



**Building Appliances  
With The Red Hat Appliance Operating System  
Bryan Kearney  
Principal Application Architect, Red Hat**

# Agenda

- Appliance Overview
- Red Hat Appliance Strategy
- Example

# Appliance Defined

An appliance is a pre-configured application and operating system bundle. Configuration options are controlled for ease of use and installation.

# Appliance Management

Appliances can be managed with on-board management tools, and can integrate with corporate-wide management consoles. These options allow for easy integration of the appliance into the virtualization environment.

# Appliance Updates

Appliances are updated via standard package management tools.

# Who Uses Appliances?

Anyone who delivers Applications to customers

# What an IT Manager Wants

Improve employee productivity by providing them with the information and services they need to be effective

# What an IT Manager Wants

Minimize IT cost and complexity

# What an IT Manager Wants

Keep confidential company information secure

# What an ISV and Solution Provider Want

Grow their customer base

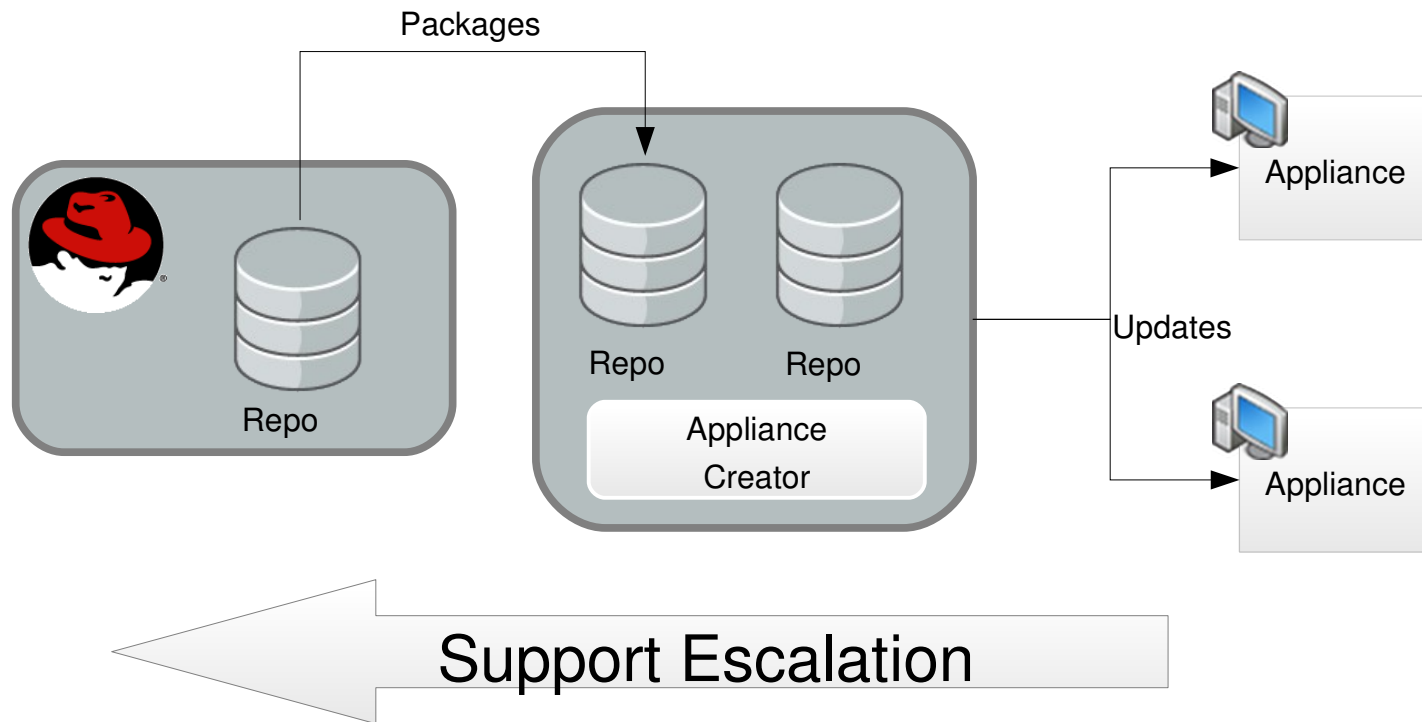
# What an ISV and Solution Provider Want

