



Customizing Your Cluster

Mason Katz
mjk@sdsc.edu



Rocks Command Line

The RCL is the primary administrative interface for a Rocks cluster

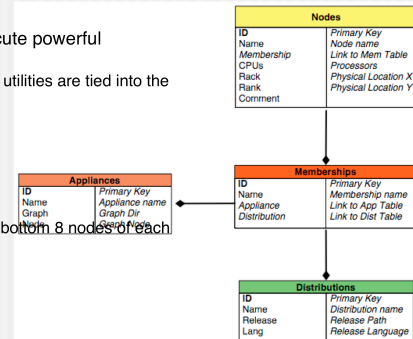


Rocks 4.x was SQL-based

- ◆ This is from RAP III (2007)
- ◆ SQL nature of Rocks
- ◆ Multi-way Joins
- ◆ Unstable Schema
- ◆ Exposed SQL to users

Node Info Stored In A MySQL Database

- ◆ If you know SQL, you can execute powerful commands
 - Rocks-supplied command line utilities are tied into the database
 - E.g., get the hostname for the bottom 8 nodes of each cabinet:



```
# cluster-fork --query="select name from nodes where rank<8" hostname
```

© 2007 UC Regents

36



Confusing Commands

```
Usage: add-extra-nic [-hvv] [-p password] [-u host] [-d database] [--help]
  [--list-rcfiles] [--list-project-info] [--verbose] [--dump] [--del] [--list]
  [--verbose] [--no-update] [--no-modify] [--dryrun] [--rcfile arg] [--host host]
  [--password password] [--db database] [--user host]
  [--if interface (default: eth1)] [--mac mac address]
  [--module linux driver module name] [--ip ip address]
  [--netmask netmask (default /24)] [--gateway ip address of gateway]
  [--name hostname on new interface] [--site client ip] node
```

```
Usage: rocks-dist [-hvcpv] [-p password] [-u host] [-d database] [-a arch]
  [-d dirname] [-g path] [-l lang] [-r release] [--help] [--list-rcfiles]
  [--list-project-info] [--verbose] [--copy] [--debug] [--graph-draw-invis-edges]
  [--graph-draw-order] [--graph-draw-edges] [--graph-draw-key] [--graph-draw-all]
  [--graph-draw-landscape] [--install] [--verbose] [--with-rolls-only] [--clean]
  [--notorrent] [--rcfile arg] [--host host] [--password password]
  [--db database] [--user host] [--arch architecture] [--comps path]
  [--dist dirname] [--graph-draw-size arg] [--graph-draw-format arg]
  [--mirror-dir dirname] [--mirror-host hostname] [--root dirname]
  [--cdrom /mnt/cdrom] [--with-roll rollname-rollversion]
  [--path single path item] command
```

Available commands:

```
dist dvd makecontrib makesitenodes copycd usb copyroll cdrom paths graph dist2mirror
```



What Bothered Us

- ◆ Lack of consistency in Rocks commands
 - ⇒ add-extra-nic (15 flags)
 - ⇒ 411put
 - ⇒ rocks-dist
 - ⇒ dbreport (~ a dozen reports)
- ◆ Not 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?



Do Over

- ◆ Consistent
 - Interface
 - Argument parsing
 - Usage / Help
- ◆ Extensible
 - Easy to add commands (3rd party rolls)
 - Easy to modify commands
- ◆ Easy to guess the right command
- ◆ Purge all –flags from Rocks
- ◆ Hide the SQL database (and underlying schema)
- ◆ Inspired by Trac

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





Grammar

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



Customizing Server Management

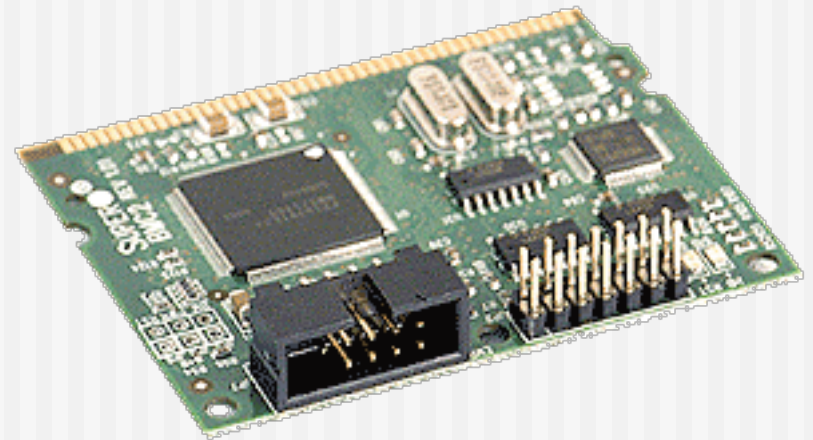
How to configure and use the remote management processor on your servers.

Case study in RCL



IPMI – Intelligent Platform Management Interface

- ◆ Available (free or low cost) on most modern servers
- ◆ Passive monitoring of sensors (temp, fan speed)
- ◆ Active control of power (on/off/reset)
- ◆ ~~It's a Standard~~





Networking

◆ Dedicated NIC

- ⇒ Does your BMC have its own Ethernet port
- ⇒ Preferred since you can isolate network traffic.

◆ Shared NIC

- ⇒ Your BMC will still have its own MAC address
- ⇒ Traffic will bridge over another Ethernet port
- ⇒ Bridging over eth0 (private network) makes sense



Step 1: Add a Network

- ◆ Every IPMI interface needs an IP address
- ◆ To isolate the traffic it should have a distinct subnet
- ◆ 192.168.0.0 / 255.255.0.0 is unused by default (your cluster may differ)



CONFIGURING IPMI



ADD NETWORK

```
rocks add network <network-name>  
  <network>  
  <subnet>
```

```
rocks add network ipmi  
  192.168.1.0  
  255.255.255.0
```



LIST NETWORK

NETWORK	SUBNET	NETMASK
private:	10.12.0.0	255.255.0.0
public:	169.228.3.0	255.255.255.240
ipmi:	192.168.1.0	255.255.255.0



Step 2: Add the interface

- ◆ Host first must be installed
- ◆ Then secondary NICs can be added
- ◆ After all hosts are configured just re-install



ADD HOST INTERFACE

```
rocks add host interface
  <host> <iface>
  ip=<address> subnet=<name>
  gateway=<address>
  name=<hostname>
```

```
rocks add host interface
  compute-0-0 ipmi
  ip=192.168.1.1 subnet=ipmi
  gateway=1 name=ipmi-0-0
```



LIST HOST INTERFACE

```
rocks list host interface compute-0-0
SUBNET  IFACE  MAC          IP          NETMASK     GATEWAY  MODULE  NAME
private eth0    00:15:17:79:d3:c0 10.12.0.12  255.255.0.0  ----- e1000e  compute-0-0
ipmi    ipmi    -----      192.168.1.2  255.255.255.0  2        ----- ipmi-0-0
```



Gateway Parameter

- ◆ Is used to specify the IPMI channel
 - May change in 5.2 final
- ◆ The channel indicates the NIC IPMI will use
- ◆ Channel 1 is eth0
- ◆ You BMC may be different, read your motherboard docs

```
rocks add host interface
compute-0-0 ipmi
ip=192.168.1.1 subnet=ipmi
gateway=1 name=ipmi-0-0
```



Step 3: Re-install

- ◆ PXE Boot
 - ⇒ Network Boot is first in BIOS boot order
 - ⇒ Set Rocks Boot action to install
 - ⇒ Reboot the host

- ◆ Otherwise use old rocks commands or just hard power cycle the host.



