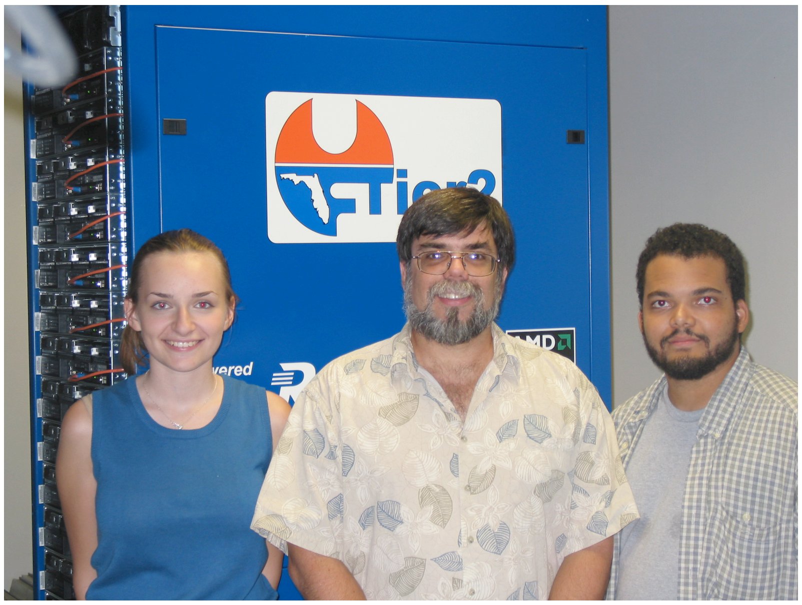
CLUSTER COMPUTING

Undergraduate Research Report



Instructor:
Marcus Hohlmann

Students:
Jennifer Helsby
Rafael David Pena

With the help of:
Jorge Rodriguez, UF

Fall 2006

CLUSTER COMPUTING

Undergraduate Research Report

Jennifer Helsby
Rafael David Pena
Fall 2006

Introduction

Cluster computing is an increasingly popular high performance computing solution. Computing clusters make up over half of the top 500 most powerful computers in the world, and the number of clusters is growing daily. At Florida Tech, we are building a computing cluster in the Physics Department.

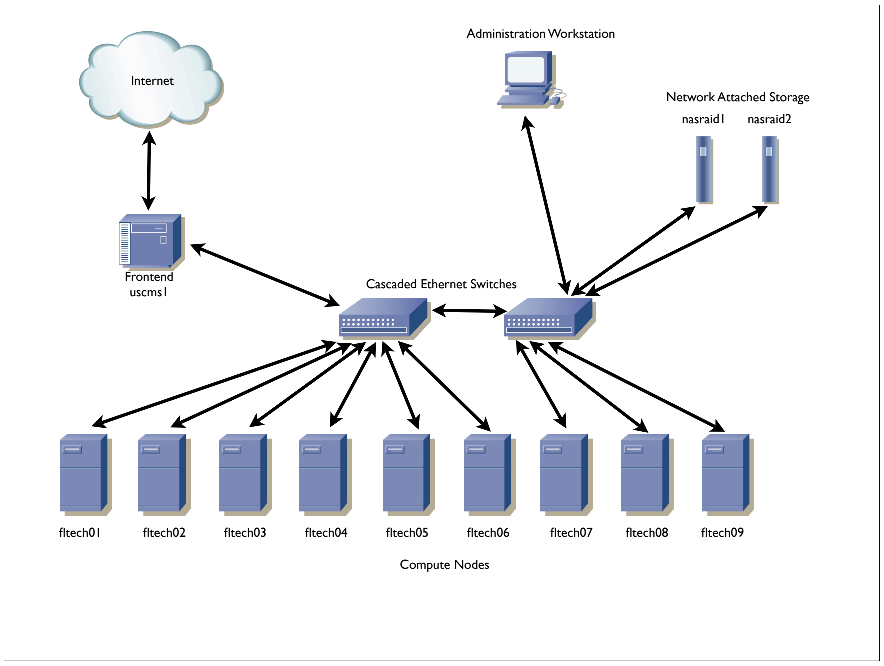
The purposes of this semester's investigations were to learn and implement the processes of creating a fully functional cluster using Rocks 3.3.0 with Condor installed - a batch job system, the end goal being the addition of Florida Tech's cluster to OSG's (Open Science Grid) Integration Test Bed. A trip to UF just before the semester started left us with one frontend and one computational node. We had learned how to customize the Rocks XML files for our purposes.

