User Session 3

Rocks-A-Palooza III

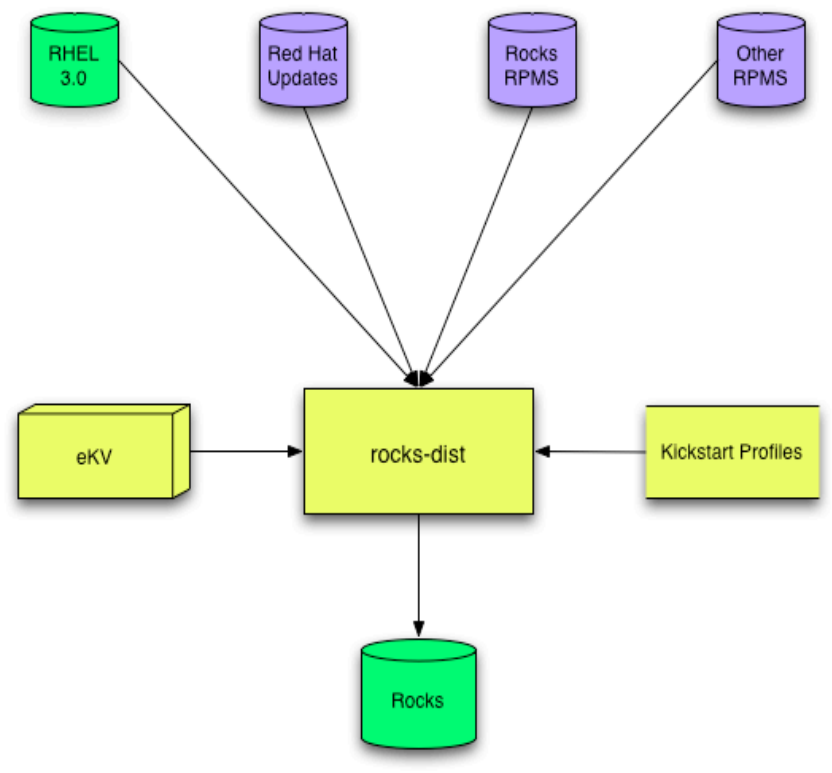
SDSC

Building on Top of Rocks

Inheritance and Rolls

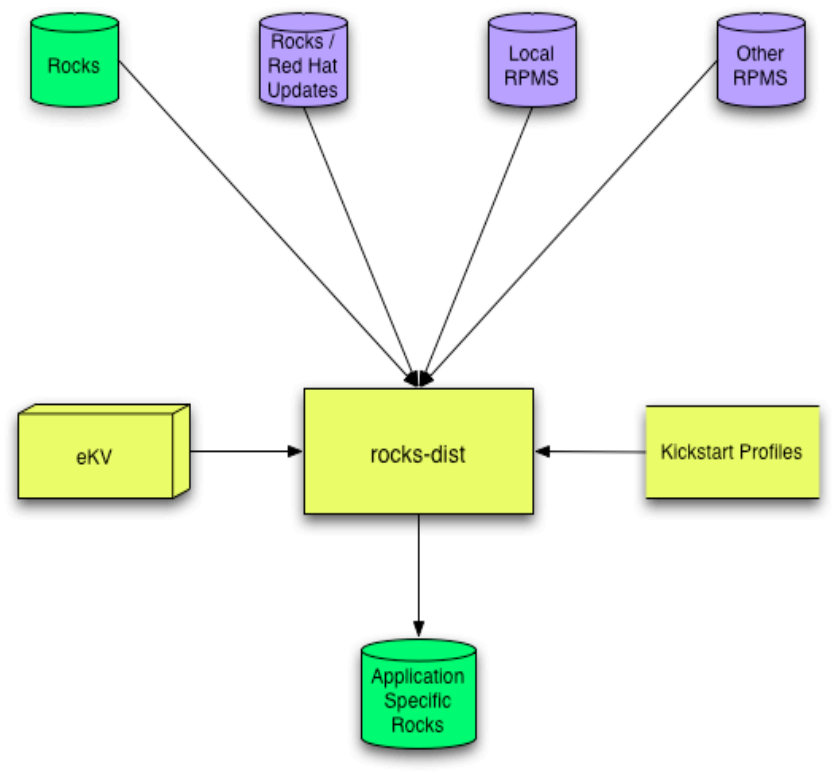
How Rocks in built

- Rocks-dist
  - Merges all RPMs
    - Red Hat
    - Rocks
  - Resolves versions
  - Creates Rocks
- Rocks distribution
  - Looks just like Red Hat
  - Cluster optimized Red Hat



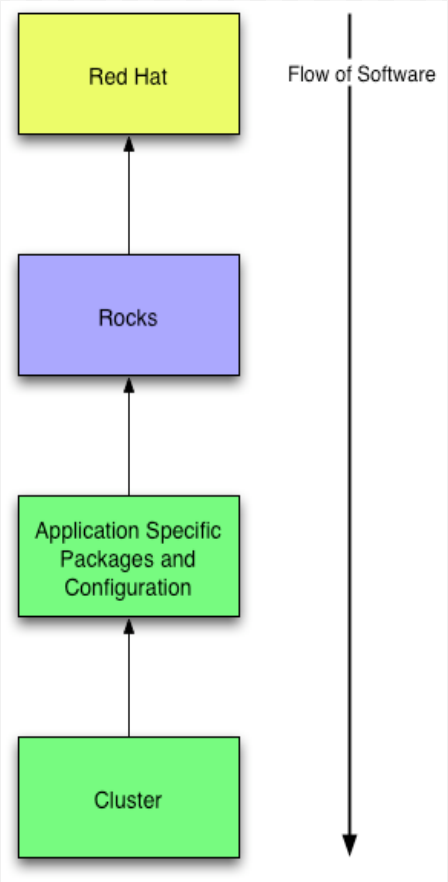
How You Create Your Own Rocks

- Rocks-dist
  - Merges all RPMs
    - Rocks
    - Yours
  - Resolves versions
  - Creates Rocks++
- Your distribution
  - Looks just like Rocks
  - Application optimized Rocks



Extension Through Inheritance

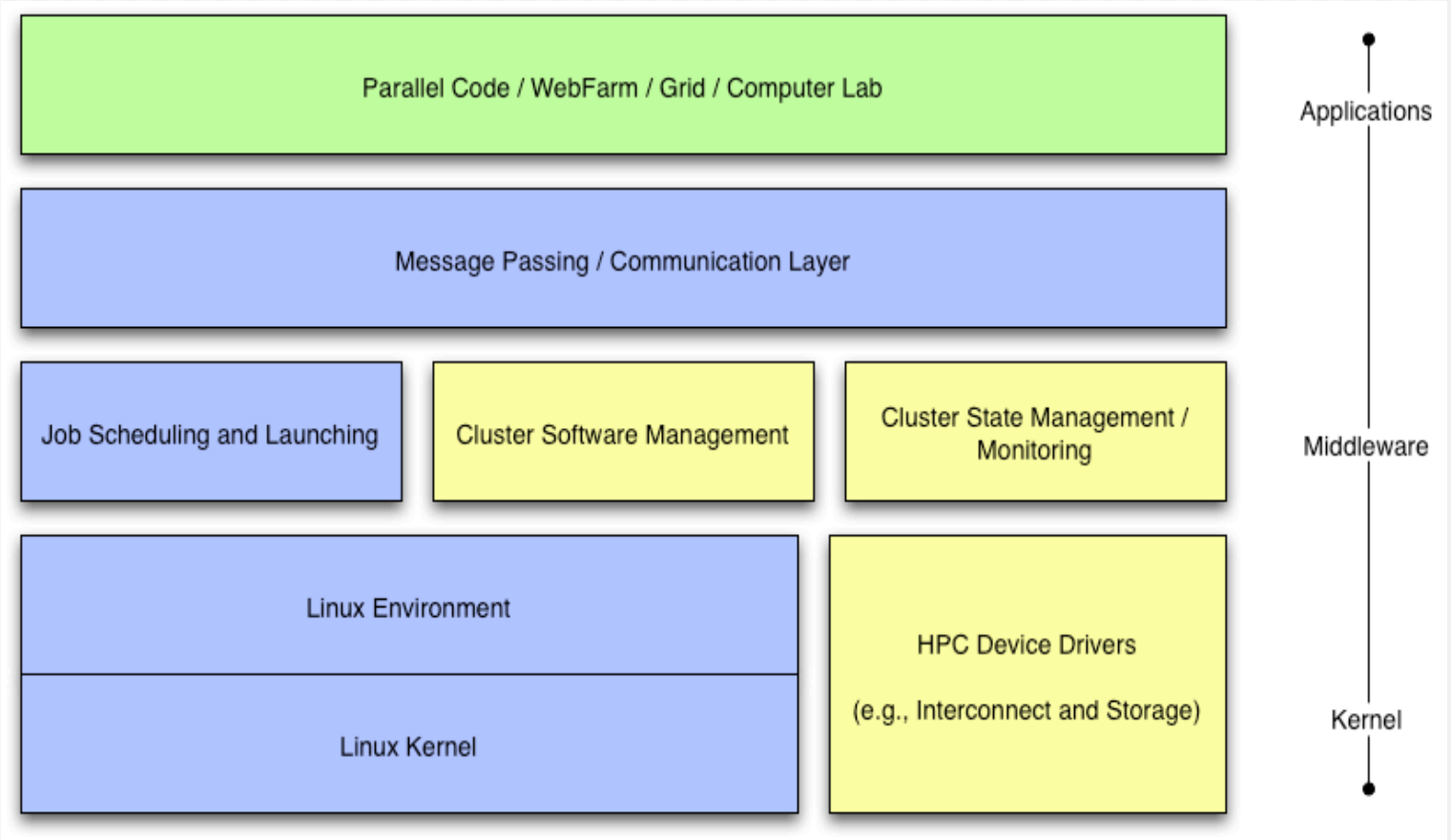
- UCSD/SDSC Rocks
  - BIRN
  - GAMESS Portal
  - GEON
  - GriPhyN
  - Camera
  - Optiputer
- Commercial
  - Other stacks “based” on Rocks
- Can also override existing functionality
  - Rocks without NFS?
  - Rocks for the desktop?



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

Rolls Break Apart Rocks



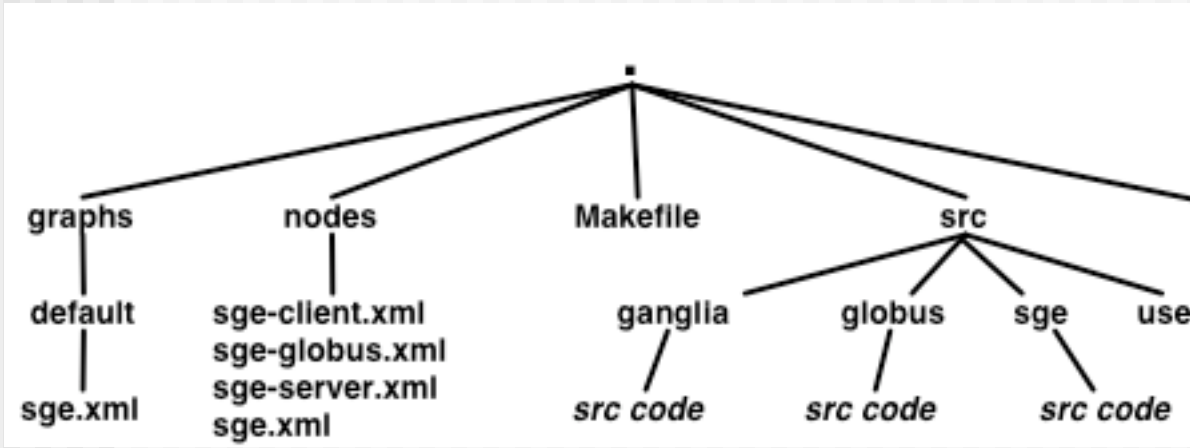
Rolls: Modifying a Standard System Installer to Support User-Customizable Cluster Frontend Appliances. Greg Bruno, Mason J. Katz, Federico D. Sacerdoti, and Phil M. Papadopoulos. *IEEE International Conference on Cluster Computing*, San Diego, California, Sep. 2004.

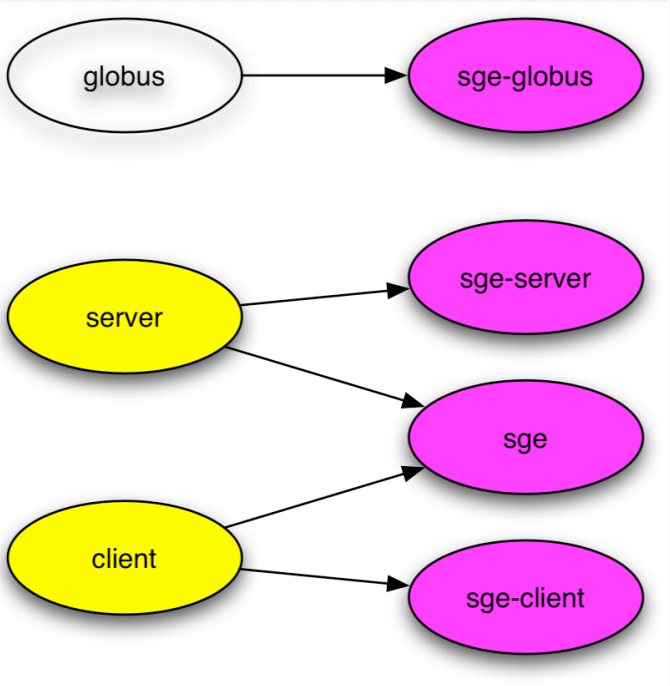
Rocks is What You Make it

- Motivation
  - "I'm concerned Rocks is becoming everything for everyone" - rocks mailing list
  - "Building a cluster should be like ordering a car. I want the sports package, but not the leather seats, …" - z4 owning rocks developer
  - We need to let go of Rocks but hold onto the core
    - Recruit more external open-source developers
    - Only trust ourselves with fundamental architecture and implementation
  - We wanted to move the SGE but need to still support PBS
- Rolls
  - Optional configuration and software
  - Just another CD for installed (think application pack)
  - SGE and PBS are different Rolls
    - User chooses scheduler
    - PBS Roll supported by Norway
    - SGE Roll supported by Singapore (and us)
  - Rolls give us more flexibility and less work
- Rocks is done
  - The core is basically stable and needs continued support
  - Rolls allow us to develop new ideas
  - Application Domain specific

