

Cluster Management

James E. Prewett

October 8, 2008



- Common Management Tools
 - OSCAR
 - ROCKS
 - Other Popular Cluster Management tools
- Software Management/Change Control
 - Cfengine
 - Getting Started with Cfengine
 - Parallel Shell Tools / Basic Cluster Scripting
 - PDSH
 - Dancer's DSH
 - Clusterit
 - C3 tools (cexec)
 - Basic Cluster Scripting
 - Backup Management
 - Logging/ Automated Log Analysis
 - Regular Expression Review

- Regular Expression
 - Meta-characters
 - Regular Expression Meta-characters (cont.)
 - SEC
 - Logsurfer+
- Security plans/procedures, Risk Analysis
 - Network Topologies and Packet Filtering
- Linux Tricks
 - Cluster-specific issues
 - Checking Your Work
- Regression Testing
 - System / Node / Software Change Management Logs
 - How to know when to upgrade, trade-offs
 - Monitoring tools



OSCAR Information

Vital Statistics:	
Version:	5.1
Date:	June 23, 2008
Distribution Formats:	tar.gz
URL:	http://oscar.openclustergroup.org/



OSCAR cluster distribution features:

- ▶ Supports X86, X86_64 processors
- ▶ Supports Ethernet networks
- ▶ Supports Infiniband networks
- ▶ Graphical Installation and Management tools
... if you like that sort of thing



OSCAR (key) Cluster Packages

Whats in the box?

- ▶ Torque Resource Manager
- ▶ Maui Scheduler
- ▶ c3
- ▶ LAM/MPI
- ▶ MPICH
- ▶ OpenMPI
- ▶ OPIUM (OSCAR User Management software)
- ▶ pFilter (Packet filtering)
- ▶ PVM
- ▶ System Imager Suite (SIS)
- ▶ Switcher Environment Switcher



OSCAR Supported Linux Distributions

- ▶ RedHat Enterprise Linux 4
- ▶ RedHat Enterprise Linux 5
- ▶ Fedora Core 7
- ▶ Fedora Core 8
- ▶ Yellow Dog Linux 5.0
- ▶ OpenSUSE Linux 10.2 (x86_64 Only!)
- ▶ “Clones of supported distributions, especially open source rebuilds of Red Hat Enterprise Linux such as CentOS and Scientific Linux, should work but are not officially tested.”



OSCAR Installation

- ▶ Install a supported Linux on the erver Node
Leave at least 4GB free in each of / and /var!
The easy way is to make 1 big partition for / !
- ▶ Create repositories for SystemInstaller

```
# mkdir /tftpboot
# mkdir /tftpboot/oscar
# mkdir /tftpboot/distro
# mkdir /tftpboot/distro/OS-version-arch
```
- ▶ Unpack the oscar-repo-common-rpms and the oscar-repo-DISTRO-VER-ARCH tarballs into /tftpboot/oscar/
- ▶ Copy your RPMs into the /tftpboot/distro/OS-version-arch directory



OSCAR Installation (cont.)

- ▶ Install yum unless your OS already has it
- ▶ Install yume:

```
# yum install createrepo
/tftpboot/oscar/common-rpms/yume*.rpm
```
- ▶ Install oscar-base RPM:

```
# yume --nogpgcheck1 --repo /tftpboot/oscar/common-rpms
install oscar-base
```

¹This is not in the documentation, but I found that the packages were not signed causing yume to barf unless you passed it the --nogpgcheck option.
YMMV



OSCAR Server Node Network Configuration

- ▶ Give your host a hostname! The default of “localhost” or “localhost.localdomain” will *not* work.
- ▶ Configure the “Public” network interface as per the requirements of your local network. This is the network that will connect to the Internet (or the lab network), so configure it appropriately.
- ▶ Configure the “Private” network interface using a “Private” IP address.
 The **IANA** has reserved the following three blocks for private internets:
 - ▶ 10.0.0.0 – 10.255.255.255 (10/8 CIDR block)
 - ▶ 172.16.0.0 – 172.31.255.255 (172.16/12 CIDR block)
 - ▶ 192.168.0.0 – 192.168.255.255 (192.168/16 CIDR block)



OSCAR Cluster Installation

Once the Server is installed and configured, start the installer!

```
# cd /opt/oscar
# ./install_cluster <device>
```

This will:

