Building Clusters With FreeBSD

Brooks Davis
The Aerospace Corporation
<brooks@{aero,freebsd}.org>
September 13, 2007

http://people.freebsd.org/~brooks/pubs/eurobsdcon2007/

THE AEROSPACE CORPORATION

Tutorial Outline

- Overview of Fellowship
- Cluster Architecture Issues
- Operational Issues
- Thoughts on a Second Cluster
- FreeBSD specifics

Overview of Fellowship

THE AEROSPACE CORPORATION

Overview of Fellowship

- The Aerospace Corporation's corporate, unclassified computing cluster
- Designed to be a general purpose cluster
  - Run a wide variety of applications
  - Growth over time
  - Remote access for maintainability
- Gaining experience with clusters was a goal
- In production since 2001
- >100 users

Overview of Fellowship System Software

- FreeBSD 6.2
- Sun Grid Engine (SGE) scheduler
- Ganglia cluster monitor
- Nagios network monitor

FreeBSD®

GRID ENGINE

Ganglia

Nagios®

THE AEROSPACE CORPORATION

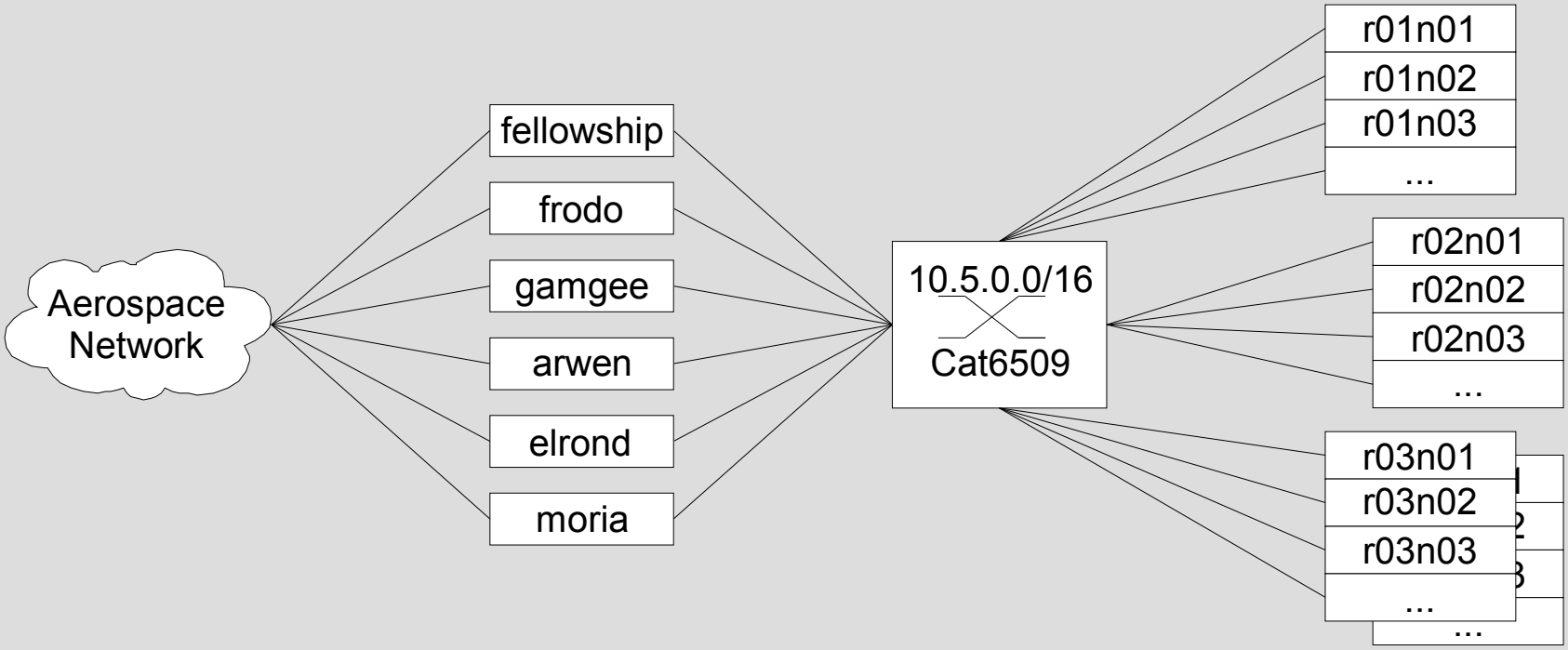
Overview of Fellowship Hardware

- 352 dual-processor nodes
  - 64 Intel Xeon nodes (soon to be quad cores)
  - 289 Opteron nodes (152 dual-core)
- 3 TB shared (NFS) disk
- >60TB total storage
- 700GB RAM
- Gigabit Ethernet
  - 32 nodes also have 2Gbps Myrinet

