File Example
any	Unsupported. For client systems that do not match any selection criteria, the AI install service		

JumpStart Rules File Keyword	AI Criteria File Directive	Command-Line Example	Criteria File Example
	provides a default AI manifest.		
arch	cpu	-c cpu=sparc	<ai_criteria name="cpu"> <value>sparc</value> </ai_criteria>
disksize	Unsupported. AI installs on the target specified in the AI manifest if that disk is at least the minimum required size.		
domainname	Unsupported.		
hostaddress	ipv4	-c ipv4=10.6.68.127	<ai_criteria name="ipv4"> <value>10.6.68.127</value> </ai_criteria>
hostname	Unsupported. To uniquely identify a host in AI, use either the IP address as described in hostaddress, or use the MAC address.	-c mac=0:3:ba:33:9d:b6	<ai_criteria_name="mac"> <value>0:3:ba:33:9d:b6</ value> </ai_criteria>
installed	Unsupported. AI installs on the target specified in the AI manifest if that disk is at least the minimum required size.		
karch	arch	-c arch=i86pc	<ai_criteria name="arch"> <value>i86pc</value> </ai_criteria>
memsize	mem	-c mem=2048	<ai_criteria name="mem"> <value>2048</value> </ai_criteria>
model	platform	-c platform=SUNW,Sun-Fire-T200	<ai_criteria_name="platform"> <value>SUNW,Sun-Fire-T200</ value> </ai_criteria>
network	network Use network, network with a range, or ipv4 with a range.	single network value: -c network="10.0.0.0"	single network value: <ai_criteria name="network"> <value>10.0.0.0</value> </ai_criteria>
	ipv4	ipv4 range: -c ipv4=10.0.0.1-10.0.0.64	ipv4 range: <ai_criteria name="ipv4">

JumpStart Rules File Keyword	AI Criteria File Directive	Command-Line Example	Criteria File Example

			`<range>10.0.0.1 10.0.0.64</range>`
			`</ai_criteria>`

`osname`	Unsupported.		
`probe`	Unsupported.		
`totaldisk`	Unsupported. AI installs on the target specified in the AI manifest if that disk is at least the minimum required size.		

Comparing Profile Keywords and AI Manifest Directives

The following table compares JumpStart profile keywords with AI manifest directives. AI uses XML manifest files to define the client installation. For more information about AI manifests, see Chapter 10, "Provisioning the Client System," in "Installing Oracle Solaris 11.2 Systems " and see the `ai_manifest`(4) man page.

To specify values that are not known until the client installation process has started, such as devices specified as `any`, consider using a derived manifests script. You can use a derived manifests script to specify swap size based on disk size or specify mirroring based on available disks, for example. For information about derived manifests scripts, see "Creating an AI Manifest at Client Installation Time" in "Installing Oracle Solaris 11.2 Systems ".

TABLE 2-2 Comparing JumpStart Profile File Keywords and AI Manifest Directives

JumpStart Profile File Keyword	AI Manifest Directives
`archive_location`	Unsupported. AI does not install from flash archives.
`backup_media`	Unsupported. The `backup_media` keyword is used only with the `upgrade` option of `install_type`. AI does not support the upgrade install type; AI supports only initial install.

`boot_device` *device*
Example:
`boot_device c1t0d0`
	Example:
	`<target>`
	`  <disk whole_disk="true">`
	`    <disk_name name="c1t0d0" name_type="ctd"/>`
	`  </disk>`
	`  <logical nodump="true" noswap="false"/>`
	`</target>`

JumpStart Profile File Keyword	AI Manifest Directives
`boot_device` *device* *eeprom* Example: `boot_device c0t0d0s0 update`	The value for the *eeprom* keyword (`update` for SPARC systems and `preserve` for x86 systems) is not supported in AI. In AI, the EEPROM on SPARC systems is always updated to the specified target device, so that the installed system automatically boots from that device. On x86 systems, the firmware is never updated.
`bootenv`	Unsupported. AI creates a boot environment and installs the Oracle Solaris 11 OS into that boot environment.
`client_arch`	Unsupported. The client architecture to be installed is defined by the AI install service, not in the AI manifest.
`client_root`	Unsupported. You can specify the type, vendor, and size of the target disk. You cannot specify the amount of root space on the target disk. See the `ai_manifest`(4) man page.
`client_swap`	Unsupported. By default, AI creates a swap volume in the root pool. You can specify a swap slice. See the `ai_manifest`(4) man page.
`cluster` *cluster-name*	Unsupported. See "Converting Software Packages" on page 23 for information about how to handle `cluster` and `package` specifications. The Oracle Solaris 11 OS uses group packages as defined in the `pkg`(5) man page. Group packages are specified just as any other package is specified in the manifest. The default AI manifest includes the packages needed for a standard Oracle Solaris 11 installation. You can customize this list of packages.
`cluster` *cluster-name* `delete`	Unsupported. The `delete` switch is used only with the `upgrade` option of `install_type`. AI does not support the upgrade install type; AI supports only initial install.
`dontuse`	Unsupported.

`fdisk` *disk-name* *type* *size*
Example:
`fdisk c0t3d0 solaris all`
	Example:
	`<target>`
	`  <disk>`
	`    <disk_name name="c0t3d0" name_type="ctd"/>`
	`    <partition action="create" name="1" part_type=`
	`"191"/>`
	`  </disk>`
	`  <logical nodump="true" noswap="false"/>`
	`</target>`
	For a full list of the disk and partition attributes supported by AI, see the `ai_manifest`(4) man page.
	Translation by `js2ai`:
	For `js2ai` translation, the value of *disk-name* must be a device. A device of `all` is not supported. The `fdisk` *type* must be `solaris`. A size of 0 or `delete` is not supported. If partitioning is `default` and

JumpStart Profile File Keyword	AI Manifest Directives
	the `rootdisk` has not been set, `js2ai` sets the first `fdisk` `solaris` partition encountered as the root disk.
`filesys`	UFS file systems are not supported. AI installs ZFS file systems. Translation by `js2ai`: If there is no other way to determine the device to be used for root, the device from the `filesys` line with the `/` mount point is used for the root pool. The local and mirrored file systems are supported when the mount point specified is `/` or `swap`. If the mount point is not `/` or `swap`, the line is logged and then ignored. JumpStart *fsoptions* are not supported. No validation of the size is performed. You might need to adjust the size specified in the resulting AI manifest to achieve a successful installation with this manifest.

Example:
`filesys c1t0d0s0 10000 /`
	The following partial AI manifest is for x86 platforms.
	`<target>`
	`  <disk>`
	`    <disk_name name="c1t0d0" name_type="ctd"/>`
	`    <partition action="create" name="1" part_type=`
	`"191">`
	`      <slice action="create" force="true"`
	`       in_vdev="rpool_vdev" in_zpool="rpool" name="0">`
	`        <size val="10000mb"/>`
	`      </slice>`
	`    </partition>`
	`  </disk>`
	`  <logical nodump="true" noswap="false">`
	`    <zpool is_root="true" name="rpool">`
	`      <vdev name="rpool_vdev" redundancy="none"/>`
	`    </zpool>`
	`  </logical>`
	`</target>`
	Translation by `js2ai`:
	The `js2ai` command only supports translations of the root file system (`/`) and swap.