This document, in addition to the logbook, is to serve as a reference guide to future students whom are to work on this project.

Cluster Architecture

Florida Tech acquired ten computers - Dual CPU Intel Pentium 1.0 GHz servers - from the University of Florida earlier this year. The plan, as shown in the figure on the next page, was to use one of these servers as the frontend, the cluster controller, and the remaining nine as computational nodes. The existing hardware was to be utilized in the cluster, but not as computational nodes. We originally had two brands of motherboards - Asus and Gigabyte - the Asus motherboards are supported by Rocks, while the Gigabyte motherboards caused problems as the version of Rocks we used did not feature support for that motherboard. Thus, the Asus motherboards have been used with the Linux distribution OpenFiler in NAS (Network Attached Storage) servers. The Gigabyte motherboards will be used in workstations. Due to the growth of our cluster, an additional switch had to be purchased to expand the internal cluster network. Ethernet switches can be cascaded if crossover cables are used. This network is not particularly fault tolerant, but the purchase of an enterprise class switch starts at about $100 per port and so is not financially viable. For a ten node cluster, the topology below is appropriate as far as fault tolerance is concerned.

Topology of the cluster (Icons courtesy of Cisco Systems)



ROCKS

Rocks is a revamped CentOS distribution that allows customization using a collection of XML files and software packages. The cornerstone of rocks is the kickstart graph, which lays out the internal structure and available appliances within Rocks (Refer to Appendix A), and the MySQL database to manage the nodes from a central location (i.e. the frontend). The connections between bubbles on the kickstart graph tell Rocks which packages to install on each appliance. Connections can be made on the graph by editing and creating Rocks' XML files, which in turn changes what packages are distributed to the nodes. The kickstart graph is thus vital in generating kickstart files. Our kickstart graph contains customized configurations, seen in red, as part of the Florida Tech Roll.

Frontend Installation

To install the frontend, one needs to boot off the Rocks 3.3.0 CD 1. There are two installation methods - CD or from a central network location - we use the CD despite its slower speed. When the Rocks prompt appears, one types `frontend` to begin the installation process. Then, the rolls on CD are loaded into memory. The information inputted during the installation fills out the MySQL database and can be edited from the web interface installed by default and accessible by browsing to localhost. The frontend has two ethernet adapters- eth0 is the cluster-side network, and eth1 is the internet adapter. Both must be configured for the cluster to function as effectively one machine.

Node Installation

To install a node, the `insert-ethers` command must first be run from the frontend. This allows the frontend to be used as a DHCP server. The frontend collects the MAC address of the node and assigns it an internal IP. These addresses are located in the cluster MySQL database. This command also enables the frontend operator to select the "Appliance Type" of the node that is to be installed, i.e. compute node, web portal, etc. In our case, we created our own appliance type - by editing the MySQL database and adding a new XML file - that installed a Rocks compute node as well as Condor (See Appendix B for detailed procedures). After the frontend has recognized the presence of a node and assigned it an IP, it sends the node a kickstart file. Kickstart files allow for automated installation - they contain the configuration information for every step in the Anaconda installation process.

System Administration

Adding a user in Rocks is done by the normal Linux command, `useradd`, which runs a series of scripts and updates the 411 database. One of the problems we encountered this semester was dealing with 411 - the authentication protocol of Rocks - and NIS - the authentication protocol of OpenFiler, our NAS operating system. Our chosen solution is to also install an NIS server on the frontend and have both NIS and 411 access the same configuration files (i.e. `/etc/passwd, /etc/group, /etc/shadow`, etc.). This is not an ideal solution, as it requires more administration, but is appropriate for a 10 node cluster. Although authentication can be scrubbed the advantages to having a authentication method allows for users to access only their own data and provides added security. We would not want someone to accidentally delete another users files for example.