Overview of Fellowship Facilities

- ~80KVA power draw
  - Average US house can draw 40KVA ***max***
- 273 kBTU/hr ~= 22Tons of refrigeration
- 600 sq ft floor space
  - Excluding HVAC and power distribution

Overview of Fellowship Network Topology



Cluster Architecture Issues

- Architecture matters
  - Mistakes are compounded when you buy hundreds of machines
- Have a requirements process
  - What are your goals?
  - What can you afford?
    - Upfront
    - Ongoing

Cluster Architecture Issues

- Operating System
- Processor Architecture
- Network Interconnect
- Storage
- Form Factor
- Facilities
- Scheduler

Cluster Architecture Issues Slide Format

- Trade offs and Considerations
  - The trade space and other things to considers
- Options
  - Concrete options
- What we did on Fellowship
- How it worked out

Operating System

Trade offs and Considerations

- Cost: Licensing, Support
- Performance: Overhead, Driver quality
- Hardware Support: Processor, Network, Storage
- Administration: Upgrade/patch process, software installation and management
- Staff experience: software porting, debugging, modification, scripting

Operating System

Options

- Linux
  - General purpose distros: Debian, Fedora, Red Hat, SuSE, Ubuntu, etc.
  - Cluster kits: Rocks, OSCAR
  - Vendor specific: Scyld
- BSD: FreeBSD, NetBSD, OpenBSD
- MacOS/Darwin
- Commercial Unix: Solaris, AIX, HPUX, Tru64
- Windows

Operating System

What we did on Fellowship

- FreeBSD
  - Started with 4.x
  - Moved to 6.x

How it worked

- Netboot works well
- Linux emulation supports commercial code (Mathematica, Matlab)
- No system scope threads in 4.x (fixed in 5.x)
- Had to port SGE, Ganglia, OpenMPI
- No parallel debugger

Processor Architecture

Trade offs and Considerations

- Cost
- Power consumption
- Heat production
- Performance: Integer, floating point, cache size and latency, memory bandwidth and latency, addressable memory
- Software Support: Operating system, hardware drivers, applications (libraries), development tools

Processor Architecture

Options

- IA32 (i386): AMD, Intel, Transmeta, Via
- AMD64 (EM64T): AMD, Intel
- IA64 (Itanium)
- SPARC
- PowerPC
- Power
- Alpha
- MIPS
- ARM

Processor Architecture

What we did on fellowship

- Intel Pentium III's for the first 86
- Intel Xeons for the next 76
- AMD Opterons for the most recent purchases (169)
- Retired Pentium III's this year

How it worked

- Pentium III's gave good service
- Xeons and Opterons performing well
- Considering 64-bit mode for the future
- Looking at Intel Woodcrest CPUs

Network Interconnects

Trade offs and Considerations

- Cost: NIC, cable, switch ports
- Performance: throughput, latency
- Form factor: cable management and termination
- Standardization: commodity vs proprietary
- Available switches: size, inter-switch links
- Separation of different types of traffic

Network Interconnects

Options

- 10/100 Ethernet
- Gigabit Ethernet
- 10 Gigabit Ethernet: fast
- Infiniband: fast, low latency
- 10 Gb Myrinet: fast, low latency
- Others: Dolphin, Fiber Channel

Network Interconnects

What we did on Fellowship

- Gigabit Ethernet
- One rack of 2Gbps Myrinet nodes

How it worked

- Gigabit Ethernet is now the default option for clusters
- Fast enough for most of our applications
- Some applications would like lower latency
- Looking at 10GbE and 10Gb Myrinet

Storage

Trade offs and Considerations

- Cost
- Capacity
- Throughput
- Latency
- Locality
- Scalability
- Manageability
- Redundancy