Example:
`filesys mirror:rpool c6t0d0s0 c6t1d0s0 60048 /`
	The following partial AI manifest is for x86 platforms. For SPARC platforms, no `partition` element is specified.
	`<target>`
	`  <disk>`
	`    <disk_name name="c6t1d0" name_type="ctd"/>`
	`    <partition action="create" name="1" part_type=`
	`"191">`
	`      <slice action="create" force="true"`
	`       in_vdev="rpool_vdev" in_zpool="rpool" name="0">`
	`        <size val="60048mb"/>`
	`      </slice>`
	`    </partition>`

JumpStart Profile File Keyword	AI Manifest Directives

	`</disk>`
	`<disk>`
	`<disk_name name="c6t0d0" name_type="ctd"/>`
	`<partition action="create" name="1" part_type=`
	`"191">`
	`<slice action="create" force="true"`
	`in_vdev="rpool_vdev" in_zpool="rpool" name="0">`
	`<size val="60048mb"/>`
	`</slice>`
	`</partition>`
	`</disk>`
	`<logical nodump="true" noswap="false">`
	`<zpool is_root="true" name="rpool">`
	`<vdev name="rpool_vdev" redundancy="mirror"/>`
	`</zpool>`
	`</logical>`
	`</target>`

`filesys` *server*:*path*	Unsupported. AI does not support remote file systems.
`forced_deployment`	Unsupported. AI does not install from flash archives.
`geo`	Unsupported. In AI, geographic regions for language support are specified in the system configuration profile. See Chapter 11, “Configuring the Client System,” in “Installing Oracle Solaris 11.2 Systems ”.
`install_type`	AI supports only `initial_install`.
`layout_constraint`	Unsupported. AI supports only `initial_install`.
`local_customization`	Unsupported. AI does not install from flash archives.
`locale`	Unsupported. In AI, locale support is specified in the system configuration profile. See Chapter 11, “Configuring the Client System,” in “Installing Oracle Solaris 11.2 Systems ”.
`metadb`	Unsupported.
`no_content_check`	Unsupported. AI does not install from flash archives.
`no_master_check`	Unsupported. AI does not install from flash archives.
`num_clients`	Unsupported.

`package` *package*
Example:
`package SUNWpampkcs11 add`
	Example:
	`<software type="IPS">`
	`<software_data action="install">`
	`<name>pkg:/SUNWpampkcs11</name>`
	`</software_data>`
	`</software>`
	See “Converting Software Packages” on page 23 for information about how to verify the package name is valid.

JumpStart Profile File Keyword	AI Manifest Directives
`package` *package* `add nfs` *server*`:`*path*	In an AI manifest, all software is retrieved from IPS (Image Packaging System) package repositories. IPS repository locations can be HTTP or file. You can specify multiple IPS repositories as software sources. See the `ai_manifest(4)` man page for information about how to specify publishers. Translation by `js2ai`: The `js2ai` command ignores the *retrieval-type* `location` argument.

`package` *package* `delete`	Example:
	`<software type="IPS">`
	`  <software_data action="uninstall">`
	`    <name>pkg:/SUNWpampkcs11</name>`
	`  </software_data>`
	`</software>`
	Because AI only supports initial installations, you should not need to use an AI manifest to delete packages.

`partitioning default`
`partitioning explicit`
	If no target installation device is specified, AI determines a default installation device and a default partitioning. To specify custom partitioning, see the `ai_manifest(4)` man page.
	Translation by `js2ai`:
	For `js2ai`, partitioning must be `default` or `explicit`. For `explicit`, only `swap` and `/` are supported

`partitioning existing`	Unsupported. For `js2ai`, partitioning must be `default` or `explicit`.
`patch`	Unsupported. AI supports only `initial_install`. To update your system or specific packages, use the `pkg update` command on the installed system.

`pool` *newpool* `auto auto auto` *device*	Example:
	`<target>`
	`  <disk>`
	`    <disk_name name="c1t0d0" name_type="ctd"/>`
	`    <partition action="create" name="1" part_type=`
	`"191">`
	`      <slice action="create" force="true"`
	`       in_vdev="rpool_vdev" in_zpool="newpool" name=`
	`"0"/>`
	`    </partition>`
	`  </disk>`
	`  <logical nodump="false" noswap="false">`
	`    <zpool is_root="true" name="newpool">`
	`      <vdev name="rpool_vdev" redundancy="none"/>`
	`    </zpool>`
	`  </logical>`

<table>
<tr><th>JumpStart Profile File Keyword</th><th>AI Manifest Directives</th></tr>
<tr><td></td><td><pre>
</target>
</pre>
<p>Translation by <code>js2ai</code>:</p>
<p>If a pool is specified in a profile, <code>js2ai</code> creates the ZFS root pool using the specified devices. The <code>pool</code> keyword supersedes all other keywords when <code>js2ai</code> determines which devices to use for the ZFS root pool.</p>
<p>The <code>js2ai</code> command does not perform any validation of the pool size, swap size, or dump size. You might need to adjust these sizes in the resulting AI manifest to achieve a successful installation with this manifest.</p></td></tr>
<tr><td><code>pool</code> <em>newpool</em> <code>auto auto auto any</code></td><td>For <code>js2ai</code>, if you specify <code>any</code> instead of a physical device name, you must provide device information prior to the specification that includes the <code>any</code> parameter. For example, you could provide a <code>root_device</code> or <code>usedisk</code> specification before this <code>pool</code> specification. See Example 2-6 for an example.</td></tr>
<tr><td><code>root_device c1t0d0s0</code></td><td><p>Example:</p>
<pre>
<target>
  <disk>
    <disk_name name="c1t0d0" name_type="ctd"/>
    <partition action="create" name="1" part_type=
"191">
      <slice action="create" force="true"
       in_vdev="rpool_vdev" in_zpool="rpool" name="0"/>
    </partition>
  </disk>
  <logical nodump="true" noswap="false">
    <zpool is_root="true" name="rpool">
      <vdev name="rpool_vdev" redundancy="none"/>
    </zpool>
  </logical>
</target>
</pre>
<p>Translation by <code>js2ai</code>:</p>
<p>When <code>root_device</code> is specified, <code>js2ai</code> sets the <code>rootdisk</code> to the specified device.</p></td></tr>
<tr><td><code>system_type</code></td><td><p>AI manifests do not differentiate system types.</p>
<p>Translation by <code>js2ai</code>:</p>
<p>Only the value <code>standalone</code> is supported.</p></td></tr>
<tr><td><code>usedisk</code></td><td><p>Translation by <code>js2ai</code>:</p>
<p>The <code>js2ai</code> command might use the specified device or devices to resolve subsequent <code>any</code> or <code>rootdisk</code> specifications. Devices specified that are not used for this purpose are added to the ZFS root pool by <code>js2ai</code> when that pool is not mirrored.</p></td></tr>
</table>

Using js2ai To Convert JumpStart Rules and Profiles to AI Criteria and Manifests

Use the `js2ai` command with the `-r` option to convert both JumpStart rules and their associated profiles to AI criteria and manifests. Initially, use the `-S` option to skip validation. This command performs a conversion operation on the `rules` file and the profiles referenced by the `rules` file. Each profile referenced in the `rules` file is processed against the AI client provisioning manifest, `/usr/share/auto_install/manifest/default.xml`. This step creates a directory named `AI_`*profile-name* for each profile specified in the JumpStart `rules` file. The `AI_`*profile-name* directory contains an AI criteria file in the form `criteria-`*rule-number*`.xml` that corresponds to the rule that referenced this profile. The `AI_`*profile-name* directory also contains AI manifest files in the form *profile-name*`.`*arch*`.xml` that correspond to the *profile-name* profile file.

When you receive a message that the conversion completed successfully, run the `js2ai` command without the `-S` option to validate the output AI manifests. Validation errors must be corrected in the AI manifest files.

If you do not see a message that the conversion was successfully completed, examine the error report and the `js2ai.log` file. The error report and the log file report warnings, process errors, unsupported items, conversion errors, and validation errors. The error report is a table output to `stdout` that shows the number of each type of error that was encountered in converting the `rules` and profile files. The log file describes the problems.

1. Correct any process errors.
2. Remove any lines from the `rules` and profile files that are listed as unsupported items.
3. Examine the conversion errors and correct the errors if possible. Otherwise, remove the lines that are causing the errors.
4. Examine any warning messages and make sure no corrections are necessary.