SET HOST BOOT

```
rocks set host boot  
  <host>  
  action=<boot-action>
```

```
rocks set host boot  
  compute-0-0  
  action=install
```



RUN HOST

```
rocks run host
```

```
<host>
```

```
<command>
```

```
rocks run host
```

```
compute-0-0
```

```
/sbin/init 6
```



Step 4: Run IPMI script

- ◆ Re-install creates an IPMI script
- ◆ This is not run by default
 - ⇒ impitool is confusing
 - ⇒ No method to reset BMC if things go wrong
- ◆ Future releases will automate this part
- ◆ In 5.2 you need to log in and source the file
 - ⇒ May change in 5.2 final



/etc/sysconfig/network-scripts/impd-<channel>

```
ipmitool lan set 1 ipaddr 192.168.1.2
ipmitool lan set 1 netmask 255.255.255.0
ipmitool lan set 1 arp respond on
ipmitool user set password 1 admin
ipmitool lan set 1 access on
ipmitool lan set 1 user
ipmitool lan set 1 auth ADMIN PASSWORD
```




USING IPMITOOL



CHASSIS STATUS

```
ipmitool -H ipmi-0-0 -P admin chassis status
System Power                : on
Power Overload              : false
Power Interlock             : inactive
Main Power Fault           : false
Power Control Fault        : false
Power Restore Policy       : always-off
Last Power Event           : ac-failed
Chassis Intrusion          : active
Front-Panel Lockout       : inactive
Drive Fault                : false
Cooling/Fan Fault         : false
```



POWER

`ipmitool -H ipmi-0-0 -P admin power off`

`ipmitool -H ipmi-0-0 -P admin power on`



Adding Software to Compute Nodes

How to create and deploy
an software without
learning anything

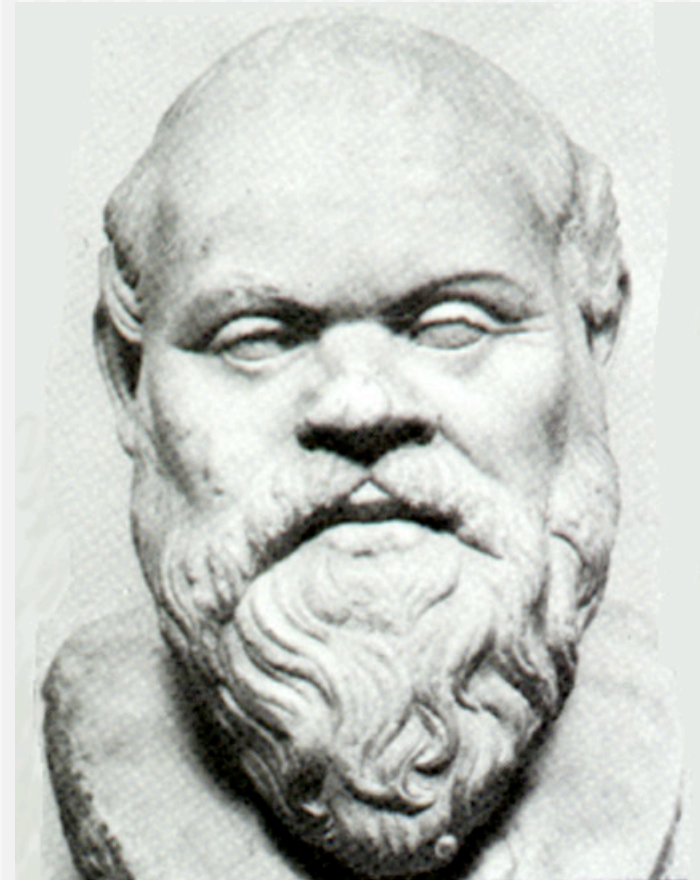
Case study in RCL
Peak at Graph XML



BUILDING AN RPM

Philosophy

- ◆ All software is installed on the local disk
- ◆ Does not require NFS or non-scalable diskless technologies
- ◆ Use the native OS packager for everything
 - Linux = rpm
 - Solaris = pkg





Violate the Rules

- ◆ You just need a few packages added and cannot find or build packages
- ◆ You want this only on your cluster and not on several clusters
- ◆ You still want to avoid NFS and benefit from Rocks management





Get a Directory Tree

- ◆ Build you software from source and install on the frontend
 - ⇒ configure
 - ⇒ make
 - ⇒ install

- ◆ Or, just untar a binary bundle



CREATE PACKAGE

```
rocks create package  
  <path>  
  <package-name>
```

```
rocks create package  
  /opt/mx  
  mx
```



Done

```
# rpm -qip mx-1.0-1.x86_64.rpm
```

```
Name           : mx                               Relocations: (not relocatable)
Version        : 1.0                               Vendor: Rocks Clusters
Release       : 1                                   Build Date: Tue 12 May 2009 04:40:00 PM
              PDT
Install Date: (not installed)                      Build Host: vizagra.rocksclusters.org
Group         : System Environment/Base             Source RPM: mx-1.0-1.src.rpm
Size          : 17588899                            License: University of California
Signature     : (none)
Summary       : A collection of Python software tools.
Description   :
The mx extensions for Python are a collection of Python software tools
which enhance Python's usability in many areas.
```



ADDING YOUR PACKAGE TO COMPUTE NODES



Step 1: Contribute the RPM

- ◆ Your distribution looks for packages from Rolls and in a contrib area
- ◆ Copy your RPMS into contrib

```
cp mx-1.0-1.x86_64.rpm  
  /export/rocks/install/contrib/5.2/  
  x86_64/RPMS
```



Step 2: Extend XML

```
cd /export/rocks/install/site-  
profiles/5.2/nodes/
```

```
cp skeleton.xml  
   extend-compute.xml
```

```
vi extend-compute.xml
```



Add Package Tag

original

```
<kickstart>

<description>
Skeleton XML Node
</description>

<changelog>
</changelog>

<!--
<package></package>
-->

<post>
</post>

</kickstart>
```

modified (with mx)

```
<kickstart>

<description>
Skeleton XML Node
</description>

<changelog>
</changelog>

<package>mx</package>

<post>
</post>

</kickstart>
```



Step 3: Rebuild Distribution

- ◆ RPM package is already contributed
- ◆ XML node file is already extended
- ◆ Now we need to rebuild the dist

- ◆ Must be done in `/export/rocks/install`



CREATE DISTRO

```
cd /export/rocks/install
```

```
rocks create distro
```




Step 4: Re-install

(repeated material 3 slides)

- ◆ PXE Boot
 - ⇒ Network Boot is first in BIOS boot order
 - ⇒ Set Rocks Boot action to install
 - ⇒ Reboot the host

- ◆ Otherwise use old rocks commands or just hard power cycle the host.



SET HOST BOOT

```
rocks set host boot  
  <host>  
  action=<boot-action>
```

```
rocks set host boot  
  compute-0-0  
  action=install
```



RUN HOST

```
rocks run host
```

```
<host>
```

```
<command>
```

```
rocks run host
```

```
compute-0-0
```

```
/sbin/init 6
```



Common User Tweaks to Rocks

Things that used to be
hard are now trivial.

A case study in Attributes



ENABLING RSH



Don't judge

- ◆ Enabling RSH is a common user request
- ◆ Requires
 - Minor XML changes
 - Rebuilding the distribution
 - Re-installing the nodes
- ◆ SSH-only is the Rocks default





Step 1: Set RSH == True

- ◆ Before you install any compute nodes
- ◆ Set the `rsh` attribute
- ◆ Compute nodes will install with `rsh`

- ◆ Still need to `rsh-ify` the frontend yourself



SET ATTR

```
rocks set attr  
  <key> <value>
```

```
rocks set attr  
  rsh true
```




ENABLING X11 ON COMPUTE HOSTS



Interactive Compute Nodes

- ◆ Another common request
- ◆ Good for computer labs
- ◆ Requires
 - ⇒ Large XML changes
 - ⇒ Rebuilding the distribution
 - ⇒ Re-installing the nodes





Step 1: Set X11 == TRUE

- ◆ Before you install any compute nodes
- ◆ Set the x11 attribute
- ◆ Compute nodes will install with X11

- ◆ This time we are only changing the compute nodes, not everything



SET APPLIANCE ATTR

```
rocks set appliance attr  
  <appliance>  
  <key> <value>
```

```
rocks set appliance attr  
  compute  
  x11 true
```



DISABLE SGE ON ONE RACK OF HARDWARE



Enable / Disable SGE

- ◆ Disable SGE and dedicate a rack to a single user
- ◆ Enable SGE on tile nodes on a Viz Wall
- ◆ Required a brand new appliance type!