Keep costs low

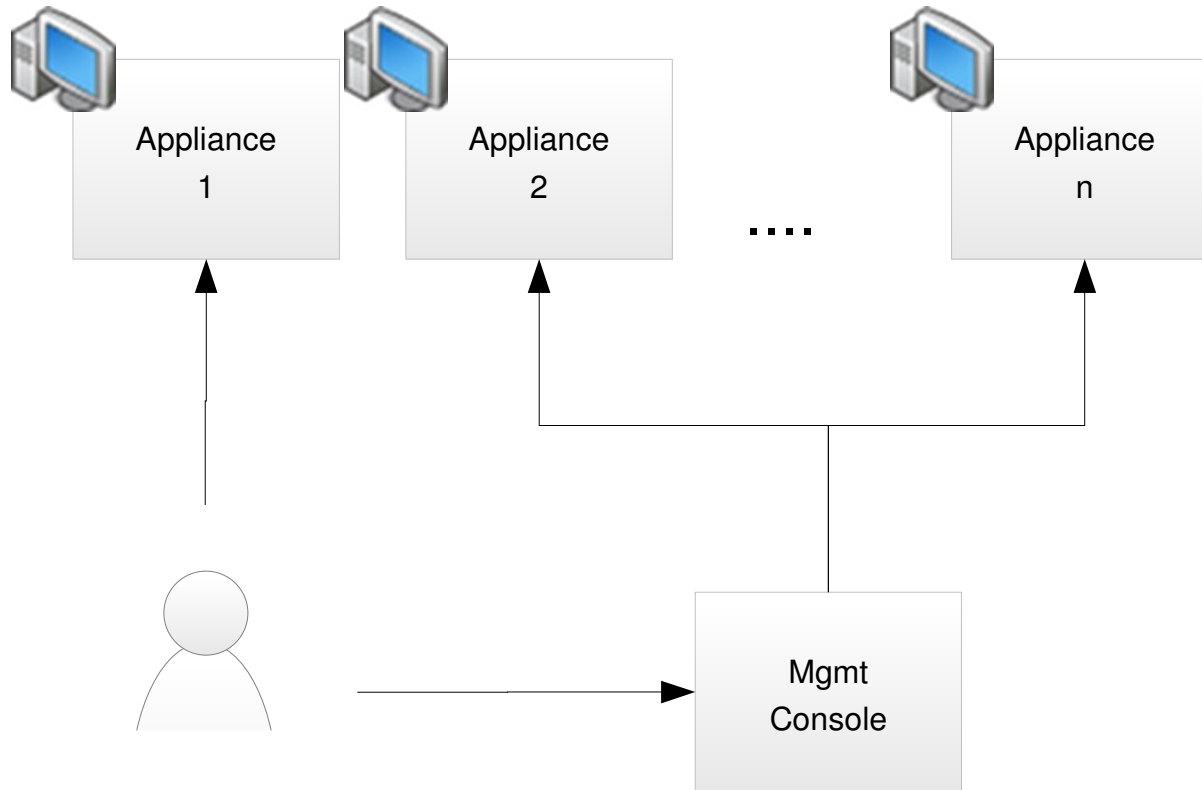
# What an ISV and Solution Provider Want

Certify Once, Deploy Anywhere

# Putting it All Together: Support and Updates



# Putting it All Together: Management



# ***THIN CRUST***

<http://www.thincrust.net>

# Appliance Operating System: AOS

- a.k.a, JEOS (Just Enough Operating System)
- Minimal package set
- Defined in a kickstart file
- Bits delivered as RPMs
- <http://www.thincrust.net/aos.html>
- We hope to provide versions via Fedora Spins and EC2 in the coming months.