Note the following options for the `js2ai` command:

- To display more information about a rule or profile file conversion, use the `-v` option. For more information, see “Displaying Additional Profile Conversion Information” on page 34.
- To validate a specific output AI manifest against the appropriate AI DTD, use the `-V` option. For more information, see “Validating an Output AI Manifest” on page 34.
- To convert one JumpStart profile file only, use the `-p` option. See the examples in “Fixing AI Manifest Files” on page 29.

How `js2ai` Converts JumpStart Device Specifications

This section describes how `js2ai` determines some target elements for the AI manifest from the JumpStart profile specifications.

How the System's Root Disk Is Determined

Because `js2ai` does not have access to the client system profile references during the profile translation process, `js2ai` attempts to determine the root disk during translation using a process that matches JumpStart as much as possible.

The `js2ai` command performs the following steps to determine what device to use for the root disk:

1. If the `root_device` keyword is specified in the profile, `js2ai` sets `rootdisk` to the device on which the slice resides.
2. If `rootdisk` is not set and the `boot_device` keyword is specified in the profile, `js2ai` sets `rootdisk` to the boot device.
3. If `rootdisk` is not set, `partitioning default` is specified, and a `solaris` `fdisk` entry is encountered, `js2ai` sets `rootdisk` to the specified disk name.
4. If `rootdisk` is not set and a `filesys` *cwtxdysz* *size* `/` entry is specified in the profile, `js2ai` sets `rootdisk` to the *cwtxdysz* disk specified in the entry.
5. If `rootdisk` is not set and a `usedisk` *disk-name* entry is specified in the profile, `js2ai` sets `rootdisk` to the *disk-name* disk specified in the entry.
6. If `rootdisk` is not set and the following specification is encountered in the profile where *size* is not 0 or `delete` and *disk-name* is not `all`, then `rootdisk` is set to this disk name.

   `fdisk` *disk-name* `solaris` *size*
7. If `rootdisk` is not set, any occurrence where the device is specified as `rootdisk` generates a conversion error.

How the `any` Device Is Translated

The `js2ai` command performs the following steps to determine what device to use when the `any` keyword is specified:

1. If the `any` device is specified and the keyword action specified (non-mirrored pool, or `filesys` with a `/` mount point), the `any` device is set to `rootdisk` if `rootdisk` is set.

2. If the any device has not been translated and a usedisk statement exists in the profile, the any device is set to the device specified by the usedisk statement.
3. If the any device has not been translated and the action where the any device is specified causes the ZFS root pool to be created, AI chooses the device, unless a mirrored pool is specified.

How the ZFS Root Pool Is Determined

The js2ai command performs the following steps to determine which device to use for the ZFS root pool. Once the ZFS root pool is determined, subsequent definitions encountered are flagged as errors if they conflict with the ZFS root pool that has already been determined.

1. If the profile specifies the pool keyword, js2ai sets the ZFS root pool to the devices specified by the pool keyword.
2. If the ZFS root pool has not been determined and the profile specifies a file system with a mount point of /, the ZFS root pool is created using the devices specified.
3. If the ZFS root pool has not been determined, all keywords in the profile have been processed, and rootdisk is set, the ZFS root pool is created using the rootdisk device.
4. If the ZFS root pool has not been determined and the partition type is default, AI chooses the device to use for the ZFS root pool.
5. If the ZFS root pool has not been determined and no errors have occurred during processing, AI chooses the device to use for the ZFS root pool.
6. If the ZFS root pool is not a mirrored pool and one or more usedisk devices that were specified have not been used for a rootdisk or any device translation, those disks are added to the ZFS root pool.

Converting Software Packages

JumpStart profiles use the cluster and package keywords to install software on the system. The cluster keyword is not supported by AI. Because in IPS the syntax to install an incorporation or group package is the same as the syntax to install any other package, if you simply change cluster to package in the JumpStart profile, the js2ai command creates the correct package installation specification in the AI manifest.

Tip - Verify the package names in the AI manifests. If a package specified for installation in an AI manifest is not available from any publisher origin specified in that AI manifest, then that client installation fails.

IPS package names are different from SVR4 package names. For example, the SVR4 package SUNWpampkcs11 is renamed to library/security/pam/module/pam-pkcs11 in IPS.

If an SVR4 package name exists in IPS, you can install the IPS package by using the SVR4 name. For example, if an AI manifest specifies installation of the SUNWpampkcs11 package, the library/security/pam/module/pam-pkcs11 package is automatically installed. In these cases, the package has been renamed.

If an SVR4 package name does not exist in IPS, you must change the package name or delete that specification from the AI manifest. For example, the SUNWCall and SUNWCuser packages have not been renamed in IPS. If the AI manifest specifies those packages, the installation fails.

Use the pkg list command on an Oracle Solaris 11 system to determine whether a particular package name can be used in your AI manifest. Be sure to use the -g option to list packages from an IPS package repository origin that is specified in the AI manifest.

The js2ai command uses the /usr/share/auto_install/manifest/default.xml AI manifest as a base to build a new AI manifest that includes specifications from the JumpStart profile file. This default AI manifest specifies installation of two packages that install the base operating system: entire and solaris-large-server. In addition to those two packages, you probably need to specify only the installation of additional tools and applications.

EXAMPLE 2-1 Determining Whether a Package Name Can Be Used

In this example, the AI manifest specifies the http://pkg.oracle.com/solaris/release repository origin.

```
$ pkg list -af -g http://pkg.oracle.com/solaris/release SUNWCall SUNWCuser
pkg list: no packages matching 'SUNWCuser, SUNWCall' known
```

This message confirms that these two packages cannot be used in this AI manifest.

EXAMPLE 2-2 Working With a Renamed Package

The pkg list command can also be used to identify packages that have been renamed.

```
$ pkg list -af -g http://pkg.oracle.com/solaris/release SUNWpampkcs11
NAME (PUBLISHER)                                        VERSION             IFO
SUNWpampkcs11                                           0.6.0-0.133         --r
```

The “r” in the last column indicates that this package is renamed. You can use this name in the AI manifest but you might want to use the pkg info command to determine the new name of the package.

See the “Renamed to” line in the following output. The SUNWpampkcs11 package has been renamed to library/security/pam/module/pam-pkcs11. You might want to specify library/security/pam/module/pam-pkcs11 in your AI manifest for greater compatibility with future Oracle Solaris updates.

```
$ pkg info -r SUNWpampkcs11
          Name: SUNWpampkcs11
```

```
       Summary:
         State: Not installed (Renamed)
    Renamed to: library/security/pam/module/pam-pkcs11@0.6.0-0.133
                consolidation/sfw/sfw-incorporation
     Publisher: solaris
       Version: 0.6.0
 Build Release: 5.11
        Branch: 0.133
Packaging Date: Wed Oct 27 18:50:11 2010
          Size: 0.00 B
          FMRI: pkg://solaris/SUNWpampkcs11@0.6.0,5.11-0.133:20101027T185011Z
$ pkg info -r pam-pkcs11
          Name: library/security/pam/module/pam-pkcs11
       Summary: The OpenSC PKCS#11 PAM Login Tools
      Category: System/Security
         State: Not installed
     Publisher: solaris
       Version: 0.6.0
 Build Release: 5.11
        Branch: 0.175.2.0.0.27.0
Packaging Date: Mon Nov 11 17:47:35 2013
          Size: 1.74 MB
          FMRI: pkg://solaris/library/security/pam/module/pam-pkcs11@0...
```

EXAMPLE 2-3 Working With an Obsolete Package

You might want to replace SUNWmysql with database/mysql-51 in your AI manifest.

```
$ pkg list -af -g http://pkg.oracle.com/solaris/release SUNWmysql
NAME (PUBLISHER)                                        VERSION             IFO
SUNWmysql                                               4.0.24-0.142        --o
```

