



NPACI Rocks Tutorial

NPACI All Hands Meeting

March 18, 2003

www.rocksclusters.org

Schedule

- Rocks 101 (45 mins)
 - **Introduction to Rocks**
 - **Cluster Architecture (HW & SW)**
 - **Rocks software overview**
- Advanced Rocks (45 mins)
 - **Modifying the default configuration**
 - **Cluster Monitoring (UCB's Ganglia)**
- Hands on Labs (90 mins)
 - **Breakout into groups and build an x86 cluster**
 - **Run an MPI job**
 - **Customization (packages, monitoring)**
 - **Compare to IA64 cluster**



Rocks 101



Make clusters easy

- Enable application scientists to build and manage their own resources
 - **Hardware cost is not the problem**
 - **System Administrators cost money, and do not scale**
 - **Software can replace much of the day-to-day grind of system administration**
- Train the next generation of users on loosely coupled parallel machines
 - **Current price-performance leader for HPC**
 - **Users will be ready to “step up” to NPACI (or other) resources when needed**
- Rocks scales to Top500 sized resources
 - **Experiment on small clusters**
 - **Build your own supercomputer with the same software!**



Past

- Rocks 1.0 released at SC2000
 - **Good start on automating cluster building**
 - **Included early prototype (written in Perl) of Ganglia**
 - **Result of collaboration between SDSC and UCB's Millennium group.**
- Rocks 2.x
 - **Fully automated installation (frontend and compute nodes)**
 - **Programmable software configuration**
 - **SCS (Singapore) first external group to contribute software patches**



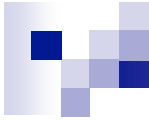
Present

- Rocks 2.3.2 (today's lab is part of our beta testing)
 - **First simultaneous x86 and IA64 release**
- Proven Scalability
 - **#233 on current Top500 list**
 - **287-node production cluster at Stanford**
 - **You can also build small clusters**
- Impact
 - **Rocks clusters on 6 continents**
 - No Antarctica... yet.
 - **4 large NSF ITR's using Rocks as core software infrastructure**



Rocks Registration Page (5 days old)

[http://www.rocksclusters.org/rocks
-register](http://www.rocksclusters.org/rocks-register)



Rocks in the commercial world

- Rocks Cluster Vendors
 - **Cray**
 - **Dell**
 - **Promicro Systems**
 - See page 79 of April's Linux Journal
 - **SCS (in Singapore)**
 - Contributed PVFS, SGE to Rocks
 - Active on the Rocks mailing list
- Training and Support
 - **Intel is training customers on Rocks**
 - **Callident is offering support services**



Promicro Systems

SSC Automatic custom installs with Kickstart

LINUX JOURNAL

Rapid development with *Kylix*
 Rescue problem photos with *The GIMP*
GNUstep for cross-platform apps

The Monthly Magazine of the Linux Community • APRIL 2003

Standardizing the Linux Desktop

Mouse from computer to computer with *Synergy*

Exploring **GNOME 2.0**

All-Linux small business with *OpenOffice* and *GnuCash*
 SANE vs. *VueScan*: power scanning tools
 USB drivers for 2.6 • Using the kernel crypto API

USA \$5.00 CAN \$6.50

ROCKS

PAPER

SCISSORS

ROCKS ALWAYS WINS

Two out of three ain't bad.....but why gamble. When deploying and installing your cluster you can't afford to waste time. Cluster deployment and management has never been easier with **Rock's** software. The **Cluster Management Tool** allows for simple installation and fast updates to the OS and other critical applications. This translates into exceptional scalability and stable, secure computing. Our high performance Linux clusters are customized to meet your specific business objectives. Based on Intel® Xeon™ and Itanium™ processors, they deliver breakneck speeds for even the most demanding applications. When it comes to cluster management, **ROCKS ALWAYS WINS.**

For more information on Promicro ROCKS clusters and a free copy of the software, please visit our web site at <http://www.promicro.com/solutions/show.asp?id=13>

promicro
S Y S T E M S
HIGH PERFORMANCE COMPUTING SOLUTIONS

Alternative Cluster Toolkits

- Cplant
 - **DOE clustering software**
- OpenMOSIX
 - **Single System Image**
- OpenSCE
 - **Kasetsart University, Thailand**
 - **Compatible with Rocks**
- OSCAR
 - **Aka cluster-in-a-box, MSC.Linux**
 - **“Best-practices of clustering”**
 - **Packaging effort, not a cluster distribution**
- Scyld
 - **Bproc based system (SSI process space)**
 - **Commercial**
- SCore
 - **Oldest clustering effort**
 - **Touches on all aspects of clustering software**



What sets us apart

- Fully automated cluster deployment
 1. **Get and burn ISO CD (DVD for IA64) image from <http://www.rockclusters.org>**
 2. **Boot frontend with CD/DVD**
 3. **Fill out 7 configuration screens (mostly Red Hat)**
 4. **Reboot frontend machine**
 5. **Integrate compute nodes with `insert-ethers`**
 6. **Ready to go!**
- Complete out of the box solution with rational default settings
- Identical environment for x86 or IA64



Testimonials and Complaints

From the Rocks-discuss mail list
and other sources



New User on Rocks Mail List

“I managed to install Rocks with five nodes. The nodes have a small HD 2.5 GB each, the cluster is in my home on a private network behind a Linux box "firewall". And it looks like everything is working fine. I can see all the nodes and the front-end in the ganglia web interface. I built it so I can learn more about clusters.

And to tell the truth I have no idea on what to do with it, I mean where to start, how to use it, what to use it for.”



Power Users

- Response to previous poster
 - **“It's sometimes scary how easy it is to install Rocks.”**
 - **This coined the phrase “Rocks, scary technology”**

- Comment from a cluster architect
 - **“You guys are the masters of the bleedin' obvious.”**
 - **This one flattered us**



Another New User

“I've set up a Rocks Cluster thanks to the bone simple installation.

Thanks for making it so easy.

The drawback, because it was so easy, I didn't learn much about clustering.”

Independent Survey of Clustering Software

- <http://heppc11.ft.uam.es/Clusters/Doc>
 - **Compares Rocks, OSCAR and others**
 - **“NPACI Rocks is the easiest solution for the installation and management of a cluster under Linux.”**
 - **“To install and manage a Linux cluster under OSCAR is more difficult than with Rocks.”**
 - “With OSCAR it is necessary to have some experience in Linux system administration, and some knowledge of cluster architecture”
- Our goal is to “make clusters easy”
 - **Automate the system administration wherever possible**
 - **Enable non-cluster experts to build clusters**



And Finally a weekly message

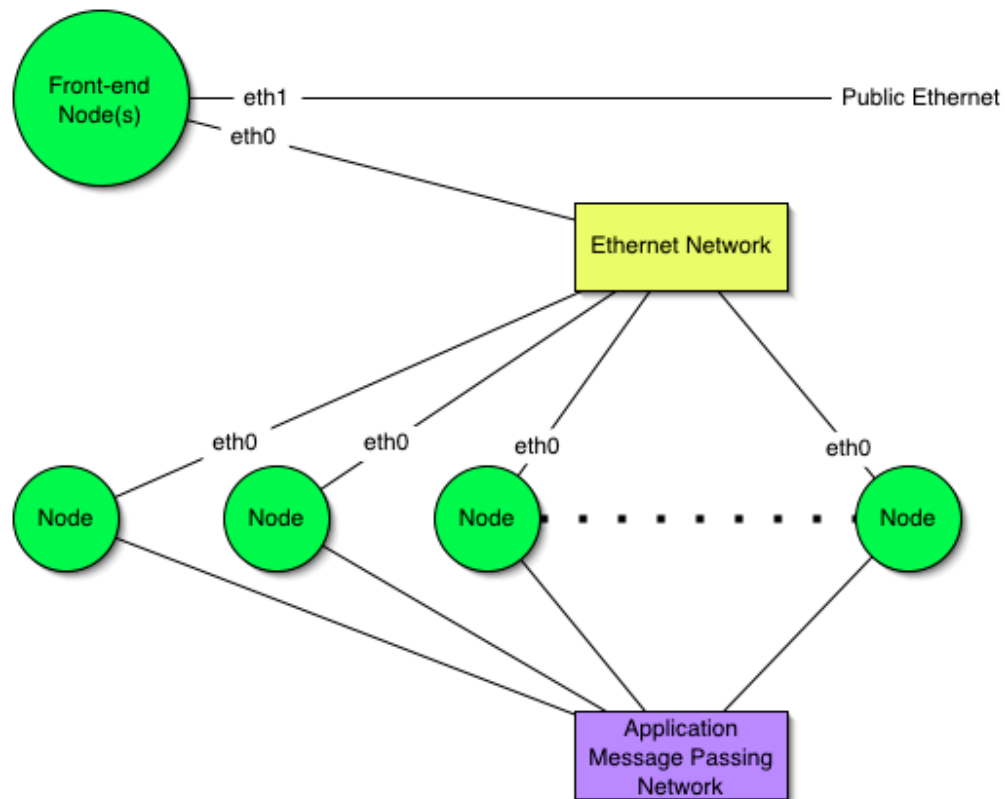
- “Your documentation sucks.”
- Guilty, but improving
 - **Rocks now installs users guide on every new cluster**
 - **Mailing list has several extremely helpful users**



Hardware



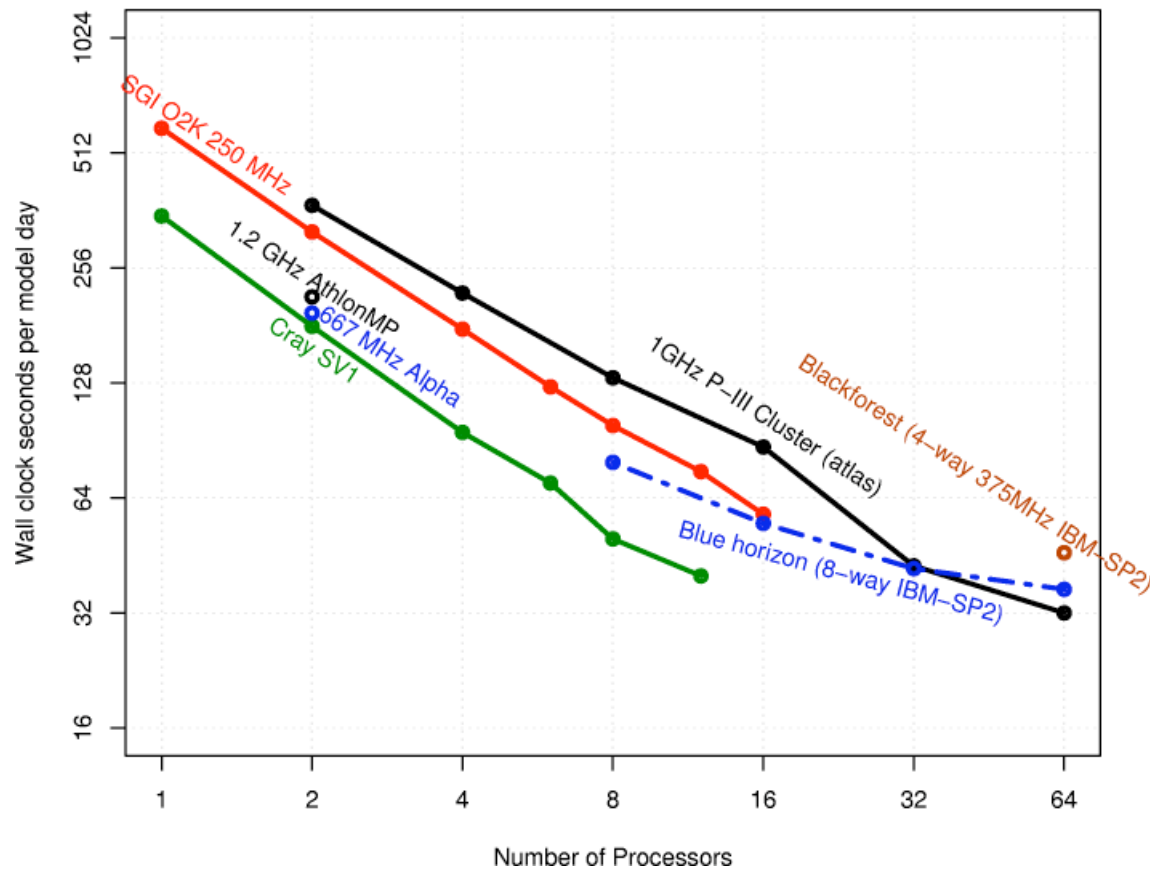
Basic System Architecture





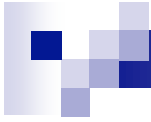
Why x86 clusters?

