Breaking the Myth of Homogeneous Clusters

Philip M. Papadopoulos
San Diego Supercomputer Center
University of California, San Diego
http://rocks.npaci.edu

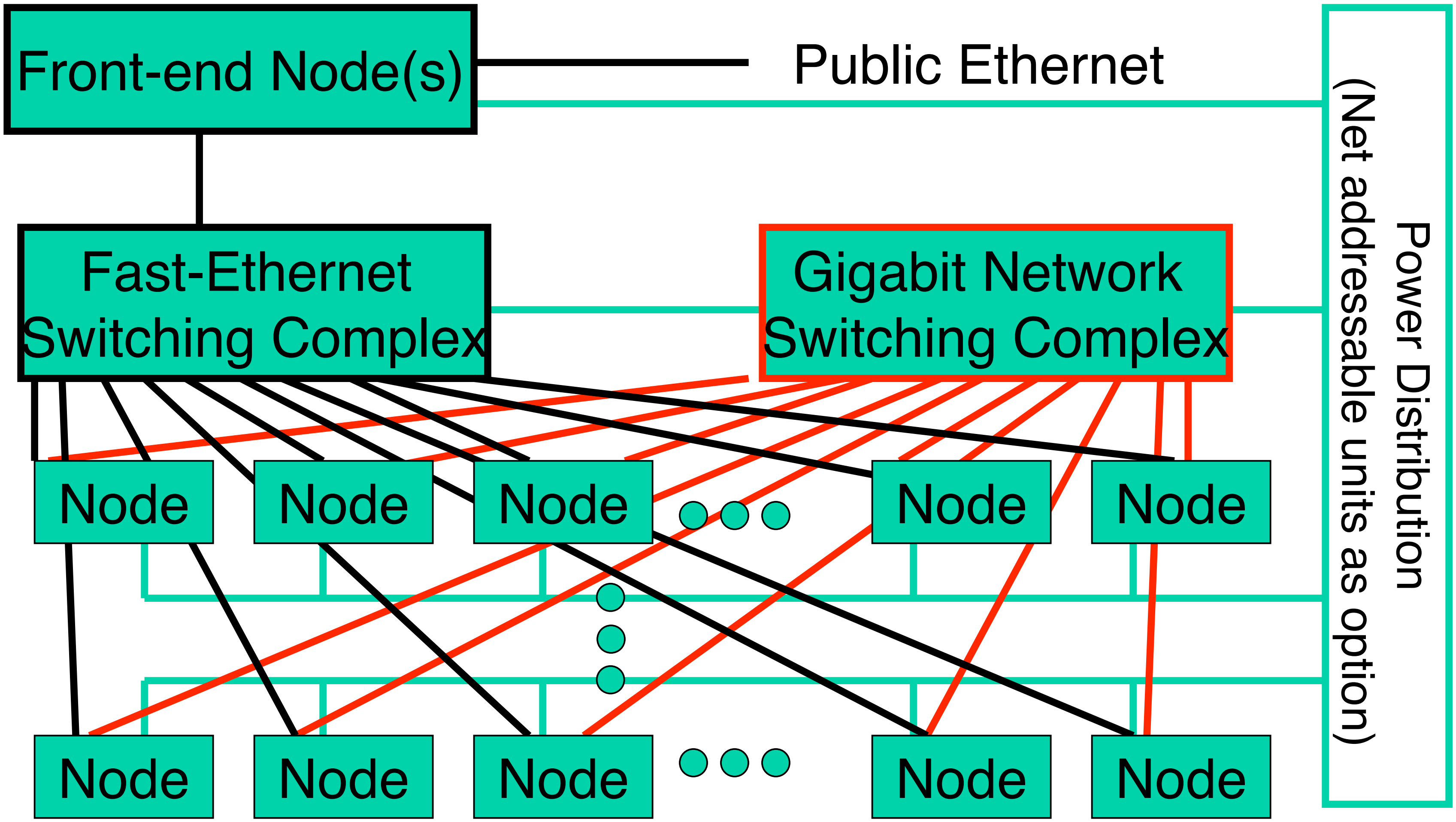
Outline

- Common (mis)perceptions about clusters
- Descriptions vs. images
- The core of the Rocks toolkits
- Things that only developers and administrators really care about

Level-Setting – Clusters 101

- Focus is on computing clusters
  - Large number of nodes that need similar system software footprints
  - MPI-style parallelism is the dominant application model
    - Workstation farming also popular
  - Not assuming homogeneity of hardware configurations
    - Do assume the same OS
    - Even "homogeneous" systems exhibit hardware differences
- General clustered endpoints should be "just as easy"
- Not high-availability clusters
  - Our techniques can help here, but we don't address the specific software needs of HA

Many variations on a basic layout



Myths

- Clusters are phenomenal price/performance computational engines, but are hard to install and manage
- Cluster management is a full-time job which gets linearly harder as one scales out.
- “Heterogeneous” Nodes are a bummer (network, memory, disk, MHz, current kernel version, PXE, CDs, Video).
- Clusters only have two types of nodes – compute and login

Beauty and the Beast

- SCSI, IDE, Integrated Raid, 100Mbit, 1000Mbit, Myrinet
- All compute nodes have the same basic configuration
- Appliances: NFS servers (dedicated), Login nodes, compute nodes, Monitoring nodes

Comprised of “appliances”

- Need to differentiate nodes on their functionality
  - Login/Compile
  - Compute
  - File server and/or Web Server
  - Grid Services (many subtypes)
  - Database Engine
- Supposition: if two nodes are the same appliance type – small differences in hardware should be *automatically* handled
- Observation – a very large percentage of the basic software configuration is common among appliances

Conventional Wisdom on Cluster install

- Head node installed by hand
- An image is built for compute nodes
  - Assumption: compute nodes are homogeneous
  - "Golden image" methods require a cluster-savvy admin to create the model node
    - Ghost, DriveImage, ImageCast, dd, …
  - Others allow an image to be created/modified using custom software
    - SystemImager, PowerCockpit, CLIC Imager, Chiba City Imager, …
- Once installed, nodes are actively managed
  - Scripts, Parallel Shells, cfEngine
  - Golden image often gets out of sync with what is actually running a newly installed node != running node

What Image-based methods Imply

- Different appliances have different images
- Substantially different hardware: each has a different image
  - E.g. SCSI vs. IDE vs. IDA (HW RAID)
- Clusters become the cross product of images: *Appliances X HW types*
- Specialized installer needed to put images on drives (from the network)
  - If image-installer (like SIS) handles some hardware differences, it must detect them
    - Commercial distros already do this for their installers (Why re-invent?)