The “o” in the last column indicates that this package is obsolete. This package name cannot be used in an AI manifest. Use the pkg list command with wildcards or the pkg search command to determine whether the package is available under a different name that can be used.

```
$ pkg list -af SUNWmysql*
NAME (PUBLISHER)                                        VERSION             IFO
SUNWmysql                                               4.0.24-0.142        --o
SUNWmysql-base                                          0.5.11-0.133        --r
SUNWmysql-python                                        0.5.11-0.162        --o
SUNWmysql-python26                                      0.5.11-0.133        --r
SUNWmysql5                                              5.0.86-0.171        --o
SUNWmysql5                                              5.0.86-0.133        --r
SUNWmysql51                                             5.1.37-0.133        --r
SUNWmysql51lib                                          5.1.37-0.133        --r
SUNWmysql51test                                         5.1.37-0.133        --r
SUNWmysql5jdbc                                          5.1.5-0.171         --o
SUNWmysql5jdbc                                          5.1.5-0.133         --r
SUNWmysql5test                                          5.0.86-0.171        --o
SUNWmysql5test                                          5.0.86-0.133        --r
SUNWmysqlt                                              4.0.24-0.142        --o
$ pkg info -r SUNWmysql51
```

```
          Name: SUNWmysql51
       Summary:
         State: Not installed (Renamed)
    Renamed to: database/mysql-51@5.1.37-0.133
                consolidation/sfw/sfw-incorporation
     Publisher: solaris
       Version: 5.1.37
 Build Release: 5.11
        Branch: 0.133
Packaging Date: Wed Oct 27 18:49:18 2010
          Size: 0.00 B
          FMRI: pkg://solaris/SUNWmysql51@5.1.37,5.11-0.133:20101027T184918Z
```

You might want to replace SUNWmysql with database/mysql-51 in your AI manifest.

Rules and Profiles Conversion Using js2ai

This section shows using a single js2ai command to convert a JumpStart rules file and all the profile files referenced by that rules file. By default, each converted profile is output to a directory named AI_*profile-filename*. The rule to select that profile is output to that same directory as an AI criteria file.

Sample JumpStart Rules File and Profile Files

The following example uses the following rules file:

```
# The following rule matches only one system:

hostname sample_host    - fdisk.profile         -

# The following rule matches only one system:

hostaddress 10.6.68.127 - mirrorfilesys.profile -

# The following rule matches any system that is on the 924.222.43.0 network:

network 924.222.43.0    - rootdisk.profile      -

# The following rule matches all x86 systems:

arch i386               - mirrorpool.profile    -
```

The fdisk.profile file has the following content:

```
install_type initial_install
system_type server
root_device c1t0d0s0
usedisk c1t0d0
fdisk rootdisk solaris all
partitioning explicit
filesys rootdisk.s1 5000 swap
filesys rootdisk.s0 10000 /
cluster SUNWCall
```

The mirrorfilesys.profile file has the following content:

```
install_type initial_install
partitioning default
filesys mirror c6t0d0s0 c6t1d0s0 60048 /
cluster SUNWCuser
```

The mirrorpool.profile file has the following content:

```
install_type initial_install
partitioning default
pool newpool auto auto auto mirror any any
cluster SUNWCuser
```

The rootdisk.profile file has the following content:

```
install_type initial_install
partitioning explicit
filesys rootdisk.s0 15000 /
filesys rootdisk.s1 1000 swap
cluster SUNWCall
```

Using js2ai to Process a rules File With Associated Profiles

Use the following command to process this rules file. In the error report, validation errors are shown as a hyphen character because validation was not done. Validation is suppressed by the -S option.

```
# js2ai -rS
                                Process  Unsupported  Conversion  Validation
Name                  Warnings  Errors   Items        Errors      Errors
-------------------   --------  -------  -----------  ----------  ----------
rules                        0        0            1           0           -
fdisk.profile                0        0            2           0           -
mirrorfilesys.profile        0        0            2           0           -
mirrorpool.profile           0        0            1           1           -
rootdisk.profile             0        0            1           2           -

Conversion completed. One or more failures and/or warnings occurred.
For errors see js2ai.log
# cat js2ai.log
rules:line 3:UNSUPPORTED: unsupported keyword: hostname
fdisk.profile:line 2:UNSUPPORTED: unsupported value for 'system_type' specified: server
fdisk.profile:line 9:UNSUPPORTED: unsupported keyword: cluster
mirrorfilesys.profile:line 3:UNSUPPORTED: unsupported mount point of 'unnamed' specified,
mount points other than '/' and 'swap' are not supported
mirrorfilesys.profile:line 4:UNSUPPORTED: unsupported keyword: cluster
mirrorpool.profile:line 3:CONVERSION: unable to convert 'any' device to physical device. Replace
 'any'
with actual device name
mirrorpool.profile:line 4:UNSUPPORTED: unsupported keyword: cluster
```

```
rootdisk.profile:line 3:CONVERSION: unable to convert 'rootdisk.s0'.  Replace'rootdisk.' with actual
device name
rootdisk.profile:line 4:CONVERSION: unable to convert 'rootdisk.s1'.  Replace'rootdisk.' with actual
device name
rootdisk.profile:line 5:UNSUPPORTED: unsupported keyword: cluster
```

Output is stored in directories named `AI_`*profile-filename*. AI criteria files created from the JumpStart rules are named for the position of the rule in the `rules` file. AI manifests are named *profile-filename*`.`*arch*`.xml`, where *arch* is `generic`, `x86`, or `sparc`.

```
# ls AI_*
AI_fdisk.profile:
fdisk.profile.x86.xml

AI_mirrorfilesys.profile:
criteria-2.xml      mirrorfilesys.profile.generic.xml

AI_mirrorpool.profile:
criteria-2.xml      mirrorpool.profile.generic.xml

AI_rootdisk.profile:
criteria-3.xml      rootdisk.profile.generic.xml
```

Replacing the hostname Keyword

The log file reports that the JumpStart rule keyword `hostname` is not supported as an AI criteria keyword for selecting an AI manifest. The `hostname` keyword specifies which clients should use the `fdisk.profile` profile. Because `hostname` is not a supported keyword for selecting AI manifests, the new `AI_fdisk.profile` directory does not contain an AI criteria file.

You could fix this problem by changing `hostname` to `hostaddress` in the `rules` file.

You could also fix this problem by creating an AI criteria file that specifies a MAC address or IP address to identify the `hostname` system. For example, the following criteria file is equivalent to the JumpStart rule `hostname sample_host` if `0:14:4F:20:53:97` is the MAC address of `sample_host`:

```
<?xml version="1.0" encoding="utf-8"?>
<ai_criteria_manifest>
  <ai_criteria name="mac">
    <value>0:14:4F:20:53:97</value>
  </ai_criteria>
</ai_criteria_manifest>
```

To find the MAC address of a system, use the `dladm` command as described in the `dladm`(1M) man page.

For the JumpStart rule `hostaddress 10.6.68.127`, the `js2ai` command automatically created the AI criteria file `AI_mirrorfilesys.profile/criteria-2.xml`, replacing the JumpStart `hostaddress` keyword with the AI `ipv4` keyword:

```
<?xml version="1.0" encoding="utf-8"?>
<ai_criteria_manifest>
  <ai_criteria name="ipv4">
    <value>
      10.6.68.127
    </value>
  </ai_criteria>
</ai_criteria_manifest>
```

For the JumpStart rule `network 924.222.43.0`, the `js2ai` command automatically created the AI criteria file `AI_rootdisk.profile/criteria-3.xml`, specifying a range of IP addresses based on the given network address:

```
<?xml version="1.0" encoding="utf-8"?>
<ai_criteria_manifest>
  <ai_criteria name="ipv4">
    <range>
      924.222.43.0 924.222.43.255
    </range>
  </ai_criteria>
</ai_criteria_manifest>
```

For the JumpStart rule `arch i386`, the `js2ai` command automatically created the AI criteria file `AI_mirrorpool.profile/criteria-4.xml`, replacing the JumpStart `arch` keyword with the AI `cpu` keyword:

```
<?xml version="1.0" encoding="utf-8"?>
<ai_criteria_manifest>
  <ai_criteria name="cpu">
    <value>
      i386
    </value>
  </ai_criteria>
</ai_criteria_manifest>
```

Fixing AI Manifest Files

The `js2ai` command often creates an AI manifest for each JumpStart profile even though errors are reported. This section describes how to address some common errors so that the output AI manifests are more complete.

