
ORACLE®

ORACLE®

A Practical Guide to Auto Install Derived Manifest Scripts

Jarod Nash
RPE Appliance

Agenda

- Kick off demo installs
- Automated Installer (AI) Overview
- AI Manifest 101
- aimanifest(1m)
- Example
- Review demo installs

- IPS Repo Management Tip

AI Lab Demonstration

- Typical lab client install time is 20-30 minutes
- Start network install of several VMs
- Continue presentation - explain Derived Manifest scripts
- Return and review results

Automated Installer Overview

Disclaimer

- Try to avoid duplicating previous Solaris SIG Events:
  - February: Andrew Watkins Solaris 11 Automated Installer Walkthrough
  - March: Chris Ridd IPS for OpenIndiana and Solaris 11
- Derived Manifests are part of AI
  - Brief mention in Andrew's talk
  - Not Solaris 11 Express, New in Solaris 11 11/11
- However, we do need some AI basics before we focus on the Derived Manifests themselves

Automated Installer

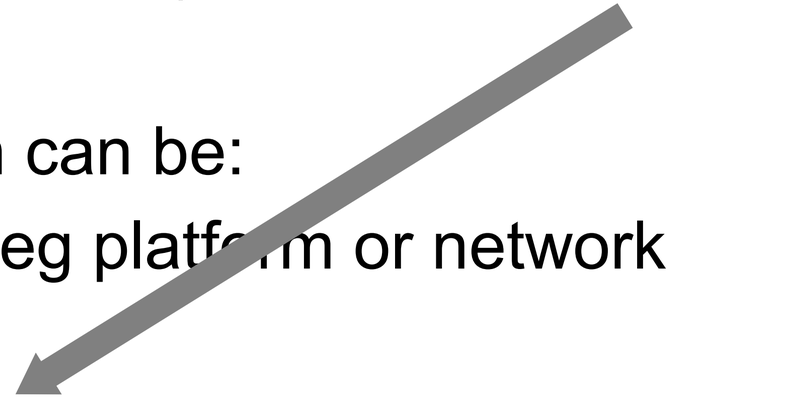
Outline

- Set up Solaris 11 AI server
  - Solaris 11 can also be Solaris 10 JumpStart server but not vice versa
  - Supports both x86 and SPARC clients
  - Clients can differ in architecture, disk and memory capacity, and other characteristics
  - Installations can differ in network configuration, packages installed, and other specifications
- Client boots and installs
  - Client gets installation specifications from AI server
  - Retrieve/Install packages from a Solaris package repository

Automated Installer

Installation Specifications

- Provisioning – AI Manifest
  - What and How to install (packages and publishers)
  - Where to install (pool/filesystem layout)
  - Each client uses only one AI manifest, which can be:
    - *Custom* - selected by a specific criteria, eg platform or network
    - *Default* - no selection criteria met
    - *Derived* - AI manifest created on-the-fly at installation time
- Configuring – System Configuration Profiles
  - Client identity, eg hostname, network, name services (optional)



AI Manifests

Selection Criteria

- Identify criteria with installadm(1m) when creating/updating
  - CLI - use -c flag one or more times
  - XML - use -C flag to indicate specified in XML <ai_criteria> elements
  - installadm checks for invalid overlapping manifest criteria
- Criteria checked at install time in order starting with:
  - mac mac=0:14:4F:20:53:94-0:14:4F:20:53:A0 (range)
  - ipv4 ipv4="10.6.68.127" (single)
  - platform platform="SUNW,Sun-Fire-T200" (uname -i output)
- Can also use criteria to select profiles, but that's another topic...

AI Manifests

The Basics

- Sections:
  - AI Settings: reboot after install, HTTP Proxy
  - Disk Layout: targets, pools, filesystems, boot environments
  - Software: publisher, origin, packages
  - Boot Configuration (x86 only): GRUB boot menu
  - Other Configuration: Non-global zones
- Minimum Requirement (AI fills in the blanks using rules)
  - Identify root pool (Disk Layout)
  - Specify Software to install (Software)

AI Manifests

Minimal Example

```
<auto_install>
  <ai_instance name="default">
    <target>
      <logical>
        <zpool name="rpool" is_root="true">
          <filesystem name="export" mountpoint="/export"/>
          <filesystem name="export/home"/>
        </zpool>
      </logical>
    </target>
    <software type="IPS">
      <source>
        <publisher name="solaris">
          <origin name="http://pkg.oracle.com/solaris/release"/>
        </publisher>
      </source>
      <software_data action="install">
        <name>pkg:/entire@latest</name>
        <name>pkg:/group/system/solaris-large-server</name>
      </software_data>
    </software>
  </ai_instance>
</auto_install>
```

AI Manifests

Advanced

- Mirroring rpool requires identifying pool disks
- Specify <disk> elements and associate with zpool/vdev
- Specify redundancy <vdev> for zpool element

```
<target>
  <logical>
    <zpool name="rpool" is_root="true"/>
  </logical>
</target>
```

```
<target>
  <disk in_vdev="mirror_vdev" in_zpool="rpool" whole_disk="true">
    <disk_name name="c0t0d0" name_type="ctd"/>
  </disk>
  <disk in_vdev="mirror_vdev" in_zpool="rpool" whole_disk="true">
    <disk_name name="c0t1d0" name_type="ctd"/>
  </disk>
  <logical>
    <zpool name="rpool" is_root=”true”>
      <vdev name="mirror_vdev" redundancy="mirror"/>
    </zpool>
  </logical>
</target>
```

AI Manifests

Manipulating XML

- aimanifest(1m) utility provided to simplify modifying XML
- Start with base/template AI manifest
  - Use default in /usr/share/auto_install, or download via HTTP (preferred)
    - Minimum: !DOCTYPE reference and a root element for this DTD
- Install environment populated with client attribute variables
  - SI_ARCH, SI_CPU, SI_NUMDISKS, etc
- Iterate modifications to *evolving manifest* in script
  - Add elements
  - Set attributes

aimanifest(1m)

Execution Model

- Environment Variables
  - AIM_MANIFEST - location of evolving manifest
  - AIM_LOGFILE - logfile for errors/debug
- Command Line
  - load [-i] - load/insert/merge into evolving manifest
  - get, add, set - retrieve, add elements, set attributes
  - return path argument -r - Nodepath of element operated on
  - validate - validate against DTD

aimanifest(5)

Nodepaths

- Nodepaths (near relation to xpaths) specify XML elements
  - /a element defined by tag
  - /a[b=6] element defined by tag, and tag and value of a child
  - /a@attr=8 (leaf) element defined by tag, and the name and value

```
<target>
 <logical>
  <zpool name="rpool" is_root="true">
   <filesystem name="export" mountpoint="/export"/>
   <filesystem name="export/home"/>
  </zpool>
 </logical>
</target>
```

```
<software type="IPS">
 <software_data action="install">
  <name>pkg:/entire@latest</name>
  <name>pkg:/group/system/solaris-large-server</name>
 </software_data>
</software>
```

```
$ aimanifest get target/logical/zpool@name
rpool
$ aimanifest get -r target/logical/zpool@name
rpool /auto_install[1]/ai_instance[1]/target[1]/logical[1]/zpool[1]
```

```
$ aimanifest get software/software_data/name
Ambiguity error:  Path matches more than one element
```

Derived Manifest Scripts

Execution Context

- Executed as aiuser role
  - Privileges of a non-privileged user plus the following privileges:
    - solaris.network.autoconf.read
    - solaris.smf.read*
  - aiuser role cannot change the system
- AIM_MANIFEST, AIM_LOGFILE and client SI_ variables setup
- Script most likely either ksh or python
  - python supports *subpathing* (using [] inside a nodepath)
- AI validates resulting manifest and may abort install

AI Manifests

Advanced Recap

```
<target>
  <logical>
    <zpool name="rpool" is_root="true"/>
  </logical>
</target>
```

```
<target>
  <disk in_vdev="mirror_vdev" in_zpool="rpool" whole_disk="true">
    <disk_name name="c0t0d0" name_type="ctd"/>
  </disk>
  <disk in_vdev="mirror_vdev" in_zpool="rpool" whole_disk="true">
    <disk_name name="c0t1d0" name_type="ctd"/>
  </disk>
  <logical>
    <zpool name="rpool" is_root=”true”>
      <vdev name="mirror_vdev" redundancy="mirror"/>
    </zpool>
  </logical>
</target>
```

Derived Manifest Scripts

Example – ksh93 script missing error trap function

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

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

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

    for ((disk_num = 1; disk_num <= $SI_NUMDISKS; disk_num++)) ; do
        eval curr_disk="$"SI_DISKNAME_${disk_num}
        disk=$(aimanifest add -r target/disk@in_vdev mirror_vdev)
        aimanifest set ${disk}@in_zpool rpool
        aimanifest set ${disk}@whole_disk true
        disk_name=$(aimanifest add -r ${disk}/disk_name@name $curr_disk)
        aimanifest set ${disk_name}@name_type ctd
    done
fi
```

Derived Manifest Scripts

Testing

```
$ head /tmp/doit
export SI_NUMDISKS=2
export SI_DISKNAME_1=c0t0d0
export SI_DISKNAME_2=c0t1d0
export AIM_MANIFEST=/tmp/mymanifest.xml

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

# Use the default if there is only one disk.
if [[ $SI_NUMDISKS -ge 2 ]] ; then
    typeset -i disk_num
$ /tmp/doit
$ cat /tmp/mymanifest.xml
...
    <target>
      <disk in_vdev="mirror_vdev" in_zpool="rpool" whole_disk="true">
        <disk_name name="c0t0d0" name_type="ctd"/>
      </disk>
      <disk in_vdev="mirror_vdev" in_zpool="rpool" whole_disk="true">
        <disk_name name="c0t1d0" name_type="ctd"/>
      </disk>
      <logical>
        <zpool name="rpool" is_root="true">
          <vdev name="mirror_vdev" redundancy="mirror"/>
```

Derived Manifest Scripts

Testing in Install Environment

- Boot without start installing
  - SPARC don't specify “- install” at ok prompt
  - x86 select “Text Installer and command line”
    NOTE: GRUB kernel line should not contain “install=true”
  - Select option 3 “Shell” from menu
- Starting installation after booting without starting:

```
# svcadm enable manifest-locator:default
# svcadm enable svc:/application/auto-installer:default
```

Lab Demo Installs

Review

IPS Management Tip

Recommended Whitepaper

- How to Create Multiple Internal Repositories for Oracle Solaris 11 by Albert White

  “Some customers connect directly to hosted Oracle Solaris package repositories to get the latest fixes, but most customers set up a local repository due to network restrictions or the desire to control which updates their systems have access to. This article provides best practices for managing local repositories through the complete software lifecycle from development and testing to production deployment.”

AI Manifests

References

- XML file, see ai_manifest(4) and DTD:
  - /usr/share/install/ai.dtd.1
- Default AI manifest:
  - /usr/share/auto_install/default.xml
- Extensively annotated example:
  - /usr/share/auto_install/manifest/ai_manifest.xml
- Full documentation:
  - Installing Oracle Solaris 11 Systems
    - Part III, Chapter 10: Provisioning the Client System

Acknowledgements

- Oracle Documentation
  - it really is very good, with a great deal of the credit going to dev team
- Peter Dennis
  - Waster^H^H^H^H^H^HSolaris 11 Tech and SRU Lead
- Robin Ridler & William “Cookie” Cooke
  - Oracle Global Lab Engineers
  - Huge experience and very helpful

Hardware and Software

ORACLE

Engineered to Work Together


ORACLE®