Description-Based Methods

- Text description of everything about a node
  - Partitioning, boot loader, packages, configuration
  - Kickstart (RedHat), YAST2 (SuSE), FAI (Debian), Jumpstart (Solaris)
    - Vendors already have extensive HW detection to find modules for Disks, NICS, Video, I/O ports, Motherboard Chipsets …
- Leverage the extensive investment in HW detection so that a cluster == #Appliances
- Rocks easily expresses commonality among appliances
  - Manage the base functionality in one place
  - Worry only about <u>differences among</u> appliances
  - Automate many places to make the installation fast and scalable

NPACI Rocks Toolkit – rocks.npaci.edu

- Techniques *and* software for easy installation, management, monitoring and update of Linux clusters
  - Cluster-aware distribution and configuration system
  - IA32 and IA64
- Installation
  - Bootable CD which contains all the packages and site configuration facilities to bring up an entire cluster
- Management and update philosophies
  - Trivial to completely reinstall any (all) nodes.
  - Nodes are 100% automatically configured
  - RedHat Kickstart to define software/configuration of nodes
  - Software is packaged in a query-enabled format
  - <u>Never</u> try to figure out if node software is consistent
  - Extensible, programmable infrastructure for all node types

Tools Integrated

- Standard cluster tools
  - MPICH, PVM, PBS, Maui (SSH, SSL -> Red Hat)
- Rocks add ons
  - Complete Myrinet support
  - Rocks-dist – distribution work horse
  - XML (programmable) Kickstart
  - eKV (console redirect to ethernet during install)
  - Automated mySQL database setup
  - Ganglia Monitoring (U.C. Berkeley and NPACI)
  - Stupid pet administration scripts
- Other tools
  - PVFS, Sun Grid Engine
  - ATLAS BLAS, High Performance Linpack
  - IOZone, Streams Benchmarks
  - MPD parallel launcher coming soon.

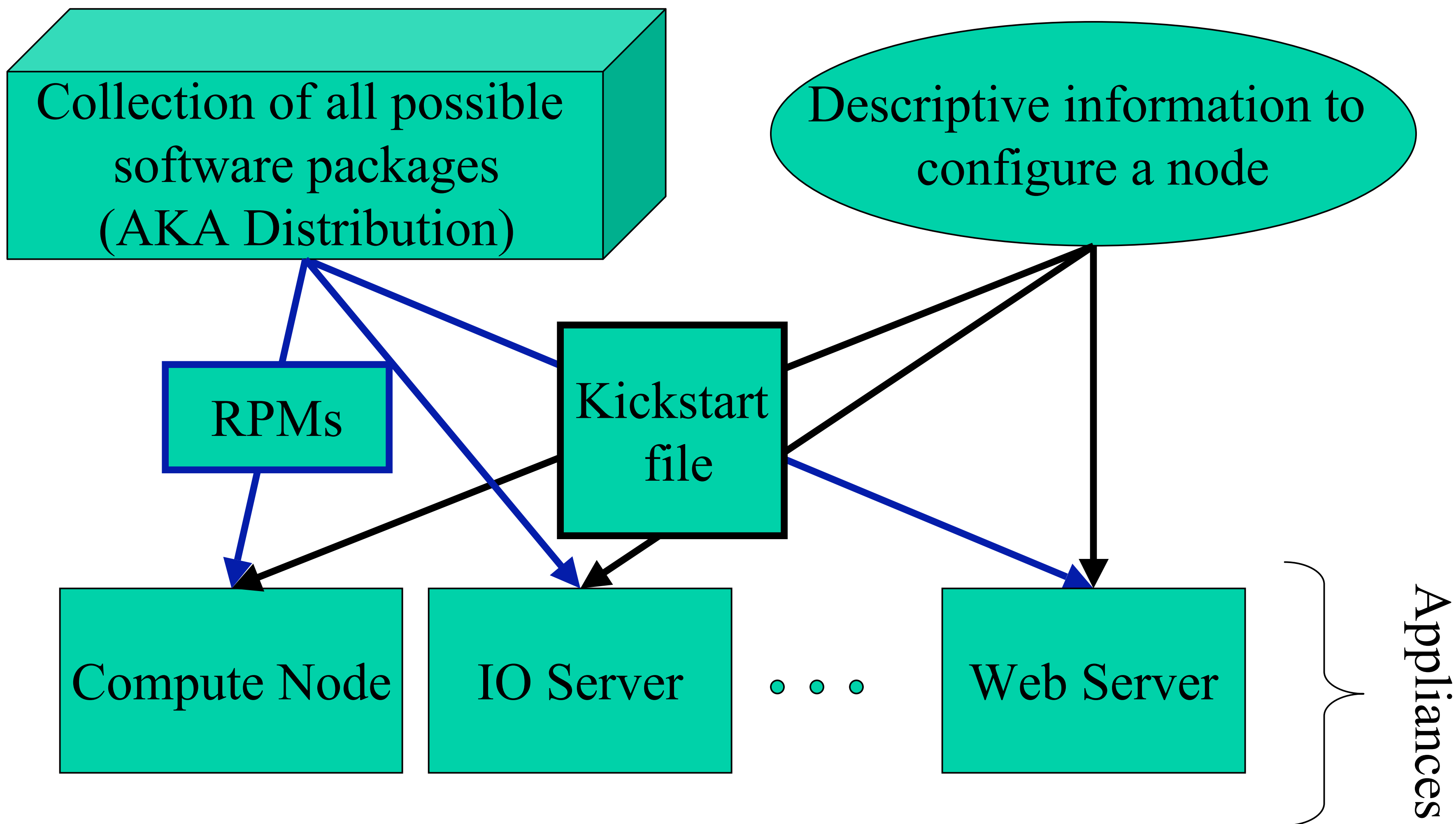
Key Ideas

- OS installation is completely disposable
  - Non-root partitions saved across reinstalls
- Software bits (packages) are separated from configuration
  - Diametrically opposite from “golden image” methods
- Description-based configuration rather than image-based
  - Installed OS is “compiled” from a graph.
- Inheritance of software configurations
  - Distribution (as in RedHat)
  - Configuration (as in appliances)
- Single step installation of updated software OS
  - Security patches pre-applied to the distribution not post-applied on the node