Condor - Batch Job System

In our cluster we use Condor as our batch job system. Condor is a piece of software that enables us to distribute the load of a computing task over the 20 CPUs in the cluster. Condor is also well suited for grid computing, as it is able to submit jobs to machines located all over the world. Also, most code does not need to be recompiled to run using Condor's computing resources. Condor is a very robust and complex application, as one can see from the 600 page manual.

Although Rocks provides a Roll for the initial installation, this is not ideal for our use due to the fact that the Rocks roll has unsupported features for OSG. Therefore we have to install Condor manually which fortunately is not too daunting a task (See Appendix C for detailed procedures). Once the RPMs are installed, configuration files located in the `/opt/condor/` directory need to be edited. Condor must be "rolled" out to the nodes via the XML files and adding the RPMs in the `/home/install/` directory tree. After the nodes are installed with their new Condor configuration the cluster is a fully qualified computing behemoth. Practically, one can see the status of the machines in condor by typing `condor_status` at a terminal:

```
Name           OpSys       Arch   State      Activity   LoadAv Mem   ActvtyTime
vm1@fltech01. LINUX       INTEL  Unclaimed  Idle       0.000   249  0+00:16:12
vm2@fltech01. LINUX       INTEL  Unclaimed  Idle       0.000   249  0+00:16:12
vm1@fltech02. LINUX       INTEL  Unclaimed  Idle       0.000   249  0+00:16:41
...
                     Machines Owner Claimed Unclaimed Matched Preempting
         INTEL/LINUX       20     2       0        18       0          0
               Total       20     2       0        18       0          0
```

Network Attached Storage with OpenFiler

We decided to do away with Rocks on our file server machines as it was proving very difficult to create a kickstart file to automate the creation of a RAID 5 array.. Instead, we are now using Openfiler, a distribution of Linux which has been designed with Network attached storage in mind (See Appendix D). We also no longer use RAID 1 - two hard drives in an array with identical data - as it has two significant drawbacks:

- Slow writing, as each hard drive must write all data.
- Inefficient use of space, i.e. for 400GB of hard drive capacity, we only get 200GB of usable space.

Thus, we decided to use RAID 5 (disk striping with distributed parity) for the following reasons:

- Faster writing. Each hard drive needs to write only 1/3 of the data.
- More efficient use of space. The efficiency of RAID 5 increases as the number of hard drives in the array increases. A minimum of three can be used.
- Fault tolerance. If any one hard drive fails, the data on that drive can be reconstructed using the data from the other two drives.