EXAMPLE 2-4 Fixing `fdisk.profile` Errors

The `js2ai` command showed the following errors for the `fdisk.profile` JumpStart profile:

```
fdisk.profile:line 2:UNSUPPORTED: unsupported value for 'system_type' specified: server
fdisk.profile:line 9:UNSUPPORTED: unsupported keyword: cluster
```

These two lines are ignored and do not affect the output AI manifest. You could delete these two lines if you want the conversion to avoid the error messages. The `fdisk.profile` file then has the following content:

```
install_type initial_install
root_device c1t0d0s0
usedisk c1t0d0
fdisk rootdisk solaris all
partitioning explicit
filesys rootdisk.s1 5000 swap
filesys rootdisk.s0 10000 /
```

Use the -p option of the js2ai command to process just this profile.

```
# js2ai -p fdisk.profile
Successfully completed conversion
```

The output AI manifest, AI_fdisk.profile/fdisk.profile.x86.xml, has the following content in the target stanza:

```
<target>
  <disk whole_disk="true">
    <disk_name name="c1t0d0" name_type="ctd"/>
    <partition action="create" name="1" part_type="191">
      <slice name="1" action="create" force="true" is_swap="true">
        <size val="5000mb"/>
      </slice>
      <slice name="0" action="create" force="true" in_zpool="rpool" in_vdev="rpool_vdev">
        <size val="10000mb"/>
      </slice>
     </partition>
  </disk>
  <logical noswap="false" nodump="true" >
    <zpool name="rpool" is_root="true">
      <vdev name="rpool_vdev" redundancy="none"/>
      <filesystem name="export" mountpoint="/export"/>
      <filesystem name="export/home"/>
      <be name="solaris"/>
    </zpool>
  </logical>
</target>
```

The software stanza is the same as in /usr/share/auto_install/manifest/default.xml because this profile does not contain any package specifications.

EXAMPLE 2-5 Fixing mirrorfilesys.profile Errors

The js2ai command showed the following errors for the mirrorfilesys.profile JumpStart profile:

```
mirrorfilesys.profile:line 3:UNSUPPORTED: unsupported mount point of 'unnamed' specified,
mount points other than '/' and 'swap' are not supported
mirrorfilesys.profile:line 4:UNSUPPORTED: unsupported keyword: cluster
```

In a JumpStart profile file, you can omit the *file-system* parameter in the filesys mirror specification. In a js2ai conversion, you cannot omit the *file-system* parameter, which must have one of the following two values: / or swap.

Edit the mirrorfilesys.profile file to add "/" at the end of the filesys specification and to delete the cluster line. The mirrorfilesys.profile file then has the following content:

```
install_type initial_install
partitioning default
filesys mirror c6t0d0s0 c6t1d0s0 60048 /
```

Save the AI_mirrorfilesys.profile/criteria-2.xml file in another location. Then, use the -p option of the js2ai command to process just this profile.

```
# js2ai -p mirrorfilesys.profile
Successfully completed conversion
```

The AI_mirrorfilesys.profile directory contains two output AI manifests: mirrorfilesys.profile.sparc.xml and mirrorfilesys.profile.x86.xml. The only difference between these two AI manifests is that mirrorfilesys.profile.x86.xml contains two partition specifications. The mirrorfilesys.profile.x86.xml manifest has the following content in the target stanza:

```
<target>
  <disk>
    <disk_name name="c6t1d0" name_type="ctd"/>
    <partition action="create" name="1" part_type="191">
      <slice action="create" force="true" in_vdev="rpool_vdev" in_zpool="rpool" name="0">
        <size val="60048mb"/>
      </slice>
    </partition>
  </disk>
  <disk>
    <disk_name name="c6t0d0" name_type="ctd"/>
    <partition action="create" name="1" part_type="191">
      <slice action="create" force="true" in_vdev="rpool_vdev" in_zpool="rpool" name="0">
        <size val="60048mb"/>
      </slice>
    </partition>
  </disk>
  <logical noswap="false" nodump="false" >
    <zpool name="newpool" is_root="true">
      <vdev name="rpool_vdev" redundancy="mirror"/>
      <filesystem name="export" mountpoint="/export"/>
      <filesystem name="export/home">
      <be name="solaris">
    </zpool>
  </logical>
</target>
```

The software stanza is the same as in /usr/share/auto_install/manifest/default.xml because this profile does not contain any package specifications.

EXAMPLE 2-6 Fixing mirrorpool.profile Errors

The js2ai command showed the following errors for the mirrorpool.profile JumpStart profile:

```
mirrorpool.profile:line 4:CONVERSION: unable to convert 'any' device to physical device.
Replace 'any' with actual device name
mirrorpool.profile:line 8:UNSUPPORTED: unsupported keyword: cluster
```

Use either of the following two methods to fix these errors. Both of these fixes result in the same AI manifest output. See also "How the any Device Is Translated" on page 22.

- Edit the mirrorpool.profile profile, replacing both any entries with physical device names. Also, delete the cluster line. The mirrorpool.profile file will then have content similar to the following:

  ```
  install_type initial_install
  partitioning default
  pool newpool auto auto auto mirror c6t0d0s0 c6t1d0s0
  ```

- If you specify any instead of a physical device name in the *vdev-list* list in the pool specification, you must provide device information prior to the specification that includes the any parameter. Edit the mirrorpool.profile profile to add a usedisk specification before the pool specification. Also, delete the cluster line. The mirrorpool.profile file will then have content similar to the following:

  ```
  install_type initial_install
  partitioning default
  usedisk c6t0d0 c6t1d0
  pool newpool auto auto auto mirror any any
  ```

Save the AI_mirrorpool.profile/criteria-4.xml file in another location. Then, use the -p option of the js2ai command to process just this profile.

```
# js2ai -p mirrorpool.profile
Successfully completed conversion
```

The AI_mirrorpool.profile directory contains two output AI manifests: mirrorpool.profile.sparc.xml and mirrorpool.profile.x86.xml. The only difference between these two AI manifests is that mirrorpool.profile.x86.xml contains two partition specifications. The mirrorpool.profile.x86.xml manifest has the following content in the target stanza:

```
<target>
  <disk>
    <disk_name name="c6t1d0" name_type="ctd"/>
    <partition action="create" name="1" part_type="191">
      <slice action="create" force="true" in_vdev="rpool_vdev" in_zpool="newpool" name="0"/>
    </partition>
  </disk>
  <disk>
    <disk_name name="c6t0d0" name_type="ctd"/>
    <partition action="create" name="1" part_type="191">
      <slice action="create" force="true" in_vdev="rpool_vdev" in_zpool="newpool" name="0"/>
    </partition>
  </disk>
  <logical noswap="false" nodump="true">
    <zpool name="rpool" is_root="true">
```

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

The software stanza is the same as in /usr/share/auto_install/manifest/default.xml because this profile does not contain any package specifications.

EXAMPLE 2-7 Fixing rootdisk.profile Errors

The js2ai command showed the following errors for the rootdisk.profile JumpStart profile:

```
rootdisk.profile:line 3:CONVERSION: unable to convert 'rootdisk.s0'.
Replace'rootdisk.' with actual device name
rootdisk.profile:line 4:CONVERSION: unable to convert 'rootdisk.s1'.
Replace'rootdisk.' with actual device name
rootdisk.profile:line 5:UNSUPPORTED: unsupported keyword: cluster
```

Use either of the following two methods to fix these errors. Both of these fixes result in the same AI manifest output.

