Inside the Viz Roll

Mason Katz
mjk@sdsc.edu
\{ base, hpc, kernel, viz \}
Early Work: NCSA

- **LCD Cluster**
  - Custom framing
  - One PC / tile
  - Portable (luggable)
  - SC 2001 Demo

- **NCSA Software**
  - Pixel Blaster
  - Display Wall In-A-Box
    - OSCAR based
    - Never fully released
NCMIR

- Using Rocks
- Hand configured a visualization cluster
- “Administered the machine to the point of instability”
  - David Lee
- Automation is needed
COTS Vis: GeoWall

- **LCD Clusters**
  - One PC / tile
  - Gigabit Ethernet
  - Optional Stereo Glasses
  - Portable
  - Commercial Frame (Reason)

- **Applications**
  - Large remote sensing
  - Volume Rendering
  - Seismic Interpretation
  - Brain mapping (NCMIR)

- **Electronic Visualization Lab**
  - Jason Leigh (UIC)
OptIPortal (SAGE)
One Node per Display
OptIPortal
Nodes Behind the Wall
Use of OptIPortal to Interactively View Microbial Genome (CGView)

Acidobacteria Bacterium Ellin345 (NCBI)
Soil Bacterium 5.6 Mb

Source: Raj Singh, UCSD
Viz Roll Source Tree
Bootstrapping

- Roll need to be build on a Rocks Frontend
- The Viz Roll needs to be build on a Rocks Frontend that has the Viz Roll install
- Is that a problem?
bootstap.sh

- Every Roll has one
- Used install building dependencies
  - From OS
  - From Roll install
- Simple shell script
- Easy to read
For Example

```
#!/bin/sh
#
# $Id: bootstrap.sh,v 1.18 2009/05/01 19:07:24 mj Exp $
. $ROLESSROOT/etc/bootstrap-functions.sh
install_os_packages viz-server
install libdmx
install libdmx-devel
compile SRPMS
install glut
install glut-devel
install joy2key
install joystick
compute_and_install cmake
compile_and_install foundation-sdl
compile_and_install foundation-sdl-image
compile_and_install foundation-sdl-mixer
compile_and_install foundation-sdl-net
compile_and_install foundation-sdl-ttf
compile_and_install chromium
compile_and_install libvox1394
compile_and_install openal
compile_and_install frealut
compile_and_install plib
compile_and_install SimGear
compile_and_install libmadv
compile_and_install libdvdrdd
compile_and_install faac
compile_and_install faad2
compile_and_install ffmpeg
compile_and_install mpeg2dec
compile_and_install wx
compile_and_install portaudio
compile_and_install glew
compile quanta
install QUANTA
```

© 2009 UC Regents
makefiles

- Every Rocks Frontend has all our build environment installed

- Roll Makefile almost empty
  - Just copy from another Roll
  - No need to change anything

- Version.mk can
  - Set version
  - Set color (graph)
  - A few other minor things
Viz Roll Sub-Graph

- Single Graph File
- 9 Nodes Files
Source Code

- Almost 50 packages
Dozen+ Command

# rocks list roll command viz
COMMAND
create viz layout
disable chromium
disable hidebezel
enable chromium
enable hidebezel
list dmx layout
list host xconfig
list viz layout
start chromium
start dmx
start xlogo
stop xlogo
sync viz
LCD Bezels
rocks enable hidebezels

- Draws pixels behind the bezels (mullions) of the LCD monitors
- Calculated offset for TwinView and normal modes
- Reset the X11 configuration on all nodes
- Great mode for moving graphics
import rocks.commands.enable

import os

class Command(rocks.commands.enable.command):

    """
    Enable Bezel Hiding mode.
    """

    <example cmd="enable hidebezels">
    </example>

    """

    MustBeRoot = 0

def run(self, params, args):

    os.system('touch ~/.hidebezels')

    # If the database videowall layout has two (or more) entries
    # for the same host and card we know we are in twinview
    # mode. In this case we need to reconfigure and restart
    # X11 for this host.

    self.db.execute("""select n.name, v.cardid
    from nodes n, videowall v where v.node=n.id""")

dict = {}
for key in self.db.fetchall():
    if dict.has_key(key):
        dict[key] = 1  # TwinView host
    else:
        dict[key] = 0  # NonTwinView host (so far)

for (host, card) in dict.keys():
    if dict[host, card]:
        os.system('ssh -f '
    '%%s /usr/X11R6/bin/xrandr -d :0 -s 1'
    % host)
rocks disable hidebezel

- All pixels are drawn and the bezels break apart the image
- Removes any offset from previous mode
- Resets the X11 configuration on all nodes
- Great for static images and text
import rocks.commands.enable
import os

class Command(rocks.commands.disable.command):
    """
    Disable Bezel Hiding mode.
    """
    <example cmd="disable hidebezels">
        </example>
    """

    MustBeRoot = 0

def run(self, params, args):
    os.system('/bin/rm ~/.hidebezels')

    # If the database videowall layout has two (or more) entries
    # for the same host and card we know we are in twinview
    # mode. In this case we need to reconfigure and restart
    # X11 for this host.

    self.db.execute("""select n.name, v.cardid
        from nodes n, videowall v where v.node=n.id"""")
    dict = {}
    for key in self.db.fetchall():
        if dict.has_key(key):
            dict[key] = 1  # TwinView host
        else:
            dict[key] = 0  # NonTwinView host (so far)

    for (host, card) in dict.keys():
        if dict[(host, card)]:
            os.system('ssh -f '
              '%%s /usr/X11R6/bin/xrandr -d :0 -s 0'
              % host)
Viz Roll Sub-Graph
Frontend Profile
<kickstart interface="public">
  <description>
  Visualization Cluster Frontend support
  </description>
  <package>curl</package>
  <package>curl-devel</package>
  ...
  <package arch="i386">cmake</package>
  <post>
  /opt/rocks/bin/rocks add appliance tile \\
    membership=Tile node=viz-tile
  /opt/rocks/bin/rocks set attr HideBezels false
  /opt/rocks/bin/rocks set appliance attr tile x11 true
  </post>
</kickstart>
<kickstart>

<description>
Visualization Cluster Database Tables
</description>

<copyright>
Copyright (c) 2000 - 2009 The Regents of the
All rights reserved. Rocks(r) v5.2  www.rocks
</copyright>