Future Work

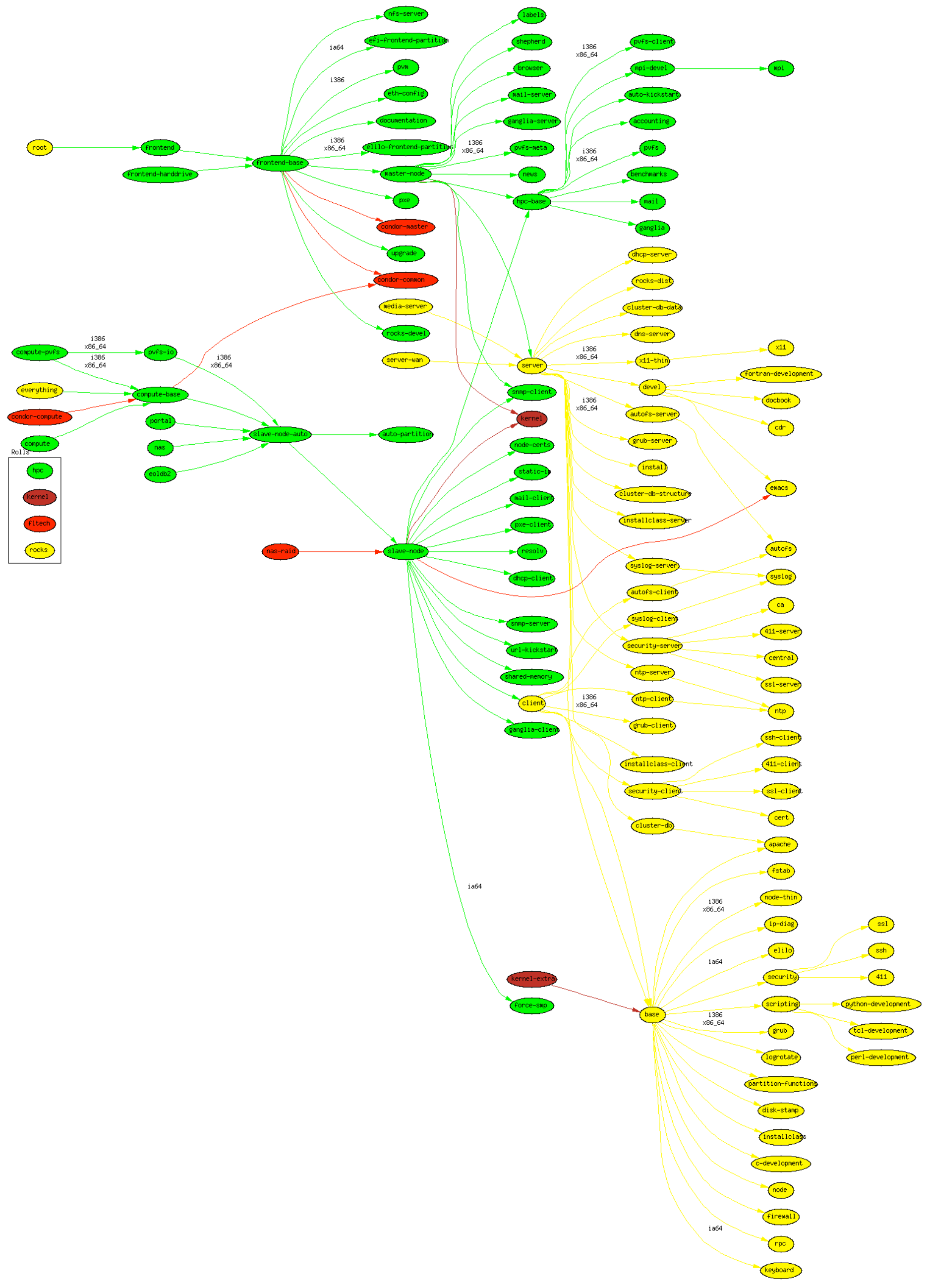
At present, we are recovering from a hard drive failure on the frontend. Ironically, we were performing a backup of the hard drive to ensure our data was safe before installing OSG when the hard drive failed. The first task is to restore or recreate lost configuration files. Rocks has been reinstalled and is now waiting upon those files. Using our custom condor-compute appliance (which we also need to recreate), we can fairly easily enter the nodes into the Rocks database. Alternatively, we may be able to actually manually enter the MAC and IP addressing information of the nodes into the frontend's MySQL database without reinstalling the nodes. This would be the most convenient solution. Once we are back up to our previous point, the second NAS - nasraid2 - can be installed using new hard drives. Then, the frontend can be backed up on both NAS servers.

After backup, the process of installing OSG can continue, with the installation of Pacman onto the frontend and the NAS to house the C.E. (Compute Element) Client software. C.E. Client software must be installed on the NAS where all nodes may access it. Then VDT (Virtual Data Toolkit) software can be installed on the frontend.

Conclusion

During this semester we were able to set up a 20 CPU cluster, with customized XML files, supporting network attached storage on a separate linux distribution, and begin the OSG installation process. With the help of Jorge Rodriguez, we have been able to make significant headway toward our goal of contributing Florida Tech computing resources to the Open Science Grid community. It has been an important learning experience in cluster computing and system administration.

Appendix A: The Kickstart Graph



Appendix B: Creation of the Condor-Compute Appliance

In detail, this is the process of creating the new appliance mentioned earlier in this paper. We want to create an appliance that rolls Condor out to the nodes with Rocks:

I. An XML file must be created for the new appliance - `condor-compute.xml` - which must be located in `/home/install/site-profiles/3.3.0/nodes` directory. A sample configuration is as follows:

```
<?xml version="1.0" standalone="no"?>
<kickstart>
<description>Condor-Compute</description>
<changelog></changelog>
<post>
<file name="/etc/motd" mode="append">Condor Compute</file>
</post>
</kickstart>
```