- Edit the rootdisk.profile profile, replacing rootdisk. with c0t0d0. Also, delete the cluster line. The rootdisk.profile file then has the following content:

  ```
  install_type initial_install
  partitioning explicit
  filesys c0t0d0s0 15000 /
  filesys c0t0d0s1 1000 swap
  ```

- Edit the rootdisk.profile profile to add a root_device specification and to delete the cluster line. The rootdisk.profile file then has the following content:

  ```
  install_type initial_install
  partitioning explicit
  root_device c0t0d0s0
  filesys rootdisk.s0 15000 /
  filesys rootdisk.s1 1000 swap
  ```

Save the AI_rootdisk.profile/criteria-3.xml file in another location. Then use the -p option of the js2ai command to process just this profile.

```
# js2ai -p rootdisk.profile
Successfully completed conversion
```

The AI_rootdisk.profile directory contains two output AI manifests: rootdisk.profile.sparc.xml and rootdisk.profile.x86.xml. The only difference between these two AI manifests is that rootdisk.profile.x86.xml contains a partition specification. The rootdisk.profile.x86.xml manifest has the following content in the target stanza:

```
<target>
  <disk>
    <disk_name name="c0t0d0" name_type="ctd"/>
    <partition action="create" name="1" part_type="191">
      <slice action="create" force="true" in_vdev="rpool_vdev" in_zpool="rpool" name="0">
        <size val="15000mb"/>
      </slice>
      <slice action="create" force="true" is_swap="true" name="1">
        <size val="1000mb"/>
      </slice>
    </partition>
  </disk>
  <logical noswap="false" nodump="true" noswap="false">
    <zpool name="rpool" is_root="true" >
      <vdev name="rpool_vdev" redundancy="none"/>
      <filesystem name="export" mountpoint="/export"/>
      <filesystem name="export/home">
      <be name="solaris">
    </zpool>
  </logical>
</target>
```

The `software` stanza is the same as in `/usr/share/auto_install/manifest/default.xml` because this profile does not contain any package specifications.

Displaying Additional Profile Conversion Information

If you want to see additional information for a rule or profile conversion or AI manifest validation, specify the `-v` option with the `js2ai` command. When you specify the `-v` option, processing steps are displayed, and the error report displays zeroes instead of omitting the `rules` file or profile from the report.

```
# js2ai -v -p rootdisk.profile
Processing profile: rootdisk.profile
Performing conversion on: rootdisk.profile
Generating x86 manifest for: rootdisk.profile
Validating rootdisk.profile.x86.xml
Generating sparc manifest for: rootdisk.profile
Validating rootdisk.profile.sparc.xml

                                   Process  Unsupported  Conversion  Validation
Name                      Warnings  Errors   Items        Errors      Errors
---------------------     --------  -------  -----------  ----------  ----------
rootdisk.profile                 0        0            0           0           0
Successfully completed conversion
```

Validating an Output AI Manifest

Use the `-V` option to validate the AI manifest that results from running the `js2ai` command.

```
# js2ai -V ./AI_rootdisk.profile/rootdisk.profile.sparc.xml
Successfully completed conversion
# js2ai -v -V ./AI_rootdisk.profile/rootdisk.profile.sparc.xml
Validating rootdisk.profile.sparc.xml

                                   Process  Unsupported  Conversion  Validation
Name                     Warnings  Errors   Items        Errors      Errors
----------------------   --------  -------  -----------  ----------  ----------
rootdisk.profile.sparc          -        -            -           -           0
Successfully completed conversion
```

♦ ♦ ♦ CHAPTER 3

Converting sysidcfg Configuration Files

This chapter describes how to convert a sysidcfg file to an AI system configuration profile. Much of the conversion can be done with the js2ai command. See the js2ai(1M) man page for more information about the js2ai command.

AI system configuration profiles are SMF XML profile files that specify system configuration. For more information about AI system configuration profiles, see Chapter 11, "Configuring the Client System," in "Installing Oracle Solaris 11.2 Systems ".

Comparing sysidcfg File Keywords to System Configuration Profile Directives

The following table compares sysidcfg file keywords with example AI system configuration profile specifications. Different SMF property value names might apply in different cases. See the examples in "sysidcfg Conversion Using js2ai" on page 41 and in "Example System Configuration Profiles" in "Installing Oracle Solaris 11.2 Systems ".

TABLE 3-1 Comparing sysidcfg File Keywords and Configuration Profile Directives

sysidcfg File Keyword	System Configuration Profile Directives
keyboard	The js2ai command does not perform any translation. Make sure the keyboard specified in the sysidcfg file is supported in Oracle Solaris 11. Service name: system/keymap Property group name: keymap Property value name: layout
name_service	AI supports DNS, NIS, and LDAP. The js2ai command supports values NONE, DNS, NIS, and LDAP. NIS+ name services are translated as NIS. If a name service is configured, the network must be configured as DefaultFixed. Service name: system/name-service/switch Property group name: config Property value name: netgroup

`sysidcfg` File Keyword	**System Configuration Profile Directives**
`network_interface`	AI supports configuring only a single interface as part of system installation. Because of this limitation, the `js2ai` command processes only the interface labeled `PRIMARY` or the first interface encountered in the `sysidcfg` file. The `js2ai` command sets the network to `DefaultFixed` if a `name_service` is specified. A properly configured `DefaultFixed` network needs to provide the host name, IP address, netmask, and gateway. Automated network configuration is only supported if no name service is specified. Oracle Solaris 11 by default uses the prefix `net` when assigning link names. The older style used in Oracle Solaris 10 is available, but is disabled by default. When the `js2ai` command does a conversion of the network interface and sees a Solaris 10 style link name, the command will disable neutral link names.To use neutral link names in Oracle Solaris 11, you must change the network interface name specified in the sysidcfg file to a Oracle Solaris neutral link name like `net0`. For more information about neutral link names, see “Network Devices and Datalink Naming in Oracle Solaris” in “Configuring and Administering Network Components in Oracle Solaris 11.2 ”. Service name: `network/install` Property group name: `install_ipv4_interface` Property value name: `static_address` Property group name: `install_ipv6_interface`
`nfs4_domain`	`nfs4_domain=dynamic` is supported for Automatic and DefaultFixed networks. `networks.nfs4_domain=`<*custom-domain-name*> is only supported for DefaultFixed networks. The conversion of the `network_interface` keyword determines whether a DefaultFixed or Automatic network is configured. Service name: `network/nfs/mapid` Property group name: `nfs-props` Property value name: `nfsmapid_domain`
`root_password`	Oracle Solaris 11 uses roles instead of the `root` user. You need to define a user with `root` role privileges in order to access the system in multi-user mode. Because the necessary structure can not completely be defined using the `root_password` keyword, you need to add the necessary structures used to define a user account with `root` role privileges to the `sc_profile.xml` file as a comment. If you do not specify the `root_password` keyword, you will also need to define the necessary datastructure for it as a comment. Service name: `system/config-user` Property group name: `root_account` Property value name: `password`
`security_policy`	The `js2ai` command supports the value `NONE`.
`service_profile`	The `js2ai` command supports the value `limited_net`.
`system_locale`	The `js2ai` command will check to ensure that the locale specified is one of the default core locales supported. For more information on core locales, see “Internationalization and Localization Changes” in “Transitioning From Oracle Solaris 10 to Oracle Solaris 11.2 ”. A warning will be generated if the locale specified is not in the core locales. The `js2ai` command will also attempt to convert non-core locales to core locales.

sysidcfg File Keyword	System Configuration Profile Directives
	Service name: system/environment Property group name: environment Property value name: LC_ALL
terminal	The js2ai command does not perform any translation. Make sure the terminal type specified in the sysidcfg file is supported in Oracle Solaris 11. Service name: system/console-login Property group name: ttymon Property value name: terminal_type
timeserver	The js2ai command supports the value localhost.
timezone	The js2ai command does not perform any translation. Service name: system/timezone Property group name: timezone Property value name: localtime

Using js2ai to Convert sysidcfg Files to System Configuration Profiles

Use the js2ai command with the -s option to convert any sysidcfg files that are associated with this JumpStart configuration to system configuration profile files. Initially, use the -S option to skip validation.