<post>

<file name="/tmp/viz-tables.sql">
DROP TABLE IF EXISTS videowall;
CREATE TABLE videowall (  
    Node    int(11) NOT NULL default '0'
    Display  varchar(8) NOT NULL default
    Resolution varchar(32) NOT NULL default
    X      int(11) NOT NULL default '0'
    Y      int(11) NOT NULL default '0'
    LeftBorder  int(11) NOT NULL default '75'
    RightBorder int(11) NOT NULL default '75'
    TopBorder   int(11) NOT NULL default '75'
    BottomBorder int(11) NOT NULL default '75'
) TYPE=MyISAM;
</file>

/opt/rocks/bin/mysql --user=root --password='' cluster &lt; /tmp/viz-  
tables.sql
</post>

</kickstart>
Ordering

- Added DB Schema
- But, need DB up first

<order head="database-data">
  <tail>viz-db</tail>
</order>
Another Look at XML

<graph>
  <edge from="client">
    <to arch="i386,x86_64">grub-client</to>
    <to>autofs-client</to>
    <to>installclass-client</to>
  </edge>
</graph>
Partial Ordering

<graph>
  <order head="autofs-client" tail="client"/>
  <edge from="client">
    <to arch="i386,x86_64">grub-client</to>
    <to>autofs-client</to>
    <to>installclass-client</to>
  </edge>
</graph>

- Forces autosf-client <post> section to run before client’s <post> section
- In order graph traversal enforces a partial ordering
- Applying standard graph theory to system installation
<kickstart interface="public">
  <description>
  Visualization Cluster support
  </description>

  <package>rocks-viz</package>
  <package>fvwm</package>
  <package>nvidia-driver</package>
  <package>freeglut</package>
  <package>chromium</package>
  <package>Cg</package>
  <package>wx</package>

  ...

  <post>
  <!-- Create the viz account on each node (avoid 411) -->
  /usr/sbin/useradd -M -u402 -c "Viz X11 Owner" -d /opt/viz/etc viz
  chown -R viz.viz /opt/viz/etc
  /sbin/chkconfig --add nvidia

  <file name="/etc/inittab"/>
  gawk \
  '/id:3:initdefault:/ { print "id:5:initdefault:"; next; } \n   { print; }' /etc/inittab &gt; /tmp/inittab
  mv /tmp/inittab /etc/inittab

  </post>
</kickstart>
useradd

- Create a new user named viz
- Used to auto-login X11 and xhost+ tiles
- Home is /opt/viz/etc
  - Directory is local to every node
  - Required for .xauthority files
- UID is hard coded
  - Avoids 411 lookups during install
  - Roll developers can request a RESERVED-UID
# Register all UIDs created by Rolls here. Third party developers may request an ID allocation from devel@rocksclusters.org.

<table>
<thead>
<tr>
<th>ID</th>
<th>NAME</th>
<th>SOURCE</th>
<th>ROLL</th>
<th>NOTES</th>
</tr>
</thead>
<tbody>
<tr>
<td>400</td>
<td>sge</td>
<td>sge</td>
<td></td>
<td></td>
</tr>
<tr>
<td>401</td>
<td>globus</td>
<td>grid</td>
<td></td>
<td></td>
</tr>
<tr>
<td>402</td>
<td>viz</td>
<td>viz</td>
<td></td>
<td>owner of X11 session</td>
</tr>
<tr>
<td>403</td>
<td>Rocksdb</td>
<td>base</td>
<td></td>
<td>foundation mysql user</td>
</tr>
<tr>
<td>404</td>
<td>Biouser</td>
<td>bio</td>
<td></td>
<td>user associated with the bioroll</td>
</tr>
<tr>
<td>405</td>
<td>jboss</td>
<td>java</td>
<td></td>
<td>jboss not started by default</td>
</tr>
<tr>
<td>406</td>
<td>gxmap</td>
<td>gx-map</td>
<td></td>
<td></td>
</tr>
<tr>
<td>407</td>
<td>Condor</td>
<td>condor</td>
<td></td>
<td>condor master user</td>
</tr>
<tr>
<td>408</td>
<td>pbs</td>
<td>pbs</td>
<td></td>
<td>pbs batch</td>
</tr>
<tr>
<td>409</td>
<td>postgres</td>
<td>postgres8</td>
<td></td>
<td>postgres version 8 roll</td>
</tr>
<tr>
<td>410</td>
<td>cvs</td>
<td>cvs-server</td>
<td></td>
<td></td>
</tr>
<tr>
<td>411</td>
<td>cvsanon</td>
<td>cvs-server</td>
<td></td>
<td></td>
</tr>
<tr>
<td>412</td>
<td>tomcat</td>
<td>java</td>
<td></td>
<td>tomcat not started by default</td>
</tr>
<tr>
<td>413</td>
<td>nagios</td>
<td>nagios</td>
<td></td>
<td></td>
</tr>
<tr>
<td>414</td>
<td>ssoadmin</td>
<td>camera</td>
<td></td>
<td>SSO admin user</td>
</tr>
<tr>
<td>415</td>
<td>wsuser</td>
<td>camera</td>
<td></td>
<td>ws user for dropbox</td>
</tr>
<tr>
<td>416</td>
<td>hg</td>
<td>------</td>
<td></td>
<td>mercurial user</td>
</tr>
<tr>
<td>417</td>
<td>boinc</td>
<td>boinc</td>
<td></td>
<td>boinc user</td>
</tr>
<tr>
<td>418</td>
<td>moab</td>
<td>moab</td>
<td></td>
<td>moab scheduler user</td>
</tr>
<tr>
<td>419</td>
<td>zfsuser</td>
<td>thumper-conf</td>
<td></td>
<td>ZFS Replication user</td>
</tr>
</tbody>
</table>
<kickstart>
<description>
Increase network buffers
</description>
<post>
<file name="/etc/sysctl.conf"/>
<file name="/tmp/sysctl.awk">
/net.core.rmem_max/ { next; }
/net.core.wmem_max/ { next; }
/net.ipv4.tcp_rmem/ { next; }
/net.ipv4.tcp_wmem/ { next; }
/net.ipv4.route.flush/ { next; }
/net.core.rmem_max/ { next; }
print; }
END {
print "net.core.rmem_max = 16777216";
print "net.core.wmem_max = 16777216";
print "net.ipv4.tcp_rmem = 4096 87380 16777216";
print "net.ipv4.tcp_wmem = 4096 16384 16777216";
print "net.ipv4.route.flush = 1";
}
</file>
awk -f /tmp/sysctl.awk /etc/sysctl.conf &gt; /tmp/sysctl.conf
cp /tmp/sysctl.conf /etc/sysctl.conf
rm /tmp/sysctl.awk /tmp/sysctl.conf
</post>
</kickstart>
Deep Pipes