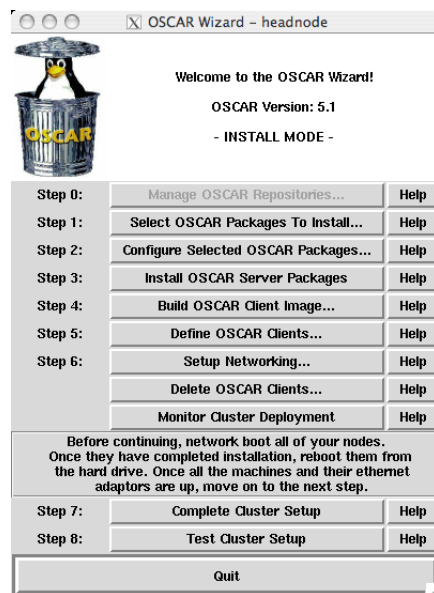
- ▶ Install all required RPMs
- ▶ update the /etc/hosts file with OSCAR aliases
- ▶ update the /etc/exports file
- ▶ update system initialization scripts (/etc/rc.d/init.d/)
- ▶ restart any affected services

Then the installer GUI will be launched.



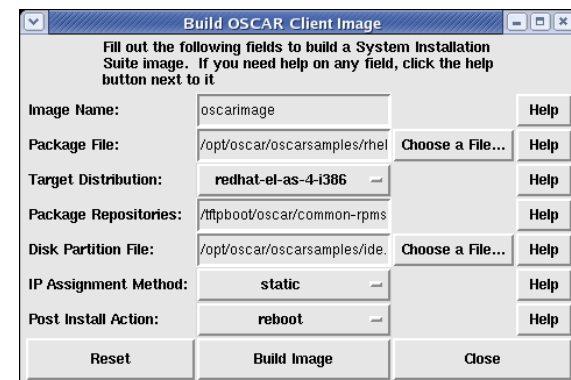
The OSCAR Installation Wizard:

- ▶ Select your packages
- ▶ Configure the packages
- ▶ Install the Server packages
- ▶ Build an image for the compute nodes
- ▶ Define the compute nodes
- ▶ Configure networking
- ▶ Complete the setup
- ▶ Test the cluster!



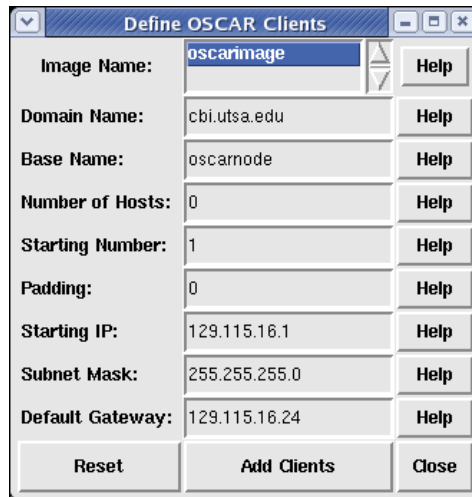
Build Client Image

- ▶ Choose an image name
- ▶ Specify Disk Partition file
- ▶ Choose a package file
- ▶ Pick IP assignment method
- ▶ Choose a Target Distribution
- ▶ Pick Post Install action
- ▶ Specify package repositories



Define OSCAR Clients (Compute Nodes)

- ▶ Pick the image to install
- ▶ Specify the domain name
- ▶ Specify the base hostname
- ▶ Specify the number of hosts
- ▶ Specify first number to append to the base hostname
- ▶ Specify the “padding”
- ▶ Specify the starting IP
- ▶ Specify the subnet mask
- ▶ Specify the default gateway

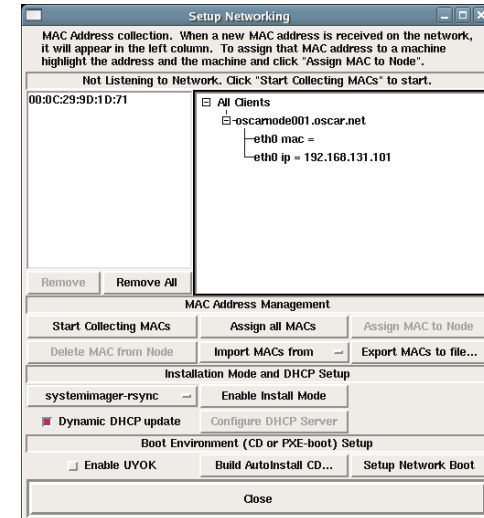


NOTE: You may only define 254 clients at a time!



Setup OSCAR Networking

- ▶ Collect MAC Addresses
- ▶ Optionally tweak SI installation mode
- ▶ Build Boot CD OR
- ▶ Setup Network Boot
- ▶ Optionally choose to Use Your Own Kernel (UYOK)



Finishing Up!

- ▶ Go to “Monitor Cluster Deployment” to monitor the progress of the installation.
- ▶ Reboot the compute nodes.
- ▶ Go to “Complete Cluster Setup”
- ▶ Run the OSCAR Test suite (unless you’re feeling brave!)
- ▶ Enjoy your new cluster!



Really, Its *that* simple!

- ▶ OSCAR comes with quite a few “standard” cluster packages.
- ▶ OSCAR uses SystemImager
- ▶ SystemImager is Good™
- ▶ RPM packages may be added by placing them in the appropriate directory, rebuilding the image, and rebooting the nodes.



ROCKS Information

Vital Statistics:	
Version:	5.0
Date:	November 12, 2006
New development:	September 2008
Distribution Formats:	tar.gz
URL:	http://oscar.opencluster.org/

ROCKS cluster distribution features:

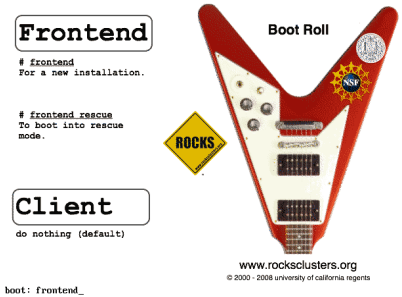
- ▶ Supports X86, X86_64 processors
- ▶ Supports Ethernet networks
- ▶ Supports Specialized networks and components (Myrinet, Infiniband, nVidia GPU)

Beginning the ROCKS Installation

For the Installation, you will need:

- ▶ Kernel/Boot Roll CD
- ▶ Base Roll CD
- ▶ Web Server Roll CD
- ▶ OS Roll CD - Disk 1
- ▶ OS Roll CD - Disk 2 OR
- ▶ ALL Red Hat Enterprise Linux 5 update CDs
- ▶ ALL CentOS 5 update 1 CDs
- ▶ ALL Scientific Linux 5 update 1 CDs

- ▶ Boot the "Kernel/Boot Roll CD" on the server
- ▶ You should see:



- ▶ Type "front-end" to begin the installation

Other Popular Cluster Management tools

- ▶ Xcat
- ▶ openMosix (RIP March 1, 2008)
- ▶ LinuxPMI Continuation of 2.6 branch of openMosix (***NOT*** Single System Image)
- ▶ OpenSSI
- ▶ Scyld
- ▶ IBM's CSM
- ▶ Also notable: Sandia's CIT²

²It may not be the most popular, but it is well designed and pretty darn cool!

What is “Change Control” ?

- ▶ Automatically manage configuration files
- ▶ Take care of maintenance tasks like running backups
- ▶ Manage things like “cron jobs” in a centralized place.

... Automate and reduce the headache of administration!

Cfengine Information

Vital Statistics:	
Version:	2.2.8
Date:	August 5, 2008
Distribution Formats:	tar.gz
URL:	http://www.cfengine.org/

What is Cfengine good for?

- ▶ Ensure proper versions of software are installed
- ▶ Template-based creation of configuration files
- ▶ Verify permissions & ownership of files and directories
- ▶ Standardize properties (netmask, domain name, etc.) of hosts
- ▶ Ensure checksums of files
- ▶ Check disk capacity

Installing Cfengine

- ▶ `tar zxf cfengine-2.2.8.tar.gz`
- ▶ `cd cfengine-2.2.8`
- ▶ `./configure`
- ▶ `make`
- ▶ `make install`
- ▶ `test: /usr/local/sbin/cfagent -v`

Getting Started with Cfengine

In order to get started with Cfengine, we will need 3 things:

- ▶ A crontab entry to run cfexecd periodically³
0 * * * * /usr/local/sbin/cfexecd -F
- ▶ An update.conf file
- ▶ A cfagent.conf file

³Cfengine can also be run as a daemon.