# Leveraging RPMs

- Identical packaging technology for Bare Metal and Appliance products
- Ideally, identical packages to reduce build and certification costs
- Exposes existing tooling (yum, rpmdb, etc) to appliance tools such as the appliance console and update model.
- Supports all Enterprise Linux Distributions.

Plug!

Tom “Spot” Callaway gave a talk on RPMs this afternoon titled “How to Make Good RPM Packages”

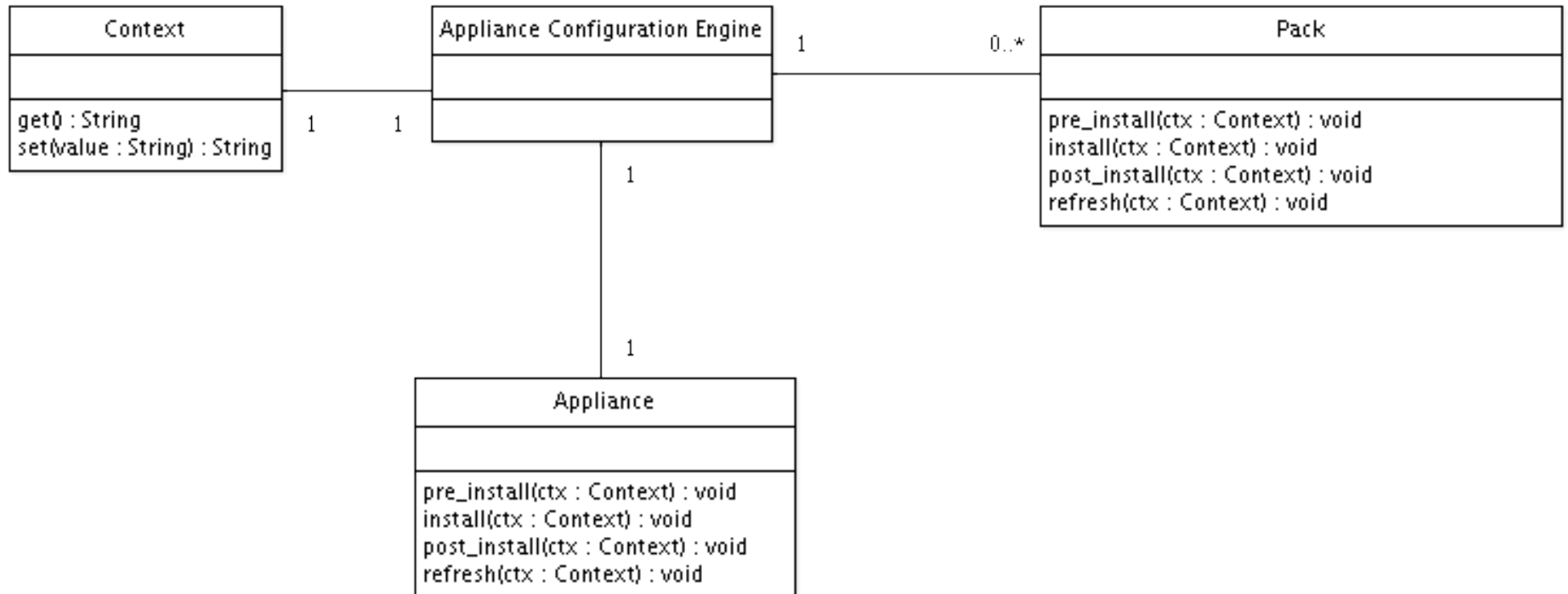
# Appliance Configuration (ACE)

- Appliance Configuration Engine is a boot time tool
- Leverages Augeas (<http://augeas.net>) to access configuration files
- Leverages Facter (<http://reductivelabs.com/projects/facter/>) to access system information
- <http://www.thincrust.net/ace.html>

Plug!

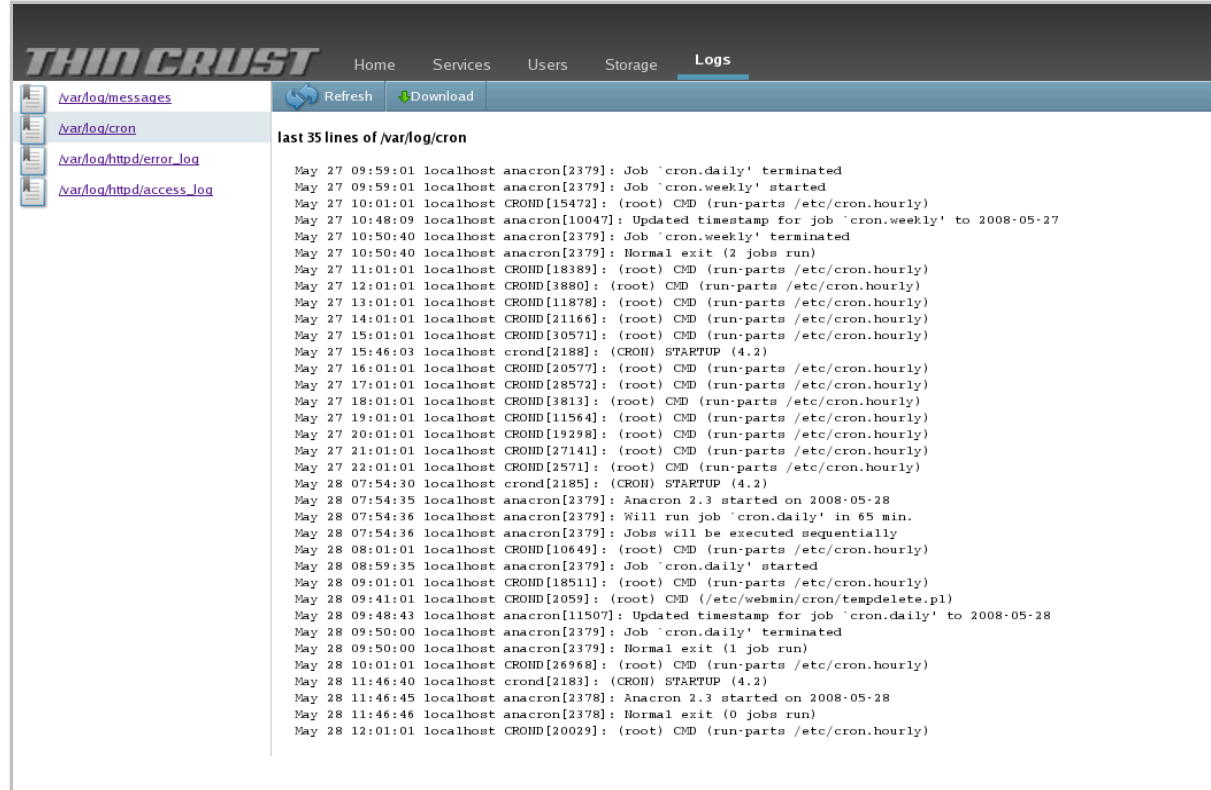
David Lutterkort will give a talk on Augeas tomorrow @ 1:30 titled “Augeas: a Linux Configuration API”

# Appliance Configuration Engine



# Configuration Console (ACC)

- Console to support management without need for shell access.

