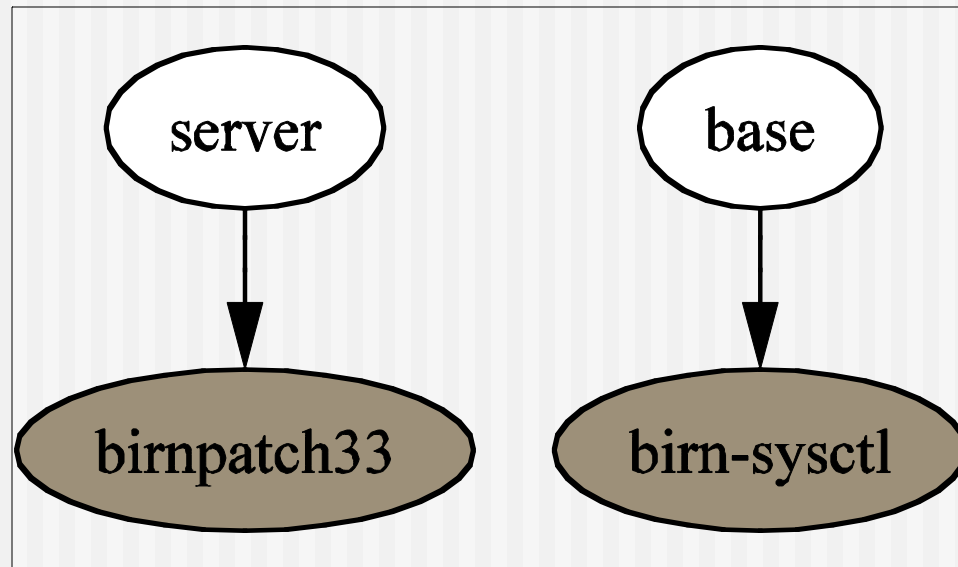




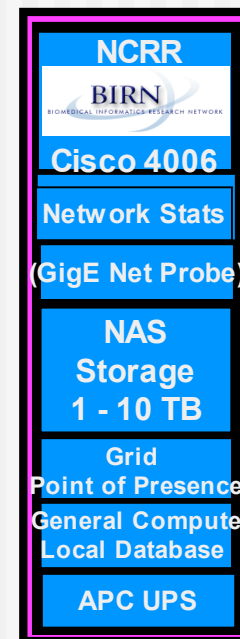
Birnpatch Roll

Philip Papadopoulos



BIRN Physical Environment

- ◆ BIRN Sites (20+) each have “clusters” with multiple appliances (nodes)
 - Grid Point of Presence
 - Storage (NAS)
 - Network monitoring
 - General Purpose
- ◆ 3 Networks
 - Private (internal)
 - Public (Internet)
 - VPN (BIRN-wide)
- ◆ Specialized Software on each System
 - BIRN Rolls (Apps, DB, Grid, Portal ...)
- ◆ Key Issues in Rocks 3.3
 - BIRN Patch Roll



At each BIRN site –
Data(Grid) Resource

BIRN Patch Roll

◆ Issues

- ⇒ Need to be able to reliably correlate ethernet Hardware Address (aa:bb:cc:dd:ee:ff) to logical ethernet (eth<n>)
 - HW Addresses Captured In DB for Compute nodes, but not for front-end
- ⇒ Wanted to be able to dump and restore node and network databases
- ⇒ Wanted automated changing of /etc/sysctl.conf for enhanced TCP support

- ◆ Basically, we needed to build a roll that would override and/or extend Rocks functionality

Rocks (Mis)Functionality

- ◆ Compute nodes automatically detect which interface is used for kickstart
 - ➔ HTTP Kickstart Request has HWAddr/Eth<n> Correlations passed to Kickstart.cgi
 - **ConfigNetworks.py** processes the request, adjusts frontend database, writes out ifcfg-eth<n> files, and modules.conf
 - ConfigNetworks did not work correctly with more than two interfaces (eth0 was always right, but eth1, eth2 were effectively random)
 - Patched ConfigNetworks.py, Roll drops the newer version on site/profiles/3.3.0/include/applets, taking precedence over the one supplied by Rocks.
- ◆ Worked for Nodes, but not frontends
 - ➔ Wrote a script that probed ethernet hardware for frontend dropped into the existing network database