Scaling of CCM3 Atmospheric GCM at T42 resolution

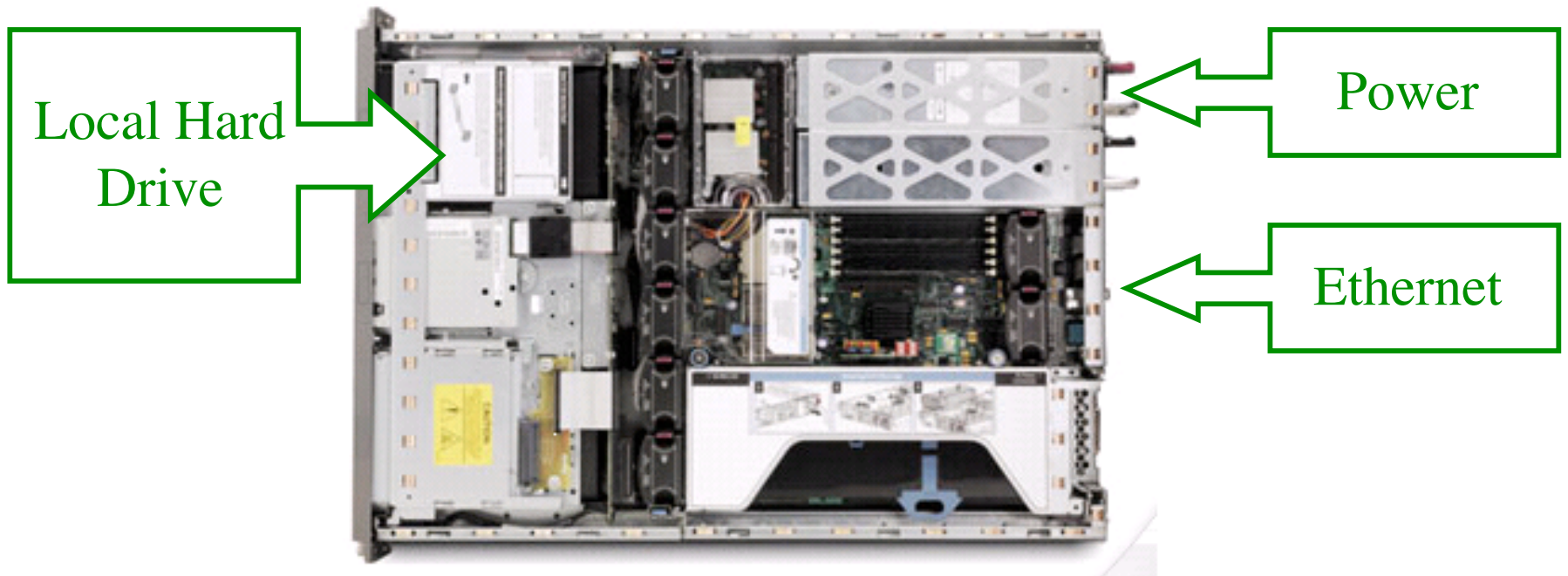


Source: Dave Pierce, SIO

Copyright © 2003 UC Regents



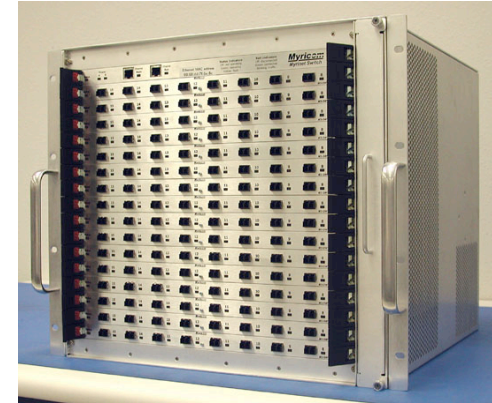
Minimum Components



x86 / IA64 server

Optional Components

- Myrinet high-performance network
- Network-addressable power distribution unit
- keyboard/video/mouse network not required
 - **Non-commodity**
 - **How do you manage your management network?**
 - **Crash carts have a lower TCO**

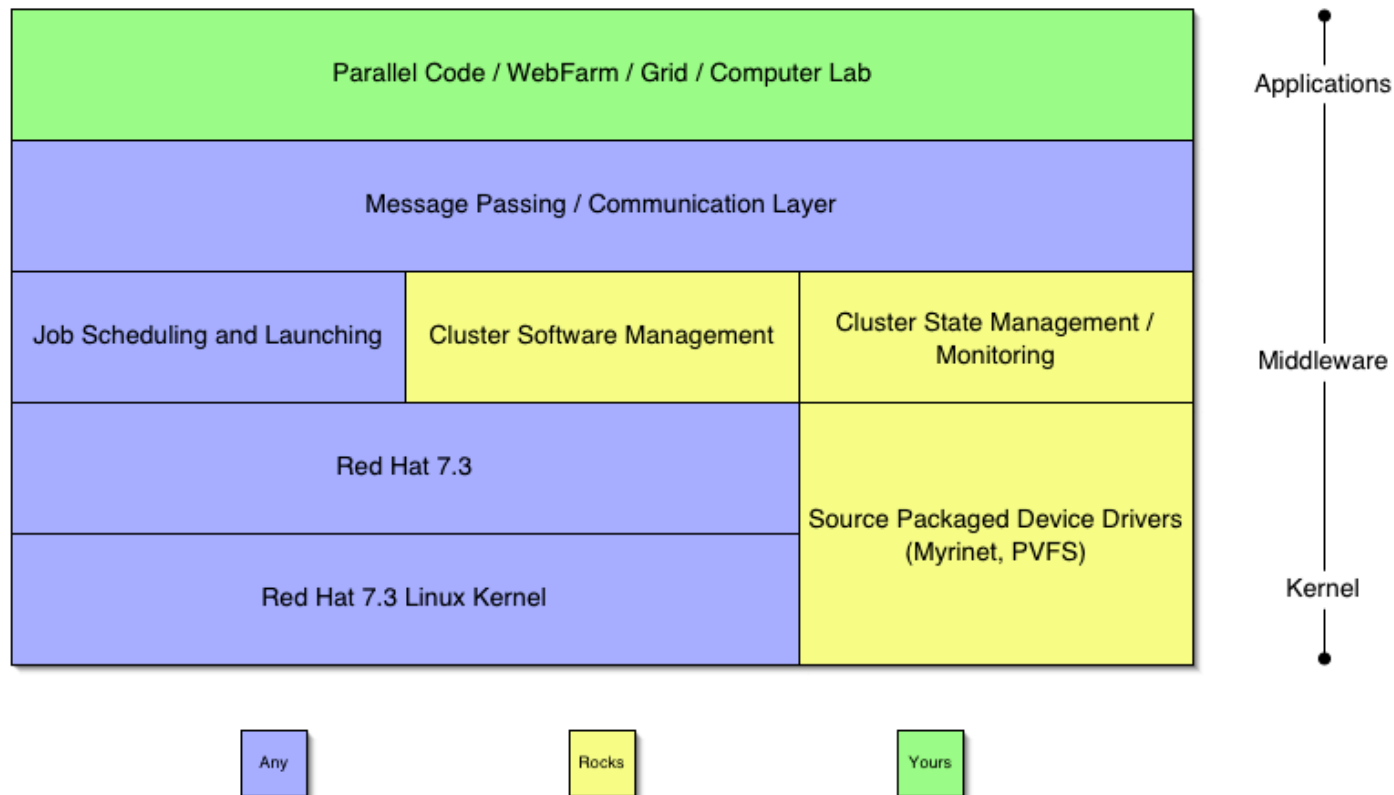




Software

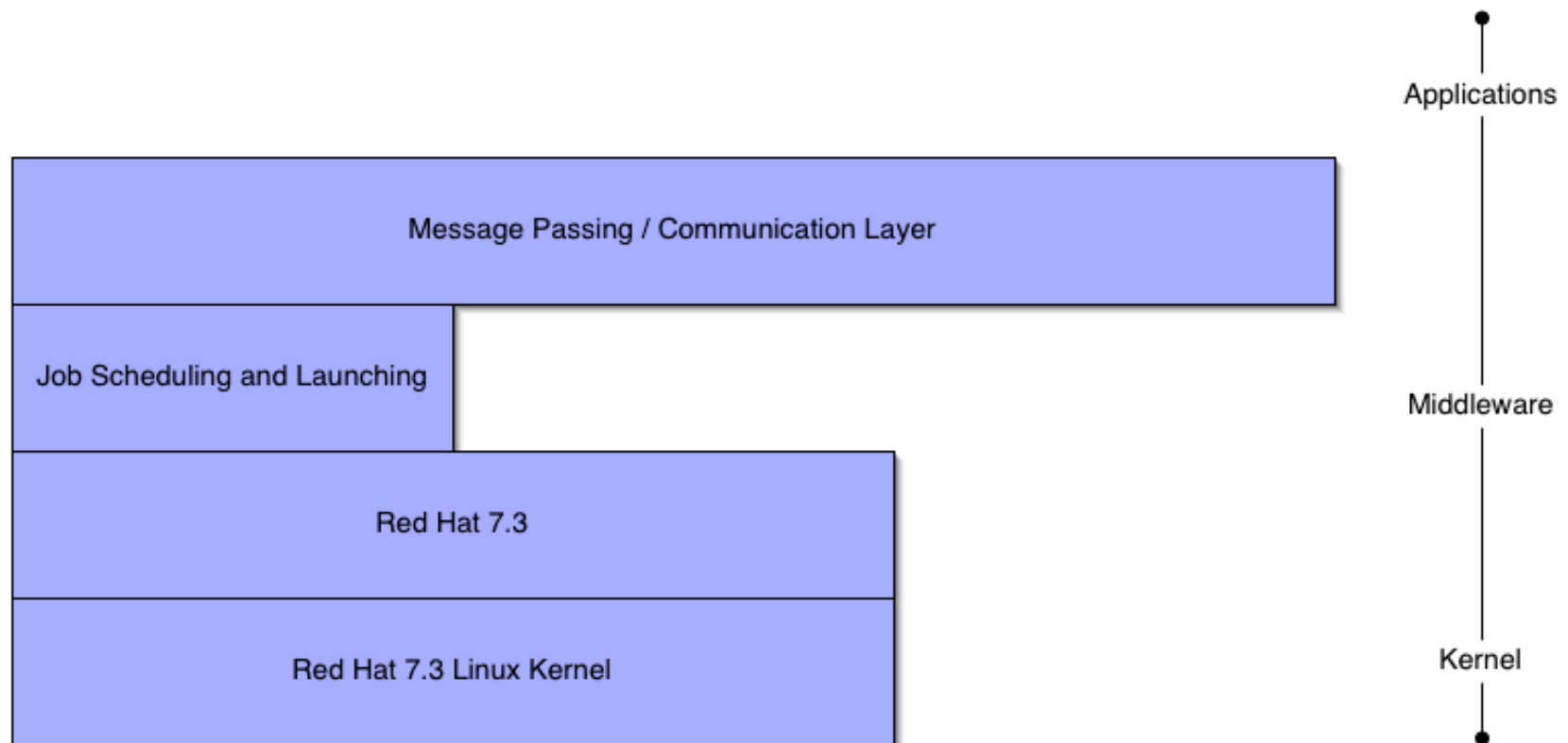


Cluster Software Stack





Common to Any Cluster





Red Hat

- Stock Red Hat 7.3 w/ updates (AW 2.1 for IA64)
- Linux 2.4 Kernel
- No support for other distributions
 - **Red Hat is the market leader for Linux**
 - In the US
 - And becoming so in Europe
- Excellent support for automated installation
 - **Scriptable installation (Kickstart)**
 - **Very good hardware detection**

