The Alligator meets the Terminator: Caiman, AI, and the other 998 ways of installing OpenSolaris

Volker A. Brandt
Brandt & Brandt Computer GmbH
vab@bb-c.de

The Big Picture
A Closer Look at Caiman
The Automated Installer
An AI Example
Random Advice
Links and Resources

The Big Picture

A Closer Look at Caiman

The Automated Installer

An AI Example

Random Advice

Links and Resources

“Interesting” Things in the Installation World:

- DC
- Caiman / Dwarf Caiman / Slim Install
- IPS
- AI
- Text Installer
- VMC

“Interesting” Things in the Installation World:

- Distribution Constructor
- Caiman + Live CD-ROM
- Image Packaging System: *pkg(5)*
- Automated Installer
- Text Installer (prototype)
- Virtual Machine Constructor (prototype)

“Interesting” Things in the Installation World:

- Distribution Constructor
- Caiman + Live CD-ROM
- Image Packaging System: *pkg(5)*
- Automated Installer
- Text Installer (prototype)
- Virtual Machine Constructor (prototype)

The Big Picture

A Closer Look at Caiman

The Automated Installer

An AI Example

Random Advice

Links and Resources

The Good Old Times:

```
What type of terminal are you using?
 1) ANSI Standard CRT
 2) DEC VT52
 3) DEC VT100
 4) Heathkit 19
 5) Lear Siegler ADM31
 6) PC Console
 7) Sun Command Tool
 8) Sun Workstation
 9) Televideo 910
 10) Televideo 925
 11) Wyse Model 50
 12) X Terminal Emulator (xterms)
 13) CDE Terminal Emulator (dtterm)
 14) Other
Type the number of your choice and press Return:
```

Caiman Design Goals

- modern look and feel
- simple to use, streamlined installation
- concentrate on OS installation tasks
- focus on user experience: don't ask too many questions, avoid reboots, etc.
- leverage Live CD concept: provide enough to run the desktop; get the rest from an IPS repository

Caiman Design Goals

- modern look and feel
- simple to use, streamlined installation
- concentrate on OS installation tasks
- focus on user experience: don't ask too many questions, avoid reboots, etc.
- leverage Live CD concept: provide enough to run the desktop; get the rest from an IPS repository
- attract the Linux developer crowd

Why the name “Caiman”?

Why the name “Caiman”?

The Caiman is
the bitter enemy
of the Anaconda.

(Anaconda is the installer from RHEL and Fedora)

Why the name “Caiman”?

Sometimes, the
Anaconda wins...

The Caiman is
the bitter enemy
of the Anaconda.

(Anaconda is the installer from RHEL and Fedora)

How does Caiman work?

- The Live CD-ROM is booted.
- A desktop with a default user (*jack)* is presented on the graphical console.
- If the user wishes to install OpenSolaris, the application `/usr/bin/gui-install` is started in `root` context.
- If the installation proceeds, the entire OS image is copied from the CD-ROM to the target disk, no pkg operations are performed.

Caiman Drawbacks

- space on the installation CD-ROM is scarce; content frequently changes
- DHCP needed for “normal” installation, NWAM too complex for novice users
- GUI required (the installer is a Gnome/GTK application)
- no SPARC version available
- most software for “real work” must be retrieved from an IPS repository via net

Remember: Caiman is Work in Progress!

Use the newest version for testing, download the preview releases from `genunix.org`!

The Big Picture
A Closer Look at Caiman
The Automated Installer
An AI Example
Random Advice
Links and Resources

AI: Automated Installer Design Goals

- hands-off installation, suitable for datacenter and remote deployment
- x86/x64 and SPARC are both first-class citizens
- driven by parameter files
- eliminate limitations of “old” installation methods (RARP, `bootparams,...`)
- leverage modern standards (XML, http, DHCP,...)