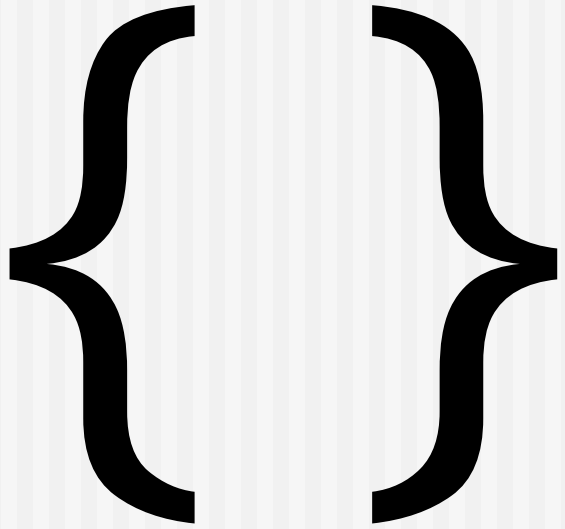
Rolls are sub-graphs

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

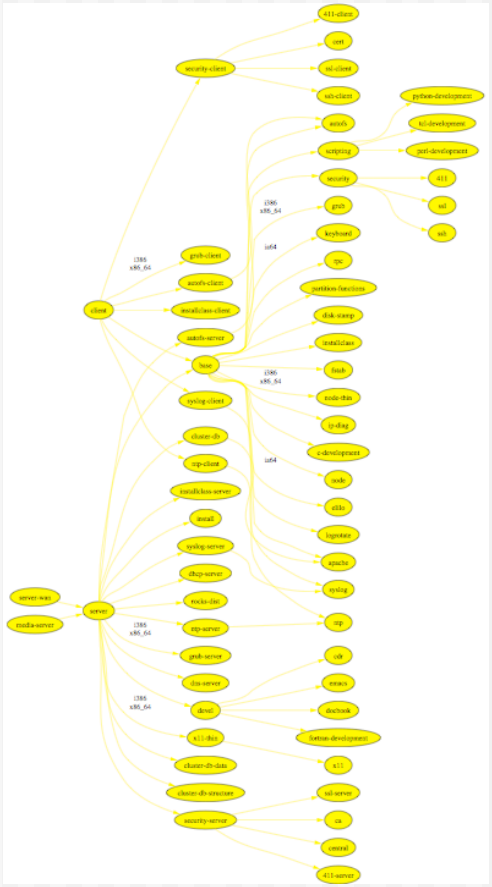




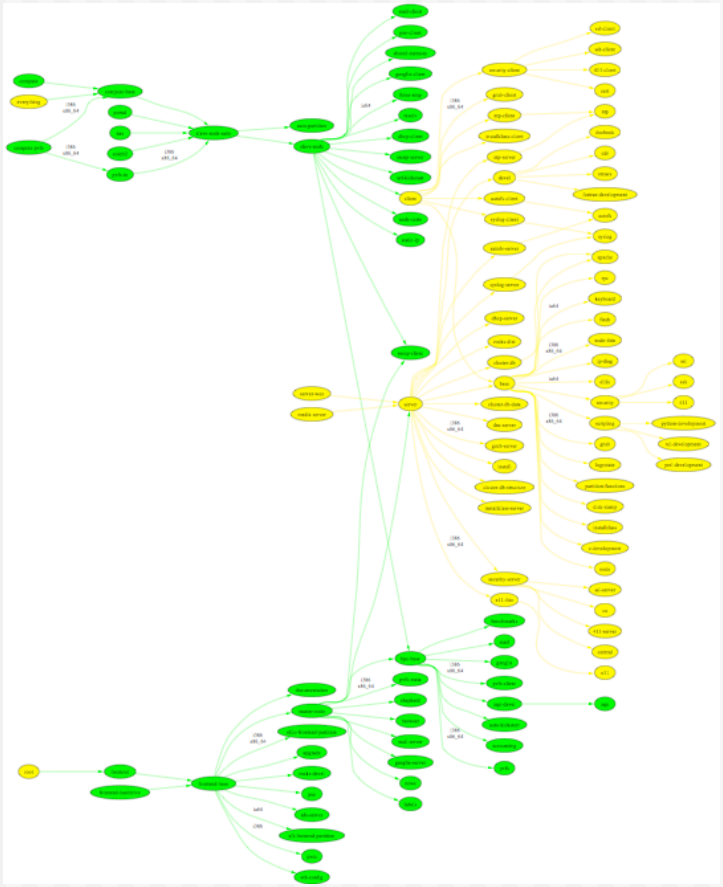
Starting from the empty set



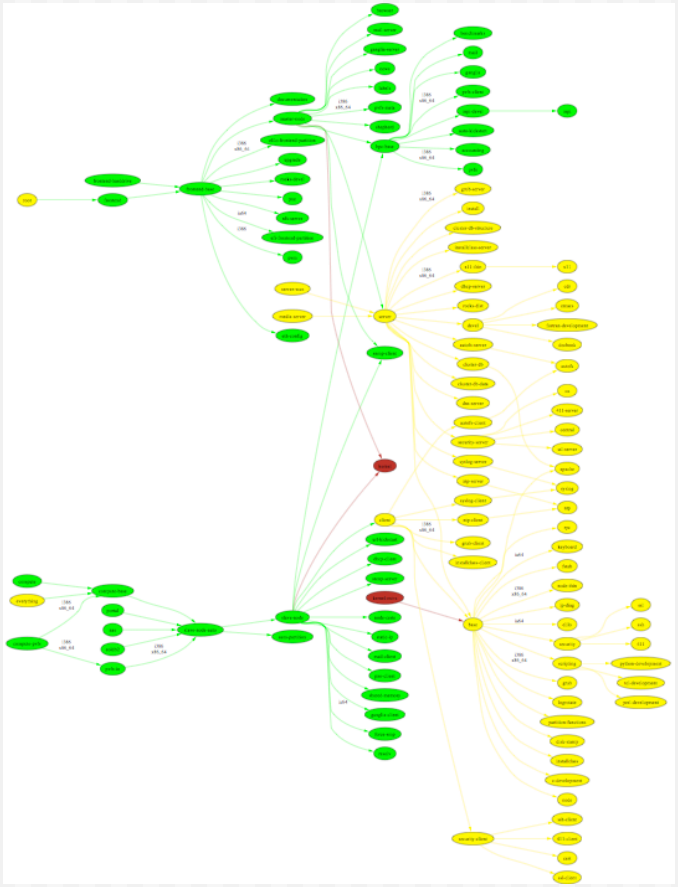
{ base }



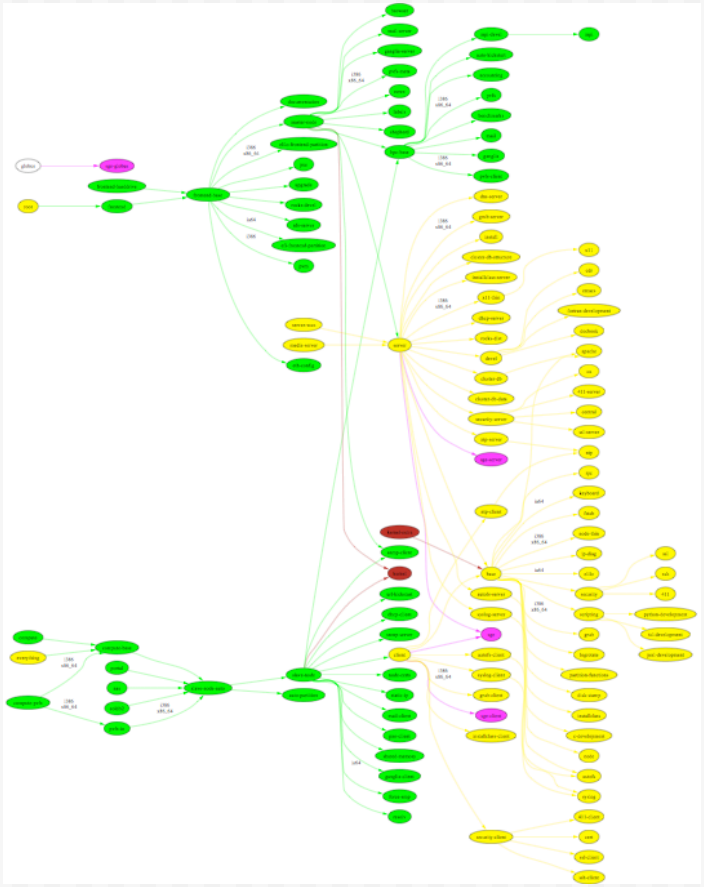
{ base, hpc }



{ base, hpc, kernel }

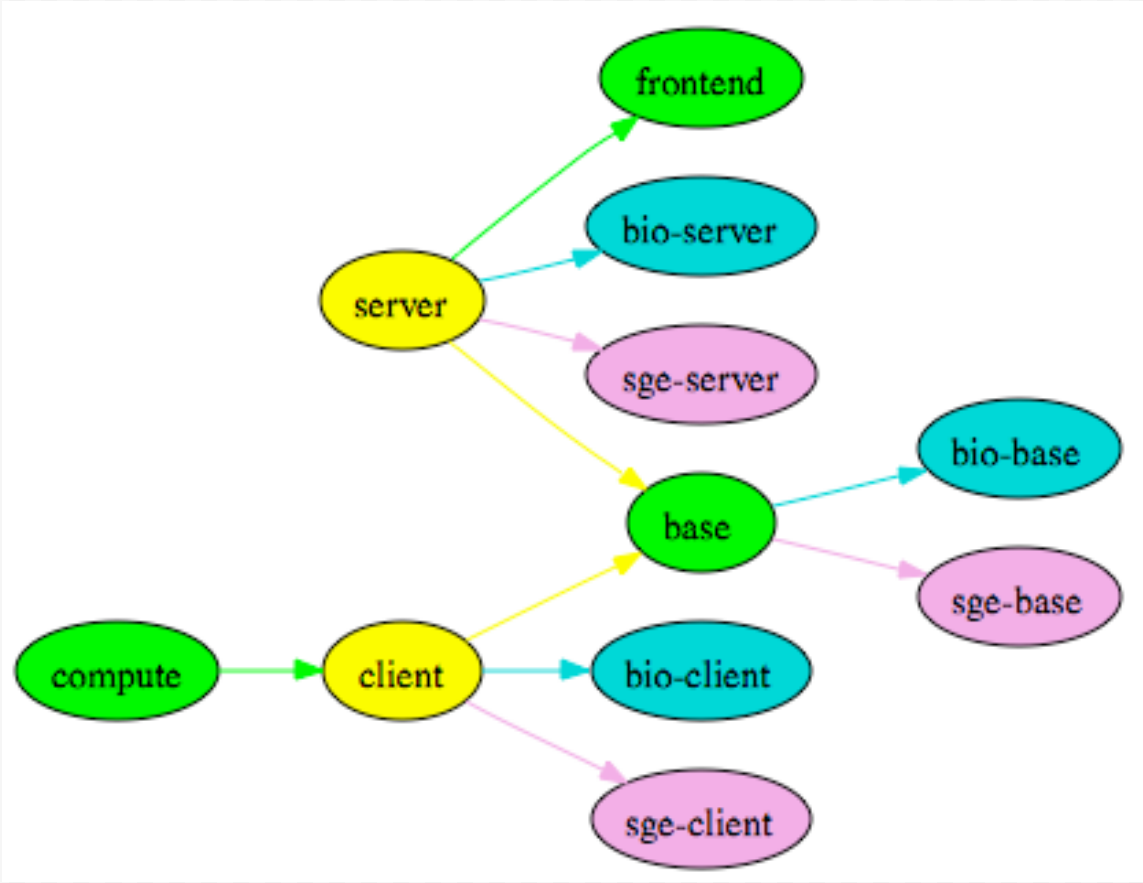


{ base, hpc, kernel, sge }



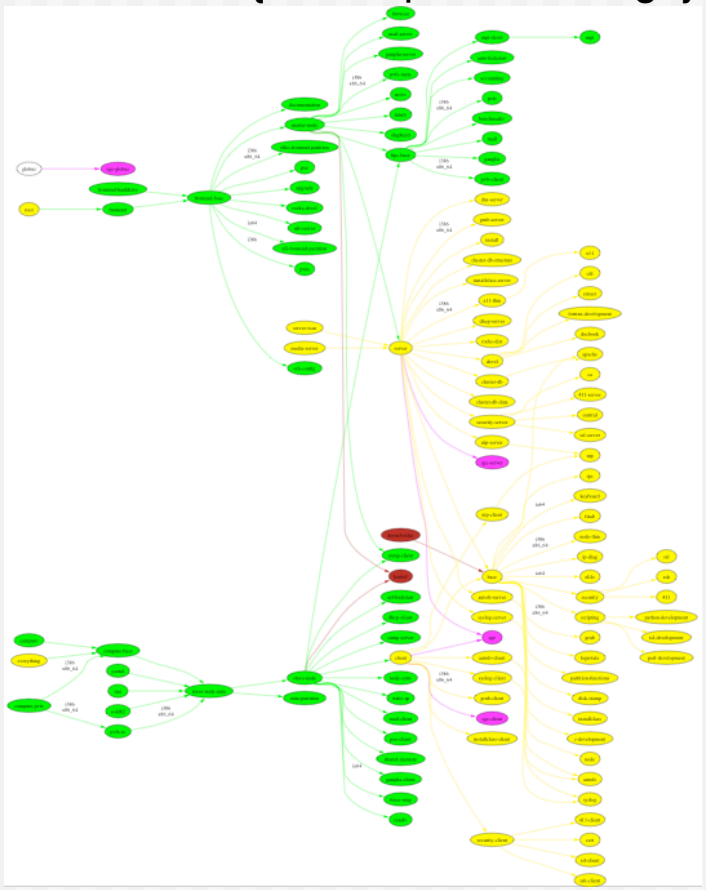
Simplified Example

{base, hpc, sge, bio}

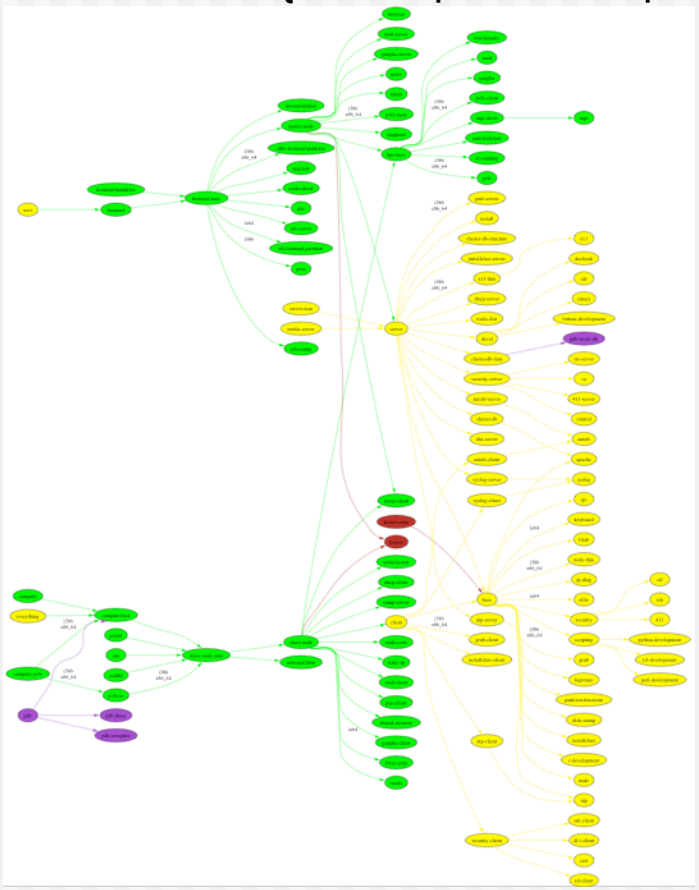


Two different Clusters

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



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



key point

Minor differences in the graph add up to large functional differences

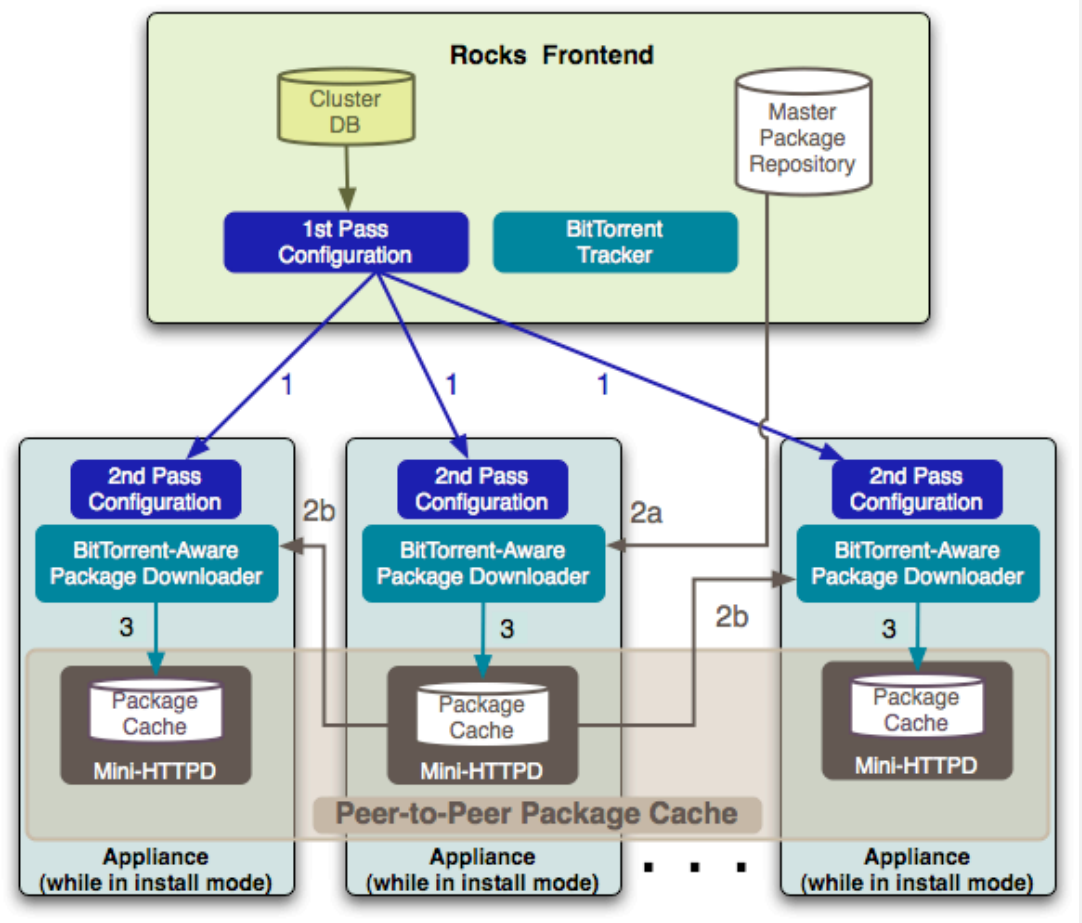
Where are the Scaling Limits?

- Time for Kickstart Generation
  - 3 - 4 s / host
  - *O*(n)
- Time to Download Packages
- Rocks uses HTTP to transport Packages
- Linux easily serves HTTP files at
  - 100MB/sec @ 1Gbit
  - 12 MB/Sec@100Mbit
- Time = <#nodes> * <total MB packages> / HTTP Speed
  - Total Packages ~ 350MB

	128 Nodes	1024 Nodes
100 Mbit	3700s (1hr)	9 hours
1 Gbit	460s (8 min)	1 hour

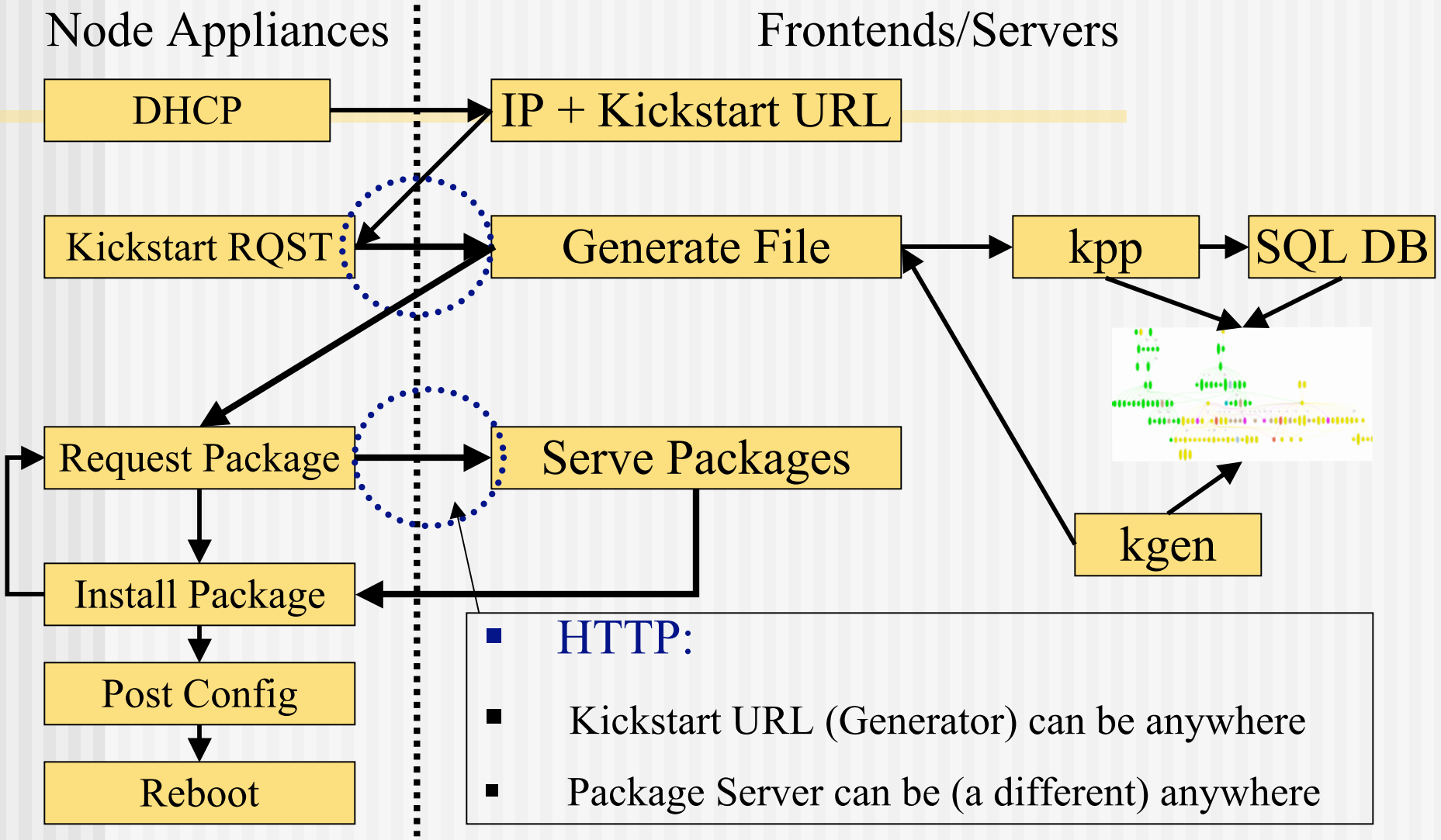
Avalanche Installer

- Install nodes from a peer-to-peer package cache
- Takes advantage of switched networks to unload the frontend
- Kickstart generation is split between frontend and nodes
- Backoff mechanisms keep the frontend load under control
- Zero administration



Pre-Avalanche

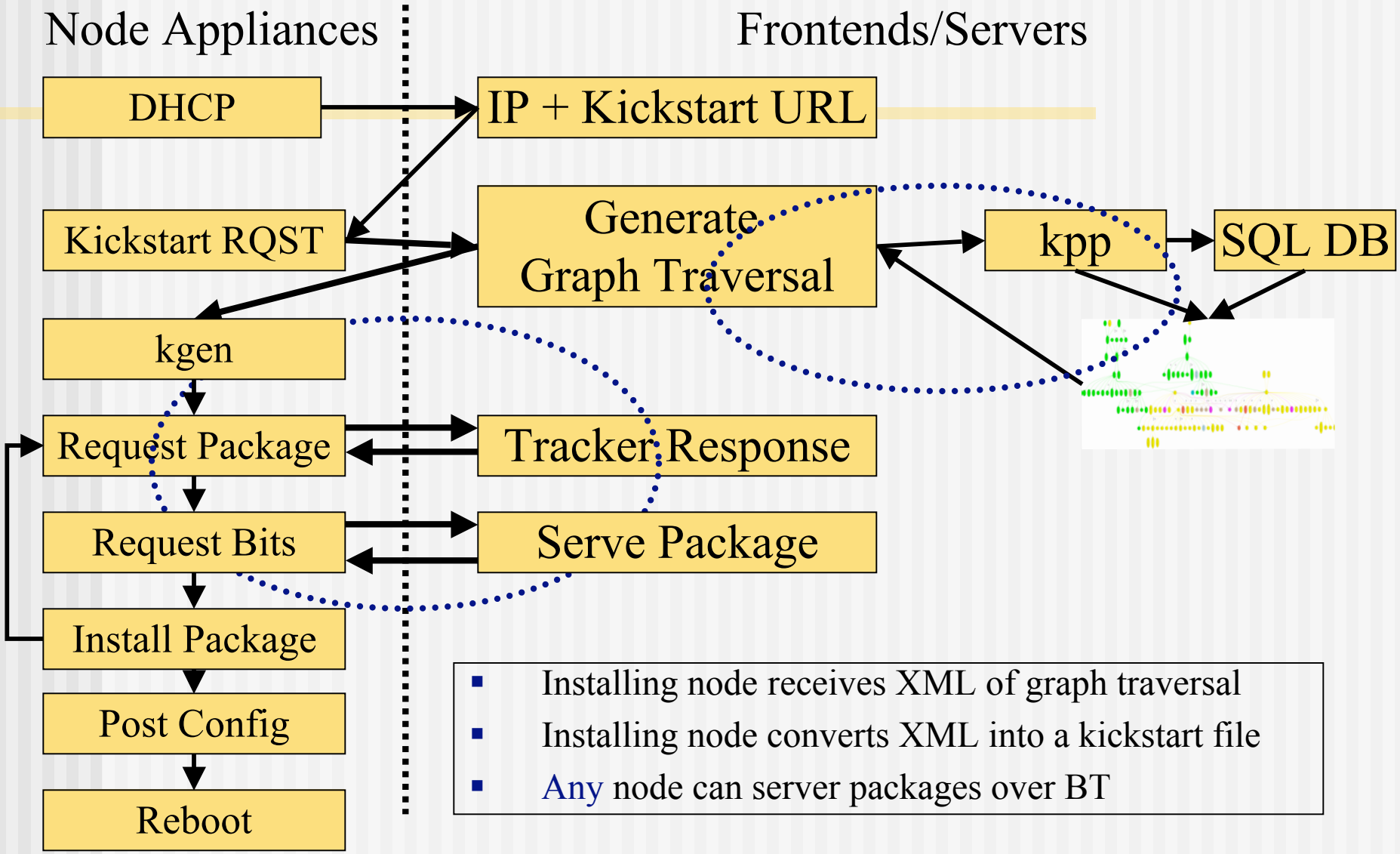
Space-Time and HTTP



- HTTP:
  - Kickstart URL (Generator) can be anywhere
  - Package Server can be (a different) anywhere

Avalanche

Space-Time and HTTP



- Installing node receives XML of graph traversal
- Installing node converts XML into a kickstart file
- Any node can server packages over BT

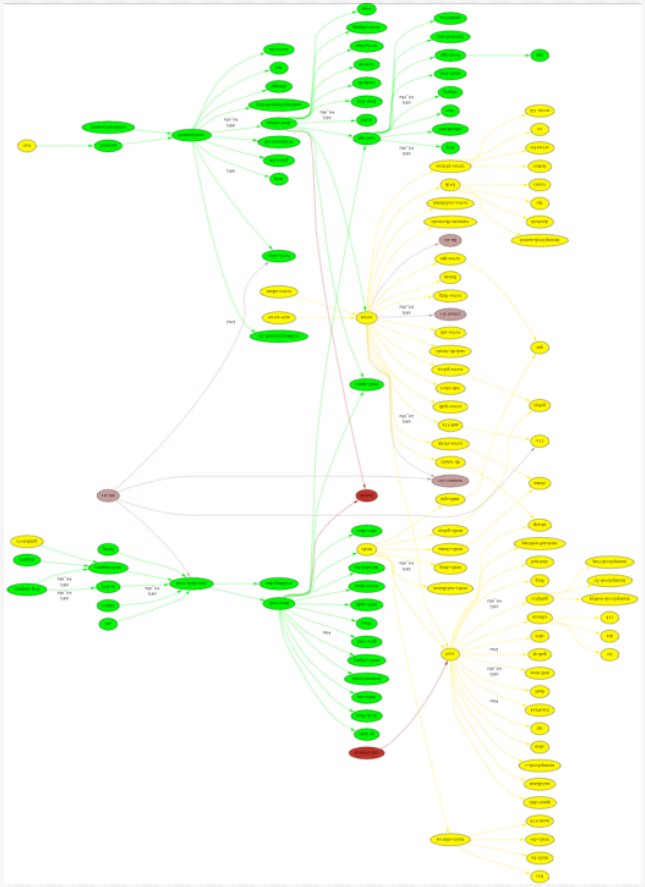
A Glimpse at Performance

- 45 Nodes – 100 Mbit
  - Old and Slow!
  - 350MB (Slim Compute Node)
- Pre-avalanche:
  - Estimate: 1600s
  - Actual: 1700s
- Avalanche:
  - Estimate: 900s
  - Actual: 1000s
- Avalanche is significantly quicker – and reduces load on the frontend

OptIPortal

viz roll

{ base, hpc, kernel, viz }

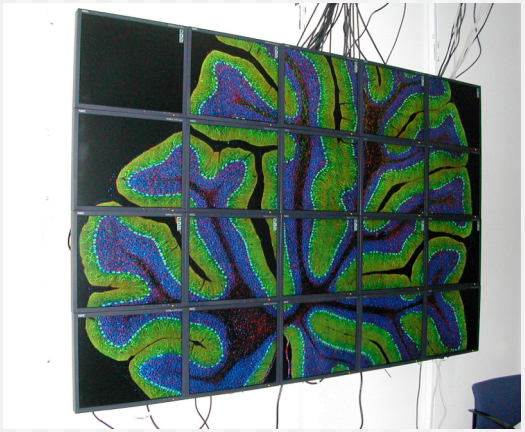


Early Work: NCSA

- 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

Genomic Map (cgview)

Building a Rocks Clusters

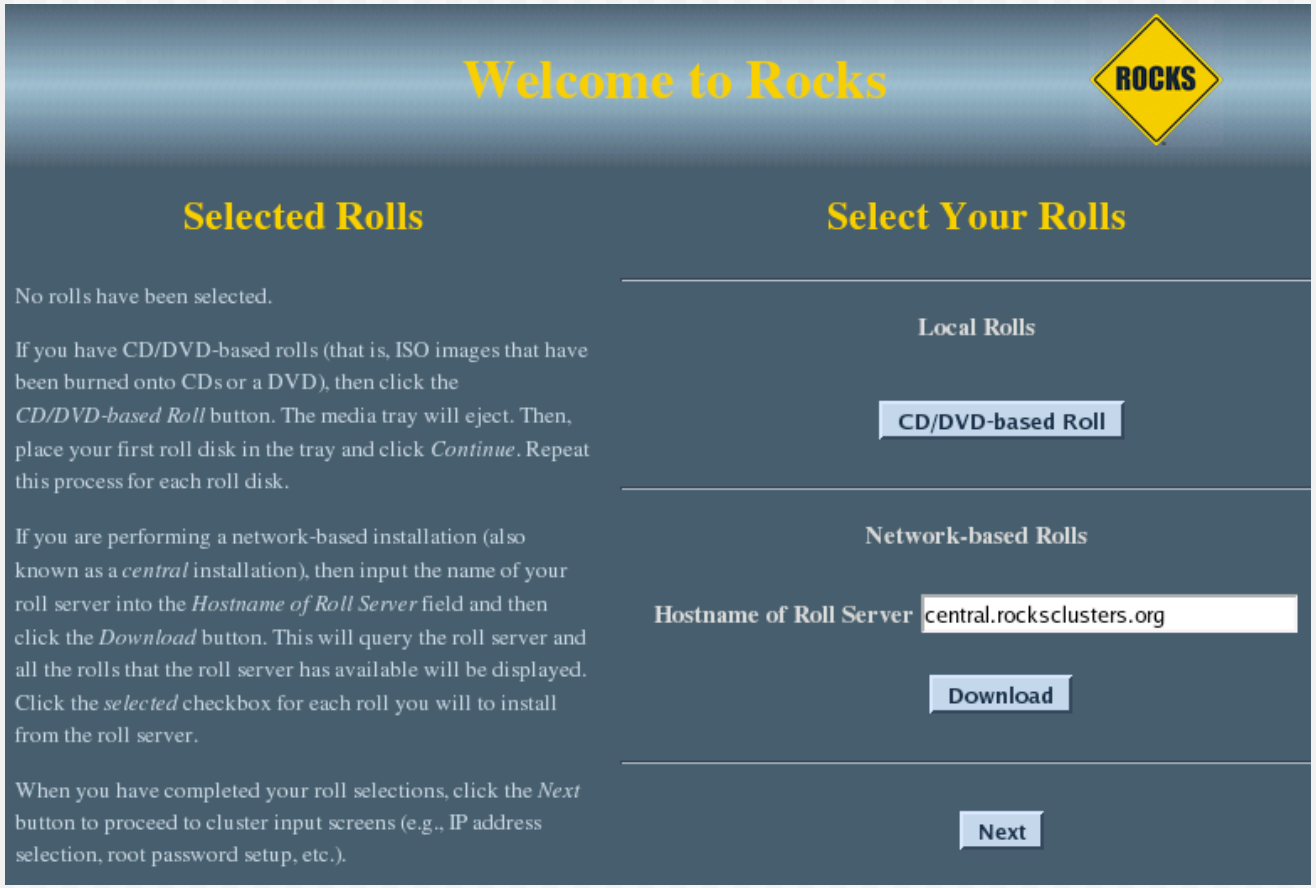
Young Frankenstein - Gene Wilder, Peter Boyle

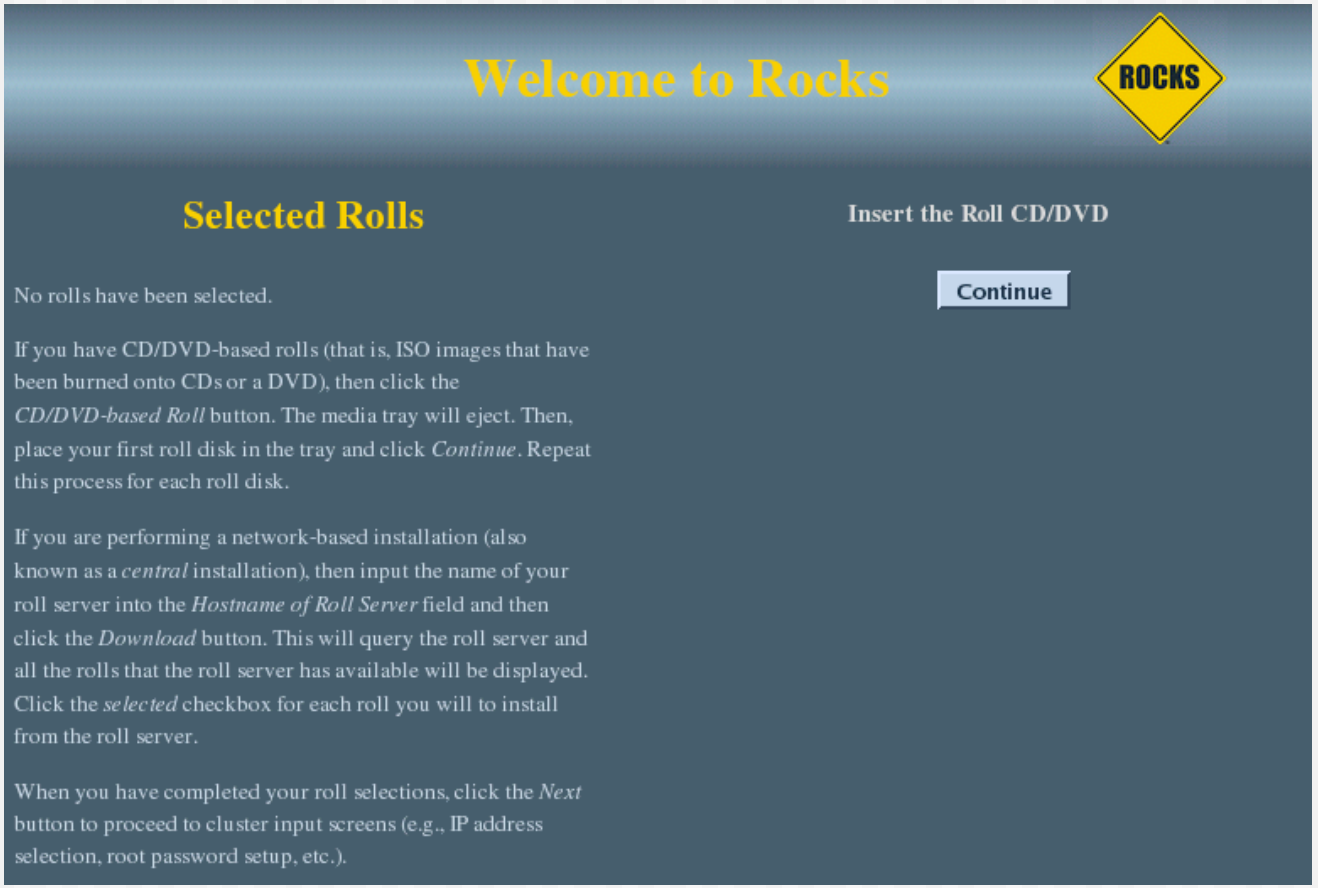
Frontend

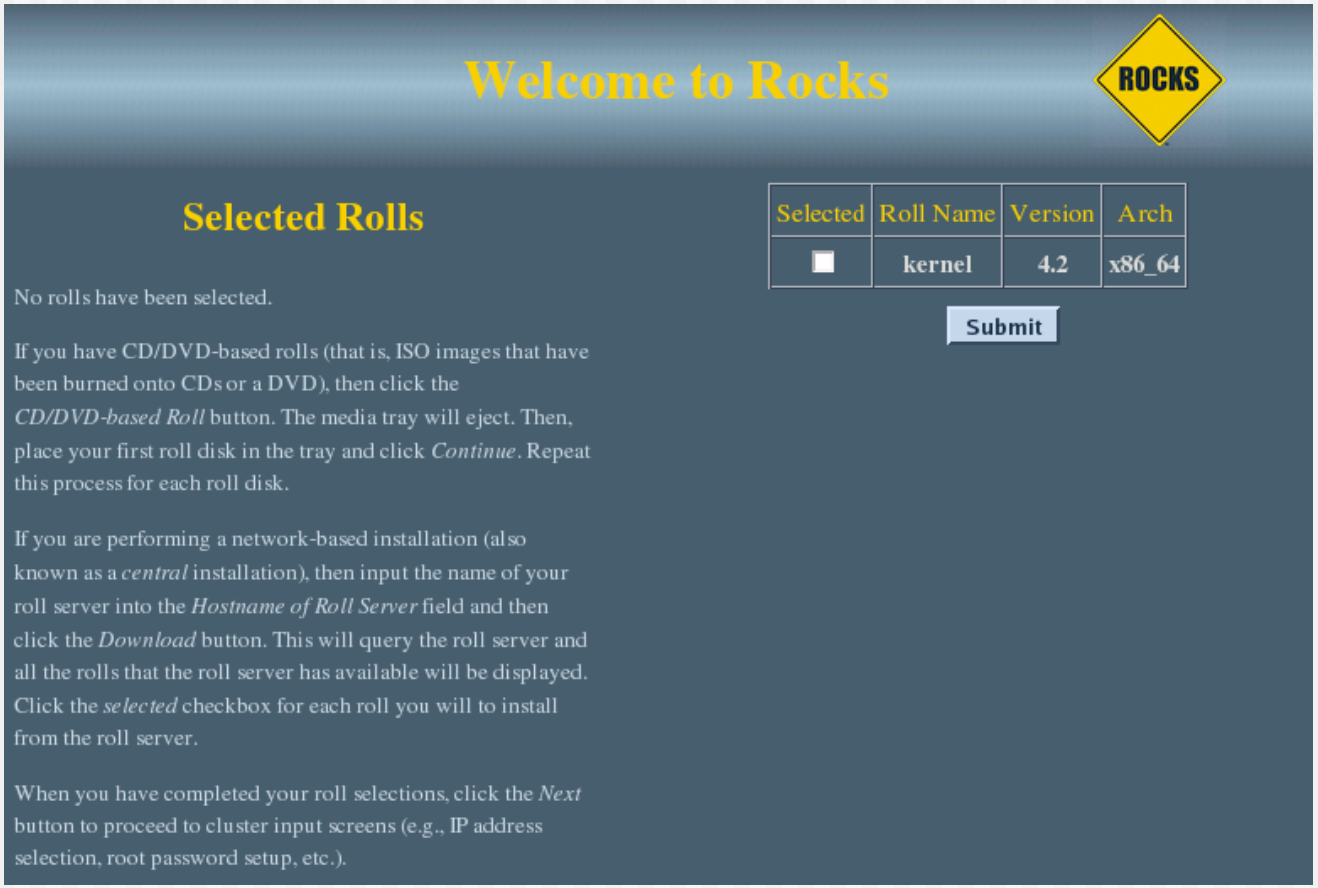
```
# frontend
For a new installation.

# frontend rescue
To boot into rescue
mode.
```

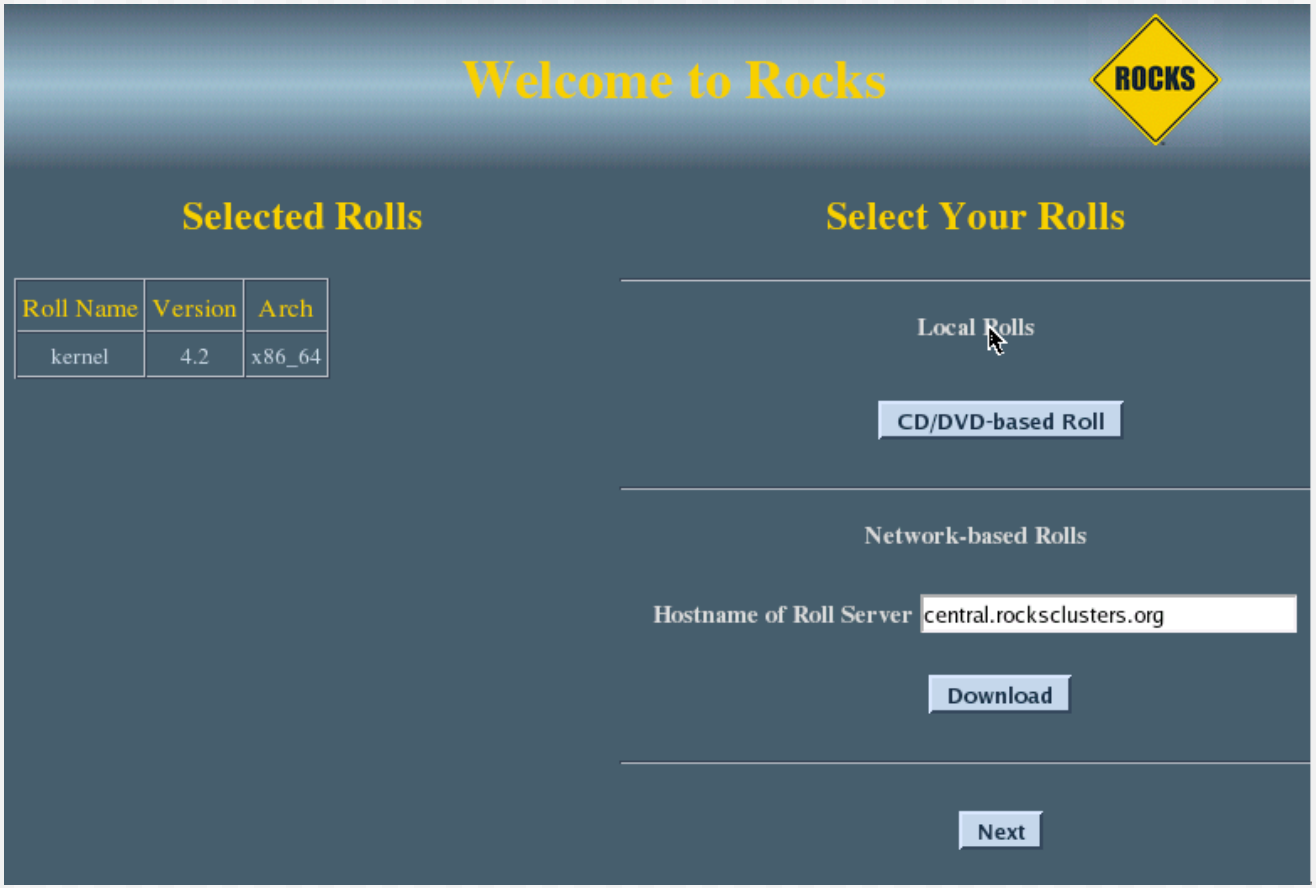
Client

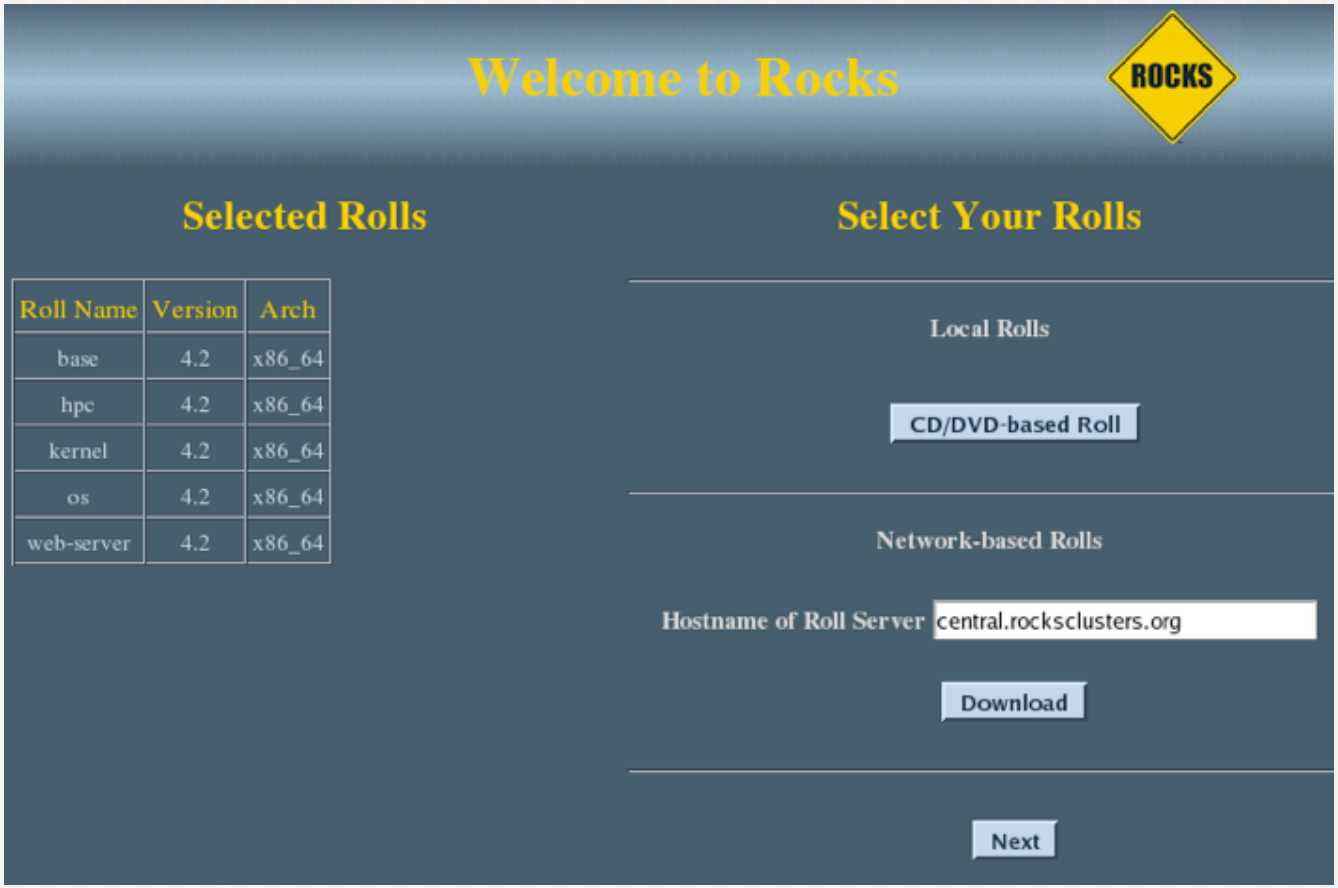
```
do nothing (default)
```

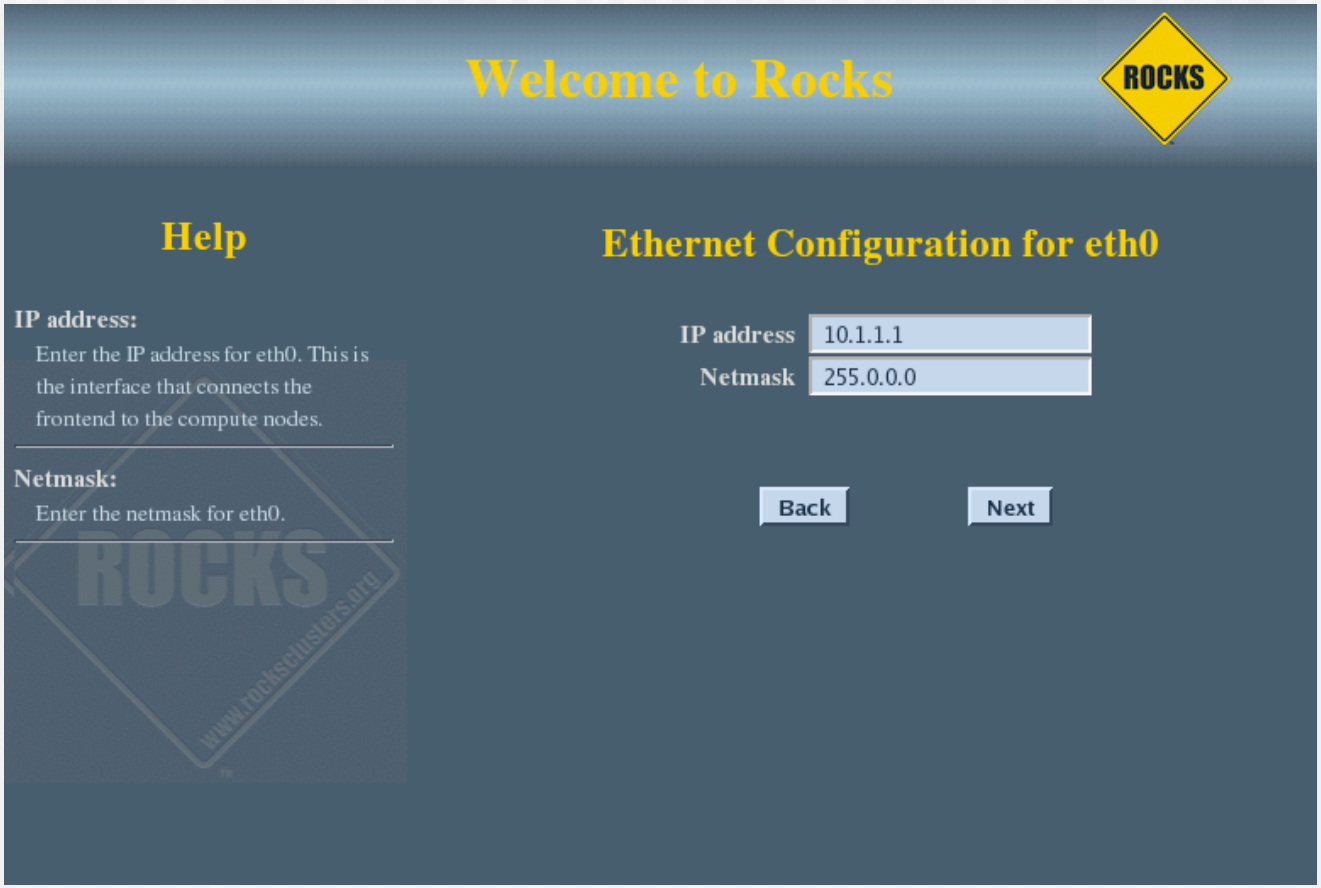


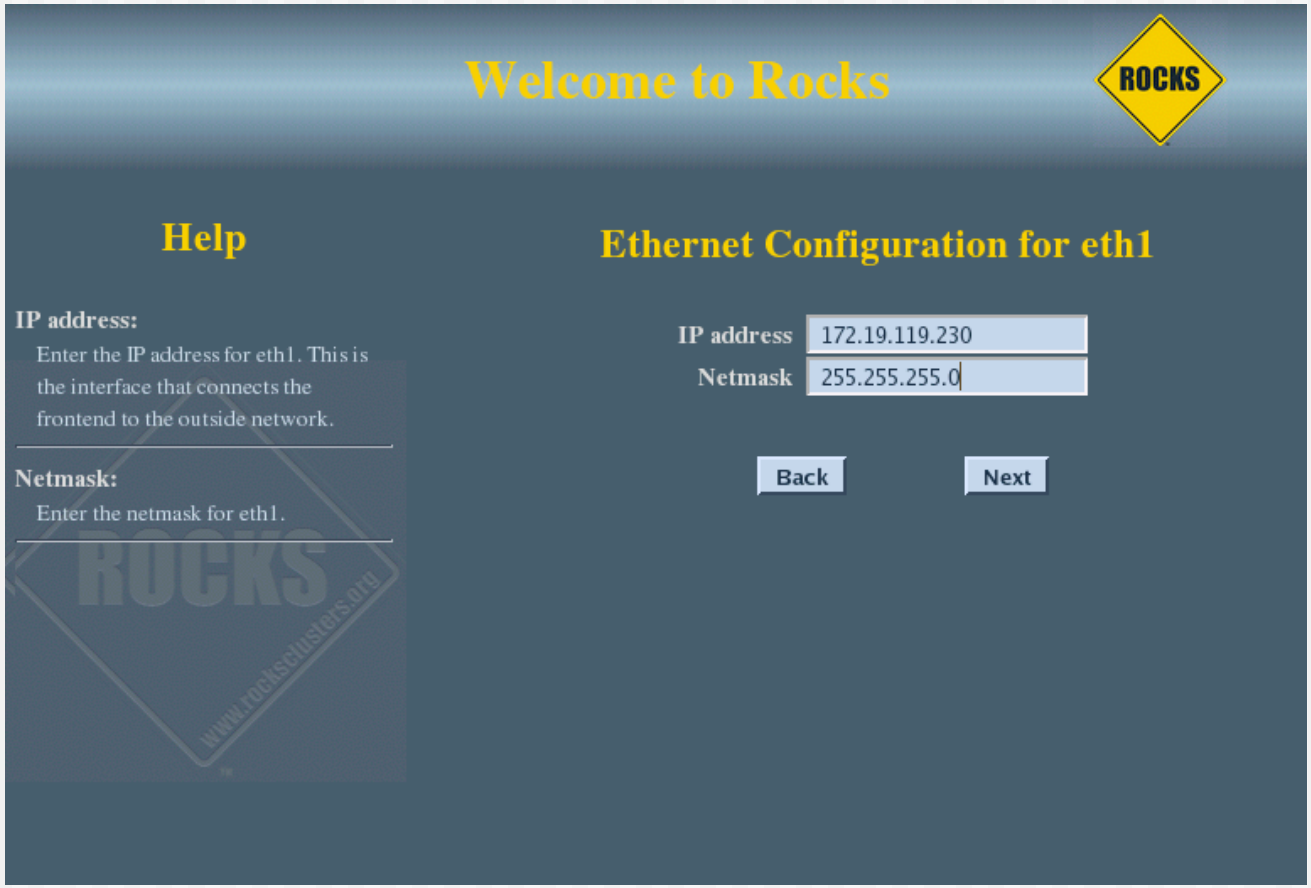


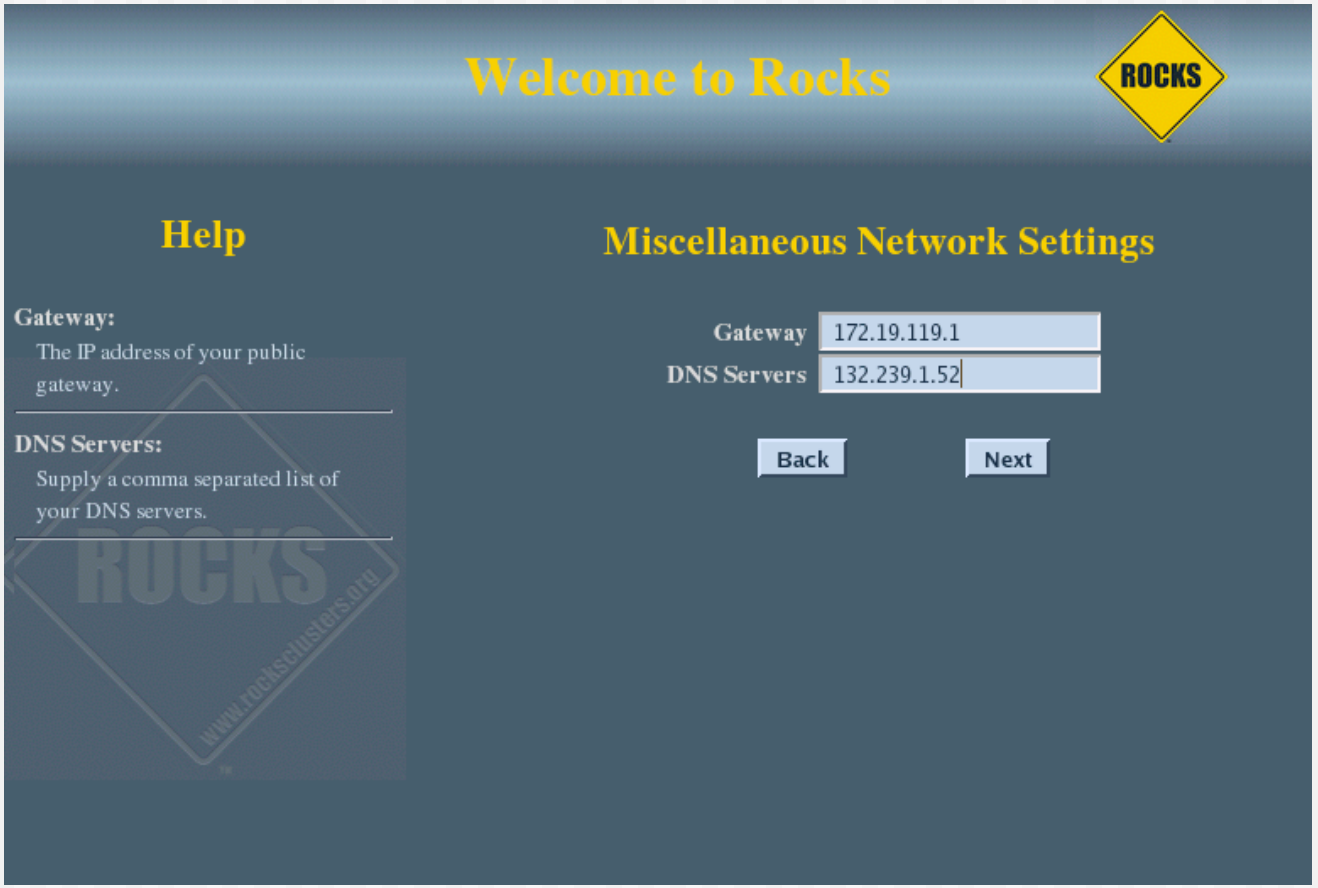


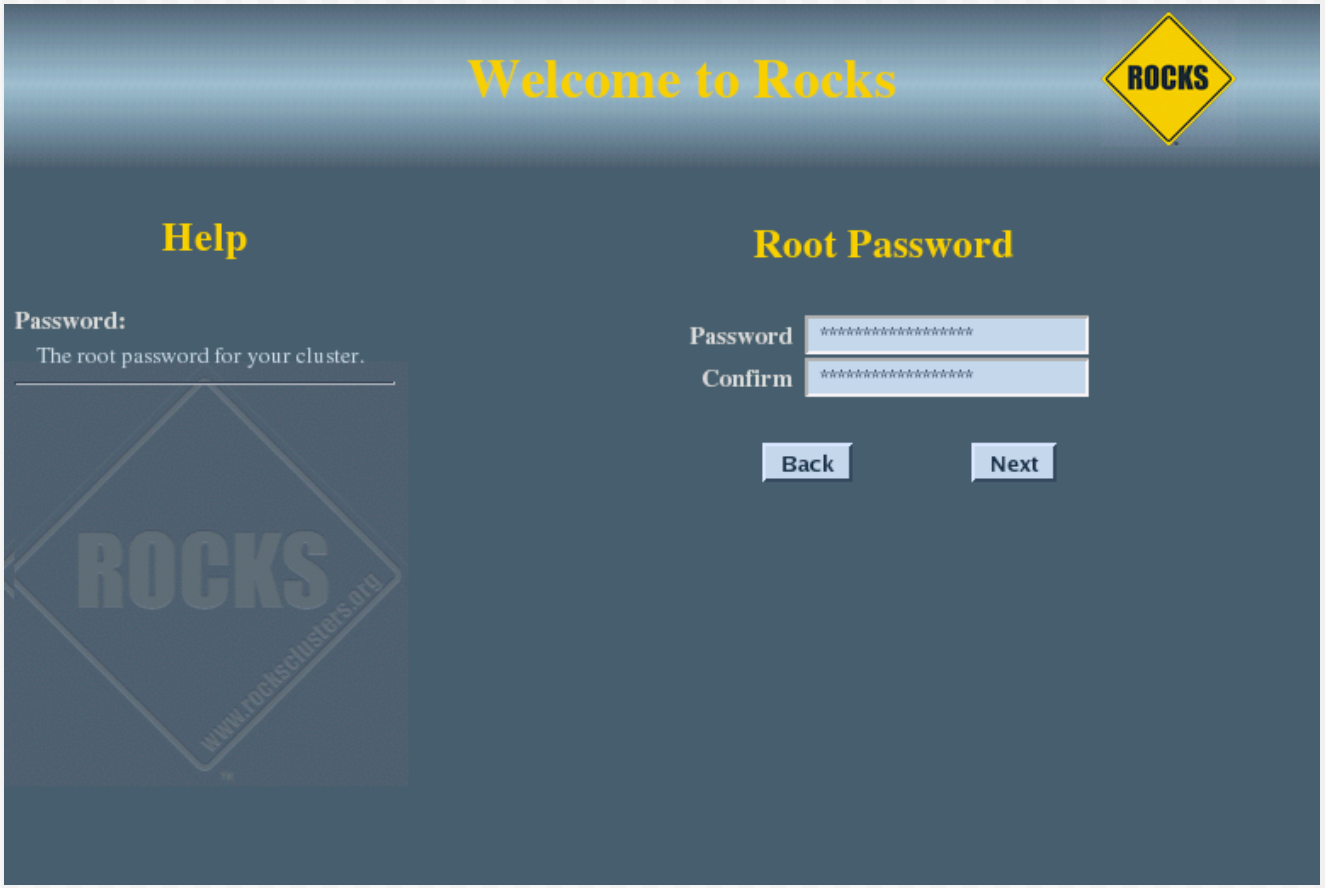


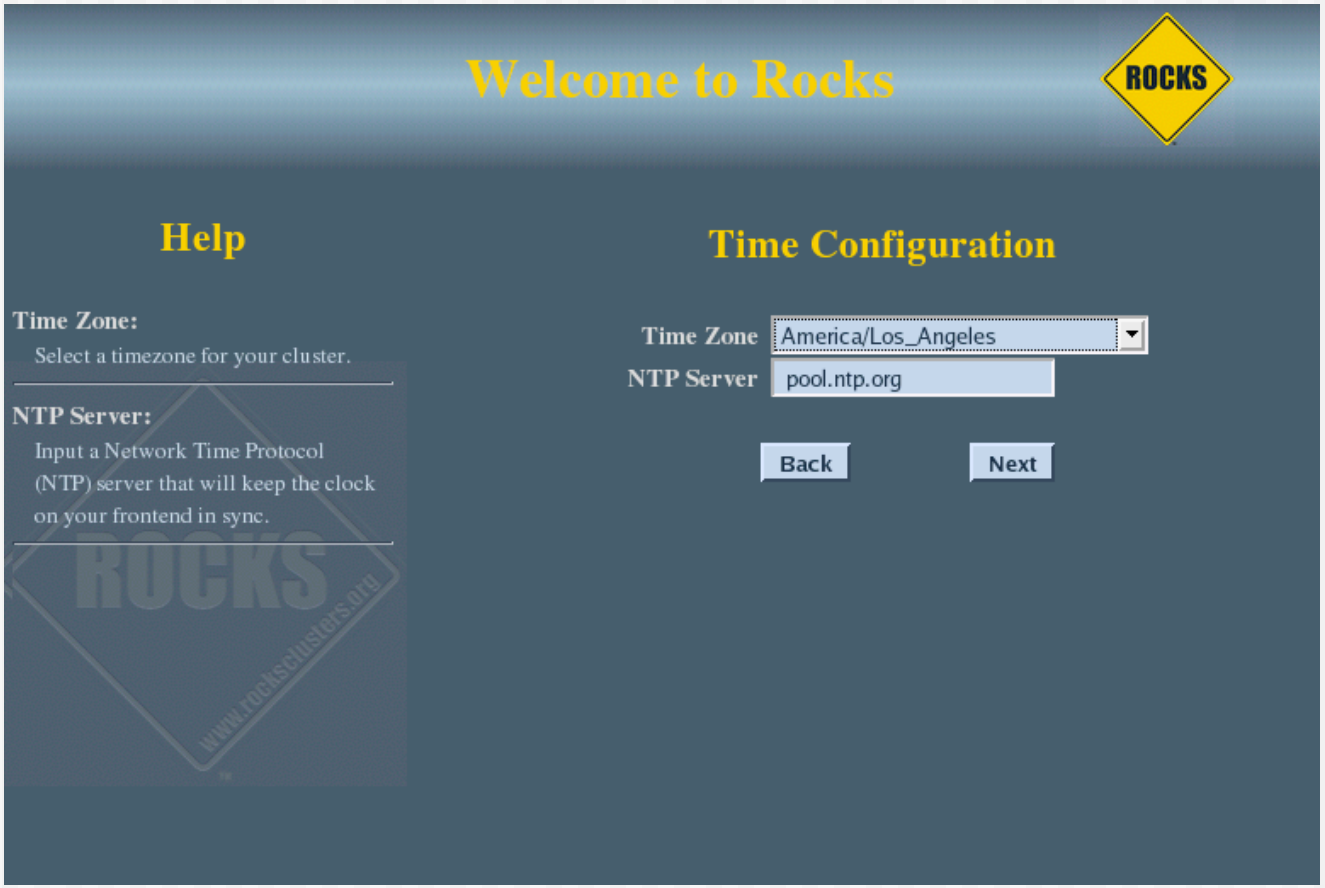


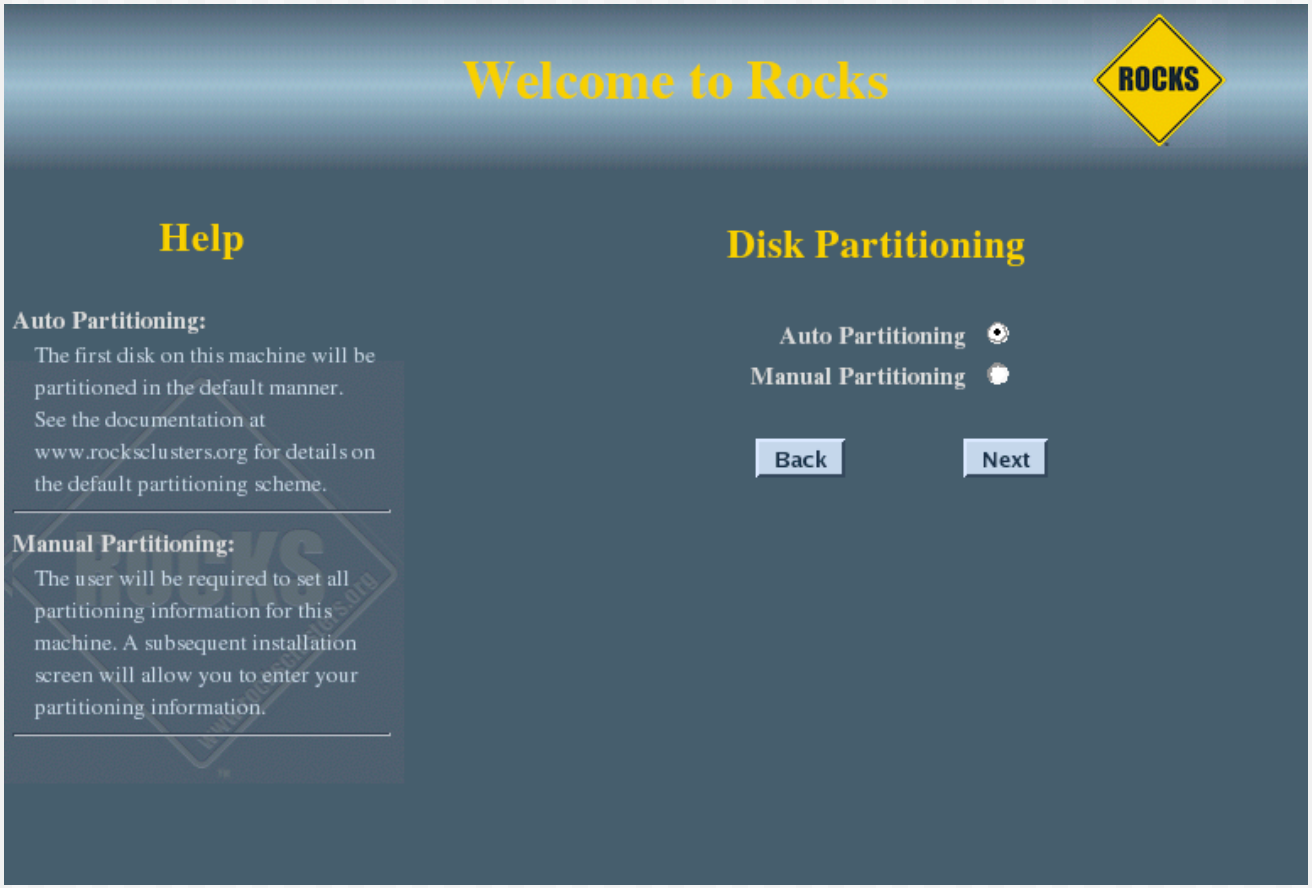






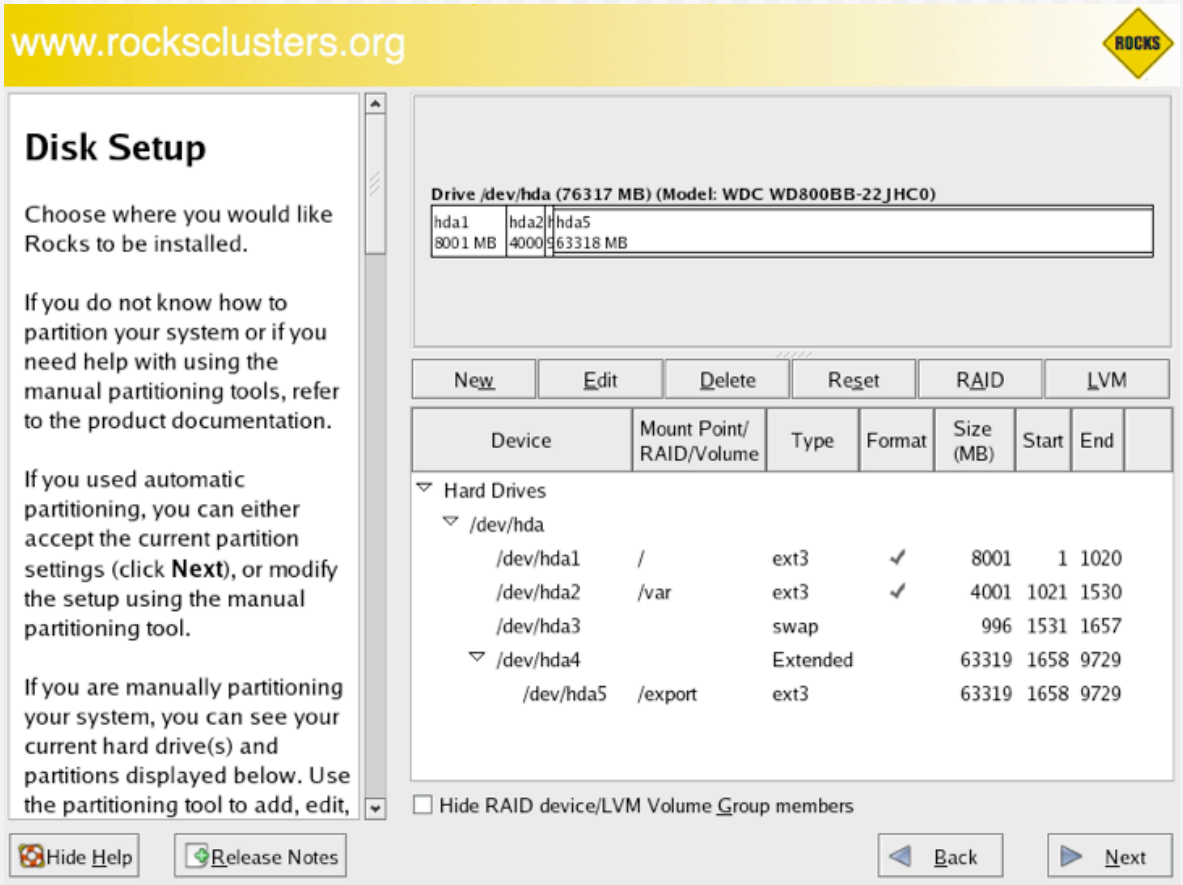


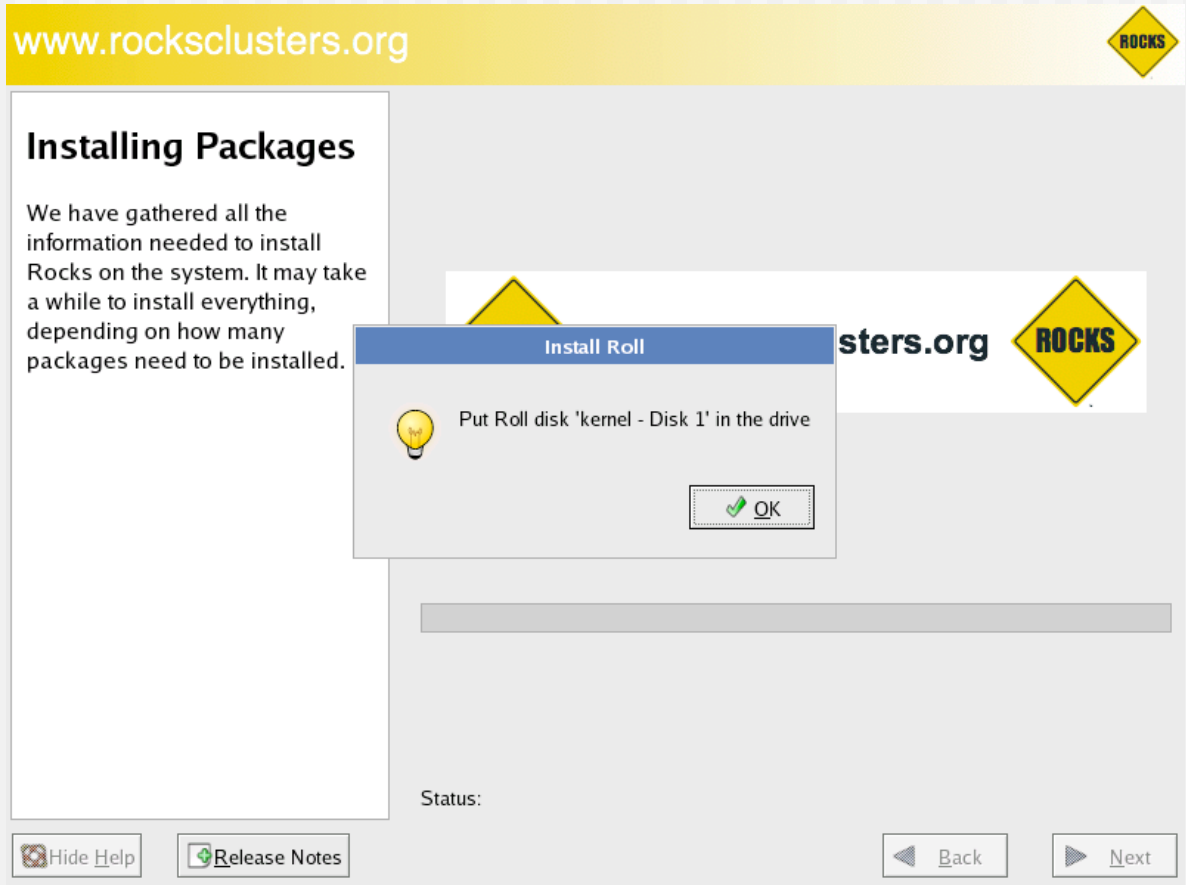


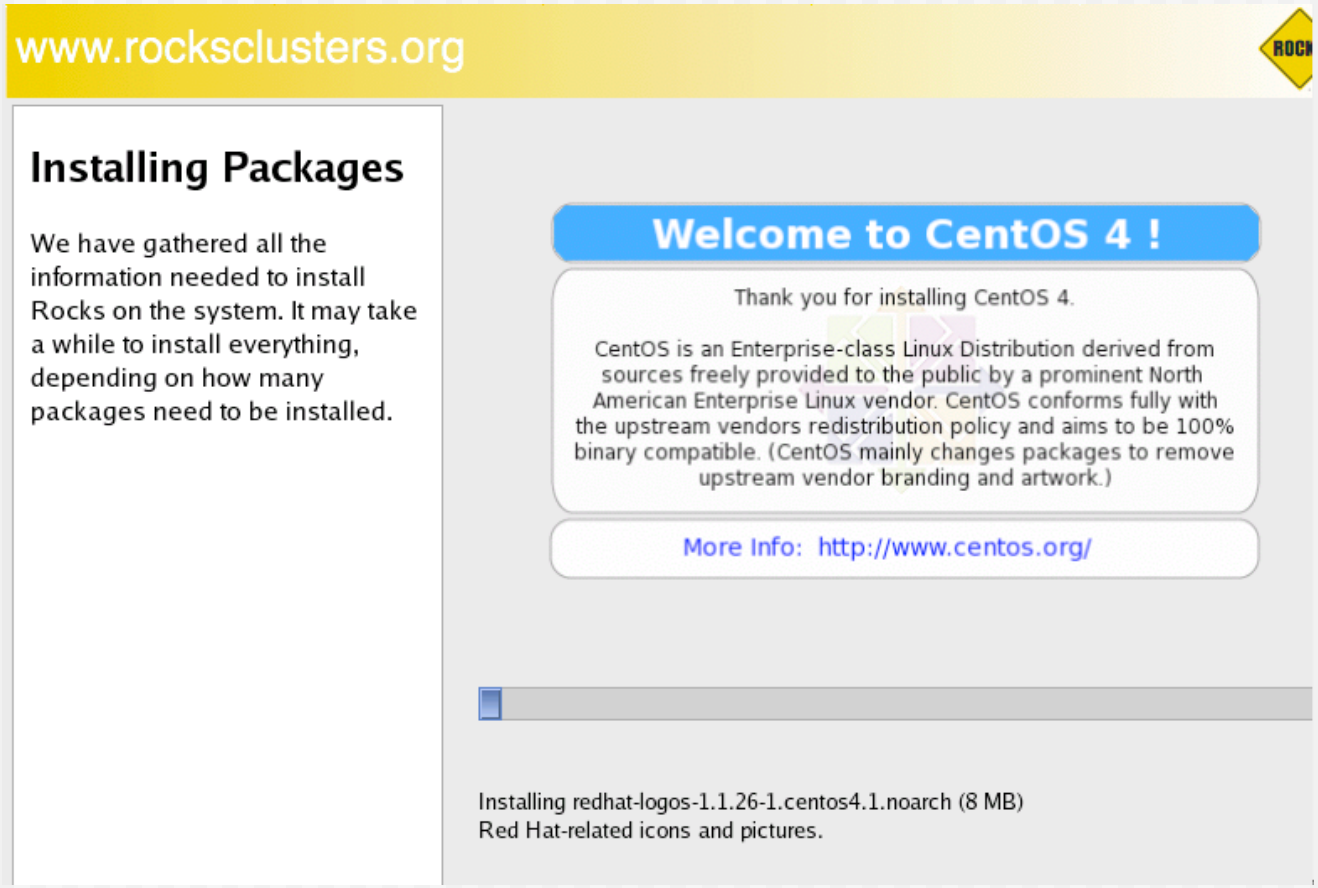


Manual Partition

not for new users





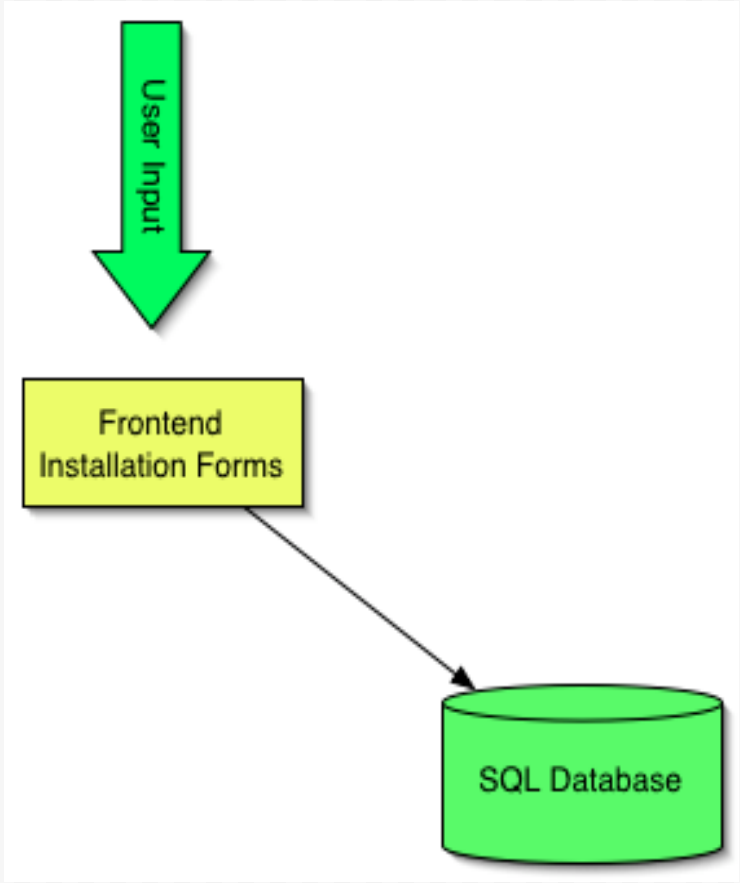


key point

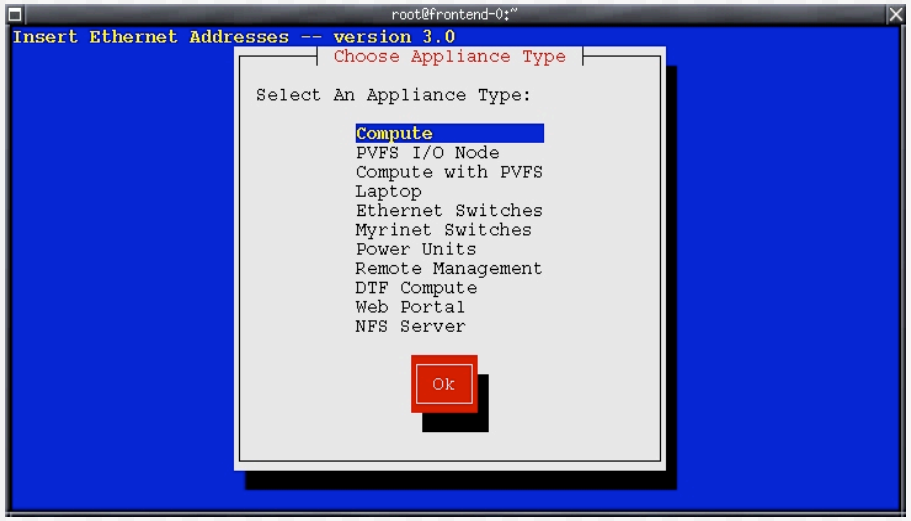
First time cluster builders should stay as close as possible to the defaults

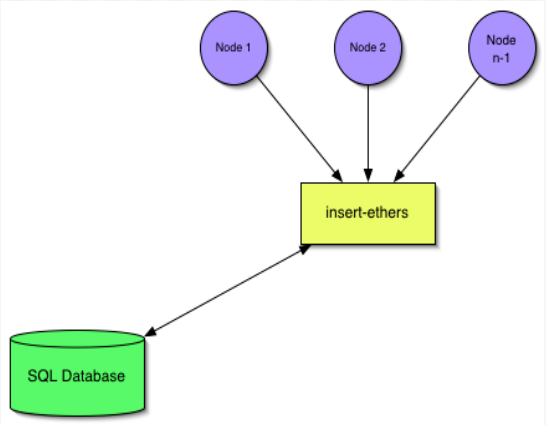
Interactive Screen

- Fill out the screens we just talked about
- Use the provided network information
- Choose your own password
- All information goes into the cluster database