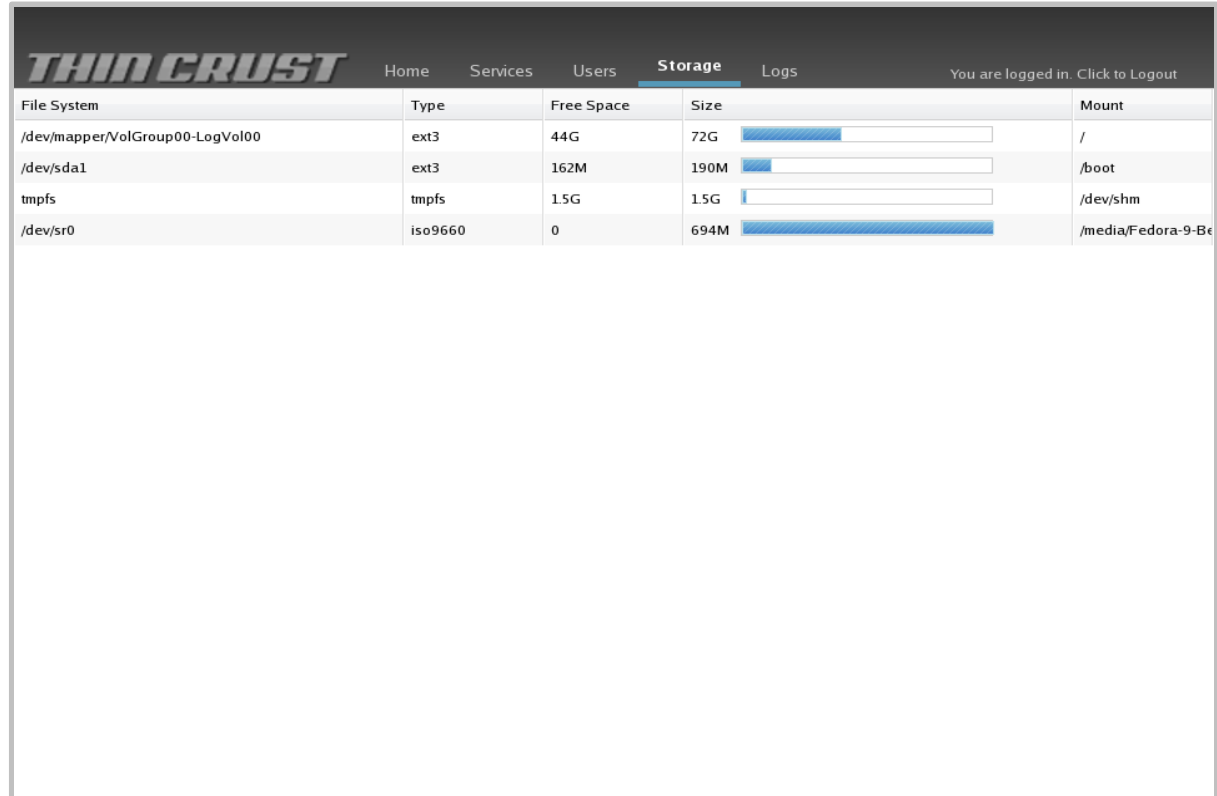


The screenshot displays the THIN CRUST Configuration Console (ACC) interface. The top navigation bar includes 'Home', 'Services', 'Users', 'Storage', and 'Logs'. The left sidebar shows a file tree with links to [/var/log/messages](#), [/var/log/cron](#), [/var/log/httpd/error\\_log](#), and [/var/log/httpd/access\\_log](#). The main content area shows the 'last 35 lines of /var/log/cron' with the following log entries:

```
May 27 09:59:01 localhost anacron[2379]: Job 'cron.daily' terminated
May 27 09:59:01 localhost anacron[2379]: Job 'cron.weekly' started
May 27 10:01:01 localhost CROHD[15472]: (root) CMD (run-parts /etc/cron.hourly)
May 27 10:48:09 localhost anacron[10047]: Updated timestamp for job 'cron.weekly' to 2008-05-27
May 27 10:50:40 localhost anacron[2379]: Job 'cron.weekly' terminated
May 27 10:50:40 localhost anacron[2379]: Normal exit (2 jobs run)
May 27 11:01:01 localhost CROHD[18389]: (root) CMD (run-parts /etc/cron.hourly)
May 27 12:01:01 localhost CROHD[3880]: (root) CMD (run-parts /etc/cron.hourly)
May 27 13:01:01 localhost CROHD[11878]: (root) CMD (run-parts /etc/cron.hourly)
May 27 14:01:01 localhost CROHD[21166]: (root) CMD (run-parts /etc/cron.hourly)
May 27 15:01:01 localhost CROHD[30571]: (root) CMD (run-parts /etc/cron.hourly)
May 27 15:46:03 localhost crond[2188]: (CRON) STARTUP (4.2)
May 27 16:01:01 localhost CROHD[20577]: (root) CMD (run-parts /etc/cron.hourly)
May 27 17:01:01 localhost CROHD[28572]: (root) CMD (run-parts /etc/cron.hourly)
May 27 18:01:01 localhost CROHD[3813]: (root) CMD (run-parts /etc/cron.hourly)
May 27 19:01:01 localhost CROHD[11564]: (root) CMD (run-parts /etc/cron.hourly)
May 27 20:01:01 localhost CROHD[19298]: (root) CMD (run-parts /etc/cron.hourly)
May 27 21:01:01 localhost CROHD[27141]: (root) CMD (run-parts /etc/cron.hourly)