SET HOST ATTR

```
rocks set host attr  
  <host(s)>  
  <key> <value>
```

```
rocks set host attr  
  rack0  
  sge false
```





All HOST Commands Accept

- ◆ No argument (all hosts)
- ◆ A list of hostnames / addresses
- ◆ A list of racks
- ◆ A list of appliance names
- ◆ Any combination of the above



No Arguments

```
# rocks list host
```

HOST	MEMBERSHIP	CPUS	RACK	RANK	COMMENT
vizagra:	Frontend	1	0	0	-----
tile-0-1:	Tile	2	0	1	-----
tile-0-0:	Tile	2	0	0	-----
tile-0-2:	Tile	2	0	2	-----
tile-0-3:	Tile	2	0	3	-----
tile-1-3:	Tile	2	1	3	-----
tile-1-2:	Tile	2	1	2	-----
tile-1-1:	Tile	2	1	1	-----
tile-1-0:	Tile	2	1	0	-----
tile-2-0:	Tile	2	2	0	-----
tile-2-1:	Tile	2	2	1	-----
tile-2-2:	Tile	2	2	2	-----



List of Hostnames / Addresses

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

HOST	MEMBERSHIP	CPUS	RACK	RANK	COMMENT
tile-0-0:	Tile	2	0	0	-----
tile-0-1:	Tile	2	0	1	-----
tile-3-0:	Tile	2	3	0	-----



List of Racks

```
# 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	-----



List of Appliance Names

```
# rocks list host tile
```

HOST	MEMBERSHIP	CPUS	RACK	RANK	COMMENT
tile-0-0:	Tile	2	0	0	-----
tile-0-1:	Tile	2	0	1	-----
tile-0-2:	Tile	2	0	2	-----
tile-0-3:	Tile	2	0	3	-----
tile-1-0:	Tile	2	1	0	-----
tile-1-1:	Tile	2	1	1	-----
tile-1-2:	Tile	2	1	2	-----
tile-1-3:	Tile	2	1	3	-----
tile-2-0:	Tile	2	2	0	-----



Any Combination