More Changes

- ◆ With ConfigNetworks.py Fixed, Architecturally wanted to create network configs with dbreports
 - ⇒ Eg. `dbreport ifcfg eth0 compute-0-0` creates a file that can go in `/etc/sysconfig/networking-scripts/ifcfg-eth0`
 - New report part of Rocks 4.0
- ◆ Wanted to integrate per-node static routes.
 - ⇒ Create another dbreport (“static-routes”) that uses the existing but not used routes table
 - New report part of Rocks 4.0

Getting Frontend Ethernet HW Mapping

- ◆ Consistent Issue for Rocks
 - Kernels “randomly” label ethernet interfaces. Probe order of kernel and BIOS are often different
 - “I want to make HWADDR X my public network”
- ◆ Frontend central=Rocks allows you to select any interface for building (good!)
 - But interface may be relabeled by kernel
- ◆ Wanted to assign “proper labels” for all interfaces before installation rebooted.
 - Needed to do this in the <post> and be kernel agnostic
 - Added “postshell” as another keyword when kickstarting
 - Added a node in the graph that holds installation until human installer removes /mnt/sysimage/tmp/postshell from the system

Logic of Postshell

- ◆ Frontend central=Rocks **postshell**
-  Probe Ether Interfaces (automatic)
-  Create file that populates networks table with HWAddress mappings
 - ◆ Generated automatically, edited by person in postshell to get desired mapping
-  Remove /mnt/sysimage/tmp/postshell to allow install to complete
-  Load new network mapping into database (automatic)
-  Run new dbreports to create new ifcfg-eth<n> files (automatic)

Enhancing Rocks Utilities

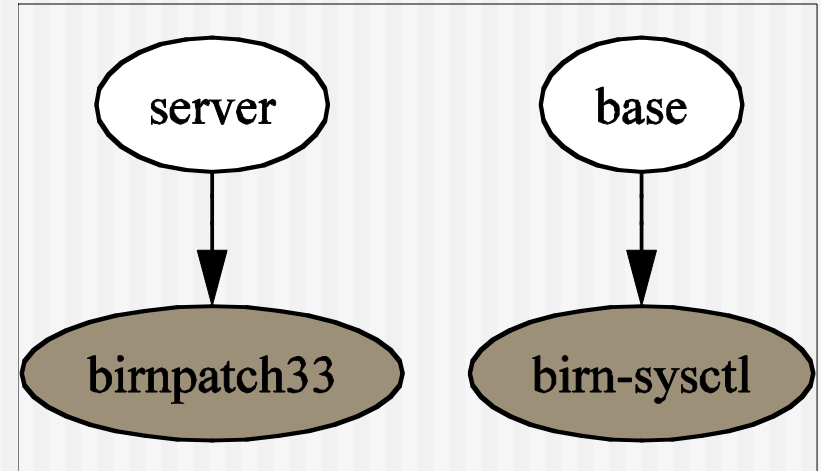
- ◆ insert-ethers, add-extra-nic
 - Added capability to specify everything from command line and insert into database eg.
 - Insert-ethers --mac="aa:bb:cc:dd:ee:ff" --appliance-type="Compute" will "discover" a new compute node and insert into database
 - All options (name, ip, netmask, rack, rank, etc) can be specified on the command line
 - Insert-ethers --dump
 - Dumps the nodes table in the form executable shell commands that look like the above
 - Mysqldump trick
- ◆ Added similar capability for add-extra-nic
- ◆ Now Part of Rocks 4.0. Very useful for users with vendor-integrated clusters that have a text file of ethernet macs.
 - Nice command-line access to key components of Rocks DB.
 - Dump allows us to change underlying schema!

Code Written

- ◆ Updated Rocks Code
 - ⇒ ConfigNetworks.py
 - ⇒ Insert-ethers.py, add-extra-nic.py
- ◆ New Code
 - ⇒ Dbreports: ifcfg, static-routes
 - ⇒ Probe ethernet HW for frontend end
- ◆ Graph Logic
 - ⇒ Added postshell handling
 - ⇒ Ordering important

Kickstart Subgraph

- ◆ Two packages added (defined in roll)
- ◆ Add Postshell handling for Servers
- ◆ Add sysctl changes for IP stack modifications to all nodes
- ◆ Total Compiled size of Roll ISO: 638KB



Where Rocks “Rocked”

- ◆ Roll Template made the creation of the shell of the roll “stupidly easy”
- ◆ Changes were surgical – Things I changed were not originally developed by me, but I could override them
- ◆ Graph changes made it clear where to add functionality depending on appliance type
 - ➔ Server Only
 - ➔ Nodes Only
 - ➔ Everything
- ◆ Other BIRN Rolls were developed separately and then these were tested together.