- Performance Tuning
- Glxgears was choppy
- Re-wrote Chromium network stack
- Still broken
- Months later increased network buffers
- Everything worked
<kickstart>

<description>
Common Setup for all SAGE nodes
</description>

<package>wx</package>

<package>libtool-libs</package>
<package>libtool</package>

<package>mpeg2dec</package>
<package>libraw1394</package>
<package>libiec61883</package>
<package>QUANTA</package>
<package>sage</package>

</kickstart>
viz-sdl

```xml
<kickstart>
  <description>
      SDL stuff
  </description>
  <!-- SDL needs stuff we trim off the base CD -->
  <package>arts</package>
  <package>qt</package>
  <!-- SDL is now in the foundation -->
  <package>foundation-sdl</package>
  <package>foundation-sdl-image</package>
  <package>foundation-sdl-mixer</package>
  <package>foundation-sdl-net</package>
  <package>foundation-sdl-ttf</package>
</kickstart>
```
Tile Profile
<kickstart interface="public">
  
  <description>
  Display Node
  </description>

  <post>
    <!-- setup autologin for viz user -->
    <file name="/etc/gdm/custom.conf">
      [daemon]
      TimedLoginEnable=true
      TimedLogin=viz
      TimedLoginDelay=5

      [security]
      DisallowTCP=false
    </file>
  </post>

</kickstart>
viz-client
viz-base ...
client -> x11

- Conditional Edge from Base Roll
- Uses x11 attribute
- Set in viz-server

<edge from="client" to="x11" cond="x11"/>
<kickstart>
<description>
Patch the nVidia driver to make it work in dom0
</description>
<package>nvidia-xen</package>
</kickstart>

<file name="/opt/viz/drivers/nvidia/install-driver" perms="755">
<![CDATA[
#!/bin/bash
if [ -e /usr/src/linux-2.6 ]
then
  ln -s /usr/src/linux-2.6 /usr/src/linux
  elif [ -e /usr/src/linux-2.4 ]
  then
    ln -s /usr/src/linux-2.4 /usr/src/linux
  fi
/opt/viz/drivers/nvidia/NVIDIA*run --no-network -x > /dev/null 2>&1
cd NVIDIA*/usr/src/nv
zcat /opt/viz/drivers/nvidia/nvidia_xenpatch.gz | \n  patch --forward -p4 --quiet > /dev/null 2>&1
make SYSSRC=/usr/src/linux module > /dev/null 2>&1
install -D -o root -g root -m 0644 nvidia.ko \n  /lib/modules/\$uname -r'/video/nvidia.ko
depmod -a
cd ..:/..:/...
./nvidia-installer --no-kernel-module -s > /dev/null 2>&1
]]></CDATA>
</file>
</post>
It’s Python Time!
Motivation

- Lack of consistency in Rocks commands
  - add-extra-nic (15 flags)
  - 411put
  - rocks-dist
  - dbreport (~ a dozen reports)

- Extensible to other groups
  - How do I add a flag to an existing command?
  - How do I add a new command?
  - How do I document my command?
Verb Based

- “add”, “set”, “enable”, …
  - Modify the cluster database
- “list”, “dump”, “report”
  - Inspect the cluster database
- About 20 verbs in the command line so far
- You can even add your own

© 2009 UC Regents
Grammar

- rocks <verb> <object…> <subject> <params…>
  - Object is general to specific
    - “host” “interface”
    - “network” “subnet”
    - “viz” “layout”
  - Subject is typed
    - host
    - appliance
    - Network

- Example:

  ```
  # rocks list host interface tile-0-0
  SUBNET  IFACE  MAC               IP             NETMASK   GATEWAY  MODULE  NAME
  private eth0  00:13:72:ba:be:42 10.255.255.254 255.0.0.0 ------- tg3    tile-0-0
  ------- eth1  00:0a:5e:1a:6d:64 -------------- --------- ------- skge   --------
  ```

© 2009 UC Regents
Implementation

◆ Python
  ✅ Similar to existing dbreport code
  ✅ Very small modules

◆ Command line is identical to the directory hierarchy
  ✅ Verbs are directories
  ✅ Objects are directories
  ✅ Subjects are __init__.py files

◆ Commands are added by adding directories
rocks add host pxeaction

network
- __init__.py
- roll
- __init__.py
- var
- __init__.py

config
- __init__.py

create

mirror

__init__.py

appliance
- __init__.py

distribution
- __init__.py

interface
- __init__.py

pxeaction
- __init__.py
Rolls Can Add Commands

- Similar to the configuration graph
- Rolls can add command line
  - Files : commands
  - Directories : verbs and objects
- Think hard before adding another verb
add

- Creates new entries in the cluster database

- Examples:
  - Hosts
  - Appliances
  - Rolls
rocks add distribution

```python
import rocks.commands

class Command(rocks.commands.DistributionArgumentProcessor,
               rocks.commands.add.command):
    ""
    Add a distribution specification to the database.
    ""

    <arg type='string' name='distribution'>
    Name of the new distribution.
    </arg>

    <example cmd='add distribution rocks-dist'>
    Adds the distribution named "rocks-dist" into the database.
    </example>

    ""

    def run(self, params, args):
        if len(args) != 1:
            self.abort("must supply one distribution")
            dist = args[0]

        if dist in self.getDistributionNames():
            self.abort("distribution "%s" exists" % dist)

        self.db.execute("""insert into distributions (name) values ('%s')""" % dist)
```

© 2009 UC Regents
dump

- Returns cluster database information in the form of rocks command lines
- Examples:
  - Hosts
  - Network
- Same as –dump flag on insert-ethers
rocks dump host

# rocks dump host
/opt/rocks/bin/rocks add host vizagra cpus=1 rack=0 rank=0 membership="Frontend"
/opt/rocks/bin/rocks add host tile-0-1 cpus=2 rack=0 rank=0 membership="Tile"
/opt/rocks/bin/rocks add host tile-0-0 cpus=2 rack=0 rank=1 membership="Tile"
/opt/rocks/bin/rocks add host tile-0-2 cpus=2 rack=0 rank=2 membership="Tile"
/opt/rocks/bin/rocks add host tile-0-3 cpus=2 rack=0 rank=3 membership="Tile"
/opt/rocks/bin/rocks add host tile-1-3 cpus=2 rack=1 rank=0 membership="Tile"
/opt/rocks/bin/rocks add host tile-1-2 cpus=2 rack=1 rank=1 membership="Tile"
/opt/rocks/bin/rocks add host tile-1-1 cpus=2 rack=1 rank=2 membership="Tile"
/opt/rocks/bin/rocks add host tile-1-0 cpus=2 rack=1 rank=3 membership="Tile"
/opt/rocks/bin/rocks add host tile-2-0 cpus=2 rack=2 rank=0 membership="Tile"
/opt/rocks/bin/rocks add host tile-2-1 cpus=2 rack=2 rank=1 membership="Tile"
/opt/rocks/bin/rocks add host tile-2-2 cpus=2 rank=2 rank=2 membership="Tile"
/opt/rocks/bin/rocks add host tile-2-3 cpus=2 rank=2 rank=3 membership="Tile"
/opt/rocks/bin/rocks add host tile-3-0 cpus=2 rank=3 rank=0 membership="Tile"
/opt/rocks/bin/rocks add host tile-3-1 cpus=2 rank=3 rank=1 membership="Tile"
/opt/rocks/bin/rocks add host tile-3-2 cpus=2 rank=3 rank=2 membership="Tile"
/opt/rocks/bin/rocks add host tile-3-3 cpus=2 rank=3 rank=3 membership="Tile"
/opt/rocks/bin/rocks add host tile-4-0 cpus=2 rank=4 rank=0 membership="Tile"
/opt/rocks/bin/rocks add host tile-4-1 cpus=2 rank=4 rank=1 membership="Tile"
/opt/rocks/bin/rocks add host tile-4-2 cpus=2 rank=4 rank=2 membership="Tile"
/opt/rocks/bin/rocks add host tile-4-3 cpus=2 rank=4 rank=3 membership="Tile"
rocks dump host

```python
import os
import sys
import string
import rocks.commands

class command(rocks.commands.HostArgumentProcessor,
               rocks.commands.dump.command):
    def run(self, params, args):
        for host in self.getHostnames(args):
            self.db.execute("select 
                n.cpus, n.rank, n.name 
                from nodes n, memberships m where 
                n.membership=m.id and n.name=""%s"" % host")
            (cpus, rank, membership) = self.db.fetchone()
            self.dump("add host %s cpus=%s rank=%s membership=%s" %
                        (host, cpus, rank, membership))
```
list

- Reports information in human readable format
- No side-effects on the database
- Examples:
  - Hosts
  - Appliances
  - Rolls
### rocks list host

```plaintext
# rocks list host

<table>
<thead>
<tr>
<th>HOST</th>
<th>MEMBERSHIP</th>
<th>CPUS</th>
<th>RACK</th>
<th>RANK</th>
<th>COMMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>vizagra:</td>
<td>Frontend</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>--------</td>
</tr>
<tr>
<td>tile-0-1:</td>
<td>Tile</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>--------</td>
</tr>
<tr>
<td>tile-0-0:</td>
<td>Tile</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>--------</td>
</tr>
<tr>
<td>tile-0-2:</td>
<td>Tile</td>
<td>2</td>
<td>0</td>
<td>2</td>
<td>--------</td>
</tr>
<tr>
<td>tile-0-3:</td>
<td>Tile</td>
<td>2</td>
<td>0</td>
<td>3</td>
<td>--------</td>
</tr>
<tr>
<td>tile-1-3:</td>
<td>Tile</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>--------</td>
</tr>
<tr>
<td>tile-1-2:</td>
<td>Tile</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>--------</td>
</tr>
<tr>
<td>tile-1-1:</td>
<td>Tile</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>--------</td>
</tr>
<tr>
<td>tile-1-0:</td>
<td>Tile</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>--------</td>
</tr>
<tr>
<td>tile-2-0:</td>
<td>Tile</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>--------</td>
</tr>
<tr>
<td>tile-2-1:</td>
<td>Tile</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>--------</td>
</tr>
<tr>
<td>tile-2-2:</td>
<td>Tile</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>--------</td>
</tr>
<tr>
<td>tile-2-3:</td>
<td>Tile</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>--------</td>
</tr>
<tr>
<td>tile-3-0:</td>
<td>Tile</td>
<td>2</td>
<td>3</td>
<td>0</td>
<td>--------</td>
</tr>
<tr>
<td>tile-3-1:</td>
<td>Tile</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>--------</td>
</tr>
<tr>
<td>tile-3-2:</td>
<td>Tile</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>--------</td>
</tr>
<tr>
<td>tile-3-3:</td>
<td>Tile</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>--------</td>
</tr>
<tr>
<td>tile-4-0:</td>
<td>Tile</td>
<td>2</td>
<td>4</td>
<td>0</td>
<td>--------</td>
</tr>
<tr>
<td>tile-4-1:</td>
<td>Tile</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>--------</td>
</tr>
<tr>
<td>tile-4-2:</td>
<td>Tile</td>
<td>2</td>
<td>4</td>
<td>2</td>
<td>--------</td>
</tr>
<tr>
<td>tile-4-3:</td>
<td>Tile</td>
<td>2</td>
<td>4</td>
<td>3</td>
<td>--------</td>
</tr>
</tbody>
</table>
```

© 2009 UC Regents
rocks list host

```python
import rocks.commands

class command(rocks.commands.HostArgumentProcessor,
               rocks.commands.list.command):
    pass

class Command(command):
    """
    List the membership, CPU count, physical position info and comment for
    a list of hosts.
    """

    <arg optional='1' type='string' name='host' repeat='1'>
    Zero, one or more host names. If no host names are supplied, info about
    all the known hosts is listed.
    </arg>

    <example cmd='list host compute-0-0'>
    List info for compute-0-0.
    </example>

    <example cmd='list host'>
    List info for all known hosts.
    </example>

    """

def run(self, params, args):
    self.beginOutput()
    for host in self.getHostNames(args):
        self.db.execute("select m.name, n.cpus, n.rank, n.rank, n.comment from
data n, memberships m where
   n.membership=m.id and n.name='%s'" % host)
        self.addOutput(host, self.db.fetchone())
    self.endOutput(header=['host', 'membership',
                          'cpus', 'rank', 'rank', 'comment'])
```
**set**

- Modifies entries in the cluster database
- Examples:
  - Network Interfaces
  - Appliance Assignment
  - Rack / Rank
- **add-extra-nic**
  - Rocks add host interface
  - Rocks set host interface
add-extra-nic is now ...

◆ **Start**

```bash
# rocks list host interface compute-1-1
```

<table>
<thead>
<tr>
<th>SUBNET</th>
<th>IFACE</th>
<th>MAC</th>
<th>IP</th>
<th>NETMASK</th>
<th>GATEWAY</th>
<th>MODULE</th>
<th>NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>private</td>
<td>eth0</td>
<td>00:0e:0c:5d:7e:5e</td>
<td>10.255.255.251</td>
<td>255.0.0.0</td>
<td>--------</td>
<td>e1000</td>
<td>compute-1-1</td>
</tr>
<tr>
<td>-------</td>
<td>-------</td>
<td>---------------</td>
<td>----------------</td>
<td>---------</td>
<td>---------</td>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>-------</td>
<td>eth1</td>
<td>00:30:1b:b2:ea:61</td>
<td>--------------</td>
<td>---------</td>
<td>--------</td>
<td>tg3</td>
<td>------------</td>
</tr>
</tbody>
</table>

◆ **Configure**

```bash
# rocks set host interface ip compute-1-1 eth1 192.168.1.1
# rocks set host interface gateway compute-1-1 eth1 192.168.1.254
# rocks set host interface name compute-1-1 eth1 fast-1-1
# rocks set host interface subnet compute-1-1 eth1 public
```

◆ **Verify**

```bash
# rocks list host interface compute-1-1
```

<table>
<thead>
<tr>
<th>SUBNET</th>
<th>IFACE</th>
<th>MAC</th>
<th>IP</th>
<th>NETMASK</th>
<th>GATEWAY</th>
<th>MODULE</th>
<th>NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>private</td>
<td>eth0</td>
<td>00:0e:0c:5d:7e:5e</td>
<td>10.255.255.251</td>
<td>255.0.0.0</td>
<td>--------</td>
<td>e1000</td>
<td>compute-1-1</td>
</tr>
<tr>
<td>public</td>
<td>eth1</td>
<td>00:30:1b:b2:ea:61</td>
<td>192.168.1.1</td>
<td>255.255.255.0</td>
<td>192.168.1.254</td>
<td>tg3</td>
<td>fast-1-1</td>
</tr>
</tbody>
</table>
import rocks.commands

class Command(rocks.commands.set.host.command):
    info = "Sets the IP address for the named interface for one host."
    shortdesc = "Example Usage:
      Rocks set host interface ip compute-0-0 eth1 192.168.0.10"
    param = ["ip", "iface"]
    args = ["ip", "iface"]

    def run(self, params, args):
        (args, iface, ip) = self.fillPositionalArgs(('iface', 'ip'))
        hosts = self.getHostnames(args)
        if len(hosts) != 1:
            self.abort("must supply one host")
        for host in hosts:
            if not iface:
                self.abort("must supply iface")
            if not ip:
                self.abort("must supply ip")
            ip = ip.upper() if null -> NULL
            self.db.execute("update networks, nodes set
                networks.ip=NEWIFC("%s", 'NULL') where
                nodes.name='%s' and networks.node=nodes.id and
                (networks.device='%s' or networks.mac='%s')
                % (ip, host, iface, iface)"" % (node, ip))
start / stop

- Start and stop something
- NULL commands
- Reserve the verbs for use on other Rolls
- Think “abstract base class”
sync

- Synchronizes the database state to software configuration files
- Similar to the old “insert-ethers – update”
- Not complete yet, see Rocks VI (5.1)
rocks sync config

```python
import os
import sys
import string
import rocks.file
import rocks.commands

class Command(rocks.commands.sync.command):
    """
    For each system configuration file controlled by Rocks, first
    rebuild the configuration file by extracting data from the
    database, then restart the relevant services.
    """
    <example cmd='sync config'>
    Rebuild all configuration files and restart relevant services.
    </example>

    def run(self, params, args):
        cmd = '/opt/rocks/sbin/insert-ethers --update'
        for line in os.popen(cmd).readlines():
            self.owner.addText(line)
```
Extensibility

- New commands
  - Add directories
  - Add \texttt{__init__.py} code

- Existing commands
  - Some commands can be extended
  - Plugins
rocks sync users

- Run after useradd
  - Populate auto.home
  - Cleanup password file
  - Send 411 files
- Two plugins
  - Fixnewusers
  - 411
- Partial Ordering
- Other Rolls can add more plugins to this command
- Command must be design for plugins (not default)
```python
import rocks.commands

class Command(rocks.commands.sync.command):
    """
    Update all user-related files (e.g., /etc/passwd, /etc/shadow, etc.)
    on all known hosts. Also, restart autofs on all known hosts.
    """
    ...

    def run(self, params, args):
        self.runPlugins()
```
411 plugin

```python
import os
import rocks.commands

class Plugin(rocks.commands.Plugin):
    
    """Force a 411 update and re-load autofs on all nodes""

    def provides(self):
        return '411'

    def requires(self):
        return ['fixnewusers']

    def run(self, args):
        
        # force the rebuild of all files under 411's control
        
        for line in os.popen('make -C /var/411 force').readlines():
            self.owner.addText(line)

        #
        # restart autofs on all known hosts
        
        cmd = '/opt/rocks/bin/tentakel "service autofs reload"'
        for line in os.popen(cmd).readlines():
            self.owner.addText(line)
```

© 2009 UC Regents
auto.home / passwd plugin

```python
import os
import string
import rocks.commands

class Plugin(rocks.commands.Plugin):
    """Relocates home directories to /export and fixes autofs.home"""

    def provides(self):
        return 'fixnewusers'

    def run(self, args):
        # scan the password file for any '/export/home' entries
        # this is the default entry as setup by useradd
        new_users = []
        default_dir = '/export/home'
        file = open('/etc/passwd', 'r')

        for line in file.readlines():
            l = string.split(line[:-1], ':')

            if len(l) < 6:
                continue

            username = l[0]
            homedir = l[5]

            if homedir == default_dir:
                new_users.append(username)

        file.close()

        hostname = '%s.%s' % 
        (self.db.getGlobalVar('Kickstart', 'PrivateHostname'),
         self.db.getGlobalVar('Kickstart', 'PrivateDNSDomain'))

        for user in new_users:
            # for each new user, change their default directory to
            # /home/<username>

            cmd = '/usr/sbin/usermod -d %s %s' % (os.path.join('/home', user), user)
            for line in os.popen(cmd).readlines():
                self.owner.addText(line)

            # then update the auto.home file
```
Argument Processing

- rocks <verb> <object…> <subject> <params…>
- Subject is typed by first object
  - host -> one or more hostname
  - roll -> one or more roll names
- Params are in key=value form
- Same as –flag=value but easier to read
Helper classes and functions

- **ArgumentProcessors**
  - Class to parse the subject in a standard way
  - Exists for hosts, rolls, appliances, ...

- **Parameters Parsing**
  - `fillPositionalArgs`
  - `fillParams`
HostArgumentProcessor

- Command must derive from `rocks.commands.HostArgumentProcessor`
- `self.getHostnames(args)`
  - Return a list of hostname as they appear in the cluster database
  - If `args = None` all the host in the cluster are returned
  - `args` can also be a group
    - Rack0, rack1
  - Or an appliance type
    - Compute, Tile, …
import rocks.commands

class command(rocks.commands.HostArgumentProcessor,
               rocks.commands.list.command):
    pass

class Command(command):
    """
    List the membership, CPU count, physical position info and comment for
    a list of hosts.
    """

    <arg optional='1' type='string' name='host' repeat='1'>
    Zero, one or more host names. If no host names are supplied, info about
    all the known hosts is listed.
    </arg>

    <example cmd='list host compute-0-0'>
    List info for compute-0-0.
    </example>

    <example cmd='list host'>
    List info for all known hosts.
    </example>
    """

def run(self, params, args):
    self.beginOutput()
    for host in self.getHostnames(args):
        self.db.execute("select m.name, n.cpus,
                         n.rack, n.rank, n.comment from
                         nodes n, memberships m where
                         n.membership=m.id and n.name='%s'" % host)
        self.addOutput(host, self.db.fetchone())
    self.endOutput(header=['host', 'membership',
                           'cpus', 'rack', 'rank', 'comment'])
```python
args = None

# rocks list host

<table>
<thead>
<tr>
<th>HOST</th>
<th>MEMBERSHIP</th>
<th>CPUS</th>
<th>RACK</th>
<th>RANK</th>
<th>COMMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>vizagra:</td>
<td>Frontend</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>--------</td>
</tr>
<tr>
<td>tile-0-1:</td>
<td>Tile</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>--------</td>
</tr>
<tr>
<td>tile-0-0:</td>
<td>Tile</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>--------</td>
</tr>
<tr>
<td>tile-0-2:</td>
<td>Tile</td>
<td>2</td>
<td>0</td>
<td>2</td>
<td>--------</td>
</tr>
<tr>
<td>tile-0-3:</td>
<td>Tile</td>
<td>2</td>
<td>0</td>
<td>3</td>
<td>--------</td>
</tr>
<tr>
<td>tile-1-3:</td>
<td>Tile</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>--------</td>
</tr>
<tr>
<td>tile-1-2:</td>
<td>Tile</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>--------</td>
</tr>
<tr>
<td>tile-1-1:</td>
<td>Tile</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>--------</td>
</tr>
<tr>
<td>tile-1-0:</td>
<td>Tile</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>--------</td>
</tr>
<tr>
<td>tile-2-0:</td>
<td>Tile</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>--------</td>
</tr>
<tr>
<td>tile-2-1:</td>
<td>Tile</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>--------</td>
</tr>
<tr>
<td>tile-2-2:</td>
<td>Tile</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>--------</td>
</tr>
<tr>
<td>tile-2-3:</td>
<td>Tile</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>--------</td>
</tr>
</tbody>
</table>
```

© 2009 UC Regents
\texttt{args} = \texttt{list of hosts}

```
# rocks list host tile-0-0 10.255.255.253 tile-3-0.local

<table>
<thead>
<tr>
<th>HOST</th>
<th>MEMBERSHIP</th>
<th>CPUS</th>
<th>RACK</th>
<th>RANK</th>
<th>COMMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>tile-0-0:</td>
<td>Tile</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>--------</td>
</tr>
<tr>
<td>tile-0-1:</td>
<td>Tile</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>--------</td>
</tr>
<tr>
<td>tile-3-0:</td>
<td>Tile</td>
<td>2</td>
<td>3</td>
<td>0</td>
<td>--------</td>
</tr>
</tbody>
</table>
```

© 2009 UC Regents
args = rack

# rocks list host rack2
HOST      MEMBERSHIP CPUS RACK RANK COMMENT
tile-2-0: Tile 2 2 0 -------
tile-2-1: Tile 2 2 1 -------
tile-2-2: Tile 2 2 2 -------
tile-2-3: Tile 2 2 3 -------
```plaintext
args = appliance type

# rocks list host tile

<table>
<thead>
<tr>
<th>HOST</th>
<th>MEMBERSHIP</th>
<th>CPUS</th>
<th>RACK</th>
<th>RANK</th>
<th>COMMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>tile-0-0:</td>
<td>Tile</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>--------</td>
</tr>
<tr>
<td>tile-0-1:</td>
<td>Tile</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>--------</td>
</tr>
<tr>
<td>tile-0-2:</td>
<td>Tile</td>
<td>2</td>
<td>0</td>
<td>2</td>
<td>--------</td>
</tr>
<tr>
<td>tile-0-3:</td>
<td>Tile</td>
<td>2</td>
<td>0</td>
<td>3</td>
<td>--------</td>
</tr>
<tr>
<td>tile-1-0:</td>
<td>Tile</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>--------</td>
</tr>
<tr>
<td>tile-1-1:</td>
<td>Tile</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>--------</td>
</tr>
<tr>
<td>tile-1-2:</td>
<td>Tile</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>--------</td>
</tr>
<tr>
<td>tile-1-3:</td>
<td>Tile</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>--------</td>
</tr>
<tr>
<td>tile-2-0:</td>
<td>Tile</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>--------</td>
</tr>
<tr>
<td>tile-2-1:</td>
<td>Tile</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>--------</td>
</tr>
<tr>
<td>tile-2-2:</td>
<td>Tile</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>--------</td>
</tr>
</tbody>
</table>
```
Any combination is fine