Storage

Options

- Local Disk
- Protocol Based Network Storage: host or NAS appliance based
- Storage Area Network
- Clustered Storage

Storage

What we did on fellowship

- Host based NFS for home directories, node roots, and some software
- Local disks for scratch and swap
- Moved home directories to a Netapp in 2005

How it worked

- NFS is scaling fine so far
- Enhanced Warner Losh's diskprep script to keep disk layouts up to date
- Users keep filling the local disks
- Disk failures are a problem

Form Factor

Trade offs and Considerations

- Cost
- Maximum performance
- Maintainability
- Cooling
- Peripheral options
- Volume (floor space)
- Looks

Form Factor

Options

- PCs on shelves
- Rackmount system
  - Cabinets
  - 4-post racks
  - 2-post racks
- Blades

Form Factor

What we did on fellowship

- 1U nodes in 2-post racks
- Core equipment in short 4-post racks
- 6 inch wide vertical cable management with direct runs from the switch in first row
- Moved to 10 inch wide vertical management in second row and patch panels in both rows
- Now installing new core equipment in cabinets

Form Factor

Form Factor

How it worked

- Node racks are accessible and fairly clean looking
- Patch panels, 10 inch cable management, and some custom cable lengths helped
- Short 4-post racks didn't work well for real servers
- Watch out for heavy equipment!

Facilities

Trade offs and Considerations

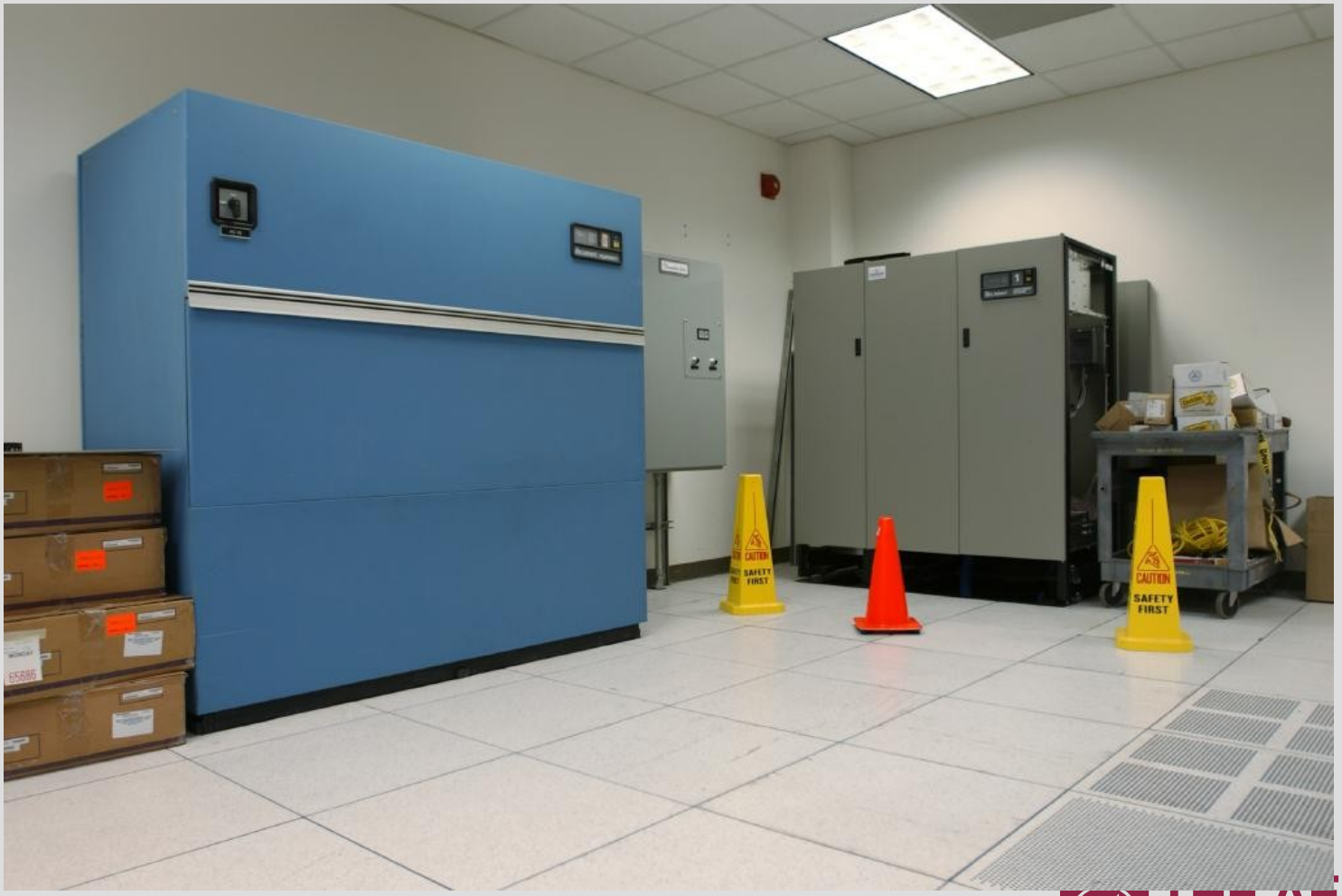
- Cost: space, equipment, installation
- Construction time
- Reliability