```
# rocks list host tile-2-0 rack1 frontend
```

HOST	MEMBERSHIP	CPUS	RACK	RANK	COMMENT
tile-1-0:	Tile	2	1	0	-----
tile-1-1:	Tile	2	1	1	-----
tile-1-2:	Tile	2	1	2	-----
tile-1-3:	Tile	2	1	3	-----
tile-2-0:	Tile	2	2	0	-----
vizagra:	Frontend	1	0	0	-----



ATTRIBUTE DETAILS



Attributes Are ...

- ◆ Cluster specific-state
- ◆ Cluster admin controlled
- ◆ Installation screen data
- ◆ Arbitrary key-value pair data
- ◆ Used to build a kickstart (or jumpstart) file



LIST HOST ATTR

HOST	ATTR	VALUE	SOURCE
tile-0-0:	HttpConf	/etc/httpd/conf	O
tile-0-0:	HttpConfigDirExt	/etc/httpd/conf.d	O
tile-0-0:	HttpRoot	/var/www/html	O
tile-0-0:	Info_CertificateCountry	US	G
tile-0-0:	Info_CertificateOrganization	CalIT2	G
tile-0-0:	Info_CertificateState	California	G
tile-0-0:	Kickstart_PrivateSyslogHost	10.1.1.1	G
tile-0-0:	Kickstart_PublicBroadcast	137.110.119.255	G
tile-0-0:	Kickstart_PublicDNSDomain	rocksclusters.org	G
tile-0-0:	Kickstart_PublicNTPHost	pool.ntp.org	G
tile-0-0:	Kickstart_PublicNetmask	255.255.255.0	G
tile-0-0:	arch	x86_64	H
tile-0-0:	hostname	tile-0-0	I
tile-0-0:	os	linux	H
tile-0-0:	rack	0	I
tile-0-0:	rank	0	I
tile-0-0:	rocks_version	5.2	G
tile-0-0:	rsh	false	G
tile-0-0:	x11	true	A



Attributes can be

- ◆ G – Global
 - ◆ O – OS
 - ◆ A – Appliance
 - ◆ H – Host
 - ◆ I – Installer (built in)
-
- ◆ Order of precedence is top to bottom



Big Change

- ◆ `app_globals` table is replaced by
 - ⇒ `global_attributes`
 - ⇒ `os_attributes`
 - ⇒ `appliance_attributes`
 - ⇒ `node_attributes`
- ◆ VAR tags are replaced by XML entities