II. Now, we need to make the links between the bubbles on the kickstart graph in order to tell Rocks which packages we want the new appliance to have (refer to Appendix A). We connect the new condor-compute node to the node that already exists - its configuration information is located in `compute.xml`. To create links between bubbles, one must create a new file in `/home/install/site-profiles/4.2.1/graphs/default` such as `links-condor-compute.xml`. It needs to contain, in XML, links between bubbles, coded as follows:

```
<?xml version="1.0" standalone="no"?>
<graph>
<description></description>
<changelog></changelog>
<edge from="condor-compute">
     <to>compute</to>
</edge>
<! --- Insert additional links here in same format -->
<order gen="kgen" head="TAIL">
     <tail>condor-compute</tail>
</order>
</graph>
```

III. These changes to the internal structure of Rocks need to be propagated throughout the cluster. To do this, `cd` up to `/home/install` and as root:

```
# rocks-dist dist
```

IV. Also, the new appliance information should be inputted into the MySQL database:

```
# add-new-appliance --appliance-name "Condor Compute" --xml-
config-file-name condor-compute
```

And that's it. Now when the `insert-ethers` command is used on the frontend, on the "Choose Appliance Type" screen, "Condor Compute" will be displayed as one of the possible choices.

Appendix C: Condor Installation and Configuration

Here we have the procedure by which we installed and configured the condor batch job system. Although Condor is available as a roll for Rocks we have, under advisement from Dr. Jorge Rodriguez, decided to install Condor via RPMs and configure the setup manually.

I. The first step is installing Condor which we do via RPM (Red Hat Package Manager). The files can be downloaded from: http://www.cs.wisc.edu/condor/downloads/. Once on this page, download version 6.6.11 to use with Rocks 3.3. Newer versions are not supported by OSG. On a terminal, go to the directory where you downloaded the file and type the following command:

```
# rpm -Uvh condor-6.6.11-<rest of filename>.rpm
```

II. After this file is installed, Condor needs to be configured by adding a file to `/opt/condor-6.6.11/` called `condor-config.local`. This file contains information about which machine will be the central manager and where to get information about the rest of the worker nodes. Here is the output of that file:

```
CONDOR_ADMIN                = jhelsby@fit.edu
CONDOR_IDS                  = 502.502
MAIL                        = /bin/mail
CPUBusy                     = False
UID_DOMAIN                  = local
FILESYSTEM_DOMAIN           = local
CONDOR_HOST                 = uscms1.local
DAEMON_LIST                 = MASTER, STARTD, SCHEDD
START                       = True
RANK                        = 0
PREEMPT                     = False
PREEMPTION_REQUIREMENTS     = False
VACATE                      = False
SUSPEND                     = False
CONTINUE                    = True
KILL                        = False
PERIODIC_CHECKPOINT         = False
WANT_SUSPEND                = False
```

```
WANT_VACATE              = False
```

III. Once this file is written, Condor can now be configured to function properly on the nodes. The rpm needs to be added to the following directory:

```
/home/install/contrib/enterprise/3/public/<arch>/RPMS/
```

IV. Rocks needs XML files to make new kickstart files. Inside `/home/install/site-profiles/3.3.0/nodes` there is a `skeleton.xml` file which needs to be copied using:

```
# cd /home/install/site-profiles/3.3.0/nodes/

# cp skeleton.xml extend-compute.xml
```

In this file, there are tags such as the following which can be modified further if need be:

```
<!-- There may be as many packages as needed here. -->
<package> condor </package>
<package> <!-- insert your 2nd package name here --> </package>
<package> <!-- insert your 3rd package name here --> </package>
```

V. Condor needs to be configured to work properly with the rest of the cluster. This is done similarly as with the frontend but this would be quite difficult to do repeatedly if say we had hundreds of nodes. We can use Rocks' XML files to configure the nodes automatically as we did the RPMs

There are two more files that must be added to the `/home/install/site-profiles/3.3.0/nodes/` directory: `condor-master.xml` and `condor-compute.xml`. These two files perform the same task as the one we performed when we created the condor-config.local only this time it is automated. At this point we need to ensure that our changes take effect. We then change directory to `/home/install` and run:

```
# rocks-dist dist
```

After we do this we can test this using the following command to reboot and reinstall a test node:

```
# /boot/kickstart/cluster-kickstart
```

If everything works correctly, the node should reboot and connect to the newly configured condor master (i.e. the frontend) and be ready to accept condor jobs.

Appendix D: