Transitioning From Oracle® Solaris 10 JumpStart to Oracle Solaris 11 Automated Installer

Part No: E21799
March 2012, E21799-02

ORACLE®

Contents

Preface

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

This book describes the following procedures and tools:

- The differences between JumpStart and AI
- How JumpStart rule and profile keywords translate to AI criteria and manifest keywords
- How `sysidcfg` files translate to AI system configuration profiles
- How to use the `js2ai` conversion utility
- How to set up one server to serve both Oracle Solaris 10 and Oracle Solaris 11 installations

Who Should Use This Book

This book is for JumpStart users who want to transition to using AI to perform automated installations of the Oracle Solaris 11 operating system (OS).

How This Book Is Organized

This book contains the following chapters:

- Chapter 1, "Comparing JumpStart and Automated Installer," provides a table of installation tasks that tells you how to do each task using JumpStart and using AI.
- Chapter 2, "Converting Rules and Profile Files," provides tables of JumpStart rule and profile keywords and the equivalent AI criteria and manifest directives. This chapter also shows example conversions using the `js2ai` utility.
- Chapter 3, "Converting Configuration Files," provides a table of `sysidcfg` file keywords and the equivalent AI system configuration profile specifications and shows an example conversion using the `js2ai` utility.
- Chapter 4, "Installing Oracle Solaris 10 Using JumpStart on an Oracle Solaris 11 Server," describes how to use an Oracle Solaris 11 server as both a JumpStart install server and an AI install server.

Related Information

Oracle Solaris 10 8/11 Installation Guide: Custom JumpStart and Advanced Installations describes how to use JumpStart.

Installing Oracle Solaris 11 Systems describes different methods for installing the Oracle Solaris 11 OS, including using AI.

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

The `pkg(5)` man page describes the Oracle Solaris Image Packaging System (IPS) feature, which enables you to store and retrieve software packages for installation. The `pkg(1)` man page explains how to install IPS packages.

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

Access to Oracle Support

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

Typographic Conventions

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

TABLE P–1 Typographic Conventions

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

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

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

TABLE P–1 Typographic Conventions *(Continued)*

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

Shell Prompts in Command Examples

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

TABLE P–2 Shell Prompts

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

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.

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 Steps

Step	JumpStart	AI
Set up an install server.	Use the `setup_install_server`(1M) command.	Use the `installadm create-service` command.
Add clients to the installation.	Use the `add_install_client`(1M) command.	Use the `installadm create-client` command.
Create derived profiles	Use a begin script.	Use the derived manifests mechanism.

TABLE 1–1 Comparing JumpStart and AI Steps *(Continued)*

Step	JumpStart	AI
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.
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`(4) files.	Use SMF system configuration profile files. Use scripts executed by an SMF service that runs once at first boot.

JumpStart to AI Migration Information

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

Get the `js2ai` utility:

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

Follow these instructions to convert JumpStart data to AI data:

- `js2ai`(1M)
- Chapter 2, "Converting Rules and Profile Files"
- Chapter 3, "Converting Configuration Files"

Use one server as both a JumpStart install server and an AI install server.

Chapter 4, "Installing Oracle Solaris 10 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 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 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: pkg.oracle.com

Make a local copy of a package repository: *Copying and Creating Oracle Solaris 11 Package Repositories*.

Provide system configuration instructions.

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

See Chapter 12, "Installing and Configuring Zones," in *Installing Oracle Solaris 11 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 Systems*.

Alternatively, create exactly the installation you want in one installation file.

See *Creating a Custom Oracle Solaris 11 Installation Image* for information about creating a custom installation image.

CHAPTER 2

Converting Rules and Profile Files

This chapter shows how to convert JumpStart rules and profile files to AI criteria files and AI manifests. Most of the conversion can be done with the `js2ai` utility. See the `js2ai`(1M) man page for more information about the `js2ai` utility.

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 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. This chapter shows how to use the `js2ai` tool to convert JumpStart `rules` files to AI criteria files. See Chapter 9, "Customizing Installations," in *Installing Oracle Solaris 11 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 Directives
`any`	Unsupported. For client systems that do not match any selection criteria, the AI install service provides a default AI manifest.

TABLE 2–1 Comparing JumpStart Rules File Keywords and AI Criteria Directives *(Continued)*

JumpStart Rules File Keyword	AI Criteria File Directives

`arch`	`cpu`
	Command option: `-c cpu=sparc`
	Criteria file:
	`<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`
	Command option: `-c ipv4=10.6.68.127`
	Criteria file:
	`<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 shown above, or use the MAC address.
	Command option: `-c mac=0:3:ba:33:9d:b6`
	Criteria file:
	`<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`
	Command option: `-c arch=i86pc`
	Criteria file:
	`<ai_criteria name="arch">`
	`  <value>i86pc</value>`
	`</ai_criteria>`

TABLE 2–1 Comparing JumpStart Rules File Keywords and AI Criteria Directives *(Continued)*

JumpStart Rules File Keyword	AI Criteria File Directives

`memsize`	`mem`
	Command option: `-c mem=2048`
	Criteria file:
	`<ai_criteria name="mem">`
	`  <value>2048</value>`
	`</ai_criteria>`

`model`	`platform`
	Command option: `-c platform=SUNW,Sun-Fire-T200`
	Criteria file:
	`<ai_criteria_name="platform">`
	`  <value>SUNW,Sun-Fire-T200</value>`
	`</ai_criteria>`

`network`	Use `network`, `network` with a range, or `ipv4` with a range.
	Command option, single `network` value: `-c network="10.0.0.0"`
	Criteria file, single `network` value:
	`<ai_criteria name="network">`
	`  <value>10.0.0.0</value>`
	`</ai_criteria>`
	Command option, `ipv4` range: `-c ipv4=10.0.0.1-10.0.0.64`
	Criteria file, `ipv4` range:
	`<ai_criteria name="ipv4">`
	`  <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 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 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*
`boot_device c1t0d0`
	`<target>`
	`  <disk whole_disk="true">`
	`    <disk_name name="c1t0d0" name_type="ctd"/>`
	`  </disk>`
	`  <logical nodump="true" noswap="false"/>`
	`</target>`

`boot_device` *device eeprom* `boot_device c0t0d0s0 update`	Same as above. The second token value for this 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 25 for information about how to handle `cluster` and `package` specifications. The Oracle Solaris 11 OS uses `pkg`(5) group packages. 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.

TABLE 2–2 Comparing JumpStart Profile File Keywords and AI Manifest Directives *(Continued)*

JumpStart Profile File Keyword	AI Manifest Directives
`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*
`fdisk c0t3d0 solaris all`
	`<target>`
	`  <disk whole_disk="true">`
	`    <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 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.

TABLE 2–2 Comparing JumpStart Profile File Keywords and AI Manifest Directives *(Continued)*

JumpStart Profile File Keyword	AI Manifest Directives

`filesys c1t0d0s0 10000 /`	The following partial AI manifest is for x86 platforms. For SPARC platforms, no `partition` element is specified.
	`<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` tool only supports translations of the root file system (`/`) and swap.

TABLE 2–2 Comparing JumpStart Profile File Keywords and AI Manifest Directives *(Continued)*

<table>
<tr><th>JumpStart Profile File Keyword</th><th>AI Manifest Directives</th></tr>
<tr><td><code>filesys mirror:rpool c6t0d0s0 c6t1d0s0 60048 /</code></td><td>The following partial AI manifest is for x86 platforms. For SPARC platforms, no <code>partition</code> element is specified.
<pre>
<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 nodump="true" noswap="false">
    <zpool is_root="true" name="rpool">
      <vdev name="rpool_vdev" redundancy="mirror"/>
    </zpool>
  </logical>
</target>
</pre></td></tr>
<tr><td><code>filesys</code> <i>server</i>:<i>path</i></td><td>Unsupported. AI does not support remote file systems.</td></tr>
<tr><td><code>forced_deployment</code></td><td>Unsupported. AI does not install from flash archives.</td></tr>
<tr><td><code>geo</code></td><td>Unsupported. In AI, geographic regions for language support are specified in the system configuration profile. See Chapter 11, “Configuring the Client System,” in <i>Installing Oracle Solaris 11 Systems</i>.</td></tr>
<tr><td><code>install_type</code></td><td>AI supports only <code>initial_install</code>.</td></tr>
<tr><td><code>layout_constraint</code></td><td>Unsupported. AI supports only <code>initial_install</code>.</td></tr>
<tr><td><code>local_customization</code></td><td>Unsupported. AI does not install from flash archives.</td></tr>
<tr><td><code>locale</code></td><td>Unsupported. In AI, locale support is specified in the system configuration profile. See Chapter 11, “Configuring the Client System,” in <i>Installing Oracle Solaris 11 Systems</i>.</td></tr>
<tr><td><code>metadb</code></td><td>Unsupported.</td></tr>
</table>