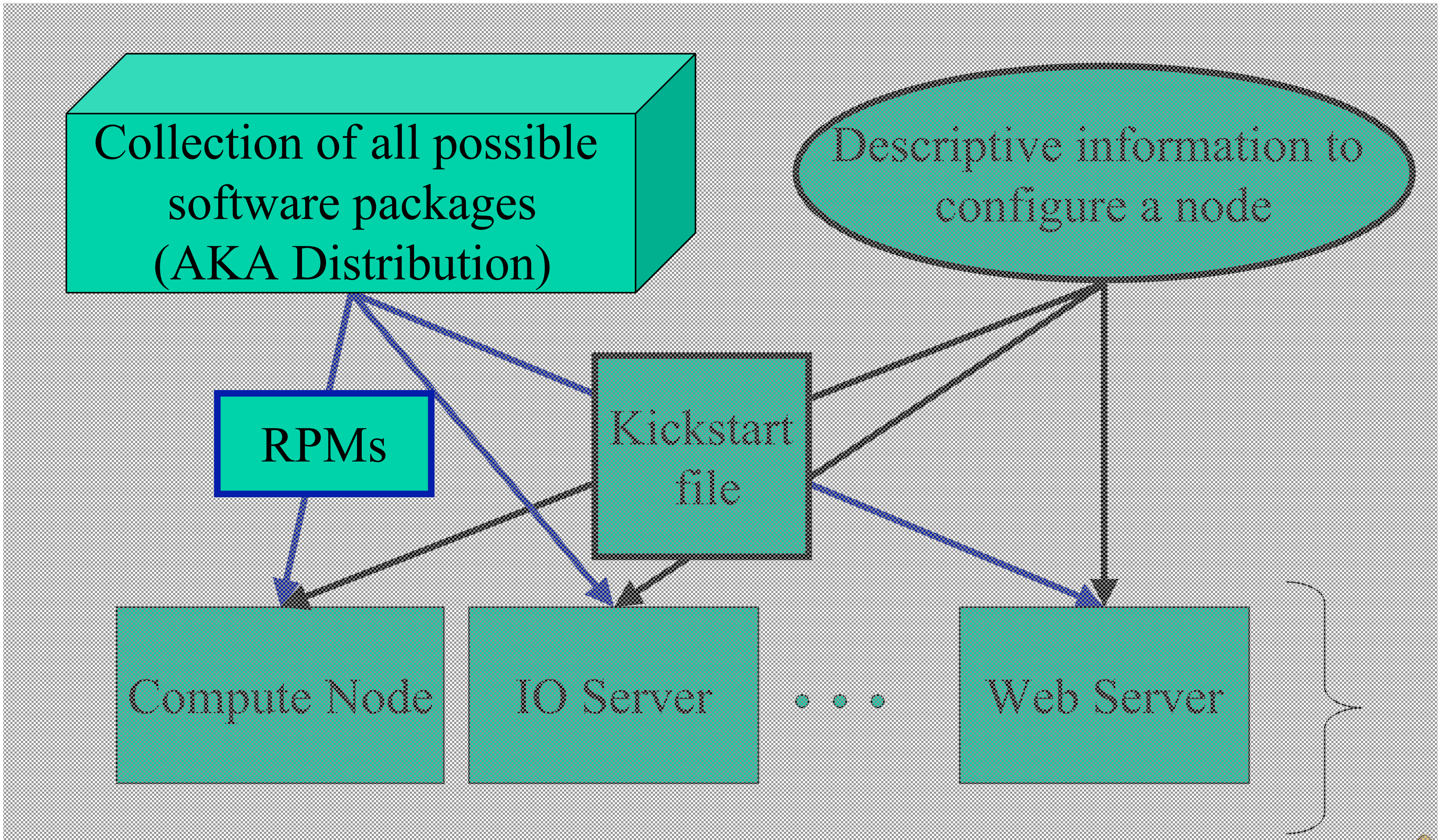
Rocks extends installation as a basic way to manage software on a cluster

It becomes trivial to insure software consistency across a cluster

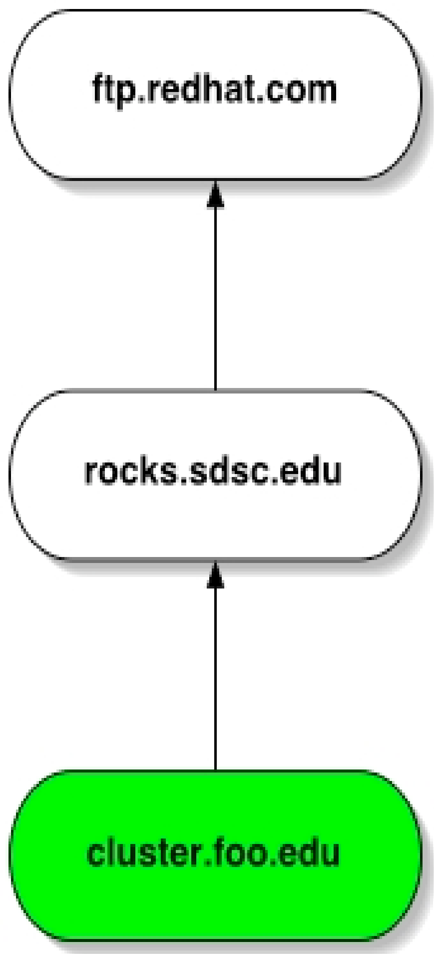
Disentangle Software Bits (distributions) and Configuration



Managing Software Distributions



Rocks-dist Repeatable process for creation of localized distributions



- # rocks-dist mirror
  - Rocks mirror
    - Rocks 2.3 release
    - Rocks 2.3 updates
- # rocks-dist dist
  - Create distribution
    - Rocks 2.3 release
    - Rocks 2.3 updates
    - **Local software**
    - Contributed software
- This is the same procedure NPACI Rocks uses.
  - Organizations can customize Rocks for their site.
- Iterate, extend as needed

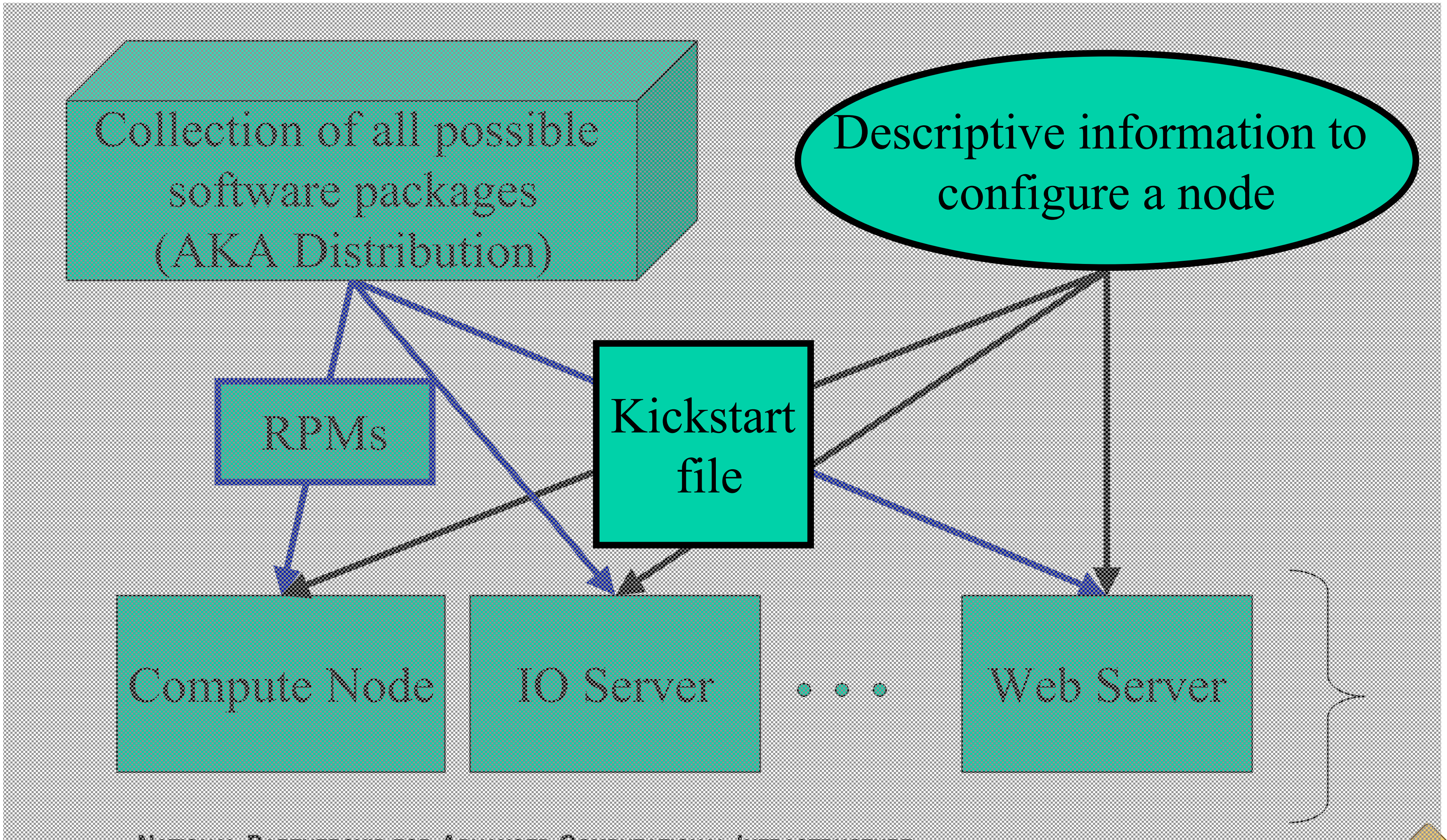
Rocks-dist

- Distribution creation tool
  - Integrate custom packages as if native to distro
    - Updates applied to the distribution – not post-install
  - Single command to build a bootable CD set
- Allows multiple distributions to reside on server
  - Development vs. Production
  - IA32 and IA64
- Simplifies mirroring (and support multiple mirrors)

Rocks-dist (more)

- RedHat has several levels of embedded "databases"
  - Originally built rocks-dist to just ingest updated rpms to eliminate the install-then-patch cycle
  - Determining "best" RPM from lots of versions is sometimes dicey
    - Contrib directories, mirrors, /usr/src/redhat/RPMS …
- It does more
  - Patches the redhat installer so the filename part of the DHCP record can be a URL
  - Patches "loader" for a variety of robustness issues (including more aggressive DHCP retries and watchdogs)
  - Single command mirroring (supports integration from multiple mirrors)
  - Single command cd set and dvd rom creation for building completely customized distributions
  - Allows multiple distributions to be built on a single server

Description-based Configuration



What is a Kickstart file?

Setup/Packages (20%)

```
cdrom
zerombr yes
bootloader --location mbr --useLilo
skipx
auth --useshadow --enablemd5
clearpart
part /boo
part swap
part / --size 4096
part /export --size 1 --grow
lang en_US
langsupport --default en_US
keyboard us
mouse genericps/2
timezone --utc GMT
rootpw --iscrypted nrDG4Vb8OjjQ.
text
install
reboot

%packages
@Base
@Emacs
@GNOME
```

Package Configuration (80%)

```
%post

cat > /etc/nsswitch.conf << 'EOF'
passwd:     files
shadow:     files
ethers:     files
EOF

cat > /etc/ntp.conf << 'EOF'
server ntp.ucsd.edu
server   127.127.1.1
fudge    127.127.1.1 stratum 10
authenticate no
driftfile /etc/ntp/drift
EOF

/bin/mkdir -p /etc/ntp
cat > /etc/ntp/step-tickers << 'EOF'
ntp.ucsd.edu
EOF

/usr/sbin/ntpdate ntp.ucsd.edu
/sbin/hwclock --systohc
```

Portable (ASCII), Not Programmable, O(30KB)

What are the Issues

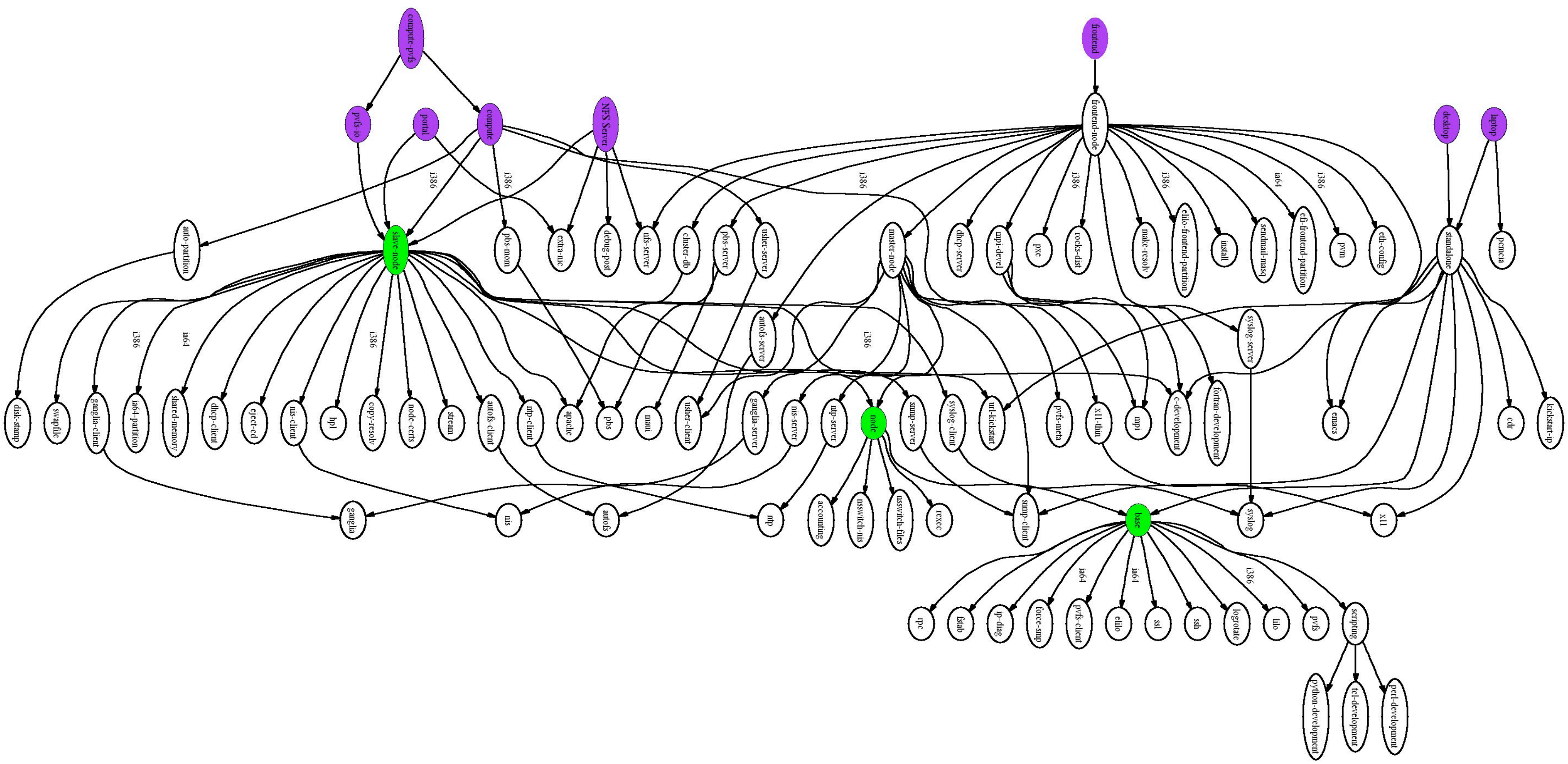
- Kickstart file is ASCII
  - There is some structure
    - Pre-configuration
    - Package list
    - Post-configuration
- Not a “programmable” format
  - Most complicated section is post-configuration
    - Usually this is handcrafted
  - Really Want to be able to build sections of the kickstart file from pieces
    - Straightforward extension to new software, different OS