May 27 22:01:01 localhost CROHD[2571]: (root) CMD (run-parts /etc/cron.hourly)
May 28 07:54:30 localhost crond[2185]: (CRON) STARTUP (4.2)
May 28 07:54:35 localhost anacron[2379]: Anacron 2.3 started on 2008-05-28
May 28 07:54:36 localhost anacron[2379]: Will run job 'cron.daily' in 65 min.
May 28 07:54:36 localhost anacron[2379]: Jobs will be executed sequentially
May 28 08:01:01 localhost CROHD[10649]: (root) CMD (run-parts /etc/cron.hourly)
May 28 08:59:35 localhost anacron[2379]: Job 'cron.daily' started
May 28 09:01:01 localhost CROHD[18511]: (root) CMD (run-parts /etc/cron.hourly)
May 28 09:41:01 localhost CROHD[2059]: (root) CMD (/etc/webmin/cron/tempdelete.pl)
May 28 09:48:43 localhost anacron[11507]: Updated timestamp for job 'cron.daily' to 2008-05-28
May 28 09:50:00 localhost anacron[2379]: Job 'cron.daily' terminated
May 28 09:50:00 localhost anacron[2379]: Normal exit (1 job run)
May 28 10:01:01 localhost CROHD[26968]: (root) CMD (run-parts /etc/cron.hourly)
May 28 11:46:40 localhost crond[2183]: (CRON) STARTUP (4.2)
May 28 11:46:45 localhost anacron[2378]: Anacron 2.3 started on 2008-05-28
May 28 11:46:46 localhost anacron[2378]: Normal exit (0 jobs run)
May 28 12:01:01 localhost CROHD[20029]: (root) CMD (run-parts /etc/cron.hourly)
```

# Configuration Console

- Designed to sit alongside the applications tool.
- API is provided to provide integration with external monitoring tools.



The screenshot shows the THIN CRUST Storage Configuration Console. The interface includes a navigation menu with 'Home', 'Services', 'Users', 'Storage' (selected), and 'Logs'. A user is logged in, with a 'Logout' link. The main content area displays a table of file systems with columns for File System, Type, Free Space, Size, and Mount. Each row includes a progress bar representing the usage of the file system.

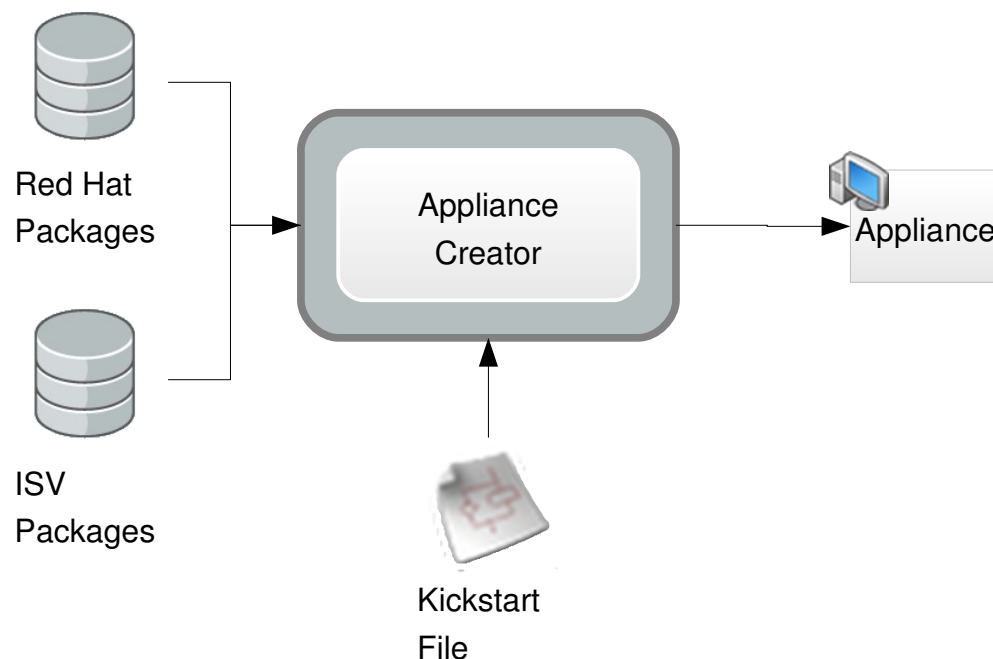
File System	Type	Free Space	Size	Mount
/dev/mapper/VolGroup00-LogVol00	ext3	44G	72G	/
/dev/sda1	ext3	162M	190M	/boot
tmpfs	tmpfs	1.5G	1.5G	/dev/shm
/dev/sr0	iso9660	0	694M	/media/Fedora-9-Be

# Appliance Creation Tool

- Command Line tool to create raw image files and libvirt config files
- <http://www.thincrust.net/tooling.html>
- Can build F8, F9, and Rawhide with no local trees
- Red Hat Enterprise Linux requires a local tree

# Appliance Creation Tool

- Live-cd based tooling to create images on local machine



Plug!

Jeremy Katz gave a talk on live cds this morning titled "Fedora – Images and Live CDs"

# Appliance Operating System / Creation Tool

- Future enhancements will include
  - Whitelisting
  - Integration with conversion tooling

# Conversion Tooling

- Tools which provide the following conversions
  - Libvirt based raw images into VMWare
  - VMWare into libvirt images
  - Libvirt based raw images into EC2 (coming soon)

Plug!

Mike Ferris will give a talk Friday @ 10:15 titled  
“Cloud Computing with Red Hat Enterprise Linux on Amazon EC2”

# Example



<http://www.sugarcrm.com>

## Example: Build an RPM

- Download the tarball from sourceforge
- Create a spec file which follows the file system hierarchy and Fedora packaging guidelines
- RPM contains files only, no %post sections

Plug!

Karsten Wade will give a talk Thursday @ 2:45 titled  
“Fedora Packages for Red Hat Enterprise Linux”

# Good RPMs

## From the File System Hierarchy

- /var gets temporary, variable data
- /user gets shared, read only data
- /lib for libraries
- /etc for configuration
- When all else fails, /opt

## Other Good Standards from Fedora

- <http://fedoraproject.org/wiki/Packaging/Guidelines>
- <http://fedoraproject.org/wiki/Packaging/Java>
- <http://fedoraproject.org/wiki/Packaging/Python>
- <http://fedoraproject.org/wiki/Packaging/Ruby>