update.conf — control section

```
#####  
#  
# Distribute the configuration files  
#  
#####  
  
control:  
# distribute the files, then clean up our mess  
workdir = ( /var/cfengine )  
actionsequence = ( copy tidy )  
policyhost = ( cfengine.hpc.unm.edu ) # master host  
domain = ( hpc.unm.edu )  
master_cfinput = ( /cfengine/inputs )  
sysadmin = root@hpc.unm.edu
```

cfagent.conf — control section

```
control:  
domain = ( hpc.unm.edu )  
netmask = ( 255.255.252.0 )  
sysadm = ( root@hpc.unm.edu )  
timezone = ( MST )  
actionsequence = (  
    mountall      # mount filesystems in /etc/fstab  
    netconfig     # check the network interface  
    resolve       # check the DNS resolver  
    tidy          # ‘tidy’ Cfengine logfiles  
    files         # check file permissions  
    directories  # ensure directories exist  
    processes )  # check processes
```

cfagent.conf — files and directories section

```
# check important files  
files:  
    /etc/passwd          mode=644 owner=root action=fixall  
    /etc/shadow          mode=600 owner=root action=fixall  
    /var/spool/torque/pbs_environment mode=644 owner=root action=fixall  
    /var/spool/torque/server_name mode=644 owner=root action=fixall  
#check that TORQUE directories exist  
directories:  
    /var/spool/torque/          owner=root mode=755 action=fixall  
    /var/spool/torque/aux/      owner=root mode=755 action=fixall  
    /var/spool/torque/mom_logs/ owner=root mode=755 action=fixall
```

(etc.)

cfagent.conf — processes section

```
# Here we define processes we want to ensure are running
# We could also define ones we wanted to kill or restart
# Strings are regular expressions used to match the name
# of the process
processes:
  "pbs_server" matches=1 # ensure PBS is running
  "maui"       matches=1 # ensure Maui is running
```



Popular Parallel Shells

- ▶ PDSH
- ▶ Dancer's DSH
- ▶ Clusterit
- ▶ C3 tools



PDSH Information

Vital Statistics:	
Version:	2.16
Date:	April 3, 2008
"Parallelism":	"sliding window" parallel algorithm
Language:	C
Distribution Formats:	RPM, tar.gz
URL:	https://computing.llnl.gov/linux/pdsh.html



PDSH Remote command modules

These are ways of accessing the remote nodes. Tune as per your security/performance requirements!

- ▶ RSH
- ▶ SSH
- ▶ Kerberos
- ▶ MRSB, QSH, MQSH, XCPU (whatever those are ;)



PDSH Node Specification

- ▶ Specify a list of hosts:
`pdsh -w node01,node05,node17 -- command`
- ▶ specify a range of hosts:
`pdsh -w node01-node100 -- command`
- ▶ Specify a range of hosts, excluding a set in the middle:
`pdsh -w node01-node100 -x node20-node30 -- command`

PDSH Node Specification (cont.)

- ▶ Specify a nodes in a netgroup “netgroup”:
`pdsh -g netgroup -- command`
- ▶ Exclude nodes in the netgroup “netgroup”:
`pdsh -X netgroup -- command`
- ▶ Execute a command on all nodes in a file:
`export WCOLL=/path/to/node-file`
`pdsh -- command`

Dancer's DSH Information

Vital Statistics:	
Version:	0.25.9
Date:	August 15, 2007
“Parallelism”:	“Hierarchical invocation technique” “4 nodes accessing 4 nodes” ...
Language:	C
Distribution Formats:	DEB, .tar.gz
URL:	http://www.netfort.gr.jp/~dancer/software/dsh.html.en

Dancer's DSH Node Specification

- ▶ Use the global nodes file, `/etc/dsh/machines.list`:
`dsh -a -c -- command`
- ▶ Use the list of nodes for “Rack 1” stored in `$HOME.dsh/group/rack1`
`dsh -g rack1 -c -- command`

Clusterit Information

Vital Statistics:	
Version: 2.5	
Date:	August 15, 2007
“Parallelism”:	N-way Fanout
Language:	C
Distribution Formats:	.tar.gz
URL:	http://clusterit.sourceforge.net/

Clusterit Node Specification (Groups and Lumps)

- ▶ Groups are sets of nodes:
 - ▶ GROUP:compute
node01
node02
- ▶ Lumps are sets of groups:
 - ▶ LUMP:cluster
compute
storage
admin

Clusterit Node Specification

- ▶ Specify a list of hosts:
dsh -w node01,node04,node23 -- command
- ▶ Exclude a list of hosts:
dsh -x node03,node09,node17 -- command
- ▶ Specify a group of hosts:
export CLUSTER=/path/to/nodefile
dsh -g compute -- command
- ▶ Specify a lump of hosts:
export CLUSTER=/path/to/nodefile
dsh -g cluster -- command

C3 Information

Vital Statistics:	
Version:	4.0.1
Date:	July 15, 2003
“Parallelism”:	“Sub-Cluster Staging”
Language:	Python
Distribution Formats:	RPM, .tar.gz
URL:	http://www.csm.ornl.gov/torc/C3/C3softwarepage.shtml

C3 Cluster Node Specification file format

/etc/c3.conf

- ▶ Specify a cluster with a head node with an external interface named “external-name” and an internal interface named “node0” and 64 compute nodes named node01-node64.