Focus on the notion of "appliances"

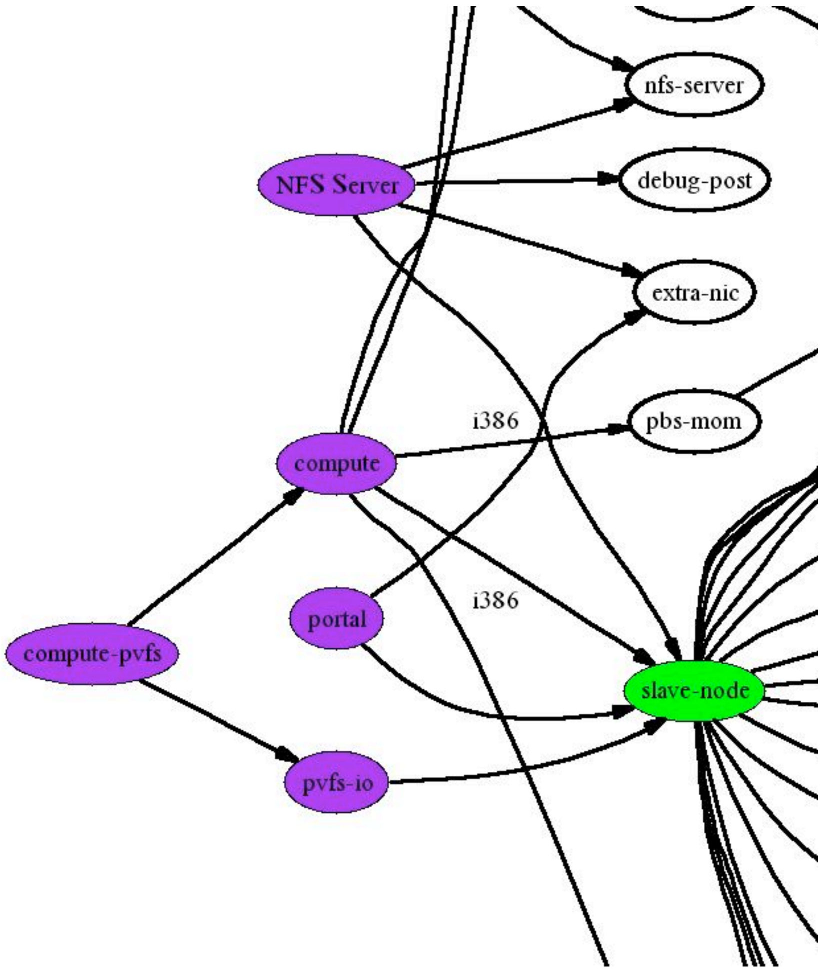
How do you define the configuration of
nodes with special attributes/capabilities

Assembly Graph of a Complete Cluster

- - "Complete" Appliances (compute, NFS, frontend, desktop, …)
- - Some key shared configuration nodes (slave-node, node, base)



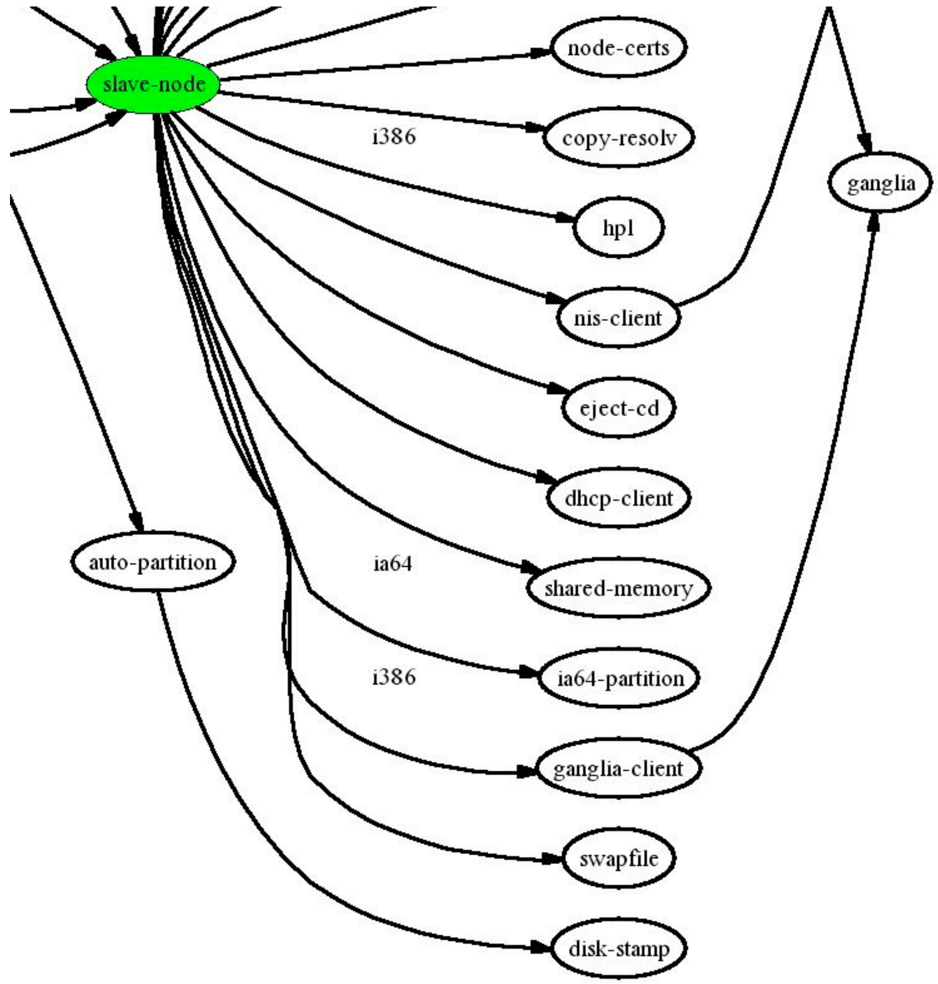
Describing Appliances



- Purple appliances all include “slave-node”
  - Or derived from slave-node
- Small differences are readily apparent
  - Portal, NFS has “extra-nic”. Compute does not
  - Compute runs “pbs-mom”, NFS, Portal do not
- Can compose some appliances
  - Compute-pvfs IsA compute and IsA pvfs-io

Architecture Dependencies

- Focus only on the differences in architectures
    - logically, IA-64 compute node is identical to IA-32
- Architecture type is passed from the top of graph
- Software bits (x86 vs. IA64) are managed in the distribution