For each sysidcfg file processed, js2ai creates an AI system configuration profile file named sc_profile.xml in the directory where the js2ai command was invoked. Use the -D option to specify a different directory for the sc_profile.xml file.

When you receive a message that the conversion completed successfully, run the js2ai command without the -S option to validate the output sc_profile.xml file. Validation errors must be corrected in the sc_profile.xml file.

If you do not see a message that the conversion was successfully completed, examine the error report and the js2ai.log file. The error report and the log file report warnings, process errors, unsupported items, conversion errors, and validation errors. The error report is a table output to stdout that shows the number of each type of error that was encountered in converting the sysidcfg file. The log file describes the problems.

1. Correct any process errors.
2. Remove any lines from the sysidcfg file that are listed as unsupported items.
3. Examine the conversion errors and correct the errors if possible. Otherwise, remove the lines that are causing the errors.

4. Examine any warning messages and make sure no corrections are necessary.

To display more information for a conversion use the -v with the js2ai command. For more information, see "Displaying Additional Configuration Conversion Information" on page 43.

To validate a specific output system configuration profile, run the js2ai command with the -V option. See the example in "Validating an Output System Configuration Profile" on page 43.

```
# js2ai -V path/sc_profile.xml
```

js2ai Conversion Warnings

All js2ai conversions of sysidcfg files will generate the following warning:

```
sysidcfg:line #:WARNING: Oracle Solaris 11 uses roles instead of root
user. An admin user with root role privileges will need to be defined in order
to access the system in multi-user mode. The necessary xml structures have
been added to sc_profile.xml as a comment. Edit sc_profile.xml to perform
the necessary modifications to define the admin user.
```

The warning has to do with the inability of the js2ai command to generate the necessary data structure from just the root_passwd keyword. If your sysidcfg file contained the root_passwd keyword in it, js2ai would generate a system/config-user structure like the following:

```
<service name="system/config-user" type="service" version="1">
  <instance enabled="true" name="default">
    <!--
Configures user account as follows:
* User account name 'jack'
* password 'jack'
* GID 10
* UID 101
* root role
* bash shell
-->
    <!--
<property_group name="user_account" type="application">
  <propval name="login" type="astring" value="jack"/>
  <propval name="password" type="astring" value="9Nd/cwBcNWFZg"/>
  <propval name="description" type="astring" value="default_user"/>
  <propval name="shell" type="astring" value="/usr/bin/bash"/>
  <propval name="gid" type="astring" value="10"/>
  <propval name="uid" type="astring" value="101"/>
  <propval name="type" type="astring" value="normal"/>
  <propval name="roles" type="astring" value="root"/>
  <propval name="profiles" type="astring" value="System Administrator"/>
  <propval name="sudoers" type="astring" value="ALL=(ALL) ALL"/>
</property_group>
-->
    <property_group name="root_account" type="application">
```

```
       <propval name="login" type="astring" value="root"/>
       <propval name="password" type="astring" value="{your_root_passwd}"/>
       <propval name="type" type="astring" value="role"/>
    </property_group>
  </instance>
</service>
```

In XML, a comment is started with `<!--`. To define the user necessary to support Oracle Solaris 11 remove the XML comment markers around the `user_account` property group structure. Next, modify the `user_account` property structure to create the user that will have `root` role privileges. If no `root_password` keyword was specified, comment out the `root_account` property group structure.

sysidcfg Conversion Using js2ai

This section shows how to use `js2ai` to convert a `sysidcfg` file to an AI system configuration profile file. For each `sysidcfg` file processed, `js2ai` creates an AI system configuration profile file named `sc_profile.xml` in the directory where the `js2ai` command was invoked. Use the `-D` option to specify a different directory for the `sc_profile.xml` file. To aid in the conversion process, make a copy of the original `sysidcfg` file and work from a copy. The conversion process might require that you edit and modify the `sysidcfg` file before it can be successfully converted.

EXAMPLE 3-1 Converting a `sysidcfg` File That Includes Errors

This example uses the following `sysidcfg` file:

```
timezone=US/Pacific
timeserver=localhost
keyboard=US-English
system_locale=en_US.UTF-8
terminal=vt100
network_interface=primary { hostname=host1
      ip_address=192.0.2.4
      netmask=255.255.255.224
      protocol_ipv6=yes
      default_route=192.0.2.3 }
name_service=DNS{domain_name=example.com
      name_server=192.0.2.1,192.0.2.2
      search=example.com,example.org}
root_password=rJmv5LUXM1OcU
security_policy=none
nfs_domain=dynamic
```

Use the following command to process this `sysidcfg` file. In the error report, validation errors are shown as a hyphen character because the `-S` option suppressed validation.

```
# js2ai -sS
                                Process  Unsupported  Conversion  Validation
```

```
Name                  Warnings  Errors   Items        Errors      Errors
-------------------   --------  -------  -----------  ----------  ----------
sysidcfg                     1        0            0           1           -

Conversion completed. One or more failures and/or warnings occurred.
For errors see js2ai.log
# cat js2ai.log

sysidcfg:line 6:CONVERSION:  DefaultFixed network configuration enabled.
Unable to complete network  configuration, replace interface PRIMARY
with the actual interface you  wish to configure.
sysidcfg:line 17:WARNING: Oracle Solaris 11 uses roles instead of root user.
An admin user with root role privileges will need to be defined in order to
access the system in multi-user mode. The necessary xml structures have been
added to sc_profile.xml as a comment.  Edit sc_profile.xml to perform the
necessary modifications to define the admin user.
```

EXAMPLE 3-2 Converting a Corrected sysidcfg File

The following modified sysidcfg file addresses the errors reported in the previous example. The PRIMARY interface specification is replaced with the interface name e1000g.

```
timezone=US/Pacific
timeserver=localhost
keyboard=US-English
system_locale=en_US.UTF-8
terminal=vt100
network_interface=e1000g { hostname=host1
      ip_address=192.0.2.4
      netmask=255.255.255.224
      protocol_ipv6=yes
      default_route=192.0.2.3 }
name_service=DNS{domain_name=example.com
      name_server=192.0.2.1,192.0.2.2
      search=example.com,example.org}
root_password=rJmv5LUXM1OcU
security_policy=none
nfs_domain=dynamic

# js2ai -sS

                               Process  Unsupported  Conversion  Validation
Name                 Warnings  Errors   Items        Errors      Errors
-------------------  --------  -------  -----------  ----------  ----------
sysidcfg                    2        0            0           0           -
Conversion completed. One or more failures and/or warnings occurred.
For details see /tmp/js2ai.log
# cat js2ai.log
sysidcfg:line 16:WARNING: Oracle Solaris 11 uses roles instead of root user.
An admin user with root role privileges will need to be defined in order to
access the system in multi-user mode.  The necessary xml structures have been
added to sc_profile.xml as a comment. Edit sc_profile.xml to perform the
necessary modifications to define the admin user.
sysidcfg:line 6:WARNING: In order to support the direct translation of the
sysidcfg interface 'e1000g', Oracle Solaris 11 neutral link name support will
```

```
be disabled.  If you wish to use neutral link names, change the interface
name specified in the sysidcfg file to a 'netx' style interface name or edit
the resulting sc_profile.xml file.
```

Although the network interface primary was replaced with an Oracle Solaris 10 interface name, you will still get a warning about neutral link names. If you want to use the old style link names then no action is required. In most instances the interfaces will map the same as they do in Oracle Solaris 10, but that outcome is not guaranteed.

If you want to use the neutral link names used by Oracle Solaris 11, you can change your network interface in the `sysidcfg` file to a `net#` style name. Typically the primary interface will get defined as `net0`. In order to address the user warning, edit the `sc_profile.xml` file as outlined in the notes for `line 6` in the preceding example.

Displaying Additional Configuration Conversion Information

If you want to see additional information for a `sysidcfg` file conversion or system configuration profile validation, specify the `-v` option with the `js2ai` command. When you specify the `-v` option, processing steps are displayed.

```
# js2ai -sv
Processing: sysidcfg
Performing conversion on: sysidcfg
Generating SC Profile
Validating sc_profile.xml

                              Process  Unsupported  Conversion  Validation
Name                 Warnings  Errors   Items        Errors      Errors
-------------------  --------  -------  -----------  ----------  ----------
sysidcfg                    2        0            0           0           0

Conversion completed. One or more failures and/or warnings occurred.
For details see /tmp/js2ai.log
```

Validating an Output System Configuration Profile

Use the `-V` option to validate the system configuration profile that results from running the `js2ai` command:

```
# js2ai -v -V ./sc_profile.xml
Validating sc_profile.xml

                              Process  Unsupported  Conversion  Validation
Name                 Warnings  Errors   Items        Errors      Errors
-------------------  --------  -------  -----------  ----------  ----------
sc_profile                  -        -            -           -           0
```

```
Successfully completed conversion
```

♦ ♦ ♦ CHAPTER 4

Installing Oracle Solaris 10 by Using JumpStart on an Oracle Solaris 11 Server

You can use JumpStart to install the Oracle Solaris 10 operating system on networked SPARC and x86 platforms, but not to install the Oracle Solaris 11 OS. However, the JumpStart install server can be an Oracle Solaris 11 system.

Your Oracle Solaris 11 server can do two different jobs:

- Serve Oracle Solaris 11 OS installations using Automated Installer. For more information, see Part III, “Installing Using an Install Server,” in “Installing Oracle Solaris 11.2 Systems ”.
- Serve Oracle Solaris 10 OS installations using JumpStart. This chapter describes how to set up a JumpStart install server on an Oracle Solaris 11 system. For more information about JumpStart, see “Oracle Solaris 10 1/13 Installation Guide: Live Upgrade and Upgrade Planning ”.

Setting Up an Oracle Solaris 11 System as an Oracle Solaris 10 JumpStart Server

The following procedure shows how to create an Oracle Solaris 11 JumpStart Install server to install the Oracle Solaris 10 OS on client systems.

▼ How to Set Up an Oracle Solaris 11 System as an Oracle Solaris 10 JumpStart Server

1. **Install the Oracle Solaris 11 OS.**

2. **Set up a static IP address.**

   For information about how to set a static IP address, see the instructions in “How to Configure an IPv4 Interface” in “Configuring and Administering Network Components in Oracle Solaris 11.2 ” and the `ipadm`(1M) man page.

3. **Install the `system/boot/network` package from the `solaris` publisher.**

```
# pkg publisher
PUBLISHER                TYPE     STATUS   URI
solaris                  origin   online   http://pkg.oracle.com/solaris11/release/
# pkg install pkg:/system/boot/network
           Packages to install:  1
     Variants/Facets to change:  3
       Create boot environment: No
Create backup boot environment: No
            Services to change:  1

DOWNLOAD                                  PKGS       FILES    XFER (MB)
Completed                                  1/1       13/13      0.0/0.0

PHASE                                        ACTIONS
Install Phase                                  34/34

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

PHASE                                          ITEMS
Reading Existing Index                           8/8
Indexing Packages                                1/1
```

4. **Download the latest Oracle Solaris 10 DVD image.**

   This image is the source of your install image and JumpStart tools. Recall that JumpStart can be used to install only the Oracle Solaris 10 OS, not the Oracle Solaris 11 OS.

5. **Set up a JumpStart install server.**

   The following example uses the Oracle Solaris 10 1/13 DVD image for SPARC.

```
# /media/SOL_10_0113_SPARC/Solaris_10/Tools/setup_install_server /export/s10u11_sparc
Verifying target directory...
Calculating the required disk space for the Solaris_10 product
Calculating space required for the installation boot image
Copying the CD image to disk...
Copying Install Boot Image hierarchy...
Copying /boot netboot hierarchy...
Install Server setup complete
```

6. **Copy the `jumpstart_sample` files to the JumpStart profile location.**

```
# cd /media/SOL_10_0113_SPARC/Solaris_10/Misc/jumpstart_sample
# mkdir -p /export/profiles/s10profile
# cp -pr * /export/profiles/s10profile
```

7. **Update the `check` script.**

   Replace the first line of the `/export/profile/s10profile/check` script with the following line:

```
#!/usr/sunos/bin/sh
```

8. **Validate your JumpStart profile.**

   In the following example, many lines of output are omitted for brevity. Note that the `-p` option does not work with all Oracle Solaris 10 update releases. Make sure you are using at least Oracle Solaris 10 Update 7.

```
# cd /export/profile/s10profile
# ./check -p /export/s10u11_sparc
45 blocks
Validating rules...
Validating profile host_class...
Validating profile zfsrootsimple...
Validating profile net924_sun4c...
Validating profile upgrade...
Validating profile x86-class...
Validating profile any_machine...
The custom JumpStart configuration is ok.
```

9. **Make sure that `udp6` is available and online.**

   The `udp6` service is delivered in the `tftp` package. Make sure the `tftp` package is installed.

```
# pkg list tftp
```

   If the `tftp` package is not installed, use the following command to install it:

```
# pkg install pkg:/service/network/tftp
```

   Once you have installed the tftp package, make sure the `udp6` service is available and online.

```
# svcs udp6
STATE          STIME    FMRI
disabled       8:36:55 svc:/network/tftp/udp6:default
# svcadm enable network/tftp/udp6
# svcs udp6
STATE          STIME    FMRI
online         8:38:28 svc:/network/tftp/udp6:default
```

10. **Link the `/tftpboot` directory to the `/etc/netboot` directory.**

    If the `/etc/netboot` directory does not exist, create it.

    JumpStart uses `/tftpboot` for network booting. AI uses `/etc/netboot` for network booting. If this Oracle Solaris 11 server might serve as both a JumpStart install server and an AI install server, then `/tftpboot` needs to be a symbolic link to `/etc/netboot` so that the two tools can coexist.

Tip - If you plan to use this server as only a JumpStart install server or only an AI install server, then this step is not required. However, you should do this step anyway to avoid failed netboot errors if you change your plan later.

```
# ls /etc/netboot
/etc/netboot: No such file or directory
# mkdir -m 755 /etc/netboot
# ln -s /etc/netboot /tftpboot
# ls -l /tftpboot
lrwxrwxrwx   1 root     root          12 Sep 14 8:46:51 /tftpboot -> /etc/netboot
```

11. **Install the client.**

    Make sure to use the fully qualified host name for the client. In this example the client host name is `line2-x4100.example.com`.

```
/export/s10u11_sparc/Solaris_10/Tools/add_install_client -e 8:0:20:fd:f2:18 \
-c line2-x4100.example.com:/export/profiles/s10profile \
-p line2-x4100.example.com:/export/profiles/s10profile line2-t1 sun4u
/export/home is already shared. However, the zfs file system
/export/home must be shared read-only with root access. Use the "zfs
set" command to set the sharenfs property for file system /export/home
as follows:Use ro and either anon=0 or
root=line2-x4100.example.com for /export/home. This must be
fixed and /export/home shared before line2-x4100.exampl.com can boot.
 .
 .
```

Because of changes to the `zfs` command, use the following commands to verify how the `/export/home` file system is shared, correct the share options if needed, and then check the how the file system is shared to verify the change. In this example, the anon=0 has not been set.

```
# share
home    /export/home    nfs     sec=sys,ro
share -F nfs -o ro,sec=sys,anon=0 /export/home
# share
home    /export/home    nfs     anon=0,sec=sys,ro
```

Index