- ▶ */etc/c3.conf contents:*
cluster my-cluster
{
external-name:node0 #head node
node[1-64] #compute nodes
}

C3 Node Specification

- ▶ Specify the default cluster:
cexec command
- ▶ Specify a subset of nodes in the default cluster:
cexec :6-53 command
- ▶ Specify a list of clusters:
cexec cluster1: cluster2: command

Basic Cluster Scripting

grep is your (best) friend

- ▶ Find the CPU count on all of the nodes:
pdsh “cat /proc/cpuinfo | grep processor | wc -l”
- ▶ Find nodes with the wrong image version:
export VER=”1.2.3”
pdsh “cat /etc/image_version | grep \ “^\$VER\$” || hostname”

More Basic Cluster Scripting

awk is a pretty good friend too!

- ▶ Find nodes where the load is greater than 2:
pdsh uptime | awk ‘{if(\$11 > 2.0){print}}’
- ▶ Find bad GM counts on all nodes:
pdsh “/opt/mx/bin/mx_counters |
awk ‘/bad/ {if (\ \$2 > 0) {print;}}’ ”

Backup anything you can't recreate

Backup anything you can recreate but can't recreate quickly

- ▶ Use backup anytime it would take longer to rebuild and reconfigure than to restore.
- ▶ “Longer” may be in terms of staff time or elapsed time or both.
- ▶ Consider:
 - ▶ User directories (not scratch!)
 - ▶ Libraries and applications you've built on site
 - ▶ Tcl module files in /usr/share/modules/modulefiles/
 - ▶ System configuration files DNS, DHCP, NIS, etc.
(Should that be everything in /etc/?)
 - ▶ Node images

Thanks to Roy Heimbach for contributing this slide!

What can we find in our logfiles?

What are we happily ignoring?

- ▶ Evidence of misconfigurations:
e.g. “/var/log/lastlog does not exist”
- ▶ Security violations
e.g. Illegal users
- ▶ Hardware/Software errors e.g. Disk failures

Logging/Automated Log Analysis Tools:

- ▶ SEC
- ▶ Logsurfer+
- ▶ splunk

Regular Expression Review

Is that line noise?

This is a quick review of [Perl Regular Expressions](#).

- ▶ Simple 'as-is' text string matching:
- ▶ “cat” or “dog”
- ▶ Meta-characters:
▶ `{ } [] () ^ $. | * + ? \`

Regular Expression Meta-characters

- ▶ . matches any single character
- ▶ * match the previous thing 0 or more times
- ▶ + match the previous thing 1 or more times
- ▶ ? match the previous thing 1 or 0 times
- ▶ ^ matches the beginning of the line
- ▶ \$ matches the end of the line
- ▶ \ 'escapes' the next character
- ▶ [] specifies a set or range of characters:
eg. [a-z,A-Z,0-9] would match all alphanumeric characters

Regular Expression Meta-characters (cont.)

- ▶ {n} match the previous thing exactly "n" times
- ▶ {n,} match the previous thing at least "n" times
- ▶ {n,m} match the previous thing at least "n" times, but not more than "m" times
- ▶ () specifies groups of things or things to "save"
the first group will be saved in \$1, the second in \$2, etc.
- ▶ | specifies "OR" inside of a group
eg. (cat|dog) would match either "cat" or "dog"

SEC Information

Vital Statistics:	
Version:	2.4.2
Date:	February 1, 2008
Language:	Perl
Distribution Formats:	.tar.gz, DEB, RPM, FreeBSD and OpenBSD ports, Gentoo portage
URL:	http://www.estpak.ee/~risto/sec/

Quick intro to SEC: SEC Components

- ▶ Messages
Single lines of text in a logfile
- ▶ Rules
Do something in response to an incoming Message
- ▶ Contexts
Passive structures to store Messages

Default SEC Rule

Match all messages and print them

```
# Print all messages
type=single
ptype=regex
pattern=.+
desc=unmatched message: $0 # note $0 is the entire message
action=logonly
```

This, or something like it, should be the last rule in your ruleset



SEC Filtering Rule

Ignore messages we're expecting

```
# This machine has 4 processors
# Ignore messages reporting what we expect!
type=single
ptype=RegExp
pattern=kernel: Total of 4 processors activated
desc=correct processors initialized
action=none
```



SEC Responding to messages

Sound the alert!

```
# This machine has 4 processors
# Report any number other than that!
# report_problem.sh is a script we wrote to report this
# to our admins
type=single
ptype=RegExp
pattern=(\S+) kernel: Total of (\d+) processors activated
desc=incorrect processor count: $2 on host: $1
action=shellcmd report_problem.sh $1 $2
```



SEC Contexts and Correlation

Finding, Blocking, and Reporting on "SSH scanners"

```
# Store "Invalid user" messages from this host unless we're blocking it
type=single
continue = TakeNext
desc = invalid login from host $2
ptype=regex
pattern = ^\S+\s+\S+\s+\S+\s+\S+\s+sshd\[d+\]: Invalid user (\S+) from (\S+)$
context = (!(block_bad_ssh-$2))
action=add bad_ssh-$2

# Block the host if we've gotten 10 "Invalid user" messages in a day
type=SingleWithThreshold
desc = invalid login from host $2
ptype=regex
pattern = ^\S+\s+\S+\s+\S+\s+\S+\s+sshd\[d+\]: Invalid user (\S+) from (\S+)$
thresh=3
action=create block_bad_ssh-$2; \
    shellcmd iptables -A INPUT --source $2 -j REJECT ; \
    report bad_ssh-$2 /usr/adm/bin/report-bad-host.pl $2 ; \
    delete bad_ssh-$2
window=1000000
```



Logsurfer+ Information:

Vital Statistics:	
Version:	1.7
Date:	December 2006
Language:	C
Distribution Formats:	.tar.gz
URL:	http://www.crypt.gen.nz/logsurfer/

System and Cluster Security!

Watch Out!

- ▶ Identify the Problem
- ▶ Security Strategies
- ▶ Dealing with Weaknesses
- ▶ Cluster Network Topologies
- ▶ Cluster Specific Issues
- ▶ Linux Tricks
- ▶ Checking Your Work

Define the Enemy

- ▶ Data thieves
- ▶ Resource thieves
- ▶ Hackers there for various reasons
- ▶ Curies script kiddies
- ▶ Malicious script kiddies

Attack Vectors

- ▶ Remote Attacks:
Network Services allow access to the machine
- ▶ Local Attacks:
Insecure Priveledged Binaries allow Priveledge escalation

Security Strategies

... besides cutting the wire

- ▶ Secure Communication
- ▶ Hunt and kill unneeded services
- ▶ Application configuration
- ▶ Protective Mechanisms

Identifying Weaknesses

The key here is to strike a balance between security and useability

- ▶ Identify and categorize running services
Are they *Really* needed?
- ▶ Identify sensitive information
Passwords, Data, etc.
- ▶ Identify protective mechanisms
TCPwrappers, iptables, firewall, etc.

Limiting Weaknesses

- ▶ Local weaknesses:
 - ▶ Limit use of installed privledged binaries
 - ▶ Removed setuid/setgid bits
 - ▶ If you don't use it, get rid of it!
- ▶ Remote weaknesses:
 - ▶ Close unused ports
 - ▶ Limit access to ports
 - ▶ If you don't use it, get rid of it!

Finding services

They *can't* hide!

- ▶ inetd(8) and xinetd(8) configuration files
- ▶ chkconfig(8)
- ▶ init(8) scripts
- ▶ ps(1)
- ▶ lsof(8) -i
- ▶ nmap(1)

Protecting a single machine with IPtables

We're *not* doing NAT

- ▶ `iptables -A INPUT -m state ESTABLISHED,RELATED -j ACCEPT`
- ▶ `iptables -A INPUT -p tcp --destination-port ssh -j ACCEPT`
- ▶ `iptables -A INPUT -j REJECT`



Protecting a network with IPtables

Hiding your cluster behind a NAT

- ▶ `iptables -A INPUT -p tcp --destination-port ssh -j ACCEPT`
- ▶ `iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT`
- ▶ `iptables -A INPUT -i INTERNAL_INTERFACE -m state --state NEW -j ACCEPT`
- ▶ `iptables -A INPUT -j REJECT`
- ▶ `iptables -A FORWARD -j REJECT`



/proc Protections

Turning on network stack security features

- ▶ Prevent address spoofing:
`echo 0 > /proc/sys/net/ipv4/conf/*/accept_source_route`
`echo 1 > /proc/sys/net/ipv4/conf/*/rp_filter`
`echo 1 > /proc/sys/net/ipv4/conf/*/log_martians`
- ▶ Disable ICMP redirects
`echo 0 > /proc/sys/net/ipv4/conf/*/accept_redirects`
- ▶ Turn off bootp packet relaying
`echo 0 > /proc/sys/net/ipv4/conf/*/bootp_relay`
- ▶ Ignore ICMP bad error responses
`echo 1 >`
`/proc/sys/net/ipv4/icmp_ignore_bogus_error_responses`
- ▶ Enable syncookie protection
`echo 1 > /proc/sys/net/ipv4/tcp_syncookies`