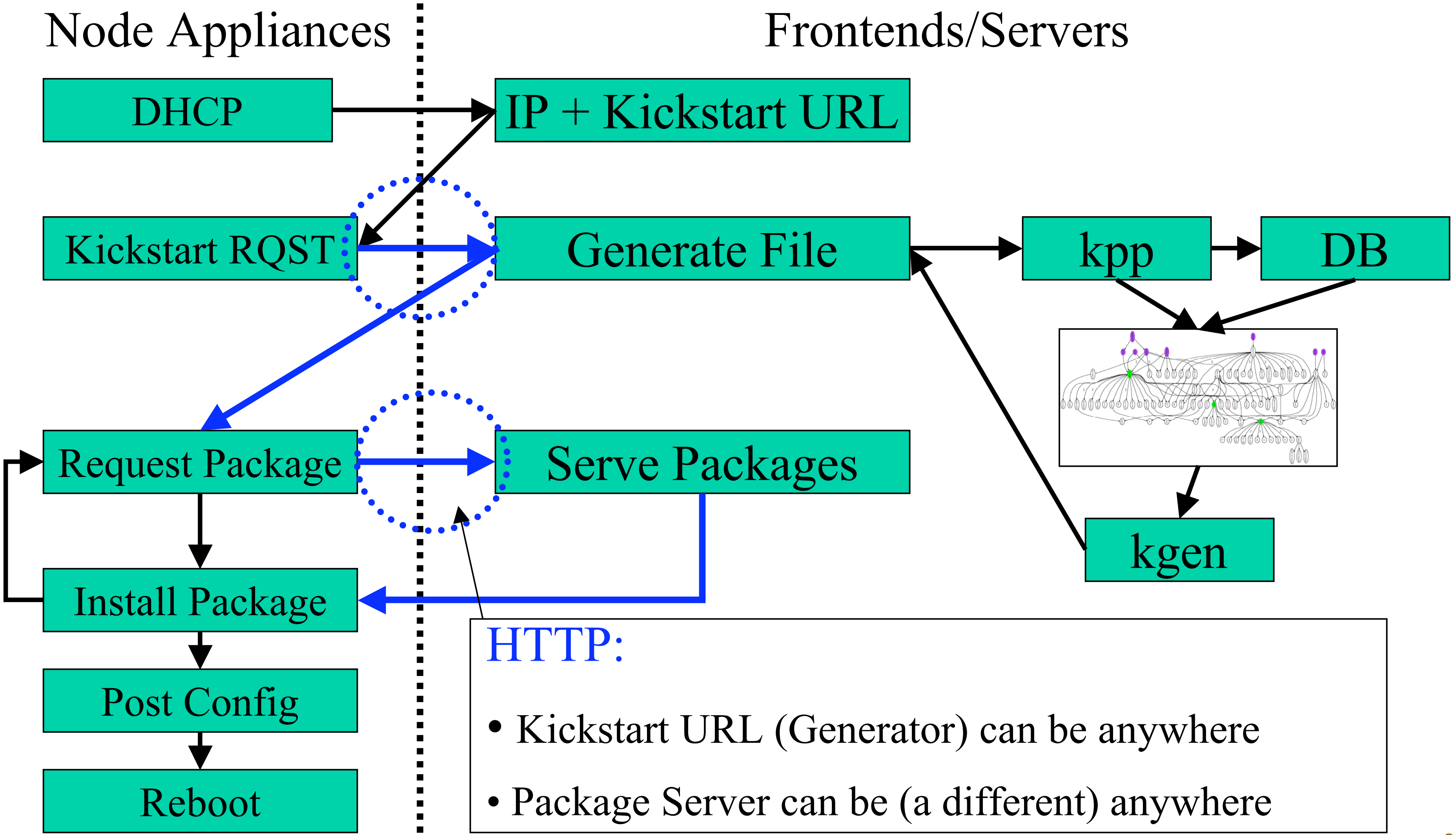
XML Used to Describe Modules

- Abstract Package Names, versions, architecture
  - ssh-client

    Not
  - ssh-client-2.1.5.i386.rpm
- Allow an administrator to encapsulate a logical subsystem
- Node-specific configuration is retrieved from our database
  - IP Address
  - Firewall policies
  - Remote access policies
  - …

```
<?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>
  <!-- include XFree86 packages for xauth -->
  <package>XFree86</package>
  <package>XFree86-libs</package>
<post>
cat &gt; /etc/ssh/ssh_config &lt;&lt; 'EOF' <!--
    default client setup -->
Host *
        CheckHostIP              no
        ForwardX11               yes
        ForwardAgent             yes
        StrictHostKeyChecking    no
        UsePrivilegedPort        no
        FallBackToRsh            no
        Protocol                 1,2
EOF </post> </kickstart>
```

Space-Time and HTTP



The Rocks Database

- Straightforward MySQL database holds cluster-wide information
  - Zero administration of the database (created automatically on frontend build)
  - MAC addresses automatically collected and appliance type assigned
  - Used to create node-specific kickstart files on-the-fly
- Node Memberships
  - (appliance type, distributions)
- Dbreport
  - Create config files from database
  - Insures information consistency
  - PBS, SGE, dhcpd.conf,/etc/hosts, …
  - Extensible ala' xinetd directory structure

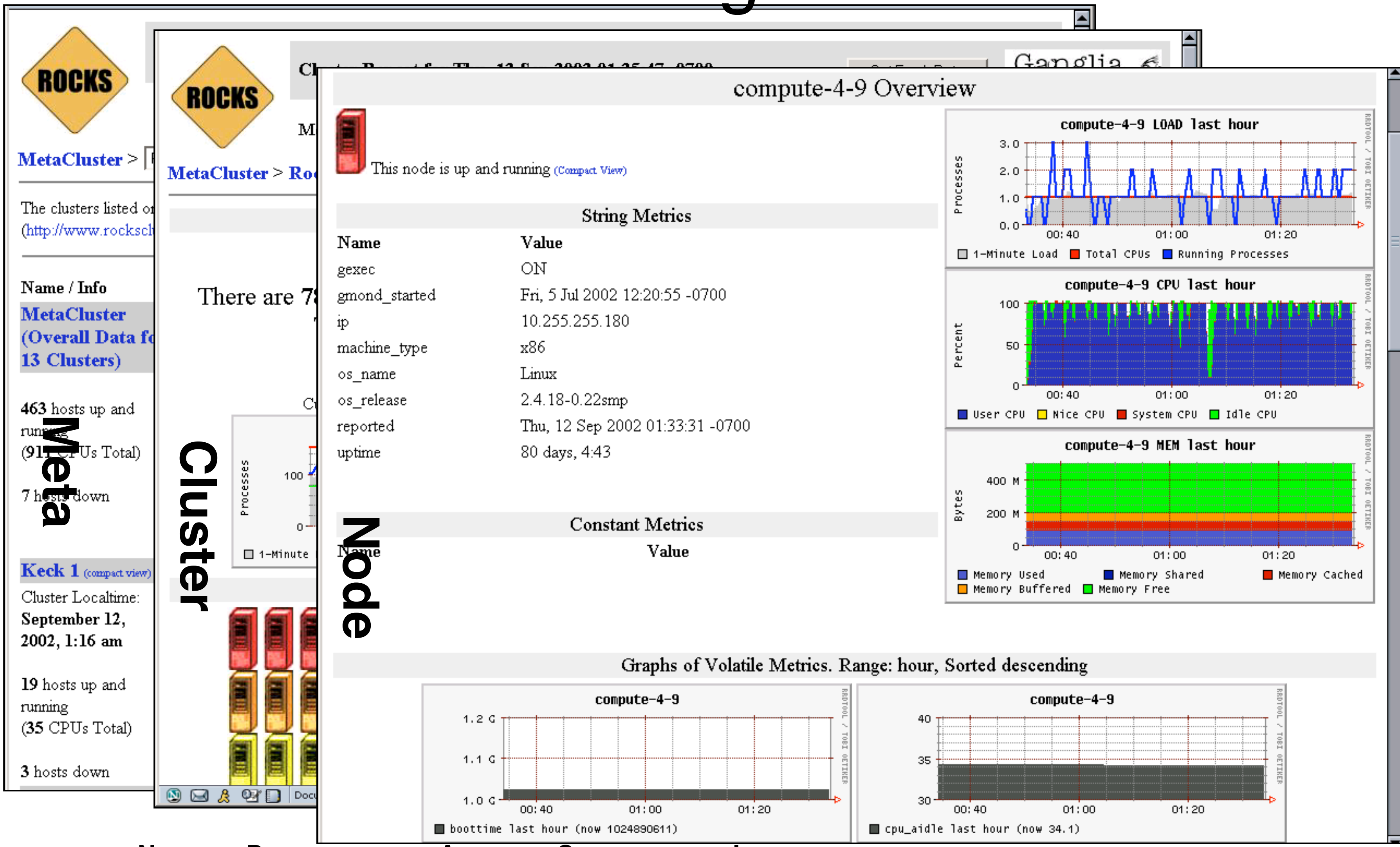
Subsystem Replacement is Easy

- Binaries are in *de facto* standard package format (RPM)
- XML module files (components) are very simple
- Graph interconnection (global assembly instructions) is separate from configuration
- Examples
  - Replace PBS with Sun Grid Engine
  - Upgrade version of OpenSSH or GCC
  - Turn on RSH (not recommended)
  - Purchase commercial compiler (recommended)
  - Add Grid Stack (In progress!)

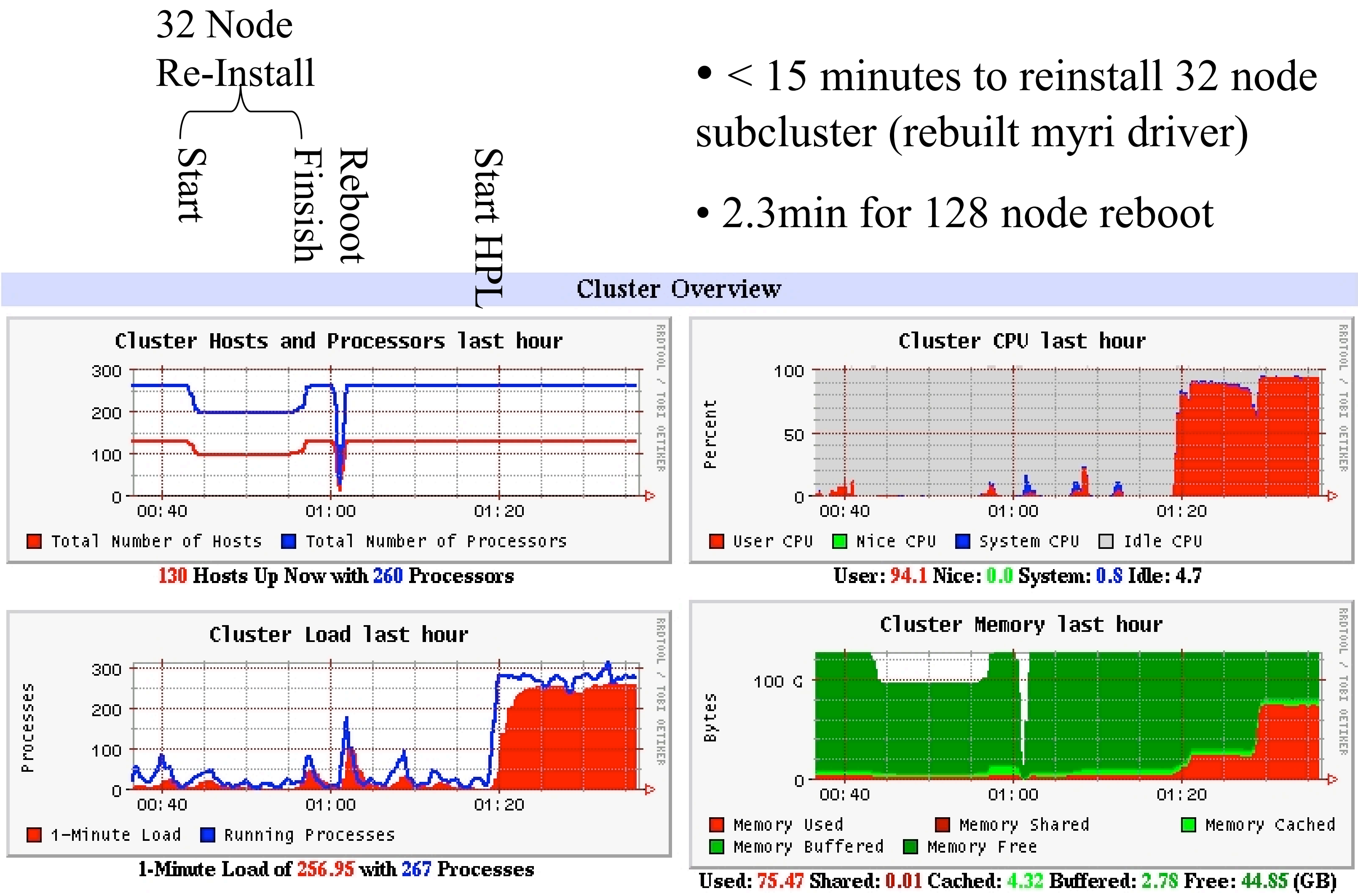
Reset

- 100s of clusters have been built with Rocks on a wide variety of physical hardware
  - Largest is 3+TF, 300 Nodes at Stanford
  - Pentium 2, Pentium III, Pentium 4, XEON, Athlon, Itanium 1 and Itanium 2
- Installation/Customization is done in a straightforward programmatic way
  - Scaling is excellent
- HTTP is used as a transport for reliability/performance
  - Configuration Server does not have to be in the cluster
  - Package Server does not have to be in the cluster

Meta Cluster Monitor Built on Ganglia



Installation, Reboot, Performance



- < 15 minutes to reinstall 32 node subcluster (rebuilt myri driver)
- 2.3min for 128 node reboot

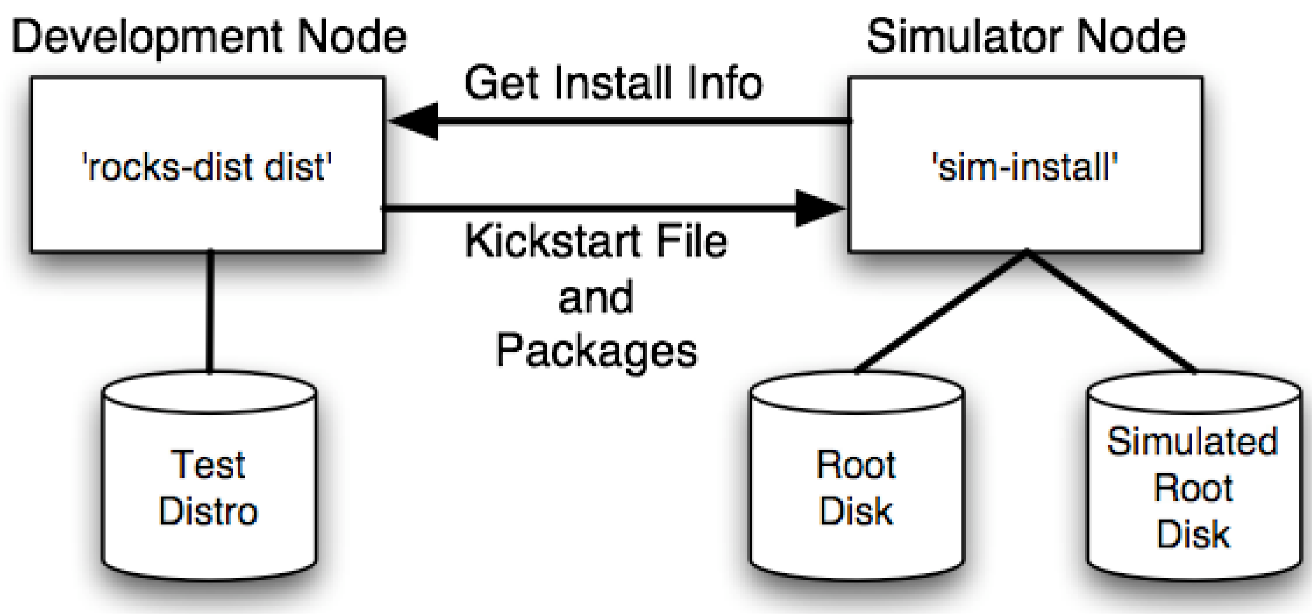
Things for admins or developers

- Handling devices that are not part of the kernel package (eg. GM and PVFS devices)
  - On rebuild, drop a source RPM in a specific directory. Automatically rebuilt after reinstall
- Need to interact with installer for testing, but don't have a KVM
  - eKV is a telnet "wedge" and is patched into the Anaconda installer so that all we need is ethernet and power
- Want to build custom distributions
  - Rocks-dist cdrom
- Want to force a specific package (kernel) version
  - Drop into a "force" directory
- Need to add new functionality
  - RPM + XML file description
  - Directory structure: drop in additional nodes and edges

Updates after the cluster is installed?

- Nothing prevents folks from doing cfEngine-style management
- Designing a way to traverse the graph to create something other than kickstart (perhaps cfEngine Instructions)
- Reinstallation is fast and surprisingly efficient
  - We've reinstalled 256 nodes in 40 Minutes
- For non-administrators, re-installation provides the "fire and forget" mechanism to get software consistency

Installation Simulator



- Develop new features on ‘development node’
- Test on standard compute node
  - Simulator installs complete operating environment on simulated root disk

Installation Simulator Benefits

- Syntax errors now cost considerably less
- Installation procedure can be examined and controlled by tools external to the installation process
  - Encourages experimentation
- The simulator can be adapted to build installation trees for diskless clusters
  - Simulator can install an “image” onto a frontend’s NFS mounted image directory

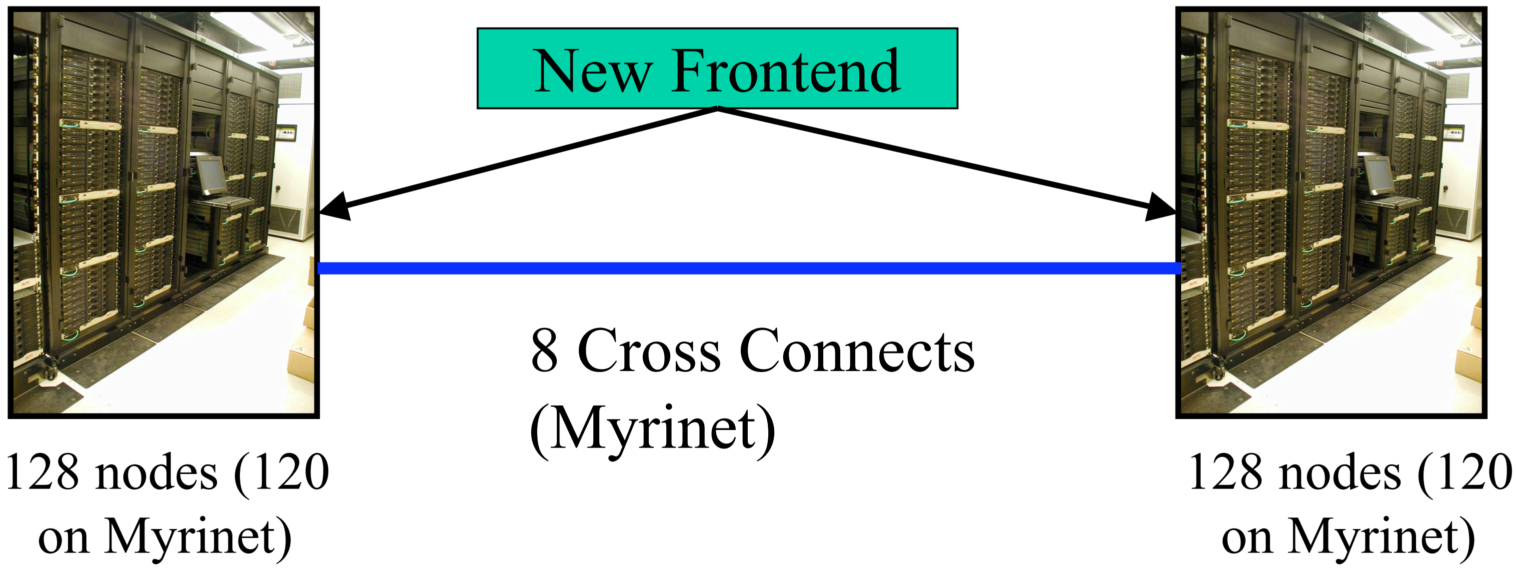
Summary

- Clusters aren't homogeneous in software configuration
- They may start out homogeneous in hardware but quickly diverge as equipment is added/replaced
- Description mechanisms + Distro's HW detection supports an extremely broad range of platforms
- IA32 and IA64
- Open Source (of course!). CVS tree available

Web Places

- http://rocks.npaci.edu
- http://rocks.npaci.edu/rocks-register
- http://Ganglia.sourceforge.net
- http://meta.rocksclusters.org

Setup



- Fri: Started 5:30pm. Built new frontend. Physical rewiring of myrinet, added ethernet switch.
- Fri: Midnight. Solved some ethernet issues. Completely reinstalled all nodes.
- Sat: 12:30a Went to sleep.
- Sat: 6:30a. Woke up. Submitted first LINPACK runs (225 nodes)

Support for Myrinet

- Myrinet device driver must be versioned to the exact kernel version (eg. SMP,options) running on a node
  - Source is compiled at reinstallation on every (Myrinet) node (adds 2 minutes installation) (a source RPM, by the way)
  - Device module is then installed (insmod).
  - GM_mapper run (add node to the network)
- Myrinet ports are limited and must be identified with a particular rank in a parallel program
  - RPC-based reservation system for Myrinet ports
  - Client requests port reservation from desired nodes
  - Rank mapping file (gm.conf) created on-the-fly
  - <u>No centralized</u> service needed to track port allocation
  - MPI-launch hides all the details of this
- HPL (LINPACK) comes pre-packaged for Myrinet
- Build your Rocks cluster, see where it sits on the Top500