TABLE 2–2 Comparing JumpStart Profile File Keywords and AI Manifest Directives *(Continued)*

JumpStart Profile File Keyword	AI Manifest Directives
`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 SUNWpampkcs11`
`package SUNWpampkcs11 add`
	`<software type="IPS">`
	`  <software_data action="install">`
	`    <name>pkg:/SUNWpampkcs11</name>`
	`  </software_data>`
	`</software>`
	See "Converting Software Packages" on page 25 for information about how to verify the package name is valid.

`package SUNWpampkcs11 add nfs golden:/packages/Solaris_10`	In an AI manifest, all software is retrieved from IPS 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` tool ignores the *retrieval_type* `location` argument.

`package SUNWpampkcs11 delete`	`<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.

TABLE 2–2 Comparing JumpStart Profile File Keywords and AI Manifest Directives *(Continued)*

<table>
<tr><th>JumpStart Profile File Keyword</th><th>AI Manifest Directives</th></tr>
<tr><td><code>pool newpool auto auto auto c1t0d0s0</code></td><td><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="newpool" name="0"/>
    </partition>
  </disk>
  <logical nodump="false" noswap="false">
    <zpool is_root="true" name="newpool">
      <vdev name="rpool_vdev" redundancy="none"/>
    </zpool>
  </logical>
</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> tool 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 newpool 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 “Fix the <code>mirrorpool.profile</code> Errors” on page 33 for an example.</td></tr>
</table>

TABLE 2–2 Comparing JumpStart Profile File Keywords and AI Manifest Directives *(Continued)*

JumpStart Profile File Keyword	AI Manifest Directives

`root_device c1t0d0s0`	`<target>`
	`  <disk whole_disk="true">`
	`    <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>`
	Translation by `js2ai`:
	When `root_device` is specified, `js2ai` sets the `rootdisk` to the specified device.

`system_type`	AI manifests do not differentiate system types. Translation by `js2ai`: Only the value `standalone` is supported.
`usedisk`	Translation by `js2ai`: The `js2ai` tool might use the specified device or devices to resolve subsequent `any` or `rootdisk` specifications. Devices specified that are not used for this purpose are added to the ZFS root pool by `js2ai`, when that pool is not mirrored.

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

Use the `js2ai` utility 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.

```
/usr/sbin/js2ai -rS [-d sysidcfg_dir] [-D destination_dir]
```

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.

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.

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.

To validate a specific output AI manifest against the appropriate AI DTD, run the js2ai command with the -V option:

```
/usr/sbin/js2ai -V manifest_file
```

AI manifests are also validated when you add them to an AI install service.

To use the js2ai utility to convert one JumpStart profile file instead of a rules file and all profiles associated with that rules file, use the -p option instead of the -r option.

```
/usr/sbin/js2ai -p JS_profile_name [-d sysidcfg_dir] [-D destination_dir]
```

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

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

The `js2ai` tool 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` tool 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. This is not applicable when a mirrored pool is specified.

How the ZFS Root Pool is Determined

The `js2ai` tool performs the following steps to determine what 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 `filesys` 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 and all keywords in the profile have been processed, and if `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. 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, `js2ai` creates the correct package installation specification in the AI manifest.

Tip – Check 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.

Some SVR4 package names exist in IPS so that 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.

Some SVR4 package names do not exist in IPS. In those cases, 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.

The `js2ai` utility 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 only need to specify installation of additional tools and applications.

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. In this example, the AI manifest specifies the `http://pkg.oracle.com/solaris11/release` repository origin.

```
$ pkg list -af -g http://pkg.oracle.com/solaris11/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.

```
$ pkg list -af -g http://pkg.oracle.com/solaris11/release SUNWpampkcs11
NAME (PUBLISHER)                                        VERSION            IFO
SUNWpampkcs11                                           0.6.0-0.133        --r
```

The “r” in the right-most 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: 0.173.0.0.0.0.487
Packaging Date: Sun Aug 28 00:16:46 2011
          Size: 1.92 MB
          FMRI: pkg://solaris/library/security/pam/module/pam-pkcs11@...
```

```
$ 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 right-most 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.

Example 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

This 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 With the Rules File Option

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 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
```

Equivalent AI Criteria Files

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 is used to specify which clients should use the `fdisk.profile` profile. Since `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 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 *Oracle Solaris Administration: Network Interfaces and Network Virtualization* and in the dladm(1M) man page.

For the JumpStart rule hostaddress 10.6.68.127, the js2ai utility 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 utility 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 utility 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>
```

Equivalent AI Manifest Files

The js2ai utility 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.

Fix the fdisk.profile Errors

The js2ai utility 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. Then the fdisk.profile file 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 utility 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 action="create" force="true" is_swap="true" name="1">
        <size val="5000mb"/>
      </slice>
      <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>
```

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

Fix the mirrorfilesys.profile Errors

The js2ai utility 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, the *file_system* parameter in the `filesys` `mirror` specification can be omitted. In a `js2ai` conversion, the *file_system* parameter cannot be omitted and 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. Then the `mirrorfilesys.profile` file 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` utility 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 nodump="true" noswap="false">
    <zpool is_root="true" name="rpool">
      <vdev name="rpool_vdev" redundancy="mirror"/>
    </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.

Fix the mirrorpool.profile Errors

The js2ai utility 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 exactly the same AI manifest output. See also “How the any Device Is Translated” on page 24.

- Edit the mirrorpool.profile profile, replacing both any with physical device names. Also, delete the cluster line. Then the mirrorpool.profile file has the following content, for example:

  ```
  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 *vdevlist* 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. Then the mirrorpool.profile file has the following content, for example:

  ```
  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 utility 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 nodump="false" noswap="false">
    <zpool is_root="true" name="newpool">
      <vdev name="rpool_vdev" redundancy="mirror"/>
    </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.

Fix the rootdisk.profile Errors

The `js2ai` utility 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 exactly the same AI manifest output.

- Edit the `rootdisk.profile` profile, replacing `rootdisk.` with `c0t0d0`. Also, delete the `cluster` line. Then the `rootdisk.profile` file 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. Then the `rootdisk.profile` file 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` utility 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 nodump="true" noswap="false">
    <zpool is_root="true" name="rpool">
      <vdev name="rpool_vdev" redundancy="none"/>
    </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 More Profile Conversion Information

If you want to see more information for a rule or profile conversion or AI manifest validation, specify the `-v` option in the `js2ai` command. When you specify the `-v` option, processing steps are displayed, and the error report displays zeroes instead of omitting that `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 resulting AI manifest:

```
# 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 Configuration Files

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

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 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 "Example sysidcfg Conversion Using js2ai" on page 40 and in "Example System Configuration Profiles" in *Installing Oracle Solaris 11 Systems*.

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

sysidcfg File Keyword	System Configuration Profile Directives
keyboard	The js2ai tool 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

TABLE 3–1 Comparing sysidcfg File Keywords and Configuration Profile Directives *(Continued)*

`sysidcfg` File Keyword	System Configuration Profile Directives
`name_service`	AI supports DNS, NIS, and LDAP. The `js2ai` tool 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`
`network_interface`	AI supports configuring only a single interface as part of system installation. Because of this limitation, the `js2ai` tool processes only the interface labeled `PRIMARY` or the first interface encountered in the `sysidcfg` file. The `js2ai` tool 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. Service name: `network/install` Property group name: `install_ipv4_interface` Property value name: `static_address` Property group name: `install_ipv6_interface`
`nfs4_domain`	Unsupported.
`root_password`	The `js2ai` tool does not perform any translation. Service name: `system/config-user` Property group name: `root_account` Property value name: `password`
`security_policy`	The `js2ai` tool supports the value `NONE`.
`service_profile`	The `js2ai` tool supports the value `limited_net`.
`system_locale`	The `js2ai` tool does not perform any translation. Make sure the locale specified in the `sysidcfg` file is supported in Oracle Solaris 11. Service name: `system/environment` Property group name: `environment` Property value name: `LC_ALL`
`terminal`	The `js2ai` tool 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` tool supports the value `localhost`.

TABLE 3–1 Comparing sysidcfg File Keywords and Configuration Profile Directives *(Continued)*

`sysidcfg` File Keyword	System Configuration Profile Directives
`timezone`	The `js2ai` tool 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` utility 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.