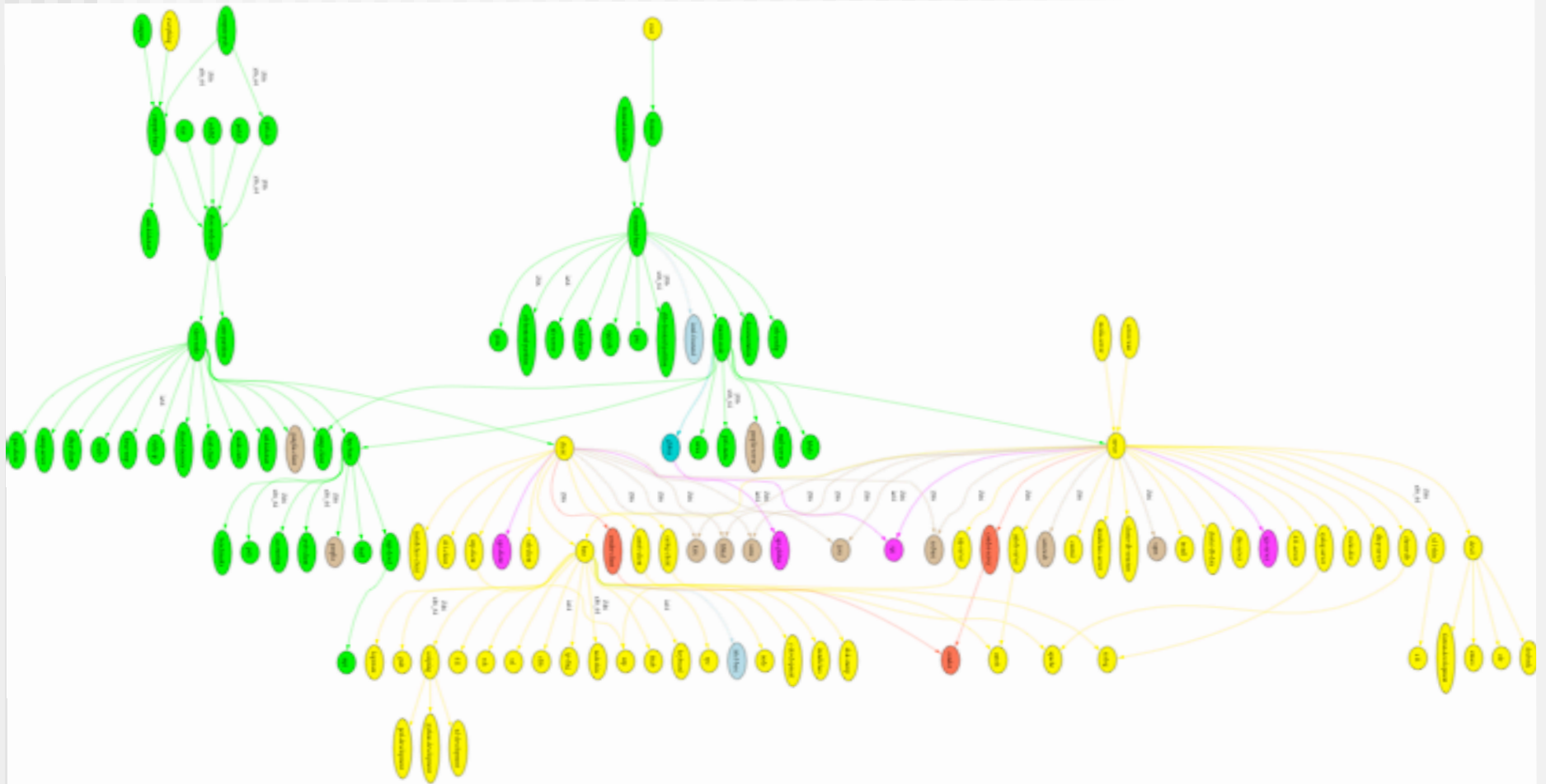


Graph XML and Rolls

The Rocks engine



It looks something like this





GRAPH FUNDAMENTALS



The XML Graph Includes

◆ Nodes

- Single purpose modules
- Kickstart file snippets (XML tags map to kickstart commands)
- Approximately 200 node files in Rocks

◆ Graph

- Defines interconnections for nodes
- Think OOP or dependencies (class, #include)
- A single default graph file in Rocks

◆ Macros

- SQL Database holds site and node specific state
- Node files may contain `&state;` entities

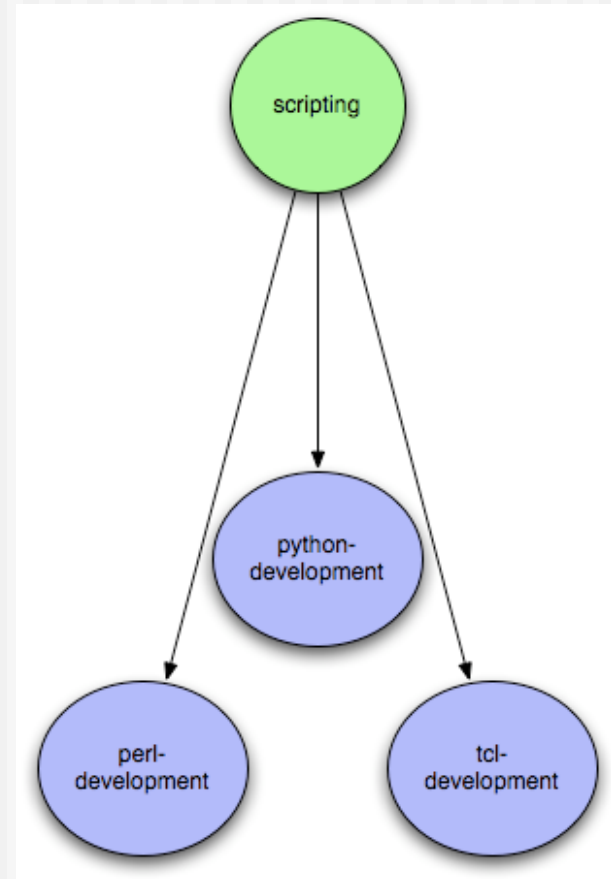


Composition

- ◆ Aggregate Functionality

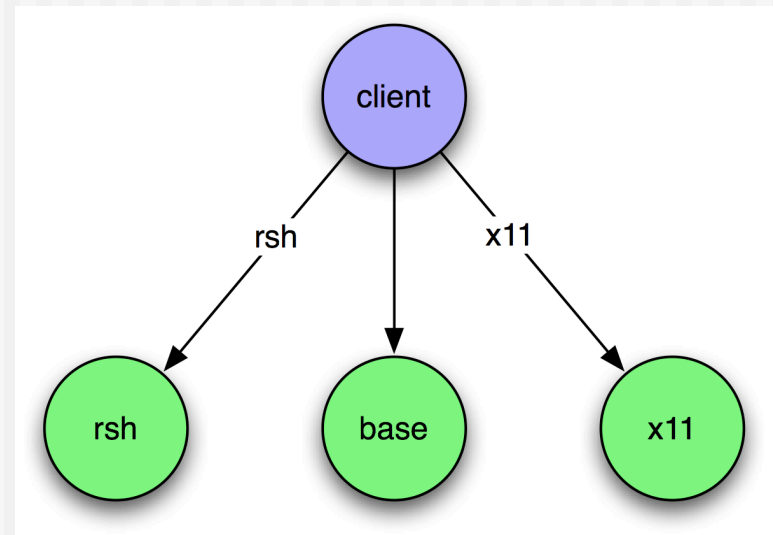
- ◆ `scripting` ISA

- `perl-development`
- `python-development`
- `tcl-development`



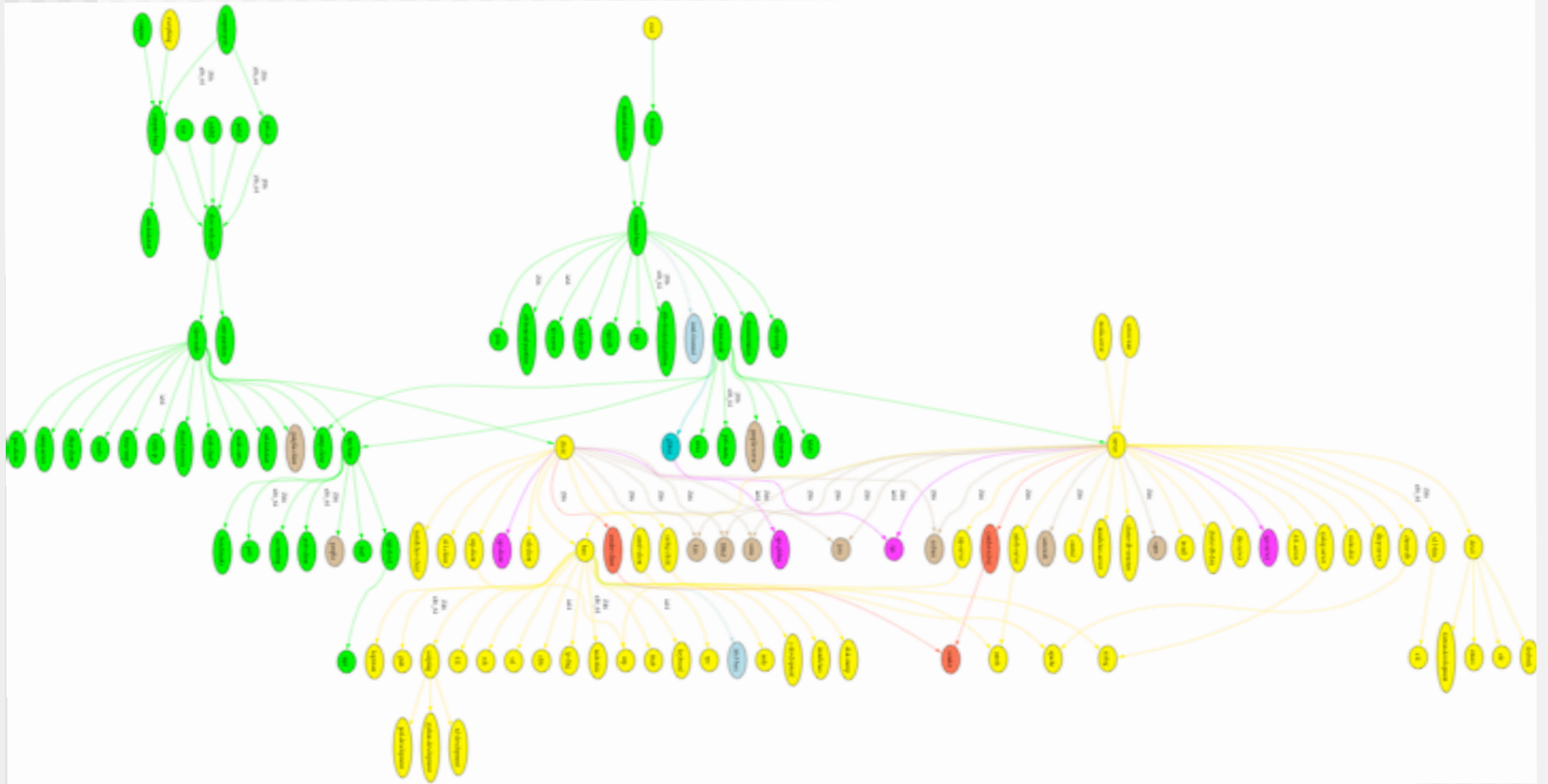
Traverse by Attributes

- ◆ if `x11 == TRUE`
 - ↳ `client IsA x11`
- ◆ if `rsh == FALSE`
 - ↳ `client IsNotA rsh`
- ◆ **Most important slide in this session**
- ◆ RCL allows you to control the graph





Putting in all together





Sample Node File

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE kickstart SYSTEM "@KICKSTART_DTD@" [<!ENTITY ssh "openssh">]>
<kickstart>
  <description>
    Enable SSH
  </description>

  <package>&ssh;</package>
  <package>&ssh;-clients</package>
  <package>&ssh;-server</package>
  <package>&ssh;-askpass</package>

<post>
cat &gt; /etc/ssh/ssh_config &lt;&lt; 'EOF' <!-- default client setup -->
Host *
    ForwardX11 yes
    ForwardAgent yes
EOF

chmod o+rx /root
mkdir /root/.ssh
chmod o+rx /root/.ssh

</post>
</kickstart>>
```