Cluster-specific issues

- ▶ System backdoors:
 - ▶ cron
 - ▶ at
- ▶ One user per node guarantee
- ▶ Passwordless authentication



One user per node

... or the right number of users per node

- ▶ Compute nodes should be wholly allocated to the user(s) that the scheduler has given them to
- ▶ Only the scheduler knows who owns the nodes
- ▶ Strategies:
 - ▶ Modify NIS maps
 - ▶ Modify /etc/passwd
 - ▶ PAM modulesWe (UNM HPC) use pam_pbsimpleauth distributed with TORQUE for most of our systems.

RSA vs. DSA (the low-down)

“In DSA, signature generation is faster than signature verification, whereas with the RSA algorithm, signature verification is very much faster than signature generation. ...”

(<http://www.rsasecurity.com/rsalabs/faq/3-4-1.html>)

In a nutshell:

RSA can be used for both encryption and digital signatures.

DSA is strictly a digital signature

Passwordless Authentication

- ▶ Job launch can't require passwords
- ▶ SSH can be used via RSAAuthentication (Public Key)
- ▶ Issues:
 - ▶ Management of host keys
 - ▶ Management of user keys

Checking Your Work

- ▶ nmap — port scanner
- ▶ Nessus — vulnerability scanner
- ▶ Securityfocus.com
 - ▶ Search for your distribution & version
 - ▶ Compare vulnerabilities to services you run
 - ▶ Compare vulnerabilities to setuid/setgid binaries on your system
- ▶ Bugtraq — for the seriously hardcore
The up-and-coming info in the security world

Finding listening services with lsof:

```
lsof shows which network files are open:  
% lsof -i | awk '/LISTEN/ print $1,$(NF-2),$(NF-1)' | sort  
| uniq  
condor_ma TCP service0.nano.alliance.unm.edu:1026  
identd TCP *:auth  
inetd TCP *:ftp  
inetd TCP *:globus-gatekeeper  
inetd TCP *:gsiftp  
inetd TCP *:klogin  
inetd TCP *:kshell  
inetd TCP *:login  
inetd TCP *:netsaint_remote
```



Finding init.d started services:

```
To find the services that will be started by default at the current runlevel  
using /etc/rc.d/init.d scripts:  
# chkconfig --list | grep 'grep :initdefault:  
/etc/inittab | awk -F: 'print $2' ':on | awk 'print $1' |  
sort | column  
atd      isdn      random  
autofs   keytable  reconfig  
condorg  netfs     sendmail  
crond    network   sshd  
globus   nfslock   syslog  
gm       pbs_mom   verifyd
```



Finding Network visible services

Nmap is your friend!

```
To find services visible from the network:  
other-host# nmap host-to-be-looked-at  
Port      State  Service  
21/tcp    open   ftp  
22/tcp    open   ssh  
23/tcp    open   telnet  
111/tcp   open   sunrpc  
113/tcp   open   auth  
513/tcp   open   login  
514/tcp   open   shell  
1026/tcp  open   nterm  
4321/tcp  open   rwhoisw
```



Regression Testing

Making sure stuff still works

Your regression tests should:

- ▶ Check your basic system components and tools
- ▶ Check your network(s)
- ▶ Check your important applications

Jim's Rule:⁴

If the cluster doesn't work for your users, the cluster *doesn't work*!

⁴Jim learned this the hard way!



You're mostly on your own :P

... but its just some shell scripts...

- ▶ You can use tools like Cfengine to automate some of your regression testing
- ▶ Your regression tests should be easy to run
- ▶ Your regression tests should produce a summary of successes and failures — a report at the end.
- ▶ Consider a suite of shell scripts
- ▶ Should the scripts attempt to repair any errors they find? (season to taste!)



System/Node/Software Change Management Logs

- ▶ Change management logs *will* save your backside!
- ▶ System administrators can be sloppy! :P :)
Where did I put that??!
- ▶ Choose a tool that works well for the administrator(s) for the system in question.



Where to keep Change Management Logs?

Somewhere that you will actually keep them!