# Example: Create Recipe

```
# ACE appliance for the Open Source Sugar CRM product
class SugarAppliance < ApplianceDefinition

    appliance_version "0.0.1"

    appliance_name "Sugar Appliance"

    packs "banners", "firewall", "httpd", "basic_site", "mysql",
        "php"

    context_variables Mysql::MYSQL_PASSWORD_KEY => "sugarcrm",
        "sugar_admin_password" => "admin"
```

# Example: Create Recipe

```
# Install lifecycle step
def install
  # Set up the symlinks from the rpm into httpd server
  ln_sf("/var/lib/sugarcrm/htdocs",
        "/var/www/html/sugarcrm")

  # Open up port 80
  open_port(80)

  # Create the main page
  create_main_page("sugarAppliance/sugarContent.erb")
```

## Example: Create Recipe

```
# Set the php memory limit to 32M
php_memory_limit="40M"
php_upload_limit="6M"

# Copy the httpd conf file over
cp_resource("sugarAppliance/sugarcrm-httpd.conf",
  "/etc/httpd/conf.d")
cp_resource("sugarAppliance/config.php", "/var/lib/sugarcrm/htdo
  cs/config.php")

end
```

# Example: Create Recipe

```
# Call the sugar silent install process
def post_install
  create_file_from_template("/var/lib/sugarcrm/htdocs/config_
    si.php",
    "sugarAppliance/config_si.php.erb", 0755)
  url = URI.parse("http://localhost/sugarcrm/install.php
    goto=SilentInstall&cli=true")
  res = Net::HTTP.get_response(url)
  ACE.log.info(res)
end

end
```

# Example: Templating

```
<a class="button" href="/sugarcrm" float="right"><span>Access SugarCRM Now!
</span></a>
```

The Sugar CRM appliance is now configured and ready for use.

```
<p/>
```

The following information is needed to utilize this appliance:

```
<br/>
```

```
<table border="0" cellpadding="2" >
```

```
  <tr><th>Parameter</th><td></td><th>Value</th></tr><tr>
```

```
  <tr><th>Root Password</th><td></td><th>thincrust</th></tr>
```

```
  <tr><th>Database User</th><td></td><th>root</th></tr>
```

```
  <tr><th>Database Password</th><td></td><th><%= mysqlrootPassword %></th></tr>
```

```
  <tr><th>IP Address</th><td></td><th><%= ipaddress %></th></tr>
```

```
</table>
```

# Create the Kickstart File

```
lang C
keyboard us
timezone US/Eastern
auth --useshadow --enablemd5
selinux --disabled
firewall --disabled
bootloader --timeout=1 --append="acpi=force"
network --bootproto=dhcp --device=eth0 --onboot=on

# Root password is thincrust
rootpw --iscrypted $1$uw6MV$m6VtUWPed4SqgoW6fKfTZ/

# Partition Information. Change this as necessary
part / --size 500 --fstype ext3 --ondisk sda
```

# Create the Kickstart File

```
# Include the repositories
%include ../../aos/kickstarts/repo-f8.ks
%include repo-thincrust.ks

# Add all the packages after the base packages
%packages --excludedocs --nobase
%include ../../aos/kickstarts/base-pkgs.ks
%include ace-pkgs.ks
sugarcrm
sugarAppliance
%end

# Add custom post scripts after the base post.
%post
%include ace-post.ks
%end
```

## Example: Build the Appliance

- Build the appliance

```
appliance-creator -v  
  --cache /home/bkearney/cache  
  -n sugar-aos  
  aos-sugar.ks
```

- Run the appliance

```
virt-image sugar-aos.xml
```

<http://www.thincrust.net/ace-examples.html>

# Contributing

- Getting the code: <http://www.thincrust.net/help.html>
- <http://www.redhat.com/mailman/listinfo/thincrust-devel>
- Email: [bkearney@redhat.com](mailto:bkearney@redhat.com)

Questions?