Facilities

Options

- Plug it in and hope
- Convert a space (office, store room, etc)
- Build or acquire a real machine room
- Use an old mainframe room

Facilities



THE AEROSPACE CORPORATION

Facilities

What we did on Fellowship

- Built the cluster in our existing 15,000 sq ft. underground machine room
  - 500KVA building UPS and two layers of backup generators
- New UPS and power distribution units (PDUs) being installed for expansion

Facilities

How it worked

- Good space with plenty of cooling
- Power was initially adequate, but is becoming limited
  - Adding a new UPS and PDUs
- Cooling issues with new UPS
- Remote access means we don't have to spend much time there

Scheduling Scheme

Trade offs and Considerations

- Cost
- Efficiency
- Support of policies
- Fit to job mix
- User expectations

Scheduling Scheme

Options

- No scheduler
- Custom or application specific scheduler
- Batch job system
- Time sharing

Scheduling Scheme

What we did on Fellowship

- None initially
- Tried OpenPBS (not stable 4 years ago, no experience since)
- Ported Sun Grid Engine (SGE) 5.3 with help from Ron Chen
- Switched to SGE 6 and mandated use in January

Scheduling Scheme

How it worked

- Voluntary adoption was poor
- Forced adoption has gone well
- Users have preconceived notions of computers that don't fit reality with batch schedulers
- We have modified SGE to add features missing from the FreeBSD port with good success

Operational Issues

- Building, Refresh and Upgrade Cycle
- User Configuration Management
- System Configuration management
- Monitoring
- Inventory Management
- Disaster Recovery

Initial Build, Refresh and Major Upgrade Cycle

Trade offs and Considerations

- Startup cost
- Ongoing cost
- Homogeneity vs Heterogeneity
- Gradual migration vs abrupt transitions

Initial Build, Refresh and Major Upgrade Cycle

Options

- Build
  - Build all at once
  - Gradual buildup
- Refresh
  - Build a new cluster before retirement
  - Build a new cluster in the same location
  - Replace parts over time
- Upgrades
  - Upgrade everything at once
  - Partion and gradually upgrade
  - Never upgrade

Initial Build, Refresh and Major Upgrade Cycle

What we did on Fellowship

- Build
  - Gradual buildup of nodes
  - Periodic purchase of new core systems for expansion and replacement
- Refresh
  - Replaced PIII's this year
  - Xeons to be replaced next year if we don't expand to a third row
- Upgrades
  - Minor OS upgrades in place
  - FreeBSD 4 to 6 and SGE 5 to 6 by partitioning

Initial Build, Refresh and Major Upgrade Cycle

How it worked

- Build
  - Most of our apps don't care
  - Different machines had different exposed serial ports which caused a problem for serial consoles
- Refresh
  - Rapid failures of Pentium III's were unexpected
- Major Upgrades
  - Partitioning allowed a gradual transition
  - New machines offered incentive to move
  - Node locked SSH keys and licenses caused problems

System Testing

Trade offs and Considerations

- Need to validate system stability and performance
  - LLNL says: “bad performance is a bug”
- “Bad batches” of hardware happen
- Lots of hardware means the unlikely is much more common

System Testing

Options

- Leave it to the vendor
- Have a burn-in period
  - No user access
  - Limited user access
- Periodic testing

System Testing

What we did on Fellowship

- Vendor burn in
  - Increasingly strict requirements to ship
- Let users decide where to run (prior to mandatory scheduling)
- Scheduler group of nodes needing testing
- Working on building up a set of performance and stress tests

System Testing

How it worked

- Ad hoc testing makes problems surprising too often
- Users find too many hardware issues before we do
- Group of nodes is easy to administer

System Configuration Management

Trade offs and Considerations

- Network Scalability
- Administrator Scalability
- Packages vs custom builds
- Upgrading system images vs new, clean images

System Configuration Management

Options

- Maintaining individual nodes
- Push images to nodes
- Network booted with shared images
  - Read only
  - Copy-on-write

System Configuration Management

What we did on Fellowship

- PXE boot node images with automatic formatting of local disks for swap and scratch
- Upgraded copies of the image in 4.x
- Building new images for each upgrade in 6.x

How it worked

- Great overall
- A package build system to help keep frontend and nodes in sync would be nice
- Network bottle neck does not appear to be a problem at this point

User Configuration Management

Trade offs and Considerations

- Maintainability
- User freedom and comfort
- Number of supported shells

User Configuration Management

Options

- Make users handle it
- Use /etc/skel to provide defaults and have users do updates
- Use a centrally located file that users source
- Don't let users do anything

User Configuration Management

What we did on Fellowship

- /etc/skel defaults plus users updates to start
- Added a central script recently
  - This script uses an sh script and some wrapper scripts to work with both sh and csh style shells
- Planning a manual update

How it worked

- Bumpy, but improving with the central script

Monitoring

Trade offs and Considerations

- Cost
- Functionality
- Flexibility
- Status vs alarms

Monitoring

Options

- Cluster management systems
- Commercial network management systems: Tivoli, OpenView
- Open Source system monitoring packages: Big Sister, Ganglia, Nagios
- Most schedulers
- SNMP

Monitoring

What we did on Fellowship

- Ganglia early on
- Added Nagios recently
- SGE

How it worked

- Ganglia provides very user friendly output
  - Rewrote most of FreeBSD support
- Nagios working well
- Finding SGE increasingly useful

Disaster Recovery

Trade offs and Considerations

- Cost up front
- Cost of recovery
- Time to recovery
- From what type of disaster
  - Hardware failure
  - Loss of building
  - Data contamination/infection/hacking

Disaster Recovery

Options

- Do nothing
- Local backups
- Off site backups
- Geographically redundant clusters
  - Transparent access to multiple clusters

Disaster Recovery

What we did on Fellowship

- Local backups (Bacula, formerly AMANDA)
- Working toward off site backups

How it worked

- No disasters yet
- Local backups are inadequate
- Looking at a second cluster
- Investigating transparent resource discovery and access

Other Issues

- Virtualization
- System Naming and Addressing
- User Access
- Administrator Access
- User Training and Support
- Inventory Management

Thoughts on a Second Cluster

- We are planning to build a second, similar cluster on the east coast
- Looking at blades for density and maintenance
- Interested in higher speed, lower latency interconnects for applications which can use them
- Considering a completely diskless approach with clustered storage to improve maintainability and scalability

FreeBSD Specifics

- Diskless booting
  - Image creation
  - Disk initialization
- Using ports on a Cluster
- Ganglia demo
- SGE installation and configuration demo

Diskless Booting: Image Creation

- Hacked copy of nanobsd Makefile
  - Removed flash image support
  - Added ability to create extra directories for use as mount points
  - Build a list of ports in the directory via chroot
    - Ports directory created with portsnap
    - Ports are built using portinstall in a chroot
    - Mount linprocfs before every chroot and unmount it afterward
    - Distfile pre-staging is supported for non-redistributable distfiles and faster rebuilds
    - Packages are also supported
    - DESTDIR support in ports will eventually make this obsolete

Diskless Booting: Image Creation

TODO

- Switch to nanobsd scripts (in place of obsolete Makefiles)
- Handle sudoers file in images
  - Copy on in place after install, extend rc.initdiskless /conf support to /usr/local/etc, or add the ability to override in port
- Find a way to keep packages in sync between nodes and front end systems

Diskless Booting Startup Process

- PXE boot with NFS root
- /etc/rc.initdiskless initializes /etc from data in /conf (mounted from /../conf to allow sharing)
  - /conf/base/etc remounts /etc
  - /conf/default/etc includes rc.conf which simply sources rc.conf.{default,bcast,ipaddr} allowing configuration to live in the right place
- /etc/rc.d/diskprep creates swap, /tmp, and /var and labels them to fstab stays consistant reguardless of disk configuration
- Normal boot from this point on

Diskless Booting: Disk Initalization

- Use sysutils/diskprep port (modified version of Warner Losh's tool for embedded deployments
  - If the right GEOM volume label doesn't exist, reconfigure the disk
- Could be improved
  - Reboot during initalization is often fatel
  - Better control of fsck at boot would be useful
    - Option to newfs file systems who's contents we don't care about
  - Alternate superblock printout in newfs too noisy

Using Ports on a Cluster

- Very good for languages and cluster tools
- Unusable for MPI ports due to the need for different ones with different compilers
  - Need a bsd.mpi.mk
- Mixed for libraries
  - Some are fine with one compiler but others could benefit from more than one version, particularly Fortran code
- Hard to keep nodes and front ends in sync
  - Need an SGE based package build system :-)

Using Ports on a Cluster

Useful Ports

- lang/gcc*, lang/icc, lang/ifc, etc.
- net-mgmt/nagios
- sysutils/diskprep
- sysutils/ganglia-monitor-core
- sysutils/ganglia-webfrontend
- sysutils/sge

Diskless Node Demo

- Building a Node Image
- Booting a Diskless Node
- Diskless Configuration
- Installing ports

Building a Node Image

```
make buildworld
make KERNCONF=SOEKRIS buildkernel
CLUSTER_ROOT=/usr/roots/
make DESTDIR=${CLUSTER_ROOT} \
  installworld
make DESTDIR=${CLUSTER_ROOT} \
  distribution
make DESTDIR=${CLUSTER_ROOT} \
  KERNCONF=SOEKRIS installkernel
```

Serial Console Changes

- /boot/loader.conf:

```
boot_multicons="YES"
boot_serial="YES"
console="comconsole"
comconsole_speed="57600"
```

- /etc/ttys:

```
47c47
< ttyd0 "/usr/libexec/getty std.9600"   dialup  off secure
---
> ttyd0 "/usr/libexec/getty std.9600"   vt100   on secure
```

Assorted Mountpoints

```
mkdir -p ${CLUSTER_ROOT}/usr/ports
mkdir -p ${CLUSTER_ROOT}/usr/home
ln -s /usr/home ${CLUSTER_ROOT}/home
```

Booting a Diskless Node

- Servers required
  - DHCP (or bootpd)
  - TFTP (via inetd)
  - NFS

DHCP Configuration

- /etc/rc.conf:

```
dhcpd_ifaces="demo-cluster"
dhcpd_enable="YES"
```

- ${LOCALBASE}/etc/dhcpd.conf:

```
# This goes in a subnet, host, or group block
server-name "coredump";
next-server 10.1.0.1;
server-identifier 10.1.0.1;
filename "varsym/boot/pxeboot";
option root-path "/usr/roots/demo-cluster";
```

TFTP Configuration

- /etc/rc.conf:

```
inetd_enable="YES"
inetd_flags="-a 10.1.0.1"
```

- /etc/inetd.conf:

```
tftp dgram udp wait root
  /usr/libexec/tftpd \
     tftpd -l -u nobody -s /usr/roots
```

NFS Configuration

- /etc/rc.conf:

```
nfs_server_enable="YES"
```

- /etc/exports:

```
/usr -alldirs -ro -maproot=root \
    -network 10.1.0.0 -mask 255.255.0.0
```

- or use ZFS

Diskless Configuration Overview

- `/etc/rc.initdiskless` uses `/conf` to override the contents of directories in `/`
  - Mostly used for `/etc`
- Mostly documented in a large comment at the top
  - Some features are not documented
- Some documentation in `diskless(8)`
- Beware: not all the documentation is correct