- ▶ A Wiki of some kind
- ▶ Emacs outline mode is nice!
- ▶ Really, whatever works for you and your staff!
- ▶ I've seen sites alias editor commands in root's environment to require the admin to make a change management log when s/he edits a config file.
- ▶ I won't tell if you're using a plain ASCII text file :)
- ▶ ...but if you do, please consider keeping it under some sort of version control :)



How to know when to upgrade, trade-offs

The Great Balancing Act!

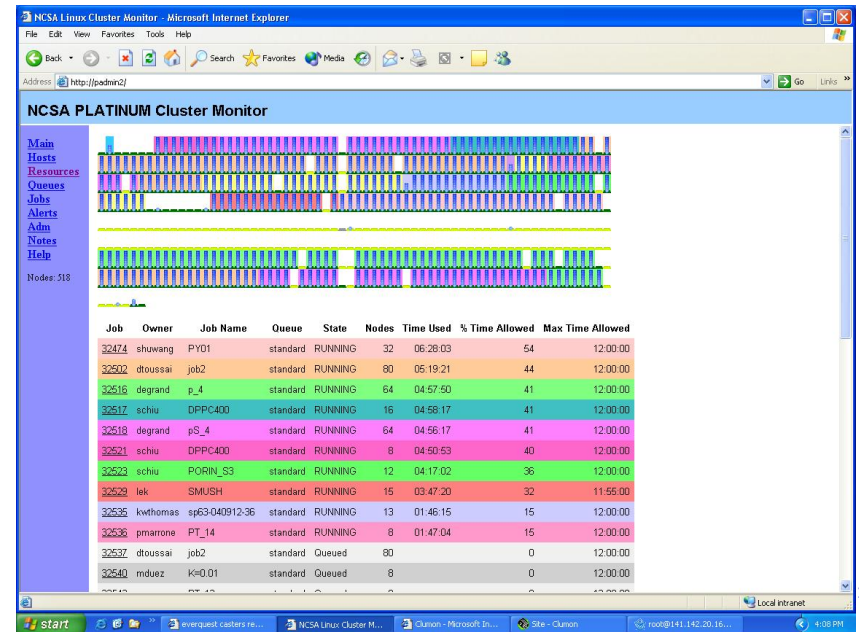
- ▶ Security upgrades
VITAL: if you have security concerns!
*VITAL: if you have ****A NETWORK CONNECTION!*****
- ▶ Required features
 - ▶ Things needed to enhance the useability/stability of the system
 - ▶ Software required by the users
- ▶ Tracking OS development
 - ▶ You don't want to fall **too** far behind
 - ▶ Upgrading several major versions is very painful!
 - ▶ Keep your upgrades *relatively* small
- ▶ Latest development may *not* be what you want!



Clumon Information:

Vital Statistics:	
Version:	2.0 Alpha
Distribution Formats:	RPM, tar.gz
URL:	http://clumon.ncsa.uiuc.edu/

Clumon

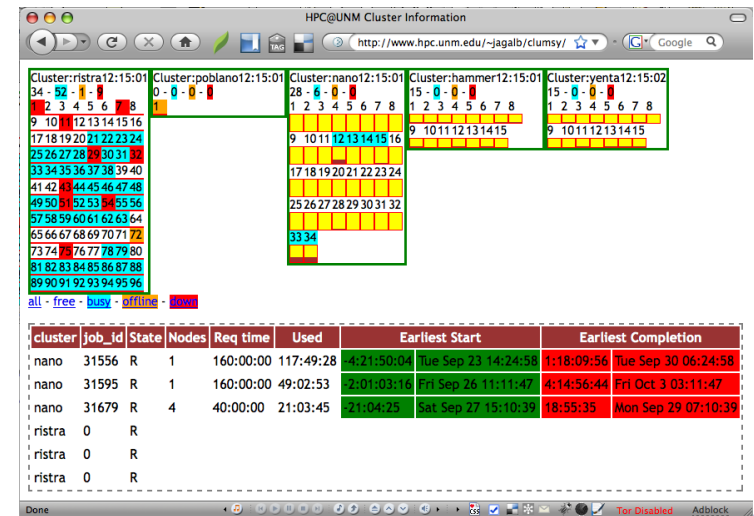


cLUMSy Information:

Vital Statistics:	
Version:	0.0.0
Distribution Formats:	UNRELEASED Bug Jim

cLUMSy

The Lightweight Universal Monitoring System
... a work in progress ...



Ganglia Information:

Vital Statistics:	
Version:	3.1.1
Distribution Formats:	RPM, .tar.gz
URL:	http://ganglia.info/

Ganglia