Sample Graph File

```
<?xml version="1.0" standalone="no"?>
<graph>
  <description>
    Default Graph for NPACI Rocks.
  </description>

  <edge from="base" to="scripting"/>
  <edge from="base" to="ssh"/>
  <edge from="base" to="ssl"/>
  <edge from="base" to="grub" arch="i386"/>
  <edge from="base" to="elilo" arch="ia64"/>

  <edge from="node" to="base"/>
  <edge from="node" to="accounting"/>
  <edge from="slave-node" to="node"/>
  <edge from="slave-node" to="nis-client"/>
  <edge from="slave-node" to="autofs-client"/>
  <edge from="slave-node" to="dhcp-client"/>
  <edge from="slave-node" to="snmp-server"/>
  <edge from="slave-node" to="node-certs"/>
  <edge from="compute" to="slave-node"/>
  <edge from="compute" to="usher-server"/>
  <edge from="master-node" to="node"/>
  <edge from="master-node" to="x11"/>
  <edge from="master-node" to="usher-client"/>
</graph>
```



Nodes XML Tools: Entities

- ◆ Get Attributes from Database

- `&Kickstart_PrivateGateway;`
- `&hostname;`

```
10.1.1.1  
compute-0-0
```

- ◆ More on attributes later

Nodes XML Tools: <eval>

- ◆ Do processing on the frontend:
 - `<eval shell="bash">`
- ◆ To insert a fortune in the kickstart file:

```
<eval shell="bash">  
/usr/games/fortune  
</eval>
```

```
"Been through Hell?  
Whaddya bring back for  
me?"  
-- A. Brilliant
```



Nodes XML Tools <file>

- ◆ Create a file on the system:
`<file name="/etc/hi-mom" mode="append">`
 How are you today?
`</file>`
- ◆ Used extensively throughout Rocks post sections
 - ➔ Keeps track of alterations automatically via RCS.

```
<file name="/etc/hi" perms="444">  
How are you today?  
I am fine.  
</file>
```

```
...RCS checkin commands...  
cat > /etc/hi << 'EOF'  
How are you today?  
I am fine.  
EOF  
chmod 444 /etc/hi-mom  
...RCS cleanup commands...
```

Fancy <file>: nested tags

```
<file name="/etc/hi">
```

Here is your fortune for today:

```
<eval>
```

```
date +"%d-%b-%Y"
```

```
echo ""
```

```
/usr/games/fortune
```

```
</eval>
```

```
</file>
```

...RCS checkin commands...

```
cat > /etc/hi << 'EOF'
```

Here is your fortune for today:

13-May-2005

**"Been through Hell? Whaddya
bring back for me?"**

-- A. Brilliant

EOF

...RCS cleanup commands...



Nodes Main

- ◆ Used to specify basic configuration:
 - timezone
 - mouse, keyboard types
 - install language
- ◆ Used more rarely than other tags
- ◆ Rocks main tags are usually a straight translation:

```
<main>

  <timezone>America/Mission_Beach
  </timezone>

</main>
```

```
...
timezone America/Mission_Beach
...
rootpw --iscrypted sndk48shdlwis
mouse genericps/2
url --url http://10.1.1.1/install/rocks-dist/..
```



Nodes Packages

- ◆ `<package>java</package>`
 - Specifies an RPM package. Version is automatically determined: take the *newest* rpm on the system with the name 'java'.
- ◆ `<package arch="x86_64">java</package>`
 - Only install this package on x86_64 architectures
- ◆ `<package arch="i386,x86_64">java</package>`

```
<package>newcastle</package>  
<package>stone-pale</package>  
<package>guinness</package>
```

```
%packages  
newcastle  
stone-pale  
guinness
```



Nodes Post

ntp-client.xml

```
<post>
```

```
/bin/mkdir -p /etc/ntp  
/usr/sbin/ntpdate &Kickstart_PrivateNTPHost;  
/sbin/hwclock --systohc
```

```
</post>
```

```
%post
```

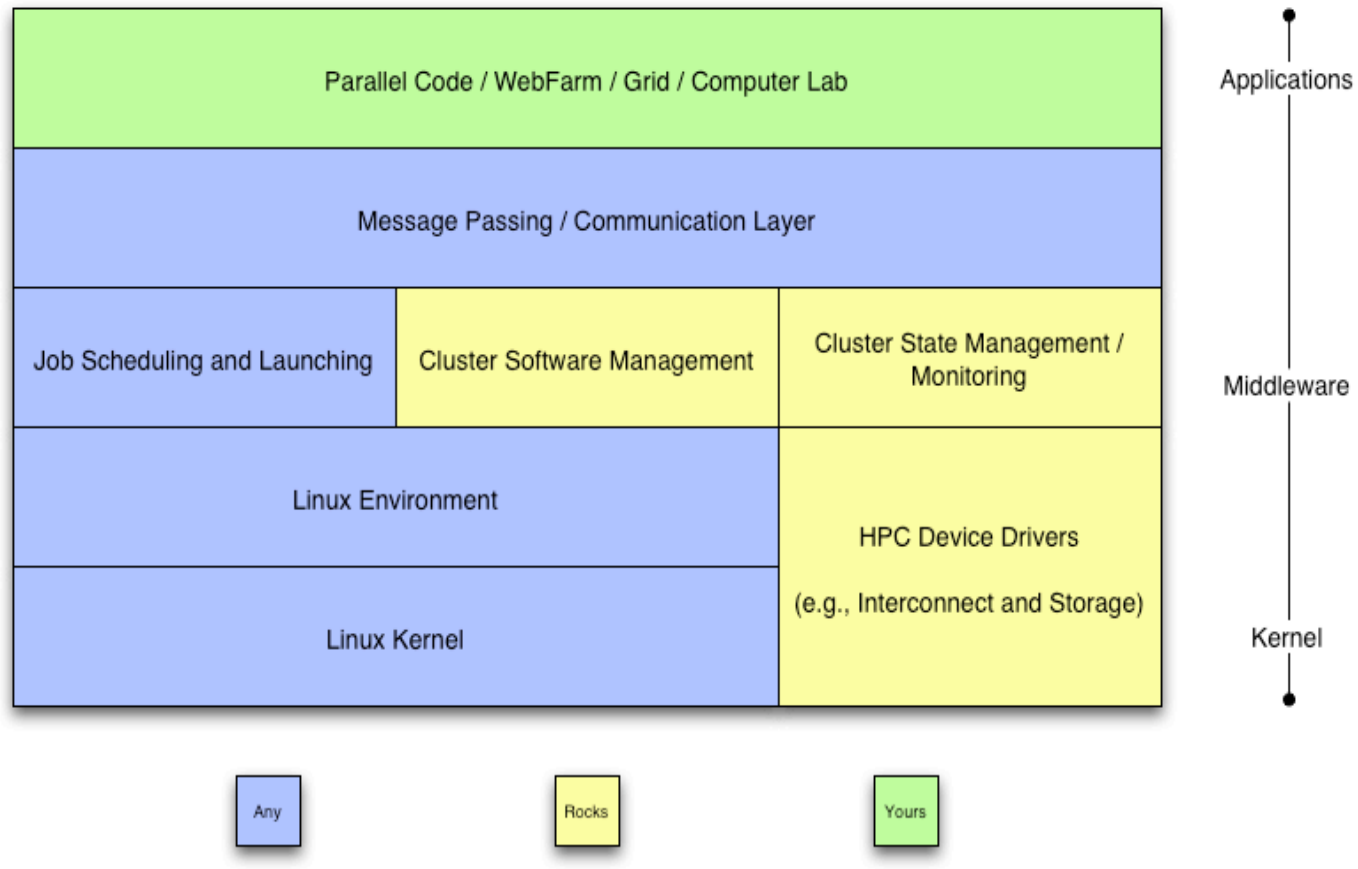
```
/bin/mkdir -p /etc/ntp  
/usr/sbin/ntpdate 10.1.1.1  
/sbin/hwclock --systohc
```



ROLL FUNDAMENTALS



Cluster Software Stack





Need Better Flexibility in Stack

◆ Issues

- Static Stack
 - Cannot redefine
 - Cannot extend
- Monolithic Stack
 - Cannot “opt out”
 - All or nothing solution
 - E.g. PBS not SGE

◆ What we need

- Dynamic Stack
- Component Based Stack
- User / Developer Extensible

PICK PACKAGES

- > COMBO #1: PREMIUM
- > COMBO #2: SPORT
- > COMBO #3: COLD WEATHER
- > NEXT STEP

MINI COOPER S

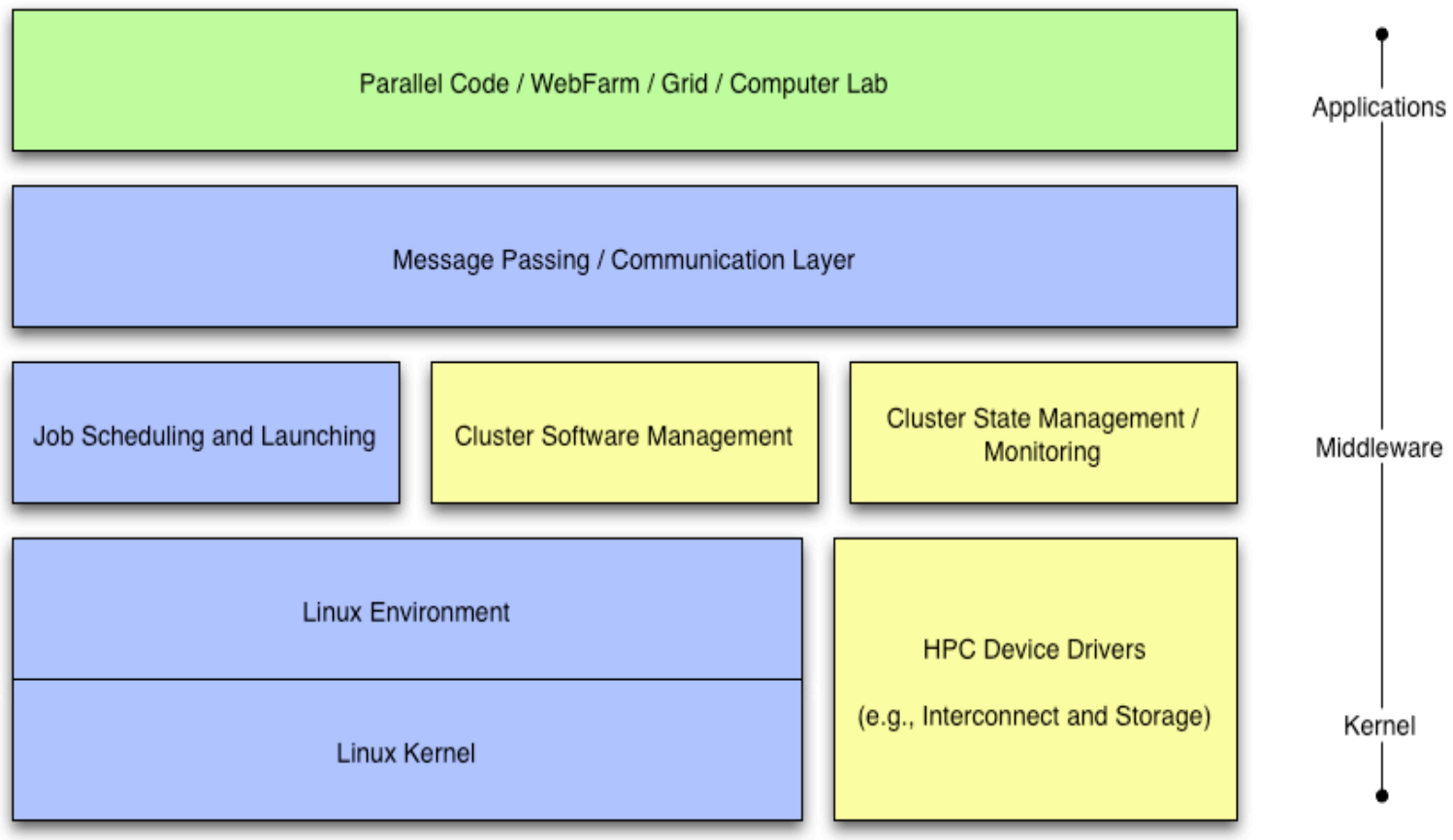
CLICK IMAGE TO ADD THE SPORT PACKAGE TO YOUR LIST.

THE SPORT PACKAGE WILL ADD:
Dynamic stability control (DSC), bonnet stripes, xenon headlamps with powerwashers, front fog lamps, 17-inch alloy 5-lite wheels with 205/45 R17 performance or all-season run-flat tires.

Sport Package (\$1350)



Rolls Break Apart Rocks



Rolls: Modifying a Standard System Installer to Support User-Customizable Cluster Frontend Appliances. Greg Bruno, Mason J. Katz, Federico D. Sacerdoti, and Phil M. Papadopoulos. *IEEE International Conference on Cluster Computing*, San Diego, California, Sep. 2004.



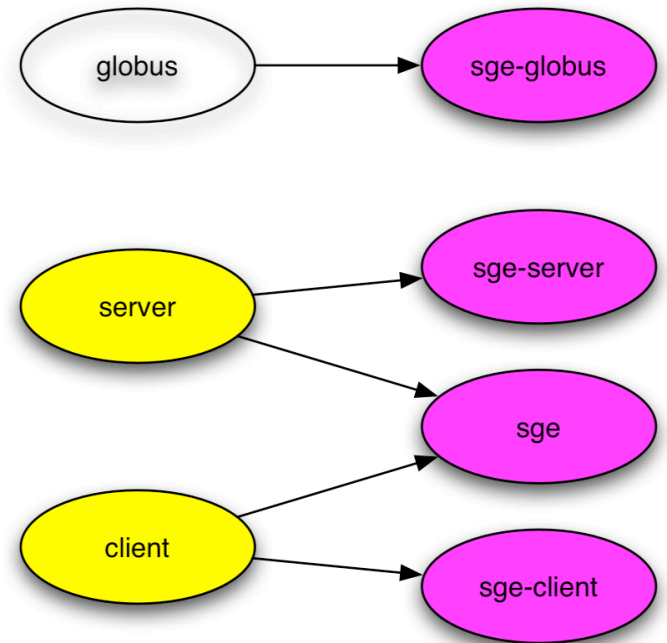
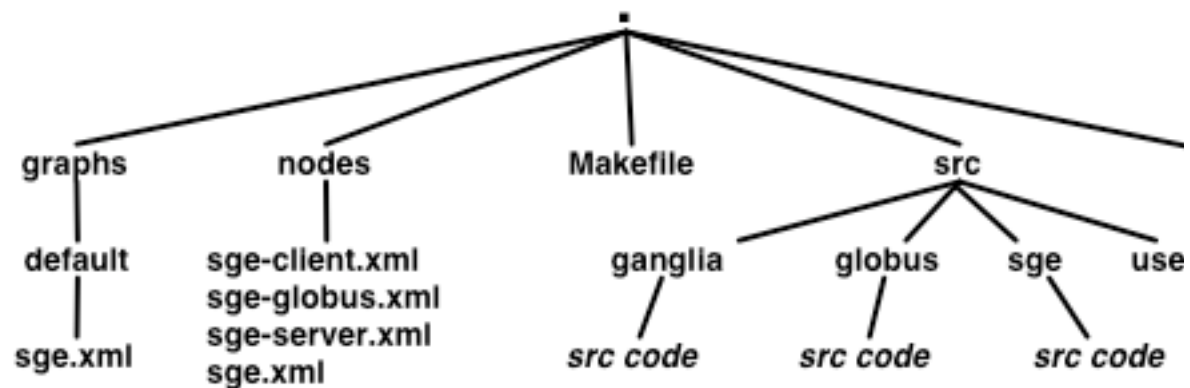
Our Graph Had Colors

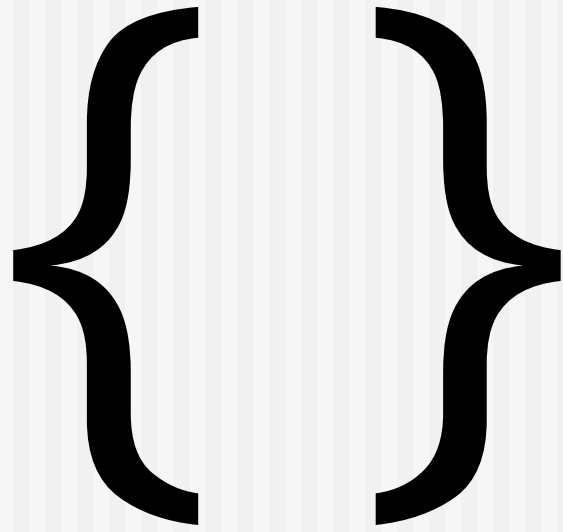




Rolls are sub-graphs

- ◆ A graph makes it easy to ‘splice’ in new nodes
- ◆ Each Roll contains its own nodes and splices them into the system graph file

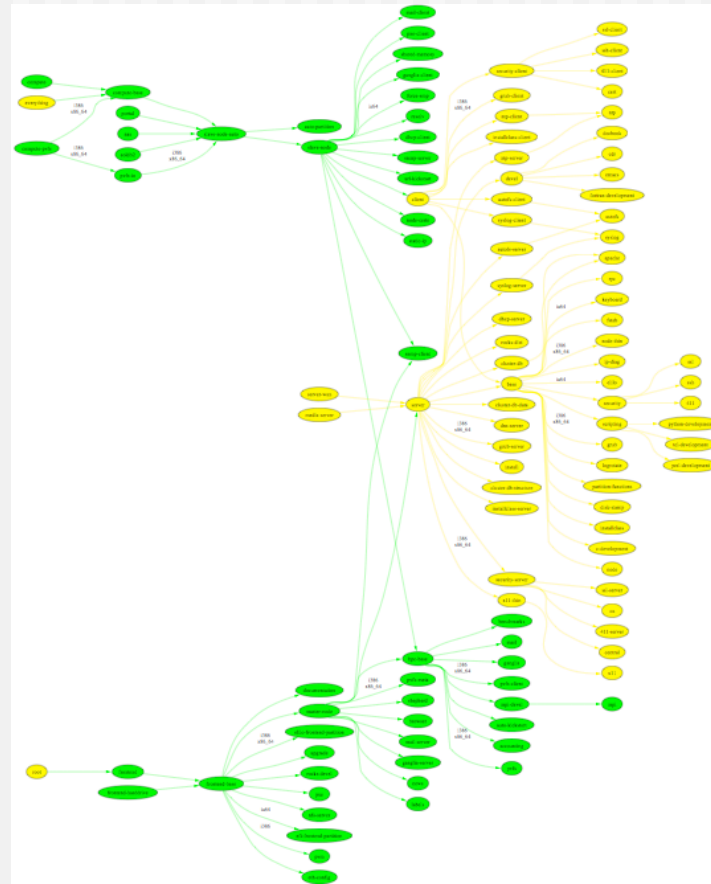




STARTING FROM THE EMPTY SET

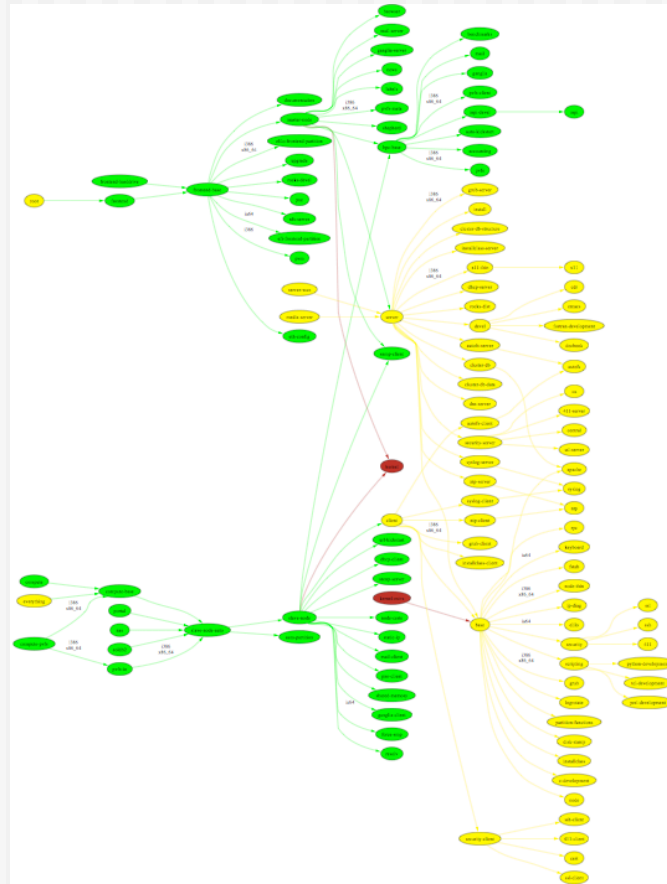


{ base, hpc }



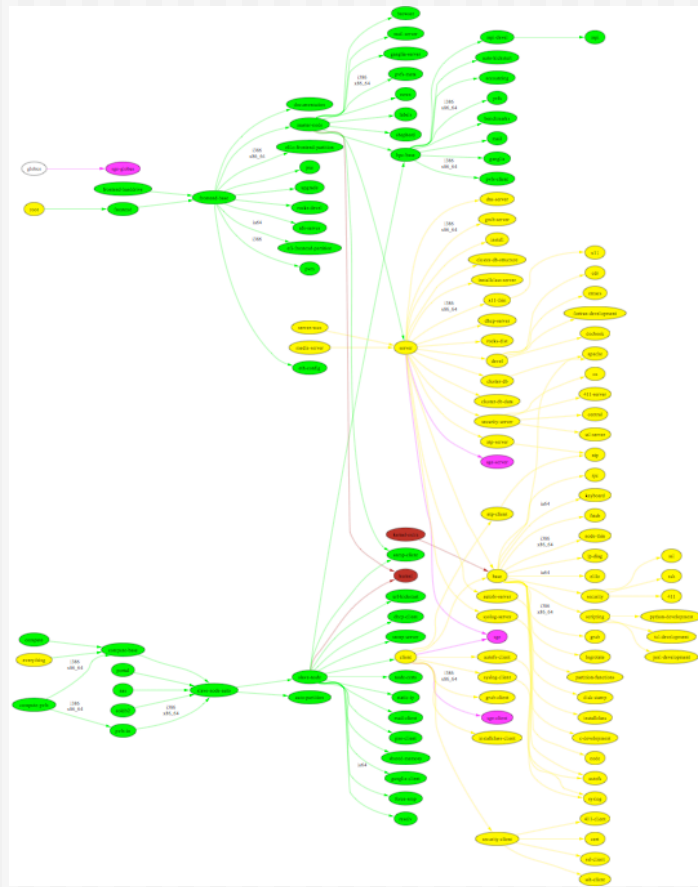


{ base, hpc, kernel }





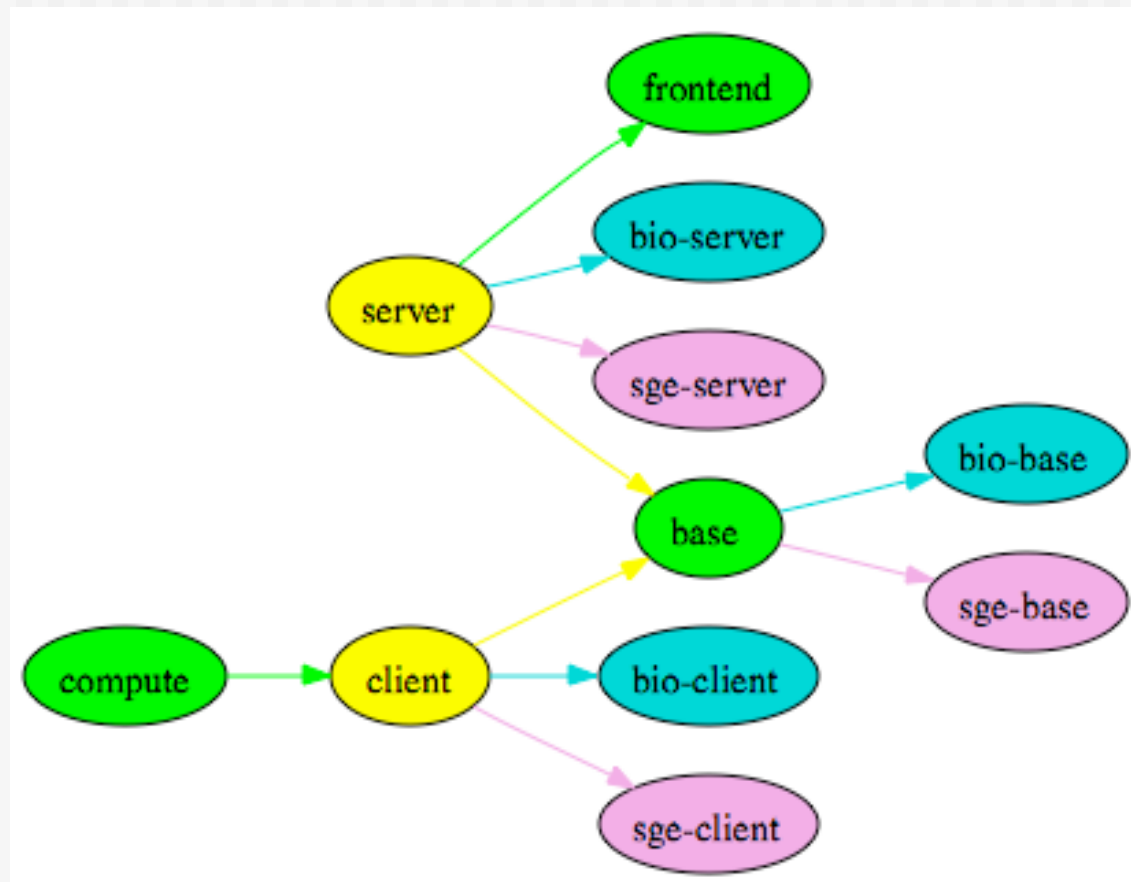
{ base, hpc, kernel, sge }





Simplified Example

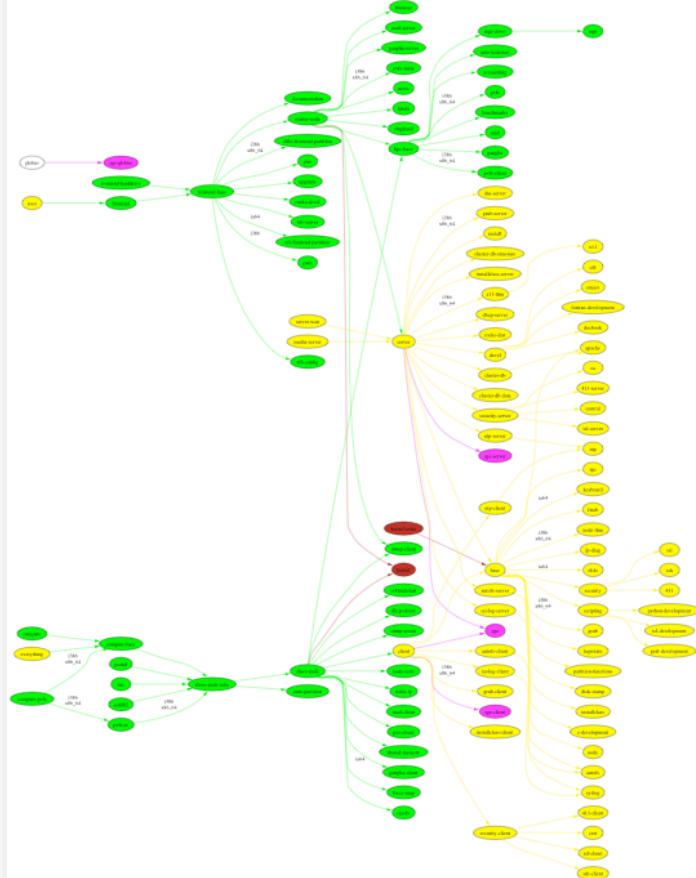
{base, hpc, sge, bio}



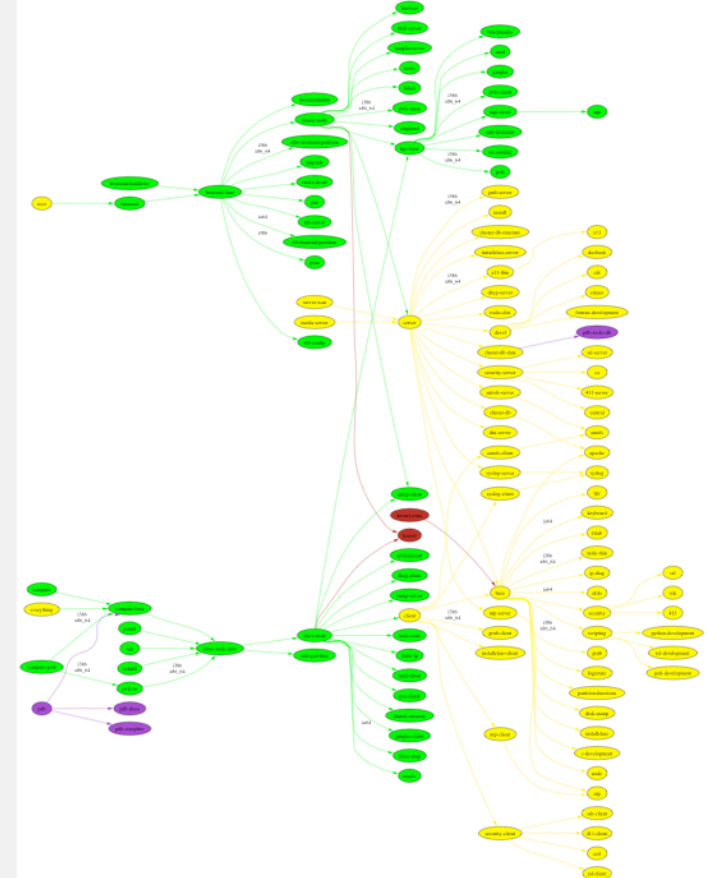


Two different Clusters

MPI Cluster:: {base, hpc, kernel, sge}



Protein Databank:: {base, hpc, kernel, pdb}





Questions?

1. Rocks Command Line
2. Attributes
3. Graph Traversal
4. Roll Sub-Graphs