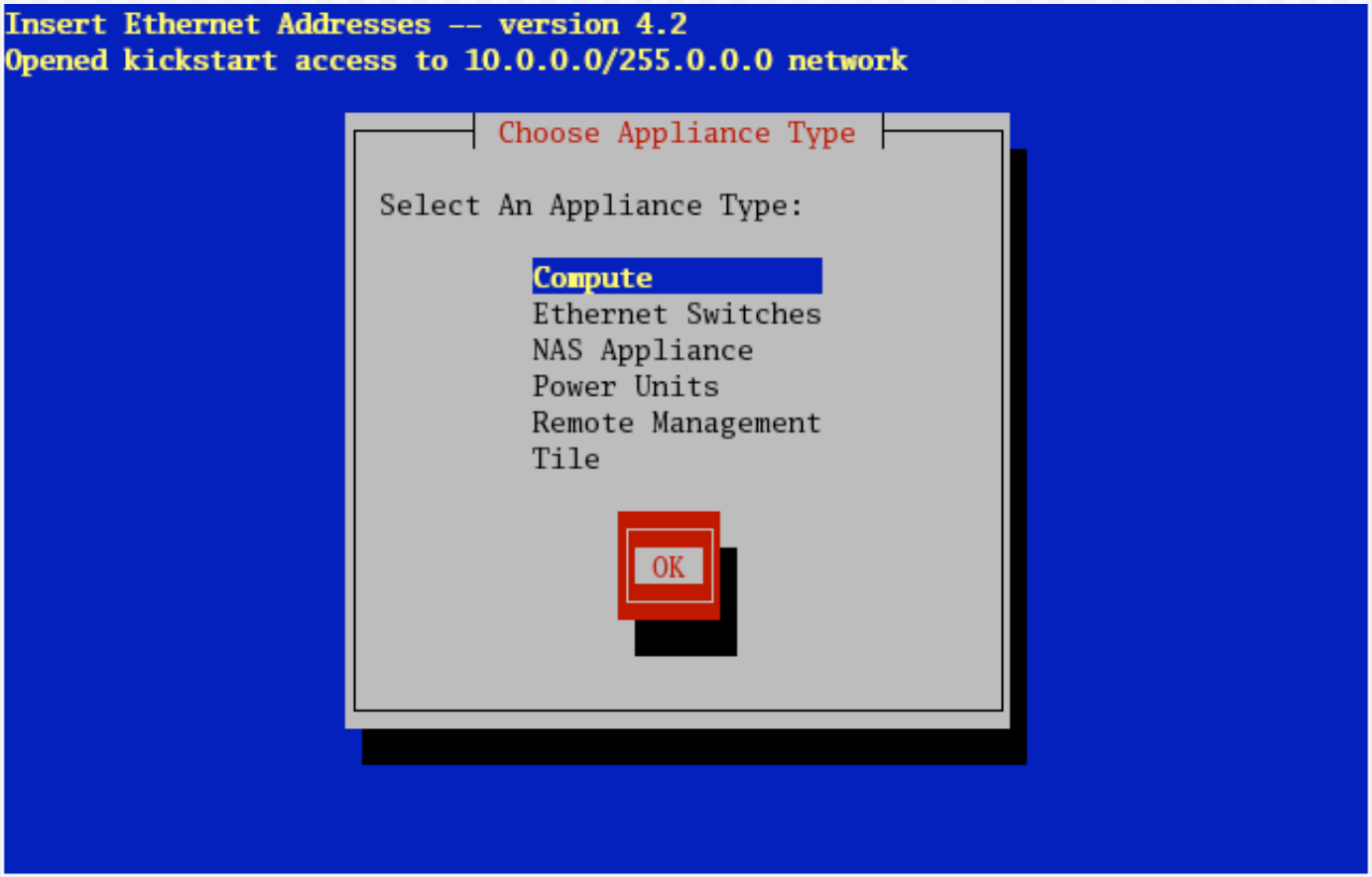
Add Compute Node with Insert-ethers

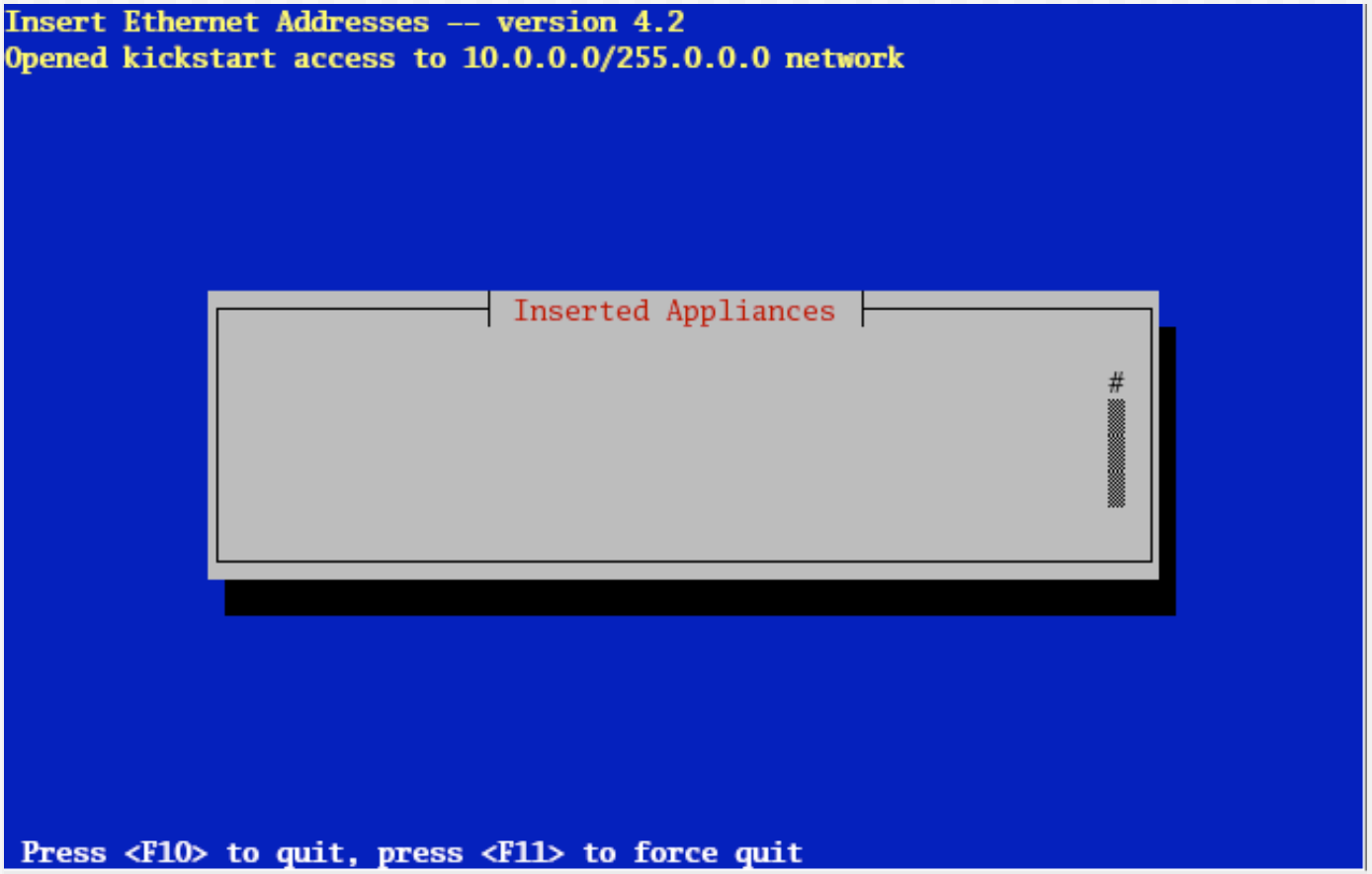


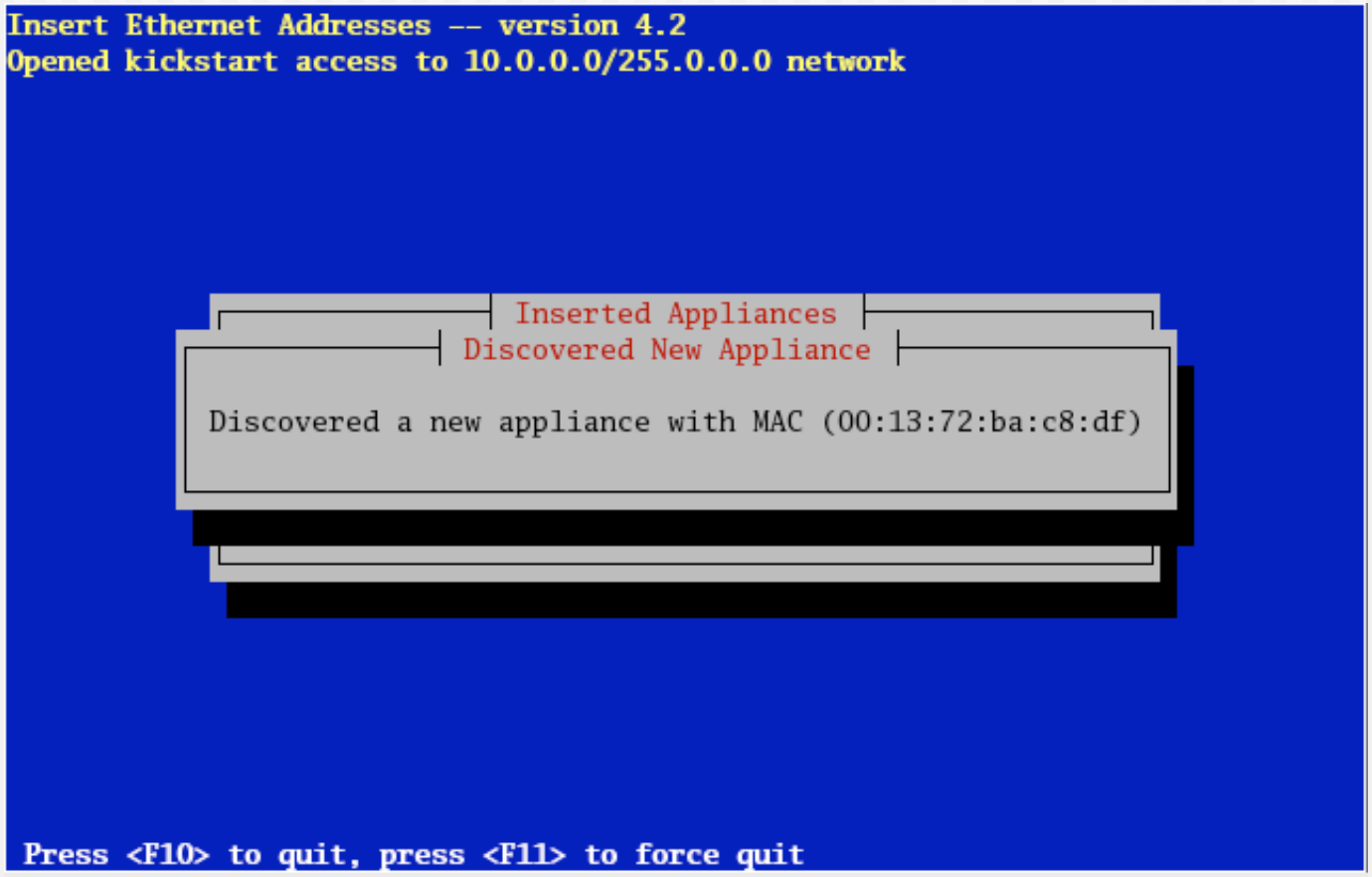


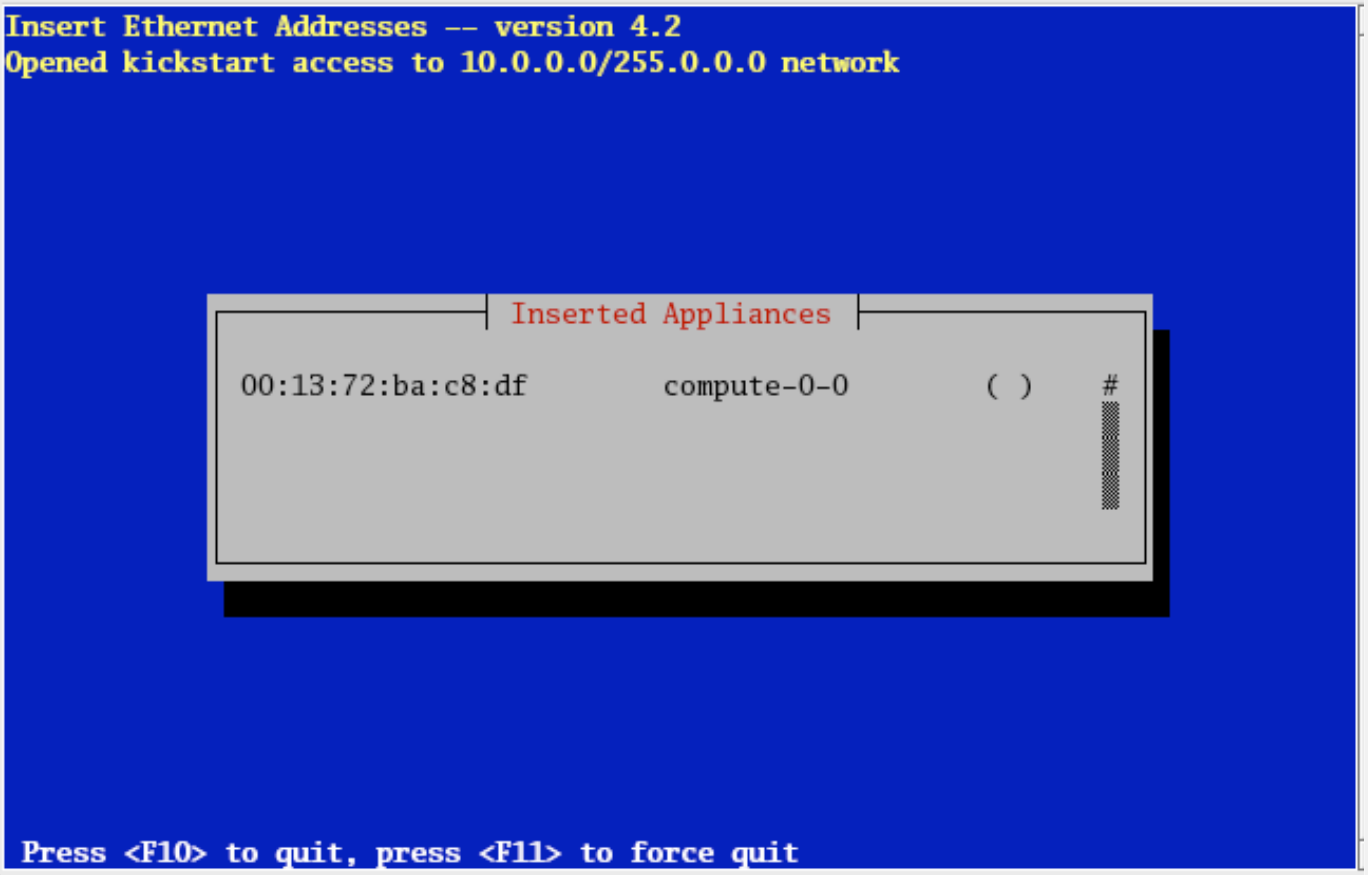
- Collect the Ethernet MAC address of cluster nodes
- Only done once, during integration
- Populates cluster database

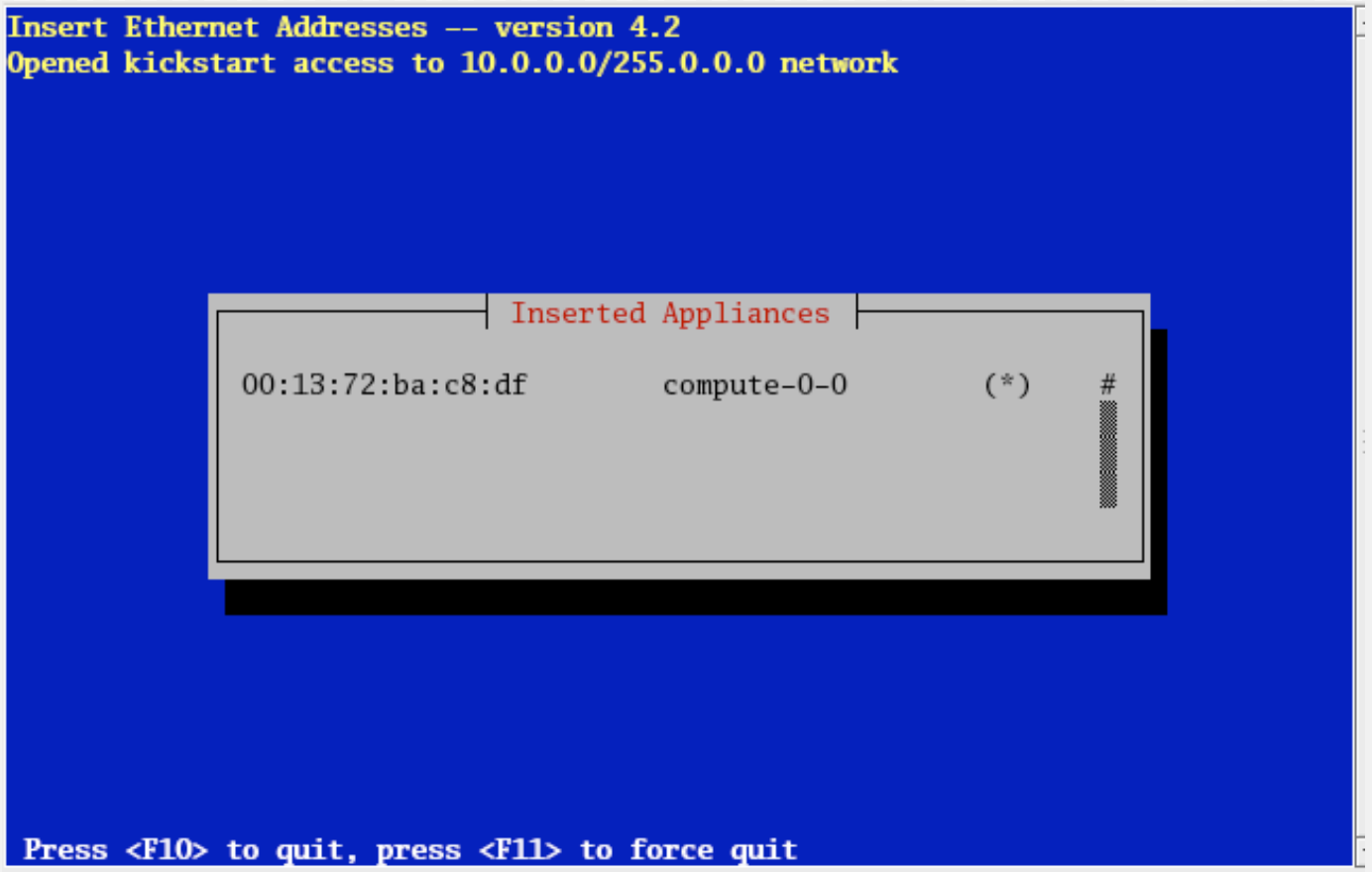
Adding Compute Nodes











Open Lab

rockstar.rocksclusters.org

- ssh access (no telnet)
- Account
  - Username: rap-01, rap-02, …
  - Password: amdrocks
- User level access only

Simple MPI Program

```
1: #include <stdio.h>
2: #include "mpi.h"
3:
4: int
5: main(int argc, char *argv[])
6: {
7:         int     numprocs;
8:         int     myid;
9:         int     namelen;
10:        char    processor_name[MPI_MAX_PROCESSOR_NAME];
11:
12:        MPI_Init(&argc, &argv);
13:
14:        MPI_Comm_size(MPI_COMM_WORLD, &numprocs);
15:        MPI_Comm_rank(MPI_COMM_WORLD, &myid);
16:        MPI_Get_processor_name(processor_name, &namelen);
17:
18:        fprintf(stderr, "Process %d on %s\n", myid, processor_name);
19:
20:        MPI_Barrier(MPI_COMM_WORLD);
21:
22:        sleep(120);
23:
24:        MPI_Finalize();
25: }
```

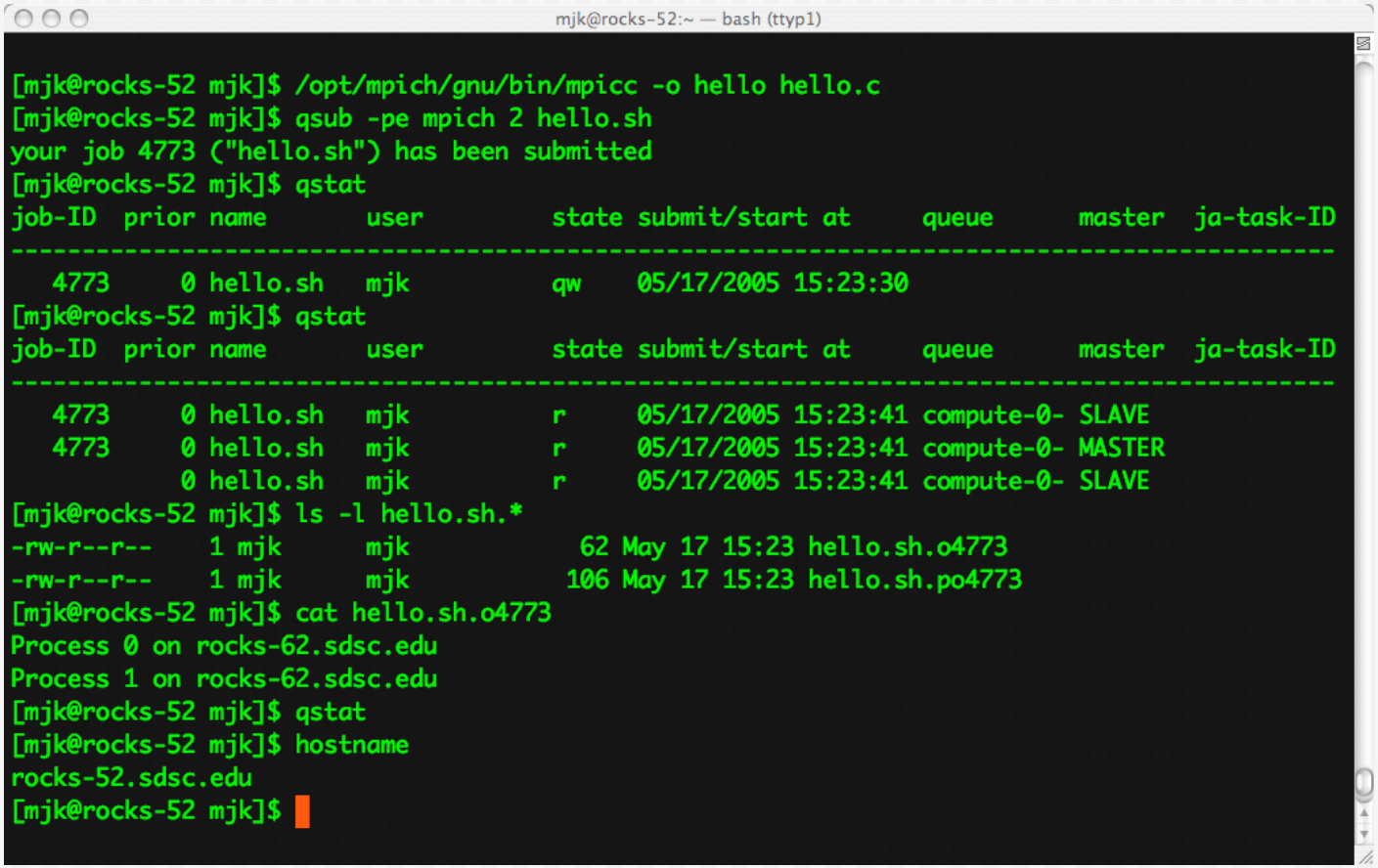
Simple MPI/SGE Submit Script

```
#! /bin/bash
#
#$ -cwd
#$ -j y
#$ -S /bin/bash

MPI_DIR=/opt/mpich/gnu

$MPI_DIR/bin/mpirun -np $NSLOTS -machinefile $TMPDIR/machines hello
```

Compile / Run

- Compile
  - `/opt/mpich/gnu/bin/mpicc -o hello hello.c`
- Run
  - `qsub -pe mpich 2 hello.sh`
- Monitor
  - `qstat`

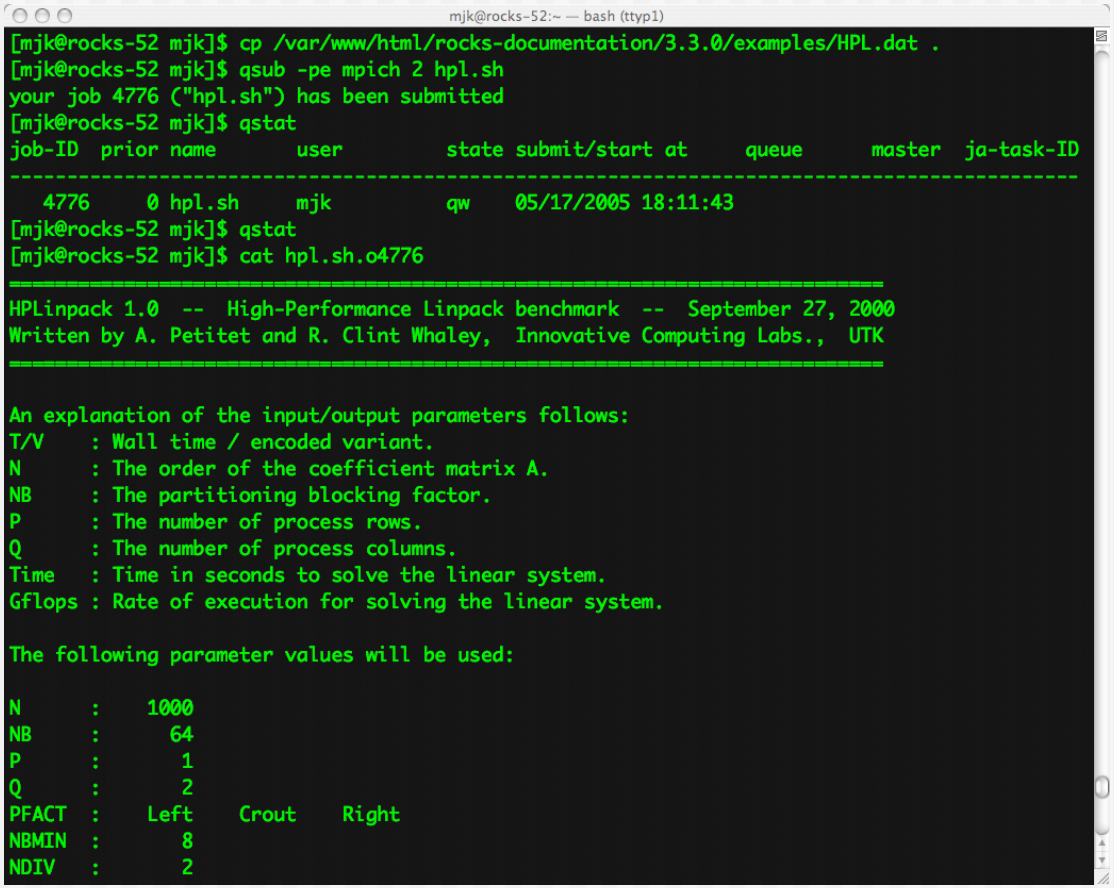
Example Run



HPL.dat

```
HPLinpack benchmark input file
Innovative Computing Laboratory, University of Tennessee
HPL.out      output file name (if any)
6            device out (6=stdout,7=stderr,file)
1            # of problems sizes (N)
1000 Ns
1            # of NBs
64 NBs
1            # of process grids (P x Q)
1 Ps
2 Qs
16.0         threshold
3            # of panel fact
0 1 2            PFACTs (0=left, 1=Crout, 2=Right)
1            # of recursive stopping criterium
8            NBMINs (>= 1)
1            # of panels in recursion
2            NDIVs
1            # of recursive panel fact.
2            RFACTs (0=left, 1=Crout, 2=Right)
1            # of broadcast
1            BCASTs (0=1rg,1=1rM,2=2rg,3=2rM,4=Lng,5=LnM)
1            # of lookahead depth
1            DEPTHs (>=0)
2            SWAP (0=bin-exch,1=long,2=mix)
80           swapping threshold
0            L1 in (0=transposed,1=no-transposed) form
0            U  in (0=transposed,1=no-transposed) form
1            Equilibration (0=no,1=yes)
8            memory alignment in double (> 0)
```

Example HPL Run



Linpack Scaling

- 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`

Others Tasks

- Globus
  - See grid roll usersguide
  - Setup user keys
  - `globus-job-run localhost /bin/hostname`
  - `globus-job-run localhost/jobmanager-sge`
- Adding RPMs to nodes
  - See usersguide for graph instructions
- Rebuild with Central/CDROM