Batch Systems

- Portable Batch System
 - **MOM**
 - Daemon on every node
 - Used for job launching and health reporting
 - **Server**
 - On the frontend only
 - Queue definition, and aggregation of node information
 - **Scheduler**
 - Policies for what job to run out of which queue at what time
 - Maui is the common HPC scheduler
- SGE - Sun Grid Engine
 - **Alternative to PBS**
 - **Integrated into Rocks by SCS (Singapore)**
- Scheduler manage scarce resources
 - **Clusters are cheap**
 - **You might not want a scheduler**

Communication Layer

- None
 - **“Embarrassingly Parallel”**
- Sockets
 - **Client-Server model**
 - **Point-to-point communication**
- MPI - Message Passing Interface
 - **Message Passing**
 - **Static model of participants**
- PVM - Parallel Virtual Machines
 - **Message Passing**
 - **For Heterogeneous architectures**
 - **Resource Control and Fault Tolerance**

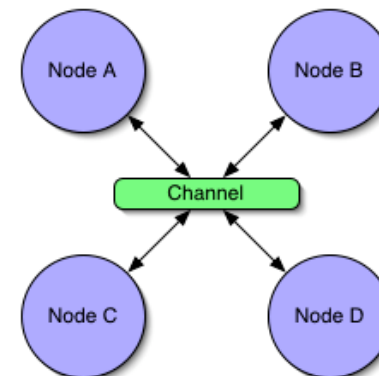
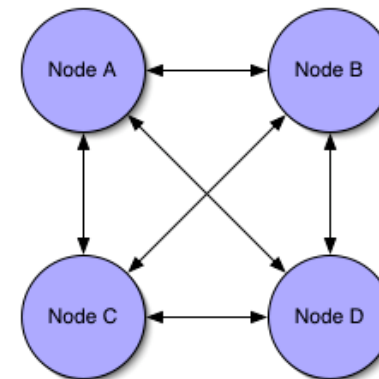
Sockets are low level

■ Sockets

- **Point-to-Point**
- **N machines = $(n^2 - n)/2$ connections**
- **1, 3, 6, 10, 15, ...**

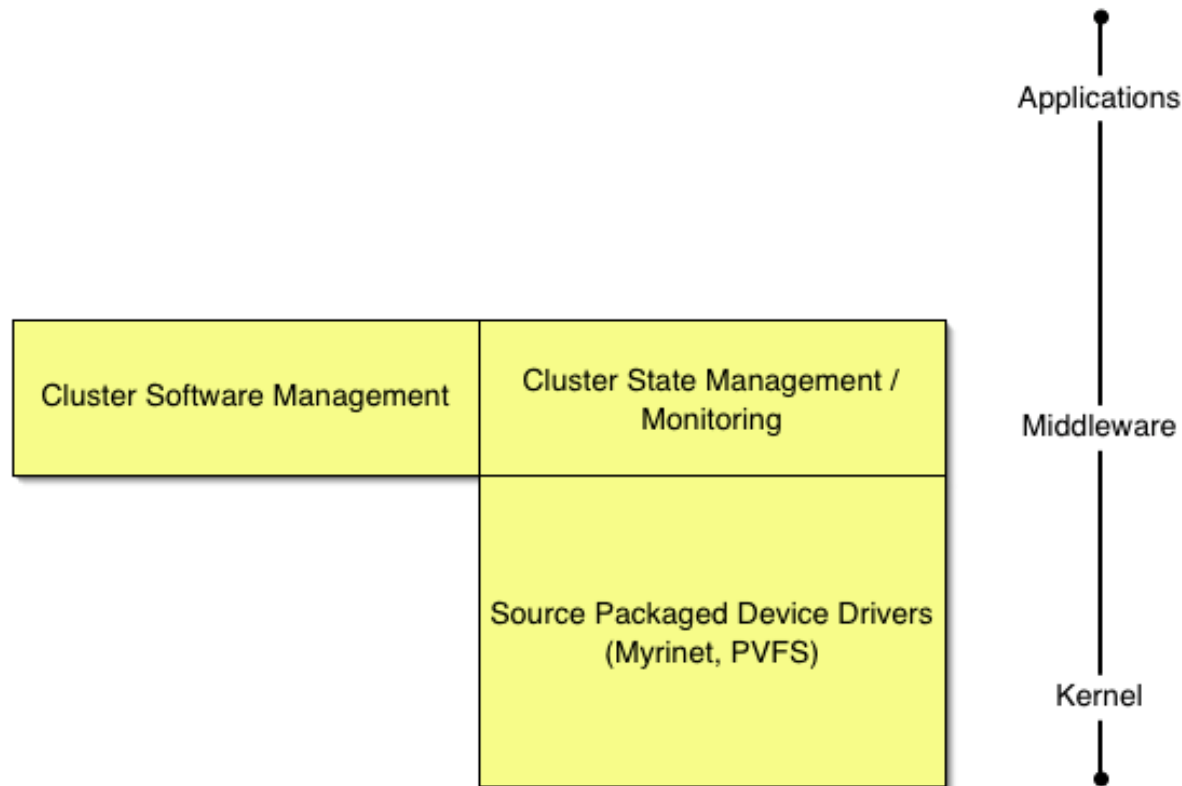
■ MPI/PVM

- **Shared virtual channel**
- **Implementation could be sockets**
- **Easier to program**





Rocks Cluster Software



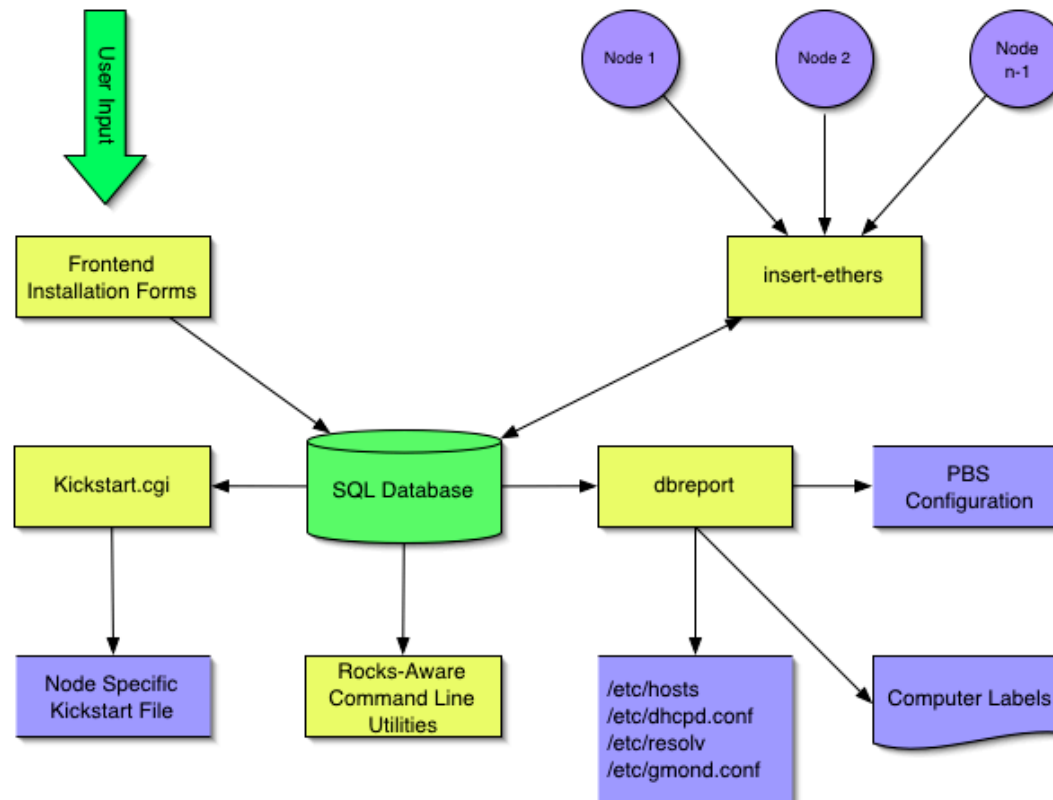
Cluster State Management

- **Static Information**
 - **Node addresses**
 - **Node types**
 - **Site-specific configuration**
- **Dynamic Information**
 - **CPU utilization**
 - **Disk utilization**
 - **Which nodes are online**



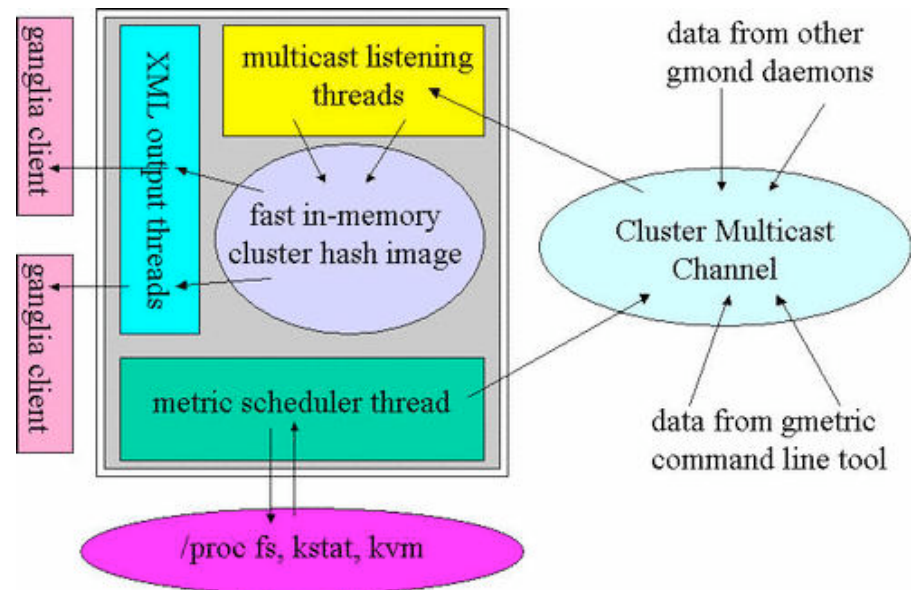


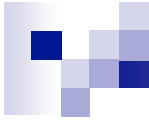
Cluster Database




Ganglia

- Scalable cluster monitoring system
 - **Based on ip multi-cast**
 - **Matt Massie, et al from UCB**
 - <http://ganglia.sourceforge.net>
- Gmond daemon on every node
 - **Multicasts system state**
 - **Listens to other daemons**
 - **All data is represented in XML**
- Ganglia command line
 - **Python code to parse XML to English**
- Gmetric
 - **Extends Ganglia**
 - **Command line to multicast single metrics**






Ganglia Screenshot




Host Report for Tue, 18 Mar 2003 01:28:58 +0000 [Get Fresh Data](#)

Last [Node View](#)

 <http://ganglia.sourceforge.net>

[Our Cluster](#) > **britannic**

britannic Overview

 This node is up and running

Time and String Metrics	
Name	Value
boottime	Tue, 18 Mar 2003 00:23:20 +0000
gexec	OFF
machine_type	ia64
os_name	Linux
os_release	2.4.18-e.12smp
sys_clock	Tue, 18 Mar 2003 00:25:34 +0000
uptime	0 day, 1:5

Constant Metrics	
Name	Value
cpu_idle	97.1 %
cpu_num	2
cpu_speed	900 MHz
mem_total	1011568 KB
mtu	1500 B
swap_total	1048544 KB

britannic LOAD last hour

Processes

1-Minute Load Total CPUs Running Processes

britannic CPU last hour

Percent

User CPU Nice CPU System CPU Idle CPU

britannic MEM last hour

Bytes

Memory Used Memory Shared Memory Cached Memory Buffered Memory Swapped Total In-Core Memory



Cluster Software Management

Software Packages

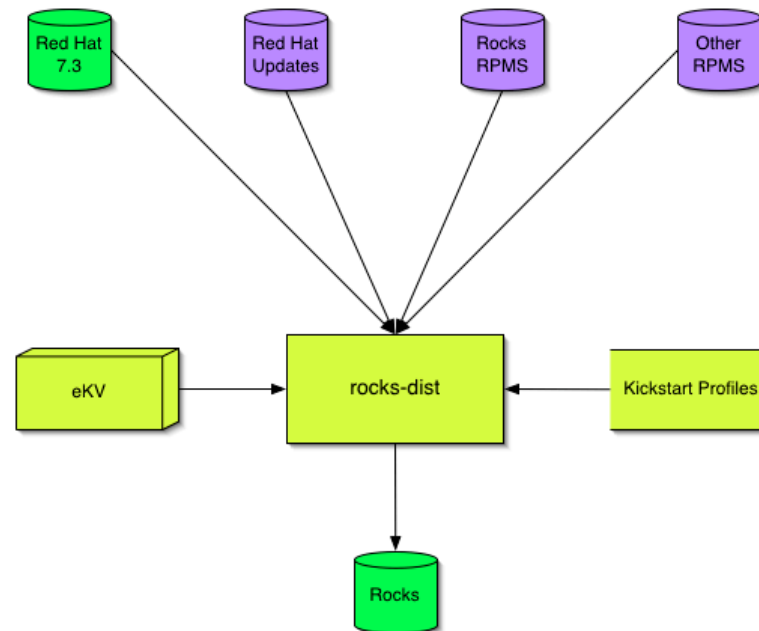
- RPMs
 - **Standard Red Hat (desktop) packaged software**
 - **Or your own addons**
- Rocks-dist
 - **Manages the RPM repository**
 - **This is the distribution**

Software Configuration

- Tuning RPMs
 - **For clusters**
 - **For your site**
 - **Other customization**
- XML Kickstart
 - **Programmatic System Building**
 - **Scalable**



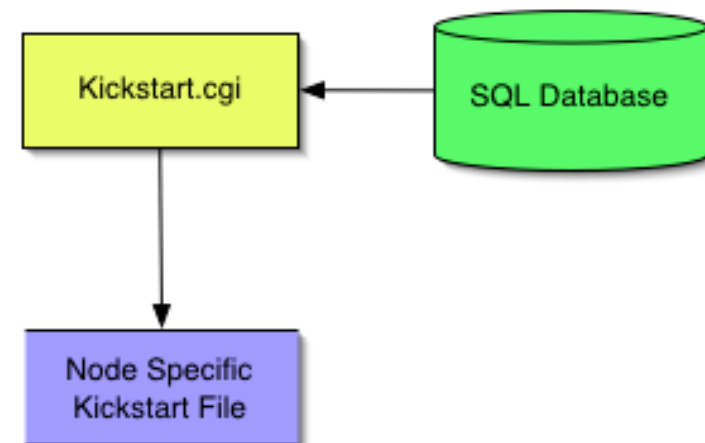
Building a Rocks Distribution

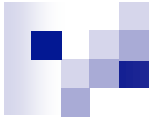


- Start with Red Hat
- Add updates, Rocks (and optional other) software
- Add Kickstart profiles
- Modify Red Hat installation boot image
- Resulting in a Red Hat compatible Rocks distribution

Kickstart

- Red Hat's Kickstart
 - **Monolithic flat ASCII file**
 - **No macro language**
 - **Requires forking based on site information and node type.**
- Rocks XML Kickstart
 - **Decompose a kickstart file into nodes and a graph**
 - Graph specifies OO framework
 - Each node specifies a service and its configuration
 - **Macros and SQL for site configuration**
 - **Driven from web cgi script**





Sample Node File

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

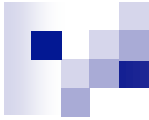
  <package>&ssh;</package>
  <package>&ssh;-clients</package>
  <package>&ssh;-server</package>
  <package>&ssh;-askpass</package>

<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

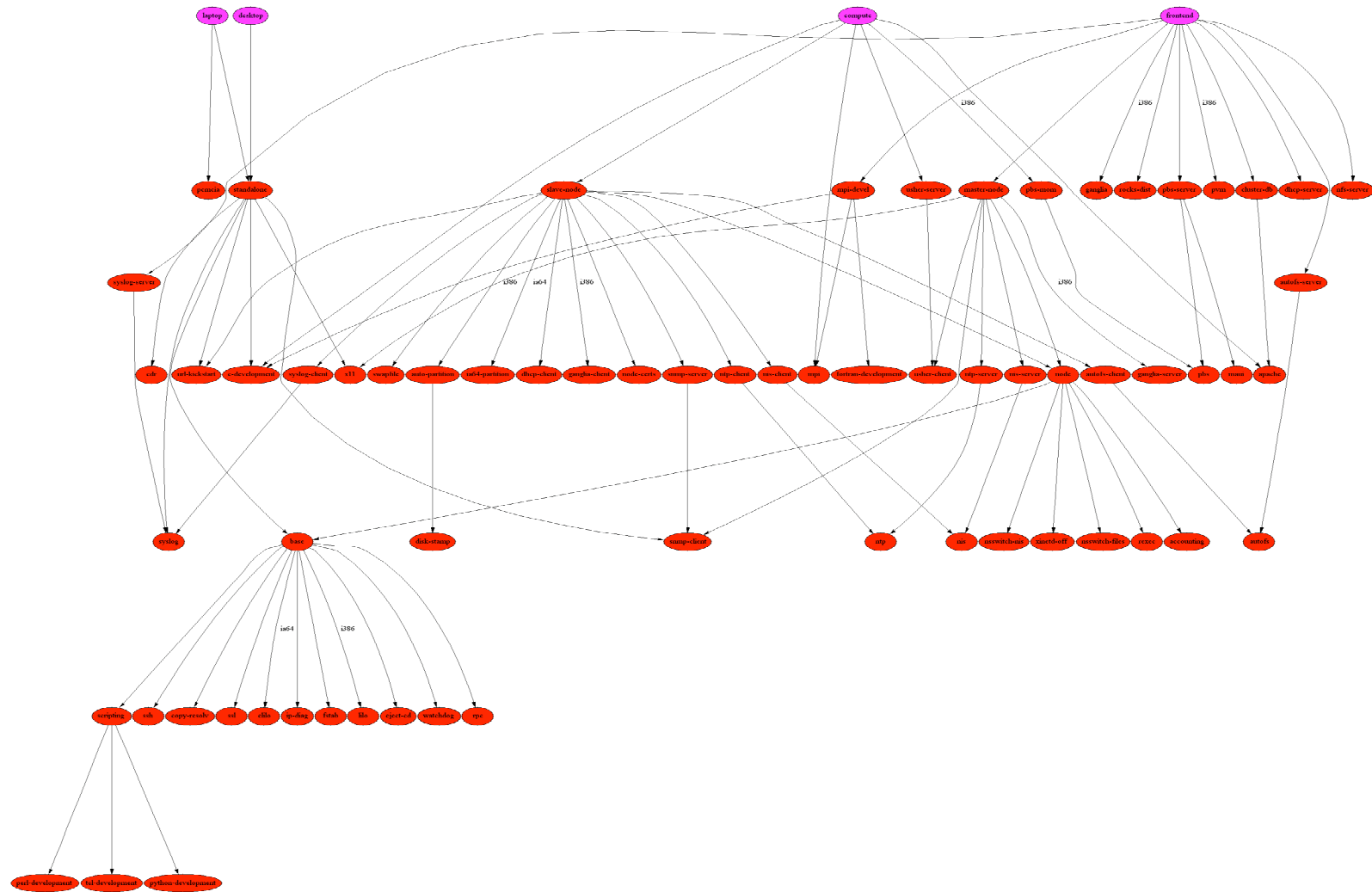
```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE kickstart SYSTEM "@GRAPH_DTD@">

<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="lilo" arch="i386"/>
  <edge from="base" to="elilo" arch="ia64"/>
  ...
  <edge from="node" to="base" weight="80"/>
  <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>
```

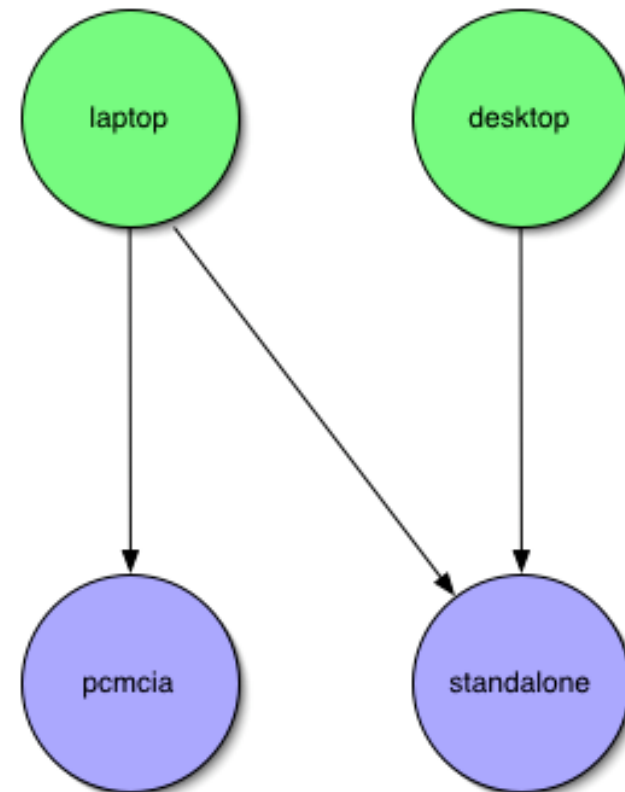


Kickstart framework



Appliances

- Laptop / Desktop
 - **Appliances**
 - **Final classes**
 - **Node types**
- Desktop IsA
 - **standalone**
- Laptop IsA
 - **standalone**
 - **pcmcia**
- Code re-use is good





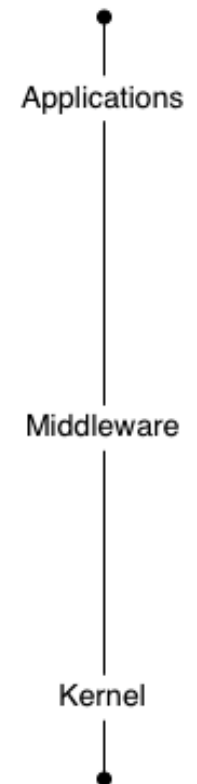
Optional Drivers

- PVFS
 - **Parallel Virtual File System**
 - **Kernel module built for all nodes**
 - **Initial support (full support in future version of Rocks)**
 - **User must decide to enable**

- Myrinet
 - **High Speed and Low Latency Interconnect**
 - **GM/MPI for user Applications**
 - **Kernel module built for all nodes with Myrinet cards**



Your Cluster Software





Let's Build a Cluster

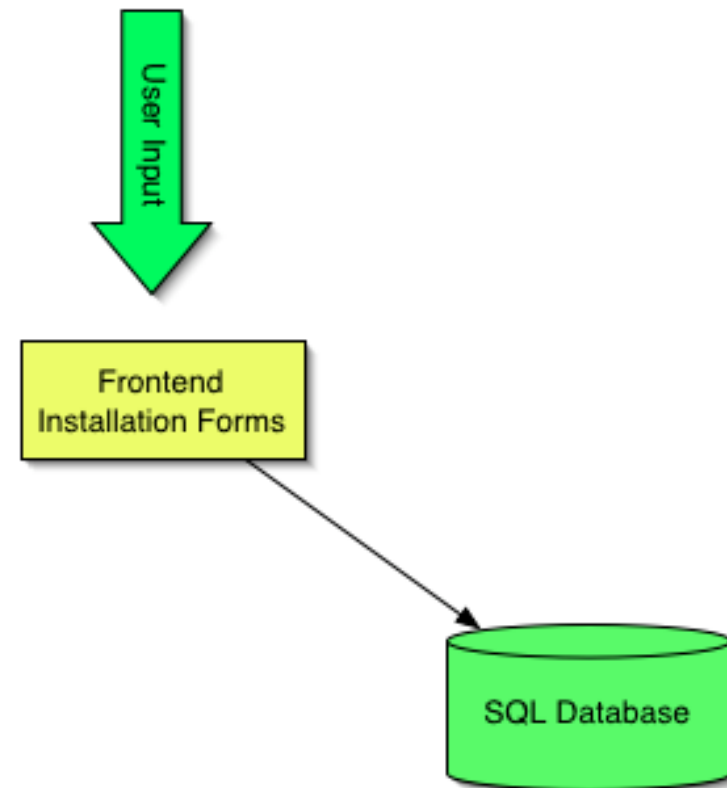


Hardware Setup

- Grab two nodes
- Use the cross-over cable to connect eth0 on the frontend to eth0 on the compute node
- Hook up a monitor and keyboard

Software Installation

- Put the CD in the frontend
- At the 'boot:' prompt, type:
 - **frontend**
- Start configuration...





Cluster Information

```
root@sedona:/home/install
Red Hat Linux (C) 2002 Red Hat, Inc.
Rocks version 2.3 -- www.rocksclusters.org

Cluster Information

These fields are optional, but fill in as many as possible.

Cluster Name:  The name of this cluster.
My Cluster
Owner:         The organization that owns this cluster.
Production
Contact:       The administrative contact (email).
admin@place.org
URL:           The website for this cluster.
http://www.place.org/
LatLong:      Your Latitude and Longitude (like 'N32.87 W117.22')
unspecified

OK  Back

<Tab>/<Alt-Tab> between elements | <Space> selects | <F12> next screen
```



Partition your Disks





Configure Private Network

```
root@sedona:/home/install
Red Hat Linux (C) 2002 Red Hat, Inc.
Rocks version 2.3 -- www.rocksclusters.org

Network Configuration for eth0 - (private cluster network)

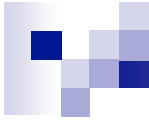
Network Device: eth0 - (private cluster network)

[ ] Use bootp/dhcp
[*] Activate on boot

IP address: 10.1.1.1
Netmask: 255.0.0.0
Default gateway (IP): 10.1.1.1
Primary nameserver: 10.1.1.1
Secondary nameserver:
Tertiary nameserver:

OK Back

<Tab>/<Alt-Tab> between elements | <Space> selects | <F12> next screen
```



Configure Public Network

```
root@sandstorm:/home/install/profiles/2.3/nodes
Red Hat Linux (C) 2002 Red Hat, Inc.
Rocks version 2.3 -- www.rocksclusters.org
Network Configuration for eth1 - (public external network)

Network Device: eth1 - (public external network)

[ ] Use bootp/dhcp
[*] Activate on boot

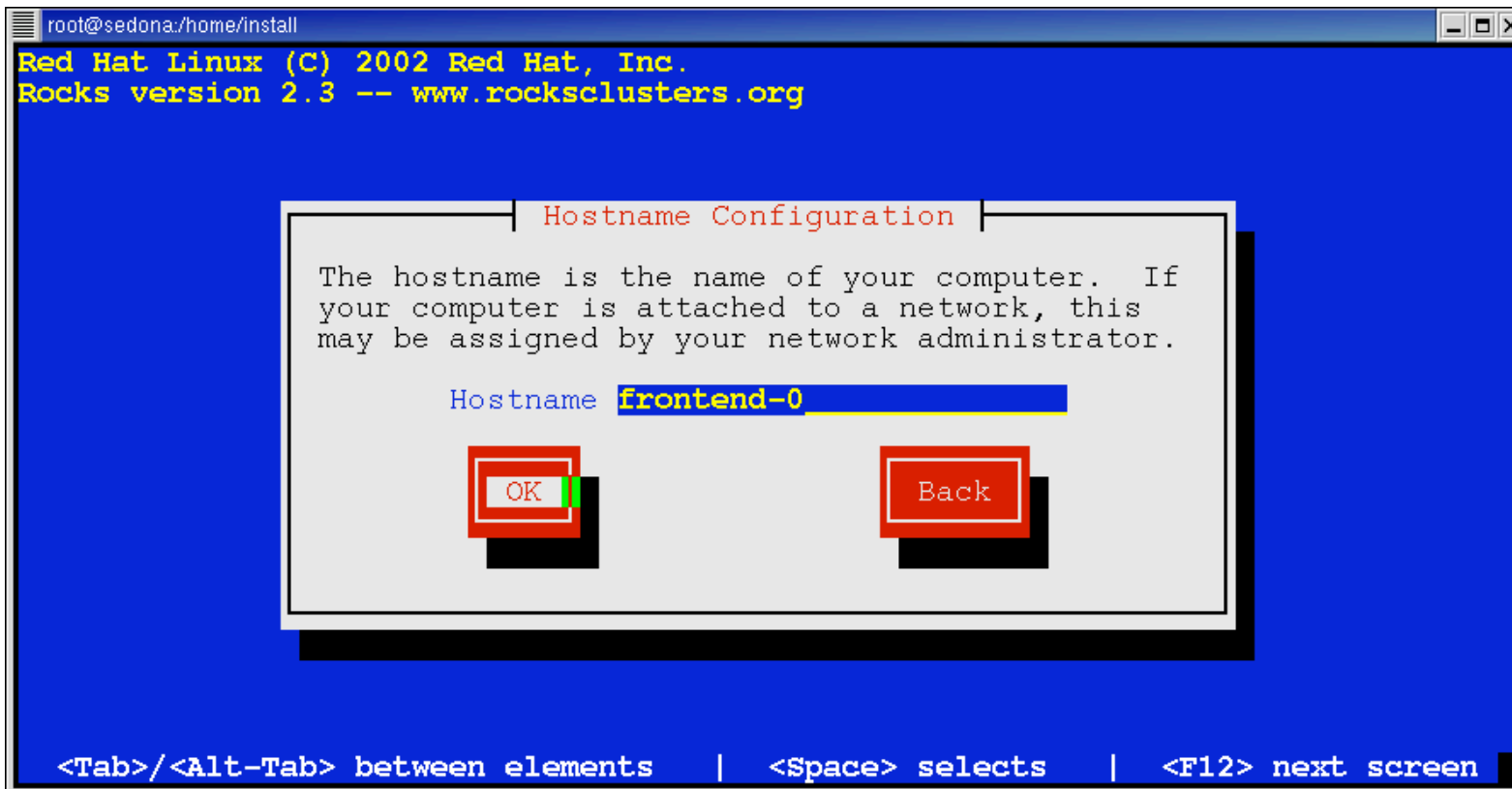
IP address:          198.202.74.156
Netmask:             255.255.255.0
Default gateway (IP): 198.202.74.10
Primary nameserver:  198.202.75.26
Secondary nameserver:
Tertiary nameserver:

OK Back

<Tab>/<Alt-Tab> between elements | <Space> selects | <F12> next screen
```

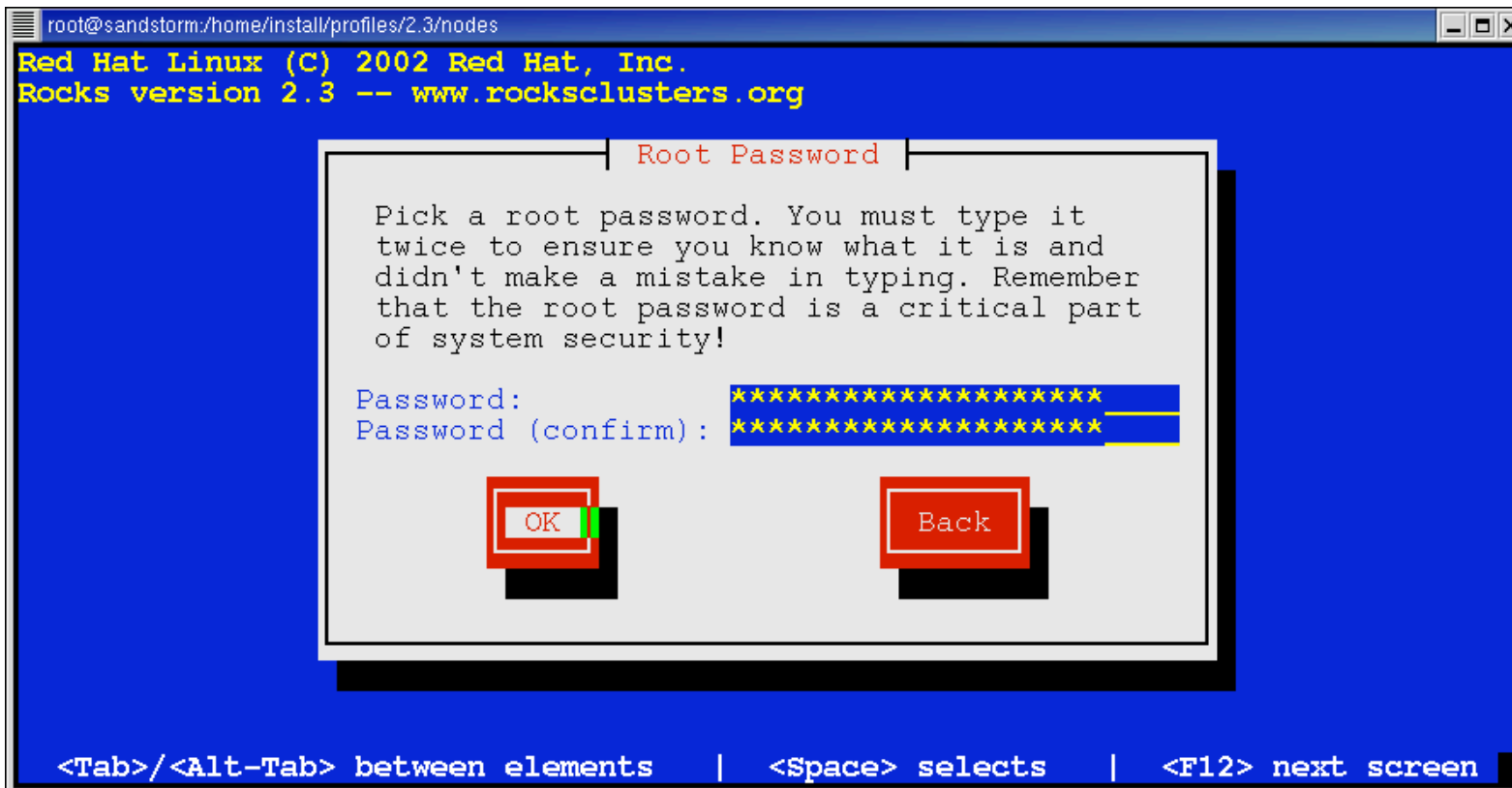


Set your hostname



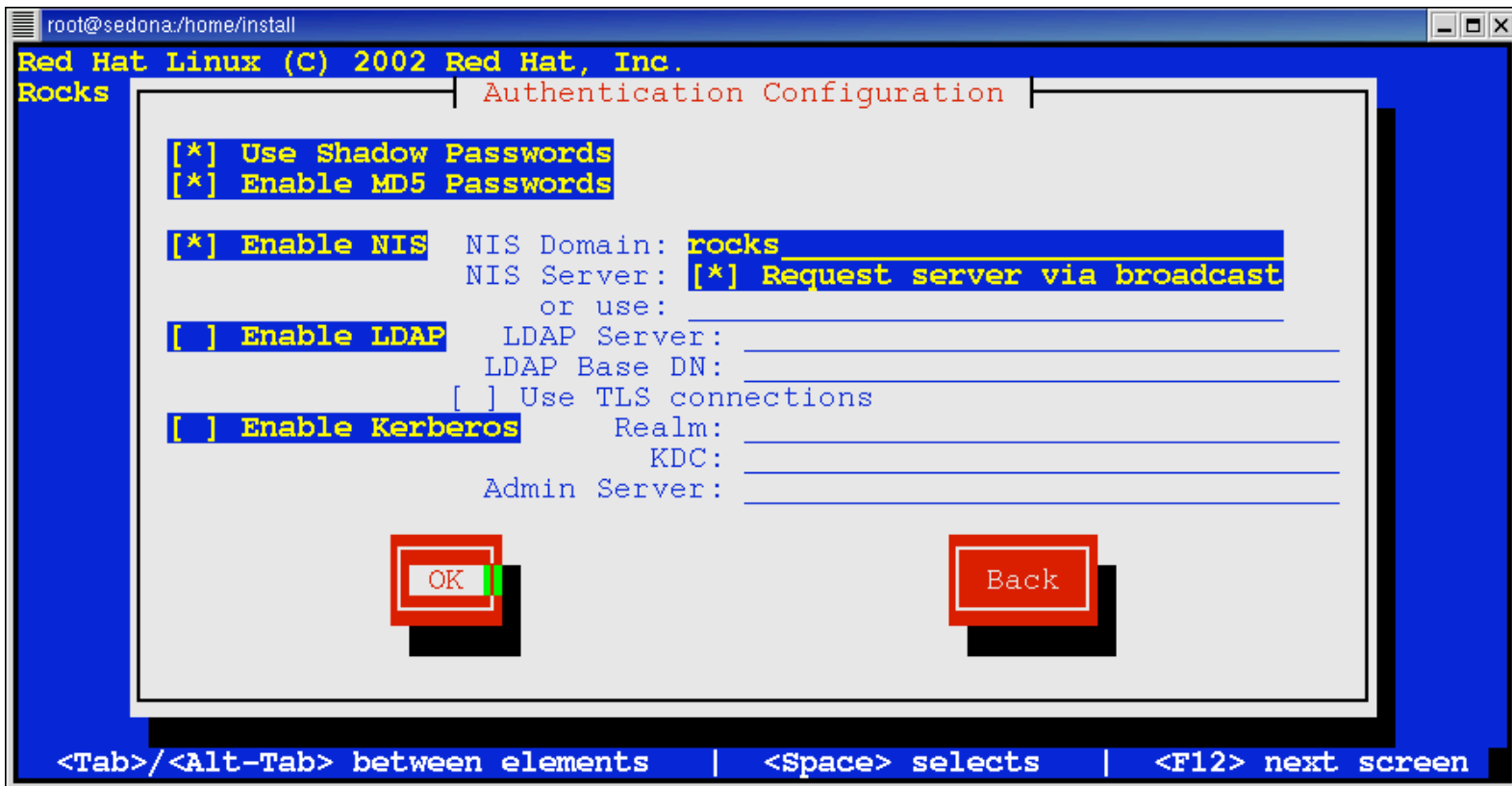


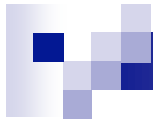
Set the root password





Configure NIS





And away we go...

```
root@sedona:/home/install
Red Hat Linux (C) 2002 Red Hat, Inc.
Rocks version 2.3 -- www.rocksclusters.org

Formatting /export filesystem...

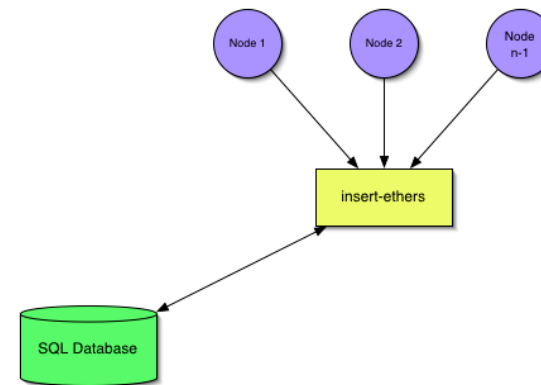
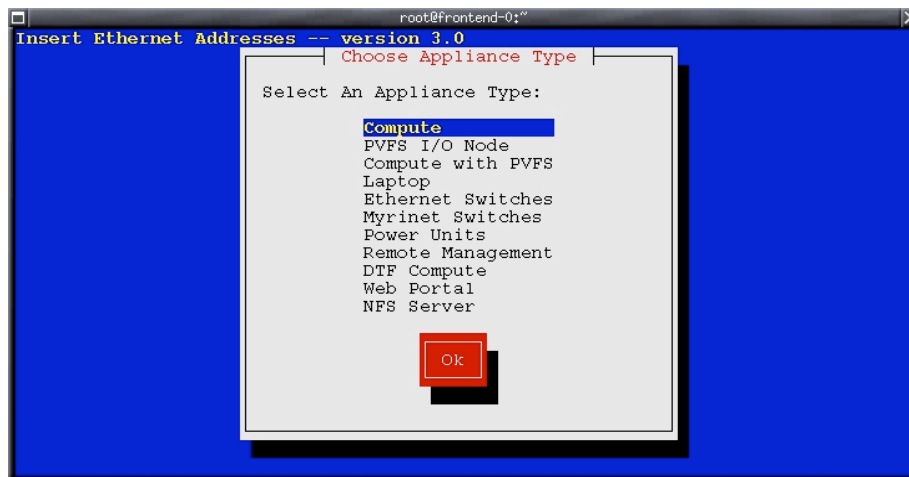
<Tab>/<Alt-Tab> between elements | <Space> selects | <F12> next screen
```



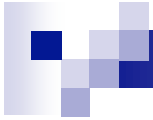
Advanced Rocks



Add Compute Node with Insert-ethers



- Collect the Ethernet MAC address of cluster nodes
- Only done once, during integration
- Populates cluster database



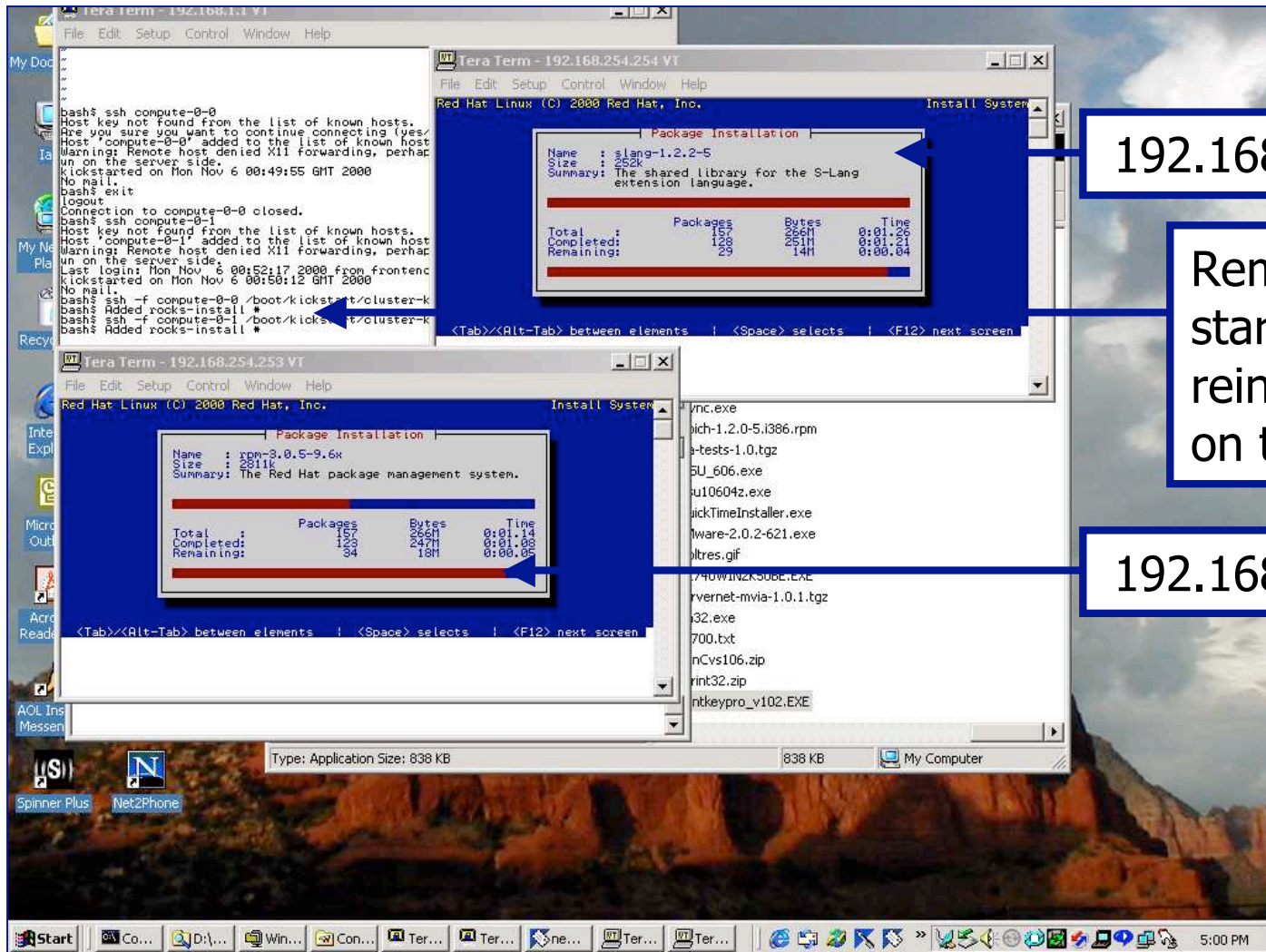
Discovered Appliance

```
root@frontend-0: /root
Insert Ethernet Addresses — version 2.1

Inserted Appliances
Discovered New Appliance
Discovered a new appliance with MAC (00:06:29:1f:41:e8)

Press <F1> to quit
```

Monitor the Install with eKV



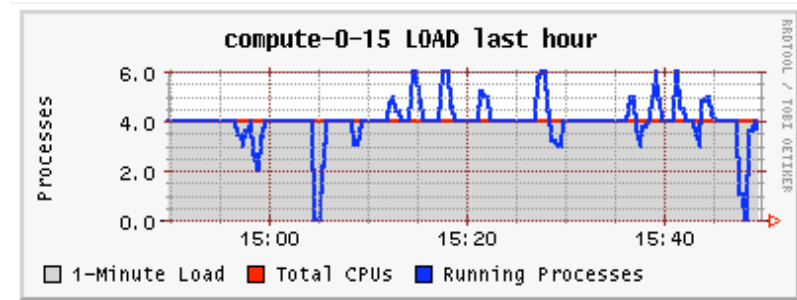
192.168.254.254

Remotely starting reinstallation on two nodes

192.168.254.253

Ganglia Gmetrics

- How to design new *metrics* for ganglia to track and graph.
- Called user-defined metrics, or *gmetrics*.
 - **1. Write gathering function in Python.**
 - **2. Push to all compute nodes.**
 - **3. Allow gschedule to call gatherer repeatedly.**
 - **4. Watch the graphs grow.**



Write Metric Gatherer in Python

- Writing a gmetric gatherer is easy in python: Rocks has native support.
- *Gschedule* is a daemon running by default on all Rocks nodes.
 - **Will call our gatherer function at regular intervals, with some randomization.**
- We just need to provide a **name** and a **value** for the metric.

packets.py

```
#
# Gmetric that publishes the number of outgoing packets
# through device eth0.
#

from gmon.gmetric import Metric
from string import count,split

class packetsOut (Metric):
    "Publishes the number of outgoing packets on eth0."

    dev = "eth0"

    def __init__(self):
        # Publish value every 2 seconds on average.
        Metric.__init__(self, 2)

    def name(self):
        return "packets-out-eth0"

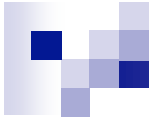
    def value(self):

        dev = open("/proc/net/dev")

        for line in dev.readlines():
            if not count(line, self.dev): continue

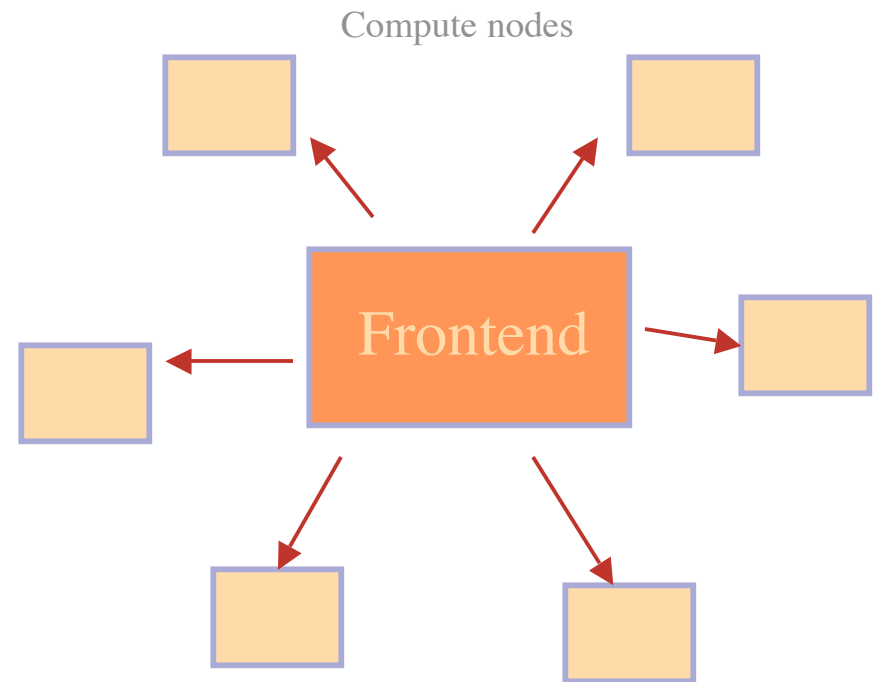
            # Discard 'dev:'
            fields = split(line,":")[1]
            values = split(fields)

            dev.close()
            return int(values[9])
```



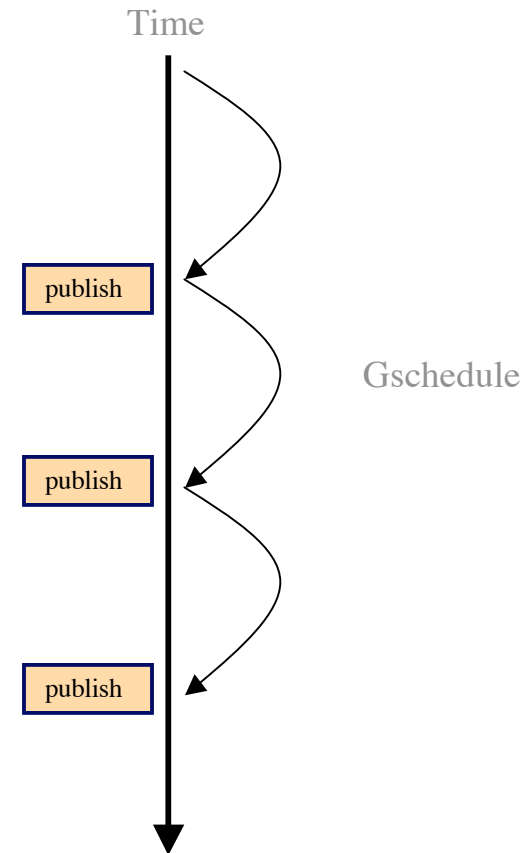
Push new Gmetric to all Nodes

- Test on individual nodes
 - **Log in (ssh)**
 - **Copy python source to /opt/ganglia/lib/python/gmon/metrics/**
- Packaging
 - **Use RPM for permanent installation**



Gschedule daemon

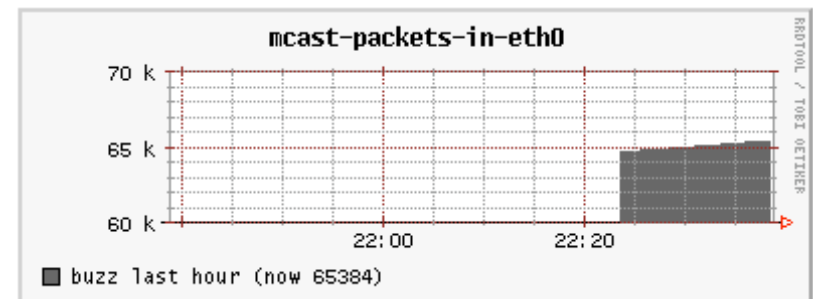
- The Rocks *gschedule* daemon will call our **Packets** metric repeatedly.
- `gschedule --debug`
Or
- `#service gschedule restart`
 - **Will publish our metric every 2 seconds.**
 - **Ganglia's gmetad will detect new metric, and graph it.**



Visualization

- Our new gmetric will be graphed by *gmetad*.
 - **Visible in webfrontend's host view.**
 - **Only metrics which are *numeric and volatile* (*slope* \neq *'zero'*) are graphed.**
- Gmetric restrictions:
 - **Total metric size < 1500 characters.**
 - **Nonetheless, offers lots of possibilities:**
 - Node temp, etc.

Graph of our metric



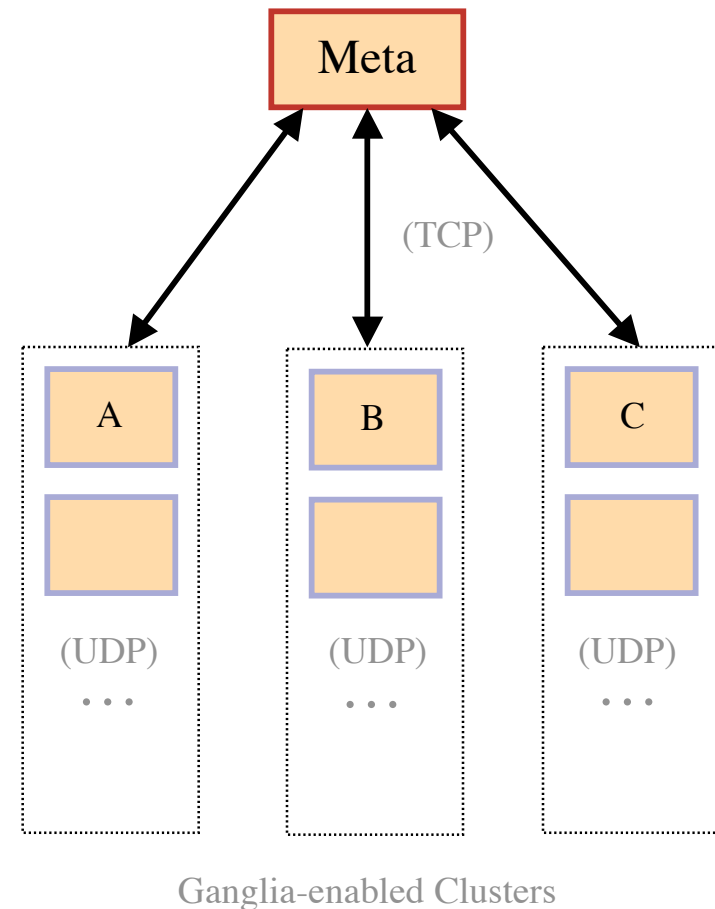
Monitoring Grids with Ganglia

- **Macro:** How to build and grow a monitoring grid.
- A *Grid* is a collection of Grids and Clusters
- A *Cluster* is a collection of nodes
 - **All ganglia-enabled**



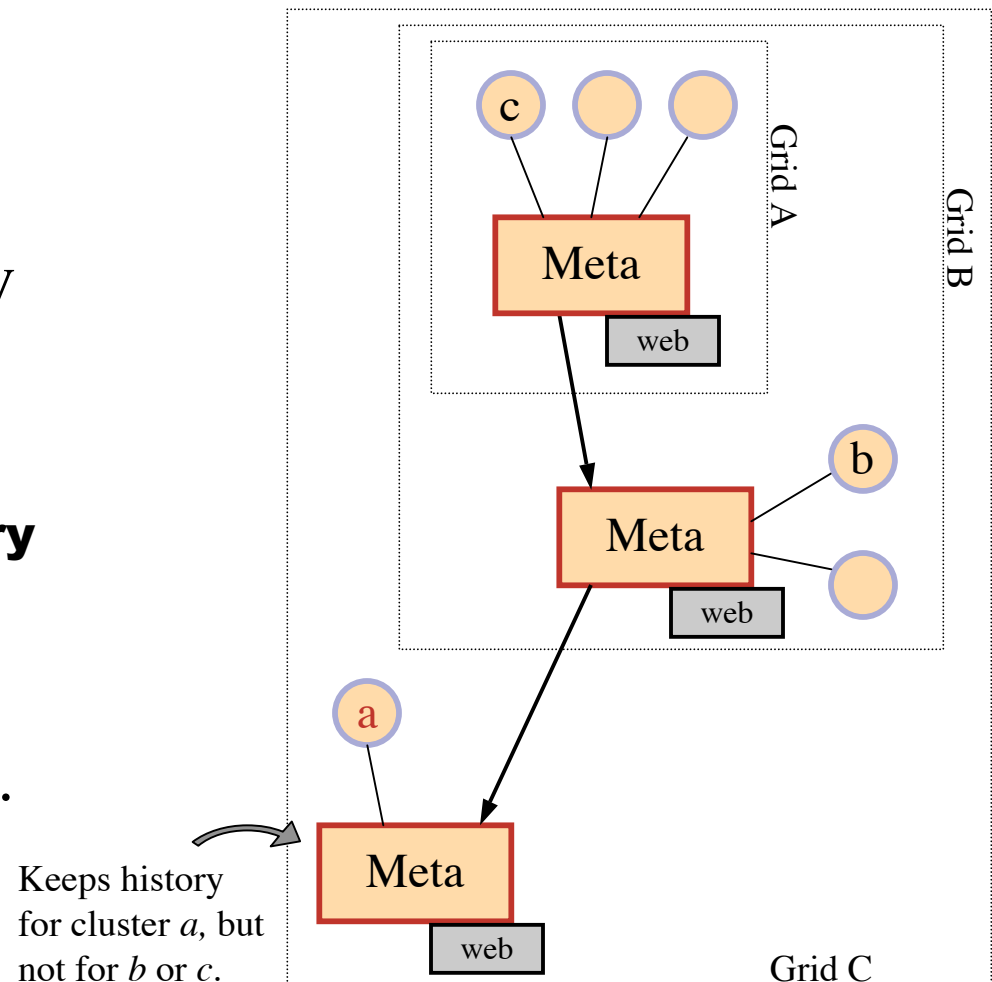
Ganglia Meta Daemon

- **Gmeta** daemon collects ganglia data from multiple clusters.
- Keeps a metric's history over time, used to generate web graphs.
- Requires the *trust* of each cluster.
- Trust is explicitly given

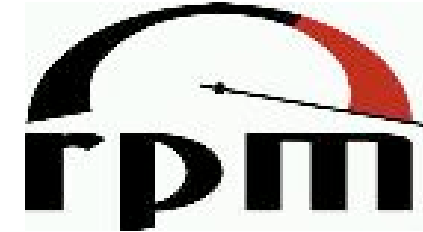


Advanced: Grid Hierarchies

- Enables arbitrarily-deep grid nesting in ganglia
- Distributes metric history collection, a proven bottleneck.
 - **Only keep metric history for child clusters.**
- Ganglia Web frontend is now a distributed system.



Creating a New RPM



- RPMs are created using ‘spec’ files
 - **‘spec’ files are an RPM’s makefile**

- We’ll use the Rocks source code infrastructure to build a new RPM

- First, get the Rocks source code:
 - **# cd /home/install**
 - **# mkdir src**
 - **# cd src**
 - **# export CVS_RSH=ssh**
 - **# cvs -d:pserver:anonymous@cvs.rocksclusters.org:/home/cvs/CVSRROOT/ login**
 - **# cvs -d:pserver:anonymous@cvs.rocksclusters.org:/home/cvs/CVSRROOT/ checkout rocks**

Create the 'contrib-new' package

- Go to the 'contrib' area of Rocks
 - **# cd /home/install/src/rocks/src/contrib/**
- Create the 'new' directory:
 - **# mkdir new**
- Populate 'new' with the 'skeleton' files:
 - **# cp skeleton/* new**
- Change names: 'skeleton' to 'new'
 - **# cd new**
 - **# mv contrib-skeleton.spec.in contrib-new.spec.in**
 - **# vi Makefile**
 - Change '/opt/skeleton' to '/opt/new'



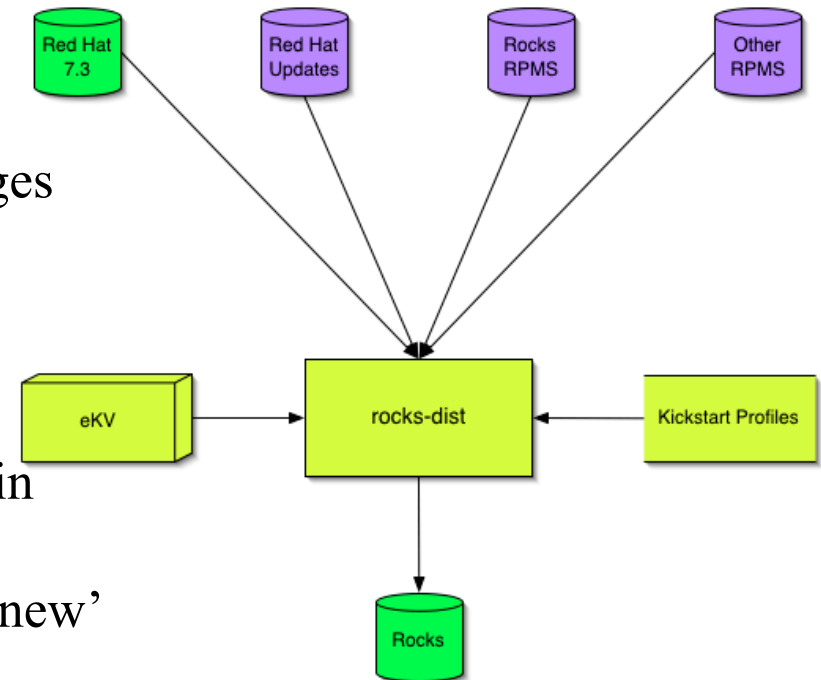
Building the Package

- Execute:
 - **# make rpm**

- This creates a binary and source RPM:
 - **/usr/src/redhat/RPMS/i386/contrib-new-1-1.i386.rpm**
 - **/usr/src/redhat/SRPMS/contrib-new-1-1.src.rpm**

Adding the New Package to the Distribution

- The distribution compiler, rocks-dist, merges RPMs from the following sources:
 - **/home/install/ftp.rockclusters.org/**
 - **/usr/src/redhat/RPMS/**
 - **/home/install/contrib/RPMS/**
- The package contrib-new-1-1.i386.rpm is in /usr/src/redhat/RPMS/i386
- To compile a new distro with the 'contrib-new' package:
 - **# cd /home/install**
 - **# rocks-dist dist**
- This creates a Red Hat compatible distro in /home/install/rocks-dist/
- Look under /home/install/rocks-dist for 'contrib-new-1-1.i386.rpm'



Assigning the New Package to a Cluster Appliance

- Extend the ‘compute’ appliance
 - **# cd /home/install/profiles/2.3.2/site-nodes**
 - **# cp skeleton.xml extend-compute.xml**

- Add the ‘contrib-new’ package
 - **# vi extend-compute.xml**
 - **Add the line:**
 - **<package>contrib-new</package>**
 - **To apply dynamic configuration to components of the ‘contrib-new’ package, write a program and place it between ‘<post> ... </post>’ tags**



Test the changes

- Run kickstart.cgi
 - # **cd /home/install**
 - # **./kickstart.cgi --client='compute-0-0' > /tmp/ks.cfg**

- Look for 'contrib-new' in '%packages' section of /tmp/ks.cfg



Reinstall Compute Nodes

- Use 'shoot-node' to reinstall compute nodes in order to apply the 'contrib-new' package
 - **# shoot-node compute-0-0**



Building a Custom Kernel RPM

- Pick a compute node
 - # ssh compute-0-0

 - **Develop on this node, doesn't "trash" the environment on the frontend**
- Create a custom '.config' file.
 - **This can be done from scratch or based on a canned Red Hat configuration file**
 - # cd /usr/src/linux-2.4
 - # cp configs/kernel-2.4.18-i686-smp.config .config
 - # vi .config



Building a Custom Kernel RPM

- Build a kernel RPM
 - **# make rpm**
- Copy the resulting kernel RPM to the frontend:
 - **# scp /usr/src/redhat/RPMS/i686/kernel.rpm frontend-0:/home/install/contrib/RPMS/public/i386**
- Rebuild the distro on the frontend
 - **# cd /home/install**
 - **# rocks-dist dist**



Compute Node Partitioning

- Creates 4 GB root partition on first drive
 - **This partition is volatile, that is, when the node is reinstalled, this partition is reformatted**
- Remainder of first drive is put into a partition called “/state/partition1”
- For each remaining drives, one partition is created per drive and named “/state/partition2”, “/state/partition3”, etc.
- All partitions labeled “/state/partition[n]” are **not reformatted** on reboots.



Example



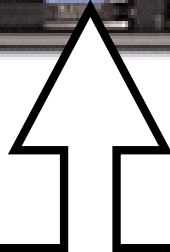
Root Drive 18 GB

/dev/sda1 / 4 GB

/dev/sda2 /state/partition1 14 GB



Example



```
Second Drive      36 GB  
  
/dev/sdb1 /state/partition2      36 GB
```



Example

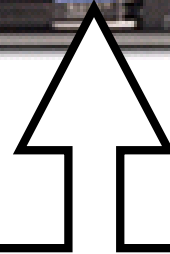
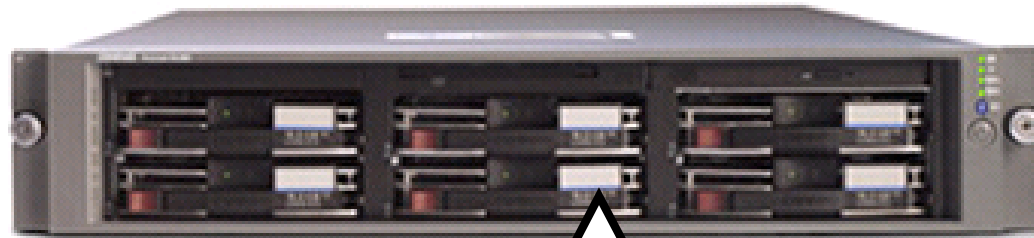


```
Third Drive      18 GB
```

```
/dev/sdc1 /state/partition3 18 GB
```



Example



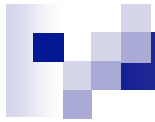
```
Fourth Drive      72 GB  
  
/dev/sdd1 /state/partition4      72 GB
```




Example



```
Fifth Drive      36 GB  
  
/dev/sde1 /state/partition5  36 GB
```



Example



Sixth Drive 181 GB

/dev/sdf1 /state/partition6 181 GB



Example



Device	Name	Size
/dev/sda1	/	4 GB
/dev/sda2	/state/partition1	14 GB
/dev/sdb1	/state/partition2	36 GB
/dev/sdc1	/state/partition3	18 GB
/dev/sdd1	/state/partition4	72 GB
/dev/sde1	/state/partition5	36 GB
/dev/sdf1	/state/partition6	181 GB

User-specified Partitioning

- Override the default partitioning configuration file:
 - **# cd /home/install/profiles/2.3.2/site-nodes**
 - **# cp skeleton.xml replace-auto-partition.xml**
- Add your partitions to 'replace-auto-partition.xml'
 - **<part> / --size 5000 --ondisk sda </part>**
 - **<part> swap --size 512 --ondisk sda </part>**
 - **<part> /mydata --size 1 --grow --ondisk sda </part>**



Building your own CD Set

- After customizing your cluster distribution, make ISO images to export it
- First, you must mirror the full Rocks release
 - **# cd /home/install**
 - **# rocks-dist mirror**
 - Or, put in the second (then third) Rocks CD and execute:
 - **# rocks-dist copycd**



Building your own CD Set

- Build a CD Set
 - **# cd /home/install**
 - **# rm -rf cdrom**
 - **# rocks-dist --dist=cdrom cdrom**

- This creates a CD set under the directory ‘/home/install/cdrom’
 - **# cd /home/install/cdrom/7.3/en/os**



Lab



Thanks

- Inspired by a lab conducted at TACC

Lab

- Building 2-node x86 clusters
- Ganglia grid
- Adding users
- Running Linpack
- Checking out rocks source
- Creating a new package
- Recompiling distro
- Reinstall compute nodes
- Creating/using a new distribution



Add New Users

- Use the standard linux command:
 - **# useradd <username>**

 - **This adds user account info to /etc/passwd, /etc/shadow and /etc/group**
 - Additionally, adds mountpoint to autofs configuration file (/etc/auto.home) and refreshes the NIS database

- To delete users, use:
 - **# userdel <username>**

 - **This reverses all the actions taken by 'useradd'**



Run Linpack

- Login as new user:
 - **# su - bruno**

 - **This prompts for information regarding a new ssh key**
 - Ssh is the sole mechanism for logging into compute nodes
- Copy over the linpack example files:
 - **\$ cp /var/www/html/rocks-documentation/2.3.2/examples/* .**



Run Linpack

- Submit a 2-processor Linpack job to PBS:
 - **\$ qsub qsub-test.sh**

- Monitor job progress with:
 - **\$ showq**

 - **Or the frontend's job monitoring web page**

Scaling Up Linpack

- To scale the job up, edit 'qsub-test.sh' and change:
 - #PBS -l nodes=2
 - **To (assuming you have dual-processor compute nodes):**
 - #PBS -l nodes=2:ppn=2
 - **Also change:**
 - /opt/mpich/ethernet/gcc/bin/mpirun -np 2
 - **To:**
 - /opt/mpich/ethernet/gcc/bin/mpirun -np 4

Scaling Up Linpack

- Then edit 'HPL.dat' and change:
 - 1 Ps
 - **To:**
 - 2 Ps
 - **The number of processors Linpack uses is $P * Q$**
- To make Linpack use more memory (and increase performance), edit 'HPL.dat' and change:
 - 1000 Ns
 - **To:**
 - 4000 Ns
 - **Linpack operates on an $N * N$ matrix**
- Submit the (larger) job:
 - **\$ qsub qsub-test.sh**

Using Linpack Over Myrinet

- Submit the job:
 - **\$ qsub qsub-test-myrin.sh**

- Scale up the job in the same manner as described in the previous slides.

A decorative graphic on the left side of the slide. It features a vertical stack of overlapping squares in various shades of light blue and white. To the right of this stack is a solid dark blue horizontal bar that spans the width of the slide.

End

Go Home