# rocks list host tile-2-0 rack1 frontend

<table>
<thead>
<tr>
<th>HOST</th>
<th>MEMBERSHIP</th>
<th>CPUS</th>
<th>RACK</th>
<th>RANK</th>
<th>COMMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>tile-1-0: Tile</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>--------</td>
</tr>
<tr>
<td>tile-1-1: Tile</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>--------</td>
</tr>
<tr>
<td>tile-1-2: Tile</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>--------</td>
</tr>
<tr>
<td>tile-1-3: Tile</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>--------</td>
</tr>
<tr>
<td>tile-2-0: Tile</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>--------</td>
</tr>
<tr>
<td>vizagra: Frontend</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>--------</td>
</tr>
</tbody>
</table>

© 2009 UC Regents
## ArgumentProcessors

<table>
<thead>
<tr>
<th>Class Name</th>
<th>Helper Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApplianceArgumentProcessor</td>
<td>getApplianceNames</td>
</tr>
<tr>
<td>DistributionArgumentProcessor</td>
<td>getDistributionNames</td>
</tr>
<tr>
<td>HostArgumentProcessors</td>
<td>getHostnames</td>
</tr>
<tr>
<td>MembershipArgumentProcessor</td>
<td>getMembershipNames</td>
</tr>
<tr>
<td>NetworkArgumentProcessor</td>
<td>getNetworkNames</td>
</tr>
<tr>
<td>RollArgumentProcessor</td>
<td>getRollNames</td>
</tr>
</tbody>
</table>
import os
import stat
import time
import sys
import string
import rocks.commands

class Command(rocks.commands.RollbackCommand, rocks.commands.list.Command):
    """
    List the status of available rolls.
    """
    def run(self, params, args):
        self.beginOutput()
        for (roll, version) in self.getRollNames(args, params):
            self.db.execute("select version, arch, enabled from rolls where name='%s' and version='%s'" % (roll, version))
            for row in self.db.fetchall():
                self.addOutput(row)
        self.endOutput(header=['name', 'version', 'arch', 'enabled'],
                        trimOwner=0)
No Parameter

```
# rocks list roll

<table>
<thead>
<tr>
<th>NAME</th>
<th>VERSION</th>
<th>ARCH</th>
<th>ENABLED</th>
</tr>
</thead>
<tbody>
<tr>
<td>viz:</td>
<td>5.0</td>
<td>i386</td>
<td>yes</td>
</tr>
<tr>
<td>sge:</td>
<td>5.0</td>
<td>i386</td>
<td>yes</td>
</tr>
<tr>
<td>kernel:</td>
<td>5.0</td>
<td>i386</td>
<td>yes</td>
</tr>
<tr>
<td>updates:</td>
<td>5.1</td>
<td>i386</td>
<td>yes</td>
</tr>
<tr>
<td>java:</td>
<td>4.3.2</td>
<td>i386</td>
<td>yes</td>
</tr>
<tr>
<td>xen:</td>
<td>5.0</td>
<td>i386</td>
<td>yes</td>
</tr>
<tr>
<td>CentOS:</td>
<td>5.1</td>
<td>i386</td>
<td>yes</td>
</tr>
<tr>
<td>ganglia:</td>
<td>5.0</td>
<td>i386</td>
<td>yes</td>
</tr>
<tr>
<td>web-server:</td>
<td>5.0</td>
<td>i386</td>
<td>yes</td>
</tr>
<tr>
<td>base:</td>
<td>5.0</td>
<td>i386</td>
<td>yes</td>
</tr>
<tr>
<td>hpc:</td>
<td>5.0</td>
<td>i386</td>
<td>yes</td>
</tr>
</tbody>
</table>
```
Version Parameter

# rocks list roll version=4.3.2

NAME    VERSION    ARCH    ENABLED
java: 4.3.2  i386    yes
Summary

- ArgumentProcessors standardize the handling of command line subjects
- Calling the helper function with an empty list returns all subject in the database
- HostArgumentProcessor knows about more than just host names
- RollArgumentProcessor can filter on versions
fillParams

- Create local variables based on command parameters (key=value)
- Argument a list of (key, default) tuples
- If the parameter is not found on the command line the default value is used
def maketorrent(self, filename, data):
    info = {}
    info['length'] = os.stat(filename)[stat.ST_SIZE]
    info['name'] = os.path.basename(filename)

    data['info'] = info
    encoded = Base64.encode(data)

    file = open('%s.torrent' % (filename), 'w')
    file.write(encoded)
    file.close()

def run(self, params, args):
    if len(args) != 1:
        self.abort('must supply one file')
    filename = args[0]

    (timestamp, ) = self.fillParams([('timestamp', time.time())])
    try:
        creation_date = int(timestamp)
    except:
        creation_date = int(time.time())

    data = {}

    # announce string
    #
    localhost = self.db.getGlobalVar('Kickstart', 'PrivateAddress')
    data['announce'] = 'http://%s:7625/announce' % (localhost)

    data['creation date'] = creation_date
    #
rocks: add host

    basename, rack, rank = host.split('-')
    self.db.execute("""select m.name from appliances a, memberships m where a.name='%s' and m.appliance=a.id"""" % basename)
    membership, = self.db.fetchone()
    rack = int(rack)
    rank = int(rank)
    except:
        membership = None
        rack = None
        rank = None

    # fillParams with the above default values
    (membership, numCPUs, rack, rank) = self.fillParams(
        [('membership', membership),
         ('cpus', 1),
         ('rack', rack),
         ('rank', rank)])

    if not membership:
        self.abort('membership not specified')
    if rack == None:
        self.abort('rack not specified')
    if rank == None:
        self.abort('rank not specified')

    self.db.execute("""insert into nodes
        (site, name, membership, cpus, rack, rank)
    values
        (0,
        '%s',
        (select id from memberships where name='%s'),
        '%d',
        '%d',
        '%d')"""
        % (host, membership, int(numCPUs), int(rack), int(rank)))
fillPositionalArgs

- Allows for parameters to have implied keys (just values on command line)
- This is an optimization for ease of use, not ease of software
- Argument is a list of keys
  - No default value processing, if a key is specified it is required
  - Use this only when a parameter is required
- Example:
  # rocks set network netmask optiputer netmask=255.255.255.0
  # rocks set network netmask optiputer 255.255.0.0
rocks set network netmask

Netmask that named networks should have.

Can be used in place of netmask argument.

Sets the netmask for the "optiputer" network to a class-c address space.

Same as above.

Sets the netmask for the "optiputer" and "cavewave" networks to a class-b address space.

---

def run(self, params, args):
    
    (args, netmask) = self.fillPositionalArgs(('netmask',))

    if not len(args):
        self.abort('must supply network')
    if not netmask:
        self.abort('must supply netmask')

    for network in self.getNetworkNames(args):
        self.db.execute("update subnets set netmask='%s' where subnets.name='%s'", (netmask, network))
rocks set host interface

Sets the MAC Address for the eth1 device on host compute-0-0.

```python
def run(self, params, args):
    (args, iface, mac) = self.fillPositionalArgs({'iface', 'mac'})

    hosts = self.getHostnames(args)

    if len(hosts) != 1:
        self.abort('must supply one host')
    if not iface:
        self.abort('must supply iface')
    if not mac:
        self.abort('must supply mac')

    for host in hosts:
        self.db.execute("'update networks, nodes set
            networks.mac=NUMBF(''{host}'',''NULL'') where
            nodes.name=''{host}'' and networks.node=nodes.id and
            (networks.device=''{host}'' or networks.mac=''{host}'')" %
            (mac, host, iface, iface))
```

© 2009 UC Regents
Help and Docstrings

- The command line is the documentation
  - No more out of date man pages
  - Still needs a cookbook document, but reference is part of the code
- We’ve been looking at this all session
- Class docstring """"text"""
- Command line has an XML format
# rocks list roll help
rocks list roll [roll]...

Description:

List the status of available rolls.

Arguments:

[roll]

List of rolls. This should be the roll base name (e.g., base, hpc, kernel). If no rolls are listed, then status for all the rolls are listed.

Examples:

$ rocks list roll kernel

List the status of the kernel roll

$ rocks list roll

List the status of all the available rolls
import os
import stat
import time
import sys
import string
import rocks.commands

class Command(rocks.commands.RollArgumentProcessor,
              rocks.commands.list.command):
    ""
    List the status of available rolls.
    ""

    <arg optional='1' type='string' name='roll' repeat='1'>
    List of rolls. This should be the roll base name (e.g., base, hpc, 
    kernel). If no rolls are listed, then status for all the rolls are 
    listed.
    </arg>

    <example cmd='list roll kernel'>
    List the status of the kernel roll
    </example>

    <example cmd='list roll'>
    List the status of all the available rolls
    </example>

    ""

def run(self, params, args):
    self.beginOutput()
    for (roll, version) in self.getRollNames(args, params):
        self.db.execute("""select version, arch, enabled from 
                        rolls where name='"%s' and version='"%s"""" 
                        %
                        (roll, version))
        for row in self.db.fetchall():
            self.addOutput(roll, row)
    self.endOutput(header=['name', 'version', 'arch', 'enabled'],
                   trimOwner=0)
<arg>

- Attributes
  - name (required)
  - optional (default = “0”)
  - type (default = “string”)
  - repeat (default = “0”)

- Example:
  
  `<arg type='string' name='network' repeat='1'>
    One or more named networks that should have the defined netmask.
  </arg>`
<param>

- Attributes
  - name (required)
  - optional (default = “1”)
  - type (default = “string”)
  - repeat (default = “0”)

- Example:
  <param type='string' name='iface'>
    Can be used in place of the iface argument.
  </param>
<example>

- Attributes
  - **cmd**(required)

- Example:

  <example cmd='set host interface mac compute-0-0 eth1 00:0e:0c:a7:5d:ff'>
  Sets the MAC Address for the eth1 device on host compute-0-0.
  </example>
<related>

Example

<related>set host interface iface</related>
<related>set host interface ip</related>
<related>set host interface gateway</related>
<related>set host interface module</related>
Help

- rocks <verb> <object…> <subject> help
  - Loads the command module
  - Parses the XML docstring
  - Format and output help as 80 column text

- Debug syntax with format= parameter
help format=raw

# rocks list roll help format=raw
1:
2: List the status of available rolls.
3:
4: <arg optional='1' type='string' name='roll' repeat='1'>
5: List of rolls. This should be the roll base name (e.g., base, hpc, kernel). If no rolls are listed, then status for all the rolls are listed.
6: </arg>
7: </example>
8: </arg>
9:
10: <example cmd='list roll kernel'>
11: List the status of the kernel roll
12: </example>
13:
14: <example cmd='list roll'>
15: List the status of all the available rolls
16: </example>
Help format=parsed

# rocks list roll help format=parsed

{"related": [], 'example': [(u'list roll kernel', u'List the status of the kernel roll
'), (u'list roll', u'List the status of all the available rolls
')], 'description': u'List the status of available rolls.

', 'param': [], 'arg': [(u'roll', u'string', 1, 1), u'List of rolls. This should be the roll base name (e.g., base, hpc, kernel). If no rolls are listed, then status for all the rolls are listed.
']}

© 2009 UC Regents
Docbook

Roll Usersguide Command Reference is generated automatically

# rocks list roll help format=docbook
<section id="rocks-list-roll" xreflabel="list roll">
<title>list roll</title>
<cmdsynopsis>
   <command>rocks list roll</command>
   <arg rep="repeat" choice="opt">roll</arg>
</cmdsynopsis>
<para>
   List the status of available rolls.
</para>
</section>