AI: Automated Installer Prerequisites

- chicken-and-egg problem: an OpenSolaris system is needed to install an OpenSolaris system
- control over DHCP (can be standalone)
- for SPARC, WAN boot capable clients
- good bandwidth to an IPS repository or a local mirror

AI: Automated Installer Prerequisites

- chicken-and-egg problem: an OpenSolaris system is needed to install an OpenSolaris system
- control over DHCP (can be standalone)
- for SPARC, WAN boot capable clients
- good bandwidth to an IPS repository or a local mirror ➜ all package data are copied in from the repository, nothing comes from the AI install image

AI: A Look at the Server

We need:

- the AI software
- the install image
- DHCP info for client address, boot file, DNS servers, default route, ...
- enabled *tftp* service
- an install service
- a matching *menu.lst* file
- a good webserver configuration
- a static IP address on the AI server

Be sure to always use current versions of everything:

```
<caiman:/usr/share/man,36# pkg image-update -nv

WARNING: pkg(5) appears to be out of date, and should be updated before
running image-update.

Please update pkg(5) using 'pfexec pkg install SUNWipkg' and then retry
the image-update.

<caiman:/usr/share/man,39# pkg install -v SUNWipkg
Creating Plan | Before evaluation:
UNEVALUATED:
+pkg:/SUNWipkg@0.5.11,5.11-0.111:20090826T200238Z

After evaluation:
pkg:/SUNWipkg@0.5.11,5.11-0.111:20090508T161015Z ->
pkg:/SUNWipkg@0.5.11,5.11-0.111:20090826T200238Z
.....
<caiman:/usr/share/man,20# pkg image-update -v
Retrieving catalog 'opensolaris.org'...
Loading catalog cache ...
Creating Plan
Creating Plan - Before evaluation:
UNEVALUATED:
+pkg:/entire@0.5.11,5.11-0.111:20090518T052643Z
+pkg:/SUNWipkg-brand@0.5.11,5.11-0.111:20090826T185654Z

After evaluation:
pkg:/entire@0.5.11,5.11-0.111:20090514T145840Z ->
pkg:/entire@0.5.11,5.11-0.111:20090518T052643Z
.....
A clone of opensolaris exists and has been updated and activated.
On the next boot the Boot Environment opensolaris-1 will be mounted on '/'.
```

Use the Dev repository:

```
<caiman:/tmp,46# pfexec pkg set-publisher -O http://pkg.opensolaris.org/dev \
   opensolaris.org
Retrieving catalog 'opensolaris.org'...
Loading catalog cache ...
.....
```

Install the Automated Installer tools:

```
<caiman:/tmp,47# pkg install SUNWinstalladm-tools
Refreshing catalog
Refreshing catalog 1/1 opensolaris.org
.....

<caiman:/tmp,134# pkg info -l SUNWinstalladm-tools
          Name: SUNWinstalladm-tools
       Summary: Automatic Installation Server Setup Tools
      Category: System/Administration and Configuration
         State: Installed
     Publisher: opensolaris.org
       Version: 0.5.11
 Build Release: 5.11
        Branch: 0.124
Packaging Date: Fri Sep 25 21:18:44 2009
          Size: 680.91 kB
          FMRI: pkg://opensolaris.org/SUNWinstalladm-
tools@0.5.11,5.11-0.124:20090925T211844Z
```

Populate an AI image directory:

```
<caiman:/tmp,49# zfs create -o mountpoint=/ai rpool/ai
<caiman:/tmp,50# df -h /ai
Filesystem             size   used  avail capacity  Mounted on
rpool/ai               685G    19K   674G     1%    /ai
<caiman:/tmp,51# md /ai/img /ai/srv
<caiman:/tmp,52# cp /data/Sun/OpenSolaris/ISO/osol-1002-124/osol-1002-124-ai-
sparc.iso /ai/img
<caiman:/tmp,53# cp /data/Sun/OpenSolaris/ISO/osol-1002-124/osol-1002-124-ai-
x86.iso /ai/img
```

Set up services:

```
<caiman:/tmp,54# installadm create-service -n 1002sparc -s \
/ai/img/osol-1002-124-ai-sparc.iso /ai/srv/osol-1002-124-ai-sparc
Setting up the target image at /ai/srv/osol-1002-124-ai-sparc ...
Registering the service 1002sparc._OSInstall._tcp.local

Detected that DHCP is not set up on this server.
If not already configured, please create a DHCP macro
named dhcp_macro_1002sparc with:
   Boot server IP (BootSrvA) : 192.168.222.47
   Boot file      (BootFile) : http://192.168.222.47:5555/cgi-bin/wanboot-cgi
If you are running Sun's DHCP server, use the following
command to add the DHCP macro, dhcp_macro_1002sparc:
   /usr/sbin/dhtadm -g -A -m dhcp_macro_1002sparc -d
:BootSrvA=192.168.222.47:BootFile=\"http://192.168.222.47:5555/cgi-
bin/wanboot-cgi\":
```

And we're in business! Here's our AI server:

```
<caiman:/tmp,140# ls -goLF /etc/netboot /tftpboot
/etc/netboot:
total 8
drwxr-xr-x   2        3 Oct 11 19:57 1002sparc/
drwxr-xr-x   3        3 Oct 11 20:15 192.168.222.0/
-rw-r--r--   1      265 Oct 11 19:57 wanboot.conf

/tftpboot:
total 1041
-rwxr-xr-x   1  139920 Oct 11 19:58 1002x86*
drwxr-xr-x   6        9 Sep 26 01:29 I86PC.OpenSolaris-1/
-rw-r--r--   1      370 Oct 11 22:14 menu.lst.1002x86
-rwxr-xr-x   1  139920 Oct 11 19:58 pxegrub.I86PC.OpenSolaris-1*
-rw-r--r--   1      130 Oct 11 19:58 rm.1002x86

<caiman:/tmp<caiman:/tmp,150# svcs -l svc:/system/install/server:default
fmri         svc:/system/install/server:default
name         Installadm Utility
enabled      true
state        online
next_state   none
state_time   Mon Oct 26 23:51:15 2009
logfile      /var/svc/log/system-install-server:default.log
restarter    svc:/system/svc/restarter:default
contract_id  86
dependency   optional_all/restart svc:/network/dns/multicast:default (online)
dependency   optional_all/none svc:/network/tftp/udp6:default (online)
dependency   optional_all/none svc:/network/dhcp-server:default (disabled)
```

(not shown: the Apache 2.2 web server that is also started, listening on port 5555)

The Big Picture
A Closer Look at Caiman
The Automated Installer
An AI Example
Random Advice
Links and Resources

An AI Example

Task is to install an SunFire V240 (SPARC) with a few extras:

- add an extra package to the target
- change some basic settings (default user and password, time zone, ...)
- monitor the installation from the console

Create a client:

```
<caiman:/tmp,49# installadm create-client -e 0:3:ba:84:23:91 -t
/ai/srv/osol-1002-124-ai-sparc -n 1002sparc
Setting up SPARC client...
Creating SPARC configuration file

Detected that DHCP is not set up on this server.
If not already configured, please create a DHCP macro
named 010003BA842391 with:
   Boot server IP (BootSrvA) : 192.168.222.47
   Boot file      (BootFile) : http://192.168.222.47:5555/cgi-bin/wanboot-cgi
If you are running Sun's DHCP server, use the following
command to add the DHCP macro, 010003BA842391:
   /usr/sbin/dhtadm -g -A -m 010003BA842391 -d
:BootSrvA=192.168.222.47:BootFile=\"http://192.168.222.47:5555/cgi-
bin/wanboot-cgi\":

Note: Be sure to assign client IP address(es) if needed
(e.g., if running Sun's DHCP server, run pntadm(1M)).
```

Manifests define the installation characteristics:

```
<caiman:/tmp,51# installadm
usage:  installadm <subcommand> <args> ...
.....
        list    [-n <svcname>]
.....
        add     -m <manifest> -n <svcname>
        remove  -m <manifest> -n <svcname>

<caiman:/tmp,52# more /var/ai/46501/AI_data/default.xml
<ai_criteria_manifest>
        <ai_embedded_manifest>
          <ai_manifest name="default">
            <ai_pkg_repo_default_authority>
                <main url="http://pkg.opensolaris.org/dev"
                   authname="opensolaris.org"/>
                <mirror url="http://repo.bb-c.de/dev"/>
            </ai_pkg_repo_default_authority>

            <!--
                By default the latest build available, in the specified IPS
                repository, is installed.
                If another build is required, the build number has
                to be appended to the 'entire' package in following
                form:

                <pkg_name="entire@0.5.11-0.build#"/>
            -->
```

Manifests define the installation characteristics:

```
                <pkg name="entire"/>
                <pkg name="SUNWcsd"/>
                <pkg name="SUNWcs"/>
                <pkg name="babel_install"/>
                <pkg name="SUNWtcsh"/>
            </ai_install_packages>
            <ai_uninstall_packages>
                <pkg name="babel_install"/>
                <pkg name="slim_install"/>
            </ai_uninstall_packages>
            <ai_auto_reboot>
                true
            </ai_auto_reboot>
        </ai_manifest>
        </ai_embedded_manifest>
        <sc_embedded_manifest name = "AI">
                <!-- <?xml version='1.0'?>
<!DOCTYPE service_bundle SYSTEM
"/usr/share/lib/xml/dtd/service_bundle.dtd.1">
<service_bundle type="profile" name="name">
  <service name="ai_properties" version="1" type="service">
    <instance name="default" enabled="true">
      <property_group name="ai" type="application">
         <propval name="username" type="astring" value="jack"/>
         <propval name="userpass" type="astring" value="9Nd/cwBcNWFZg"/>
         <propval name="description" type="astring" value="default_user"/>
         <propval name="rootpass" type="astring" value="..."/>
         <propval name="timezone" type="astring" value="MET"/>
```

Manifests define the installation characteristics:

```
                <pkg name="entire"/>
                <pkg name="SUNWcsd"/>
                <pkg name="SUNWcs"/>
                <pkg name="babel_install"/>
                <pkg name="SUNWtcsh"/>
            </ai_install_packages>
            <ai_uninstall_packages>
                <pkg name="babel_install"/>
                <pkg name="slim_install"/>
            </ai_uninstall_packages>
            <ai_auto_reboot>
                true
            </ai_auto_reboot>
        </ai_manifest>
        </ai_embedded_manifest>
        <sc_embedded_manifest name = "AI">
                <!-- <?xml version='1.0'?>
<!DOCTYPE service_bundle SYSTEM
"/usr/share/lib/xml/dtd/service_bundle.dtd.1">
<service_bundle type="profile" name="name">
  <service name="ai_properties" version="1" type="service">
    <instance name="default" enabled="true">
      <property_group name="ai" type="application">
         <propval name="username" type="astring" value="jack"/>
         <propval name="userpass" type="astring" value="9Nd/cwBcNWFZg"/>
         <propval name="description" type="astring" value="default_user"/>
         <propval name="rootpass" type="astring" value="..."/>
         <propval name="timezone" type="astring" value="MET"/>
```

Manifests define the installation characteristics:

```
                <pkg name="entire"/>
                <pkg name="SUNWcsd"/>
                <pkg name="SUNWcs"/>
                <pkg name="babel_install"/>
                <pkg name="SUNWtcsh"/>
            </ai_install_packages>
            <ai_uninstall_packages>
                <pkg name="babel_install"/>
                <pkg name="slim_install"/>
            </ai_uninstall_packages>
            <ai_auto_reboot>
                true
            </ai_auto_reboot>
        </ai_manifest>
        </ai_embedded_manifest>
        <sc_embedded_manifest name = "AI">
                <!-- <?xml version='1.0'?>
<!DOCTYPE service_bundle SYSTEM
"/usr/share/lib/xml/dtd/service_bundle.dtd.1">
<service_bundle type="profile" name="name">
  <service name="ai_properties" version="1" type="service">
    <instance name="default" enabled="true">
      <property_group name="ai" type="application">
         <propval name="username" type="astring" value="jack"/>
         <propval name="userpass" type="astring" value="9Nd/cwBcNWFZg"/>
         <propval name="description" type="astring" value="default_user"/>
         <propval name="rootpass" type="astring" value="..."/>
         <propval name="timezone" type="astring" value="MET"/>
```

Manifests define the installation characteristics:

```
                <pkg name="entire"/>
                <pkg name="SUNWcsd"/>
                <pkg name="SUNWcs"/>
                <pkg name="babel_install"/>
                <pkg name="SUNWtcsh"/>
            </ai_install_packages>
            <ai_uninstall_packages>
                <pkg name="babel_install"/>
                <pkg name="slim_install"/>
            </ai_uninstall_packages>
            <ai_auto_reboot>
                true
            </ai_auto_reboot>
        </ai_manifest>
        </ai_embedded_manifest>
        <sc_embedded_manifest name = "AI">
                <!-- <?xml version='1.0'?>
<!DOCTYPE service_bundle SYSTEM
"/usr/share/lib/xml/dtd/service_bundle.dtd.1">
<service_bundle type="profile" name="name">
  <service name="ai_properties" version="1" type="service">
    <instance name="default" enabled="true">
      <property_group name="ai" type="application">
         <propval name="username" type="astring" value="jack"/>
         <propval name="userpass" type="astring" value="9Nd/cwBcNWFZg"/>
         <propval name="description" type="astring" value="default_user"/>
         <propval name="rootpass" type="astring" value="..."/>
         <propval name="timezone" type="astring" value="MET"/>
```

Sample Installation Log

The Big Picture
A Closer Look at Caiman
The Automated Installer
An AI Example
Random Advice
Links and Resources

Random Advice

- use Caiman and especially AI, it's the future

Random Advice

- use Caiman and especially AI, it's the future
- always use the newest versions

Random Advice

- use Caiman and especially AI, it's the future
- always use the newest versions
- lern some Python, it will make things easier in the long run

Random Advice

- use Caiman and especially AI, it's the future
- always use the newest versions
- lern some Python, it will make things easier in the long run
- become familiar with XML, it's the new format for all configuration data

Random Advice

- use Caiman and especially AI, it's the future
- always use the newest versions
- lern some Python, it will make things easier in the long run
- become familiar with XML, it's the new format for all configuration data
- understand DHCP and set it up manually, don't let AI do it

Random Advice

- use Caiman and especially AI, it's the future
- always use the newest versions
- lern some Python, it will make things easier in the long run
- become familiar with XML, it's the new format for all configuration data
- understand DHCP and set it up manually, don't let AI do it
- remember: it's Work in Progress

Extras

- Distribution Constructor
- Text Installer
- Virtual Machine Constructor

Distribution Constructor

- build OpenSolaris-based distributions from a package repository
- much more flexible than the old *SUNWCreq*, *SUNWCall*, ... scheme
- driven by manifest files in XML format
- output includes boot media and network install images
- planned: build an installable image from an existing installed OpenSolaris instance

Text Based Installer

- currently prototype, not available yet
- sample implementation done by students outside of Sun
- not only recreate *gui-install*, but provide more fine-grained configuration screens
- identical functionality for SPARC and x86
- first implementation: boot from media only
- later: interactive network boot

The Big Picture
A Closer Look at Caiman
The Automated Installer
An AI Example
Random Advice
Links and Resources

Links and Resources

OpenSolaris Installation and Packaging community:

http://hub.opensolaris.org/bin/view/Community+Group+install/

OpenSolaris Project Caiman:

http://hub.opensolaris.org/bin/view/Project+caiman/

Source Code (Mercurial Repository):

```
hg clone ssh://anon@opensolaris.org/hg/caiman/slim_source
```

The Author's Link List for this Presentation:

http://www.bb-c.de/osdevcon2009/

Questions and Discussion

Questions and Discussion

Thank you!

Volker A. Brandt
Brandt & Brandt Computer GmbH
vab@bb-c.de

Questions and Discussion

Thank you!

Volker A. Brandt
Brandt & Brandt Computer GmbH
vab@bb-c.de