Diskless Configuration How it Works

- Warning: highly simplified
- An md(4) (aka MFS) file system is created for /etc
- For each of the directories `/conf/base/etc`, `/conf/default/etc`, `/conf/<node_bcast_address>/etc`, `/conf/<node_ip_address>/etc`:
  - if `${dir}/diskless_remount` mount the NFS path in the file over the top of the directory
  - Copy the contents in to the md(4) file system

Diskless Configuration Shared /conf

- Simplifies images upgrades

```
mkdir ${CLUSTER_ROOT}/conf
echo "/../conf" \
    ${CLUSTER_ROOT}/conf/diskless_remount
```

- Actual /conf in ${CLUSTER_ROOT}/../conf

Diskless Configuration (/conf) Interesting Files

- `base/etc/diskless_remount`:
  `/etc`
- `default/etc/fstab` (Dump and Pass fields not shown):

```
# Device                Mountpoint FStype  Options
# No / entry, it's unnecessary
10.1.0.1:/usr/home /usr/home  nfs      ro,bg,tcp
```

- `default/etc/ttys`
  - Override defaults if you use serial or firewire consoles
- `default/etc/sysctl.conf`
  - Set alternate limits, etc

Diskless Configuration (/conf) Interesting Files

- default/etc/rc.conf:

```
if [ -r /etc/rc.conf.default ]; then
        . /etc/rc.conf.default
fi
if [ -r /etc/rc.conf.bcast ]; then
        . /etc/rc.conf.bcast
fi
if [ -r /etc/rc.conf.ip ]; then
        . /etc/rc.conf.ip
fi
```

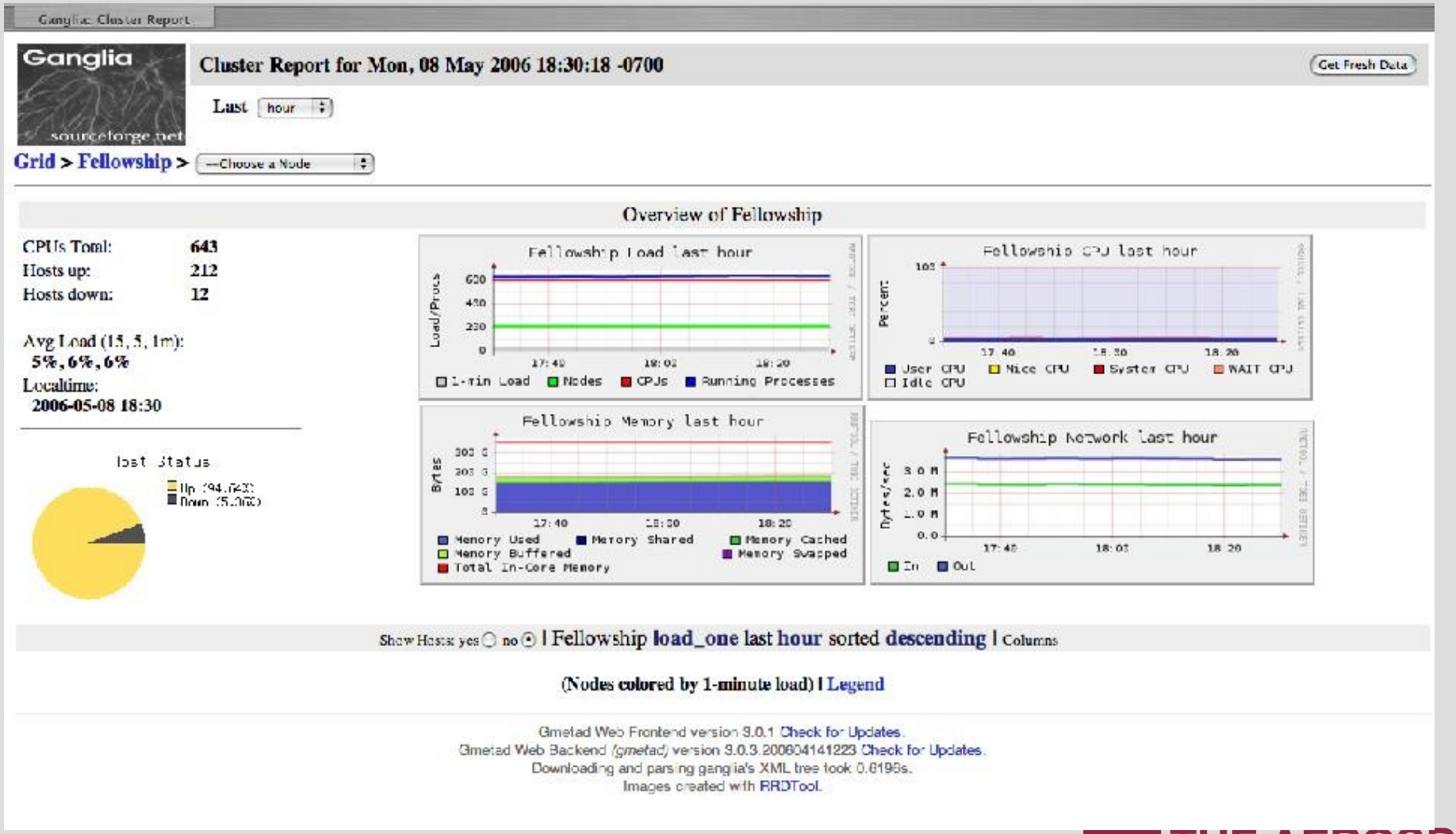
Diskless Configuration (/conf) Interesting Files

- `default/etc/rc.conf.default`:

```
sge_execd_enable="YES"
gmond_enable="YES"
sshd_enable="YES"
```

- `default/etc/ssh/ssh_host*_key*`
  - We use one set of keys for all nodes to simplify `known_hosts` file maintenance
- `default/etc/periodic.conf`:
  - disable mail for non-critical issues and disable expensive operations like updating the locate db

Ganglia Demo



THE AEROSPACE CORPORATION

Ganglia Configuration

- On client and server:
  - `/etc/rc.conf`

    ```
    gmond_enable="YES"
    ```
  - ${LOCALBASE}/etc/gmond.conf
    - defaults work on most systems
- On server
  - `/etc/rc.conf`

    ```
    gmetad_enable="YES"
    ```
  - ${LOCALBASE}/etc/gmetad.conf
    - defaults work on most systems

SGE Configuration

- Prerequisites
  - Physical SGE install
    - port: `sysutils/sge`
  - A shared file system
  - Entries in `/etc/services`
    - `sge_qmaster` and `sge_execd`
    - Default to 6444 and 6445 respectively in upcoming releases (IANA assignments)

Installing SGE qmaster

- `cd /usr/local/sge`
- `./install_qmaster`
  - Generally take the defaults
  - Group id range
    - enter a range of 10-100 unused gids
  - qmaster/scheduler startup script
    - say **no** if using the port
  - Adding admin and submit hosts
    - probably add the local host
  - shadow host
    - probably not needed
- Add `sge_qmaster_enable="YES"` to `/etc/rc.conf`

Installing SGE execd (the official way)

- `cd /usr/local/sge`
- `./install_execd`
  - Generally take the defaults
  - startup script
    - say no if using the port
- Add `sge_execd_enable="YES"` to `/etc/rc.conf`
- Repeat on **every** node...

Installing SGE execd (the scriptable way)

```
!#/bin/sh
HOST=$1
FQDN=${HOST}.cluster.example.com
SGE_CELL=${SGE_CELL-default}
SPOOLDIR=${SGE_ROOT}/${SGE_CELL}/spool/${HOST}

qconf -aattr hostgroup hostlist $FQDN @allhosts
qconf -as ${FQDN}
qconf -ah ${FQDN}

mkdir -p ${SPOOLDIR}
mkdir -p ${SPOOLDIR}/active_jobs
mkdir -p ${SPOOLDIR}/jobs
mkdir -p ${SPOOLDIR}/job_scripts
chown -R sgeadmin ${SPOOLDIR}
```

Adding a Parallel Environment

- Add the PE
  - `qconf -Ap mpich.template`
- Add the PE to the PE list for a queue
  - `qconf -mq`
  - Edit the *`pe_list`* variable

Questions?

- http://people.freebsd.org/~brooks/pub/eurobsdcon2007/eurobsdcon2007-cluster-tutorial.pdf

Disclaimer

- All trademarks, service marks, and trade names are the property of their respective owners.