```
/usr/sbin/js2ai -sS [-d sysidcfg_dir] [-D destination_dir]
```

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.

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.

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.

To validate a specific output system configuration profile, run the `js2ai` command with the `-V` option:

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

System configuration profiles are also validated when you add them to an AI install service or when you run the following command on your AI install server:

```
# installadm validate -n install_service_name sc_profile.xml
```

See "Validate a System Configuration Profile" in *Installing Oracle Solaris 11 Systems* for more information about the `installadm validate` command.

Example sysidcfg Conversion Using js2ai

This section shows using `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.

Sample sysidcfg File

This example uses the following `sysidcfg` file:

```
timezone=US/Pacific
terminal=xterms
timeserver=localhost
network_interface=primary {
                        hostname=host1
                        ip_address=10.80.127.35
                        netmask=255.255.255.224
                        protocol_ipv6=no
                        default_route=10.80.127.33}
root_password=rJmv5LUXM1OcU
security_policy=none
```

Using js2ai With the sysidcfg File Option

Use the following command to process this `sysidcfg` 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 -sS
                                Process  Unsupported  Conversion  Validation
Name                  Warnings  Errors   Items        Errors      Errors
-------------------   --------  -------  -----------  ----------  ----------
sysidcfg                     0        0            0           1           -

Conversion completed. One or more failures occurred.
For errors see js2ai.log
# cat js2ai.log
sysidcfg:line 4:CONVERSION: when the PRIMARY interface is specified, by default the
system will be configured for both IPv4 and IPv6 via automatic network configuration.
The options specified will be ignored. If you wish to configure the interface with
the specified options replace PRIMARY with the name of the interface that should be
configured.
```

Equivalent System Configuration Profile File

The following modified `sysidcfg` file addresses the error reported in the `js2ai.log` file. The `PRIMARY` interface specification is replaced with the interface name `e1000g`.

```
timezone=US/Pacific
terminal=xterms
timeserver=localhost
network_interface=e1000g {
                        hostname=host1
                        ip_address=10.80.127.35
                        netmask=255.255.255.224
                        protocol_ipv6=no
                        default_route=10.80.127.33}
root_password=rJmv5LUXM1OcU
security_policy=none
```

```
# js2ai -sS
Successfully completed conversion
```

This time no error report is output and the `js2ai.log` file is empty.

The following `sc_profile.xml` file is produced.

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE service_bundle
  SYSTEM ’/usr/share/lib/xml/dtd/service_bundle.dtd.1’>
<service_bundle name="system configuration" type="profile">
  <service name="system/timezone" type="service" version="1">
    <instance enabled="true" name="default">
      <property_group name="timezone" type="application">
        <propval name="localtime" type="astring" value="US/Pacific"/>
      </property_group>
    </instance>
  </service>
  <service name="system/console-login" type="service" version="1">
    <instance enabled="true" name="default">
      <property_group name="ttymon" type="application">
        <propval name="terminal_type" type="astring" value="xterms"/>
      </property_group>
    </instance>
  </service>
  <service name="system/config-user" type="service" version="1">
    <instance enabled="true" name="default">
      <property_group name="root_account" type="application">
        <propval name="password" type="astring" value="rJmv5LUXM1OcU"/>
      </property_group>
    </instance>
  </service>
  <service name="system/identity" type="service" version="1">
    <instance enabled="true" name="node">
      <property_group name="config" type="application">
        <propval name="nodename" type="astring" value="host1"/>
      </property_group>
    </instance>
  </service>
  <service name="network/install" type="service" version="1">
    <instance enabled="true" name="default">
      <property_group name="install_ipv4_interface" type="application">
        <propval name="name" type="astring" value="e1000g/v4"/>
        <propval name="address_type" type="astring" value="static"/>
        <propval name="static_address" type="net_address_v4" value="10.80.127.35/27"/>
        <propval name="default_route" type="net_address_v4" value="10.80.127.33"/>
      </property_group>
    </instance>
  </service>
  <service name="network/physical" type="service" version="1">
    <instance enabled="true" name="default">
      <property_group name="netcfg" type="application">
        <propval name="active_ncp" type="astring" value="DefaultFixed"/>
      </property_group>
    </instance>
  </service>
</service_bundle>
```

Displaying More Configuration Conversion Information

If you want to see more information for a sysidcfg file conversion or system configuration profile validation, specify the -v option in the js2ai command. When you specify the -v option, processing steps are displayed, and the error report displays zeroes if no errors occurred instead of omitting the error report.

```
# 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                     0        0            0           0           0
Successfully completed conversion
```

Validating an Output System Configuration Profile

Use the -V option to validate the resulting system configuration profile:

```
# 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 Using JumpStart on an Oracle Solaris 11 Server

If you are familiar with using JumpStart to install the Oracle Solaris 10 operating system on networked SPARC and x86 platforms, then you probably know that JumpStart can be used to install only the Oracle Solaris 10 OS, not 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 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 8/11 Installation Guide: Custom JumpStart and Advanced Installations*.

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

The following steps describe how to create an Oracle Solaris 11 JumpStart install server to install the Oracle Solaris 10 OS on client systems.

1. Install the Oracle Solaris 11 OS on the machine that will be the JumpStart install server.
2. Set up a static IP address on the Oracle Solaris 11 machine that will be the JumpStart install server.

   For information about how to set a static IP address, see the instructions in "How to Configure an IP Interface" in *Oracle Solaris Administration: Network Interfaces and Network Virtualization* 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. Use the Oracle Solaris 10 DVD image and the `setup_install_server`(1M) command to set up a JumpStart install server.

   The following example uses the Oracle Solaris 10 9/10 DVD image for SPARC.

```
# /media/SOL_10_0910_SPARC/Solaris_10/Tools/setup_install_server /export/s10u9_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 from the DVD image to the JumpStart profile location.

```
# cd /media/SOL_10_0910_SPARC/Solaris_10/Misc/jumpstart_sample
# mkdir -p /export/profiles/s10profile
# cp -pr * /export/profiles/s10profile
```

7. Perform the workaround described in "`rm_install_client` Script Issue" on page 46.

8. Perform the workaround described in "`check` Script Issue" on page 46.

9. Run the `check`(1M) command with the `-p` *s10-image* option to 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 Oracle Solaris 10 Update 7 or later. (Reference CR 6728067.)

```
# cd /export/profile/s10profile
./check -p /export/s10u9_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.
```

10. Make sure that udp6 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
```

11. Link the /tftpboot directory to the /etc/netboot directory. If the /etc/netboot directory does not exist, create it.

```
# 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
```

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.

12. Run add_install_client(1M) to install the client.

```
/export/s10u9_sparc/Solaris_10/Tools/add_install_client -e 8:0:20:fd:f2:18 \
-c line2-x4100:/export/profiles/s10profile \
-p line2-x4100:/export/profiles/s10profile line2-t1 sun4u
Adding Ethernet number for line2-t1.sfbay.sun.com to /etc/ethers
making /tftpboot
enabling network/rarp service
enabling network/rpc/bootparams service
updating /etc/bootparams
copying boot file to /tftpboot/inetboot.SUN4U.Solaris_10-1
```

Workarounds for Known Issues

This section gives instructions for avoiding particular problems.

rm_install_client Script Issue

```
CR 6646677: The rm_install_client script does not work with Solaris 10 dig based
nslookup
```

To work around this issue, make the following changes in the `rm_install_client` script in your Oracle Solaris 10 net image. For example, if you have set up an Oracle Solaris 10 net image at `/export/s10u9/dvds/latest`, then make the following changes in the `rm_install_client` script at `/export/s10u9/dvds/latest/Solaris_10/Tools/rm_install_client`:

Change line 168 from this:

```
ANS=‘nslookup ${K} 2>&1‘
```

to this:

```
ANS=‘nslookup ${K} | /bin/sed ’/^;;/d’ 2>&1‘
```

Change line 273 from this:

```
ANS=‘echo $ANS | sed -e ’s/#.*$//’‘
```

to this:

```
ANS=‘echo $ANS | sed -e ’s/ #.*$//’‘
```

check Script Issue

```
CR 6838095: opensolaris jumpstart server doesn’t run the check script correctly
```

To work around this issue, change the `check` script in the JumpStart profile directory `/export/profile/s10profile` to use `/usr/sunos/bin/sh`. Replace the first line of the `check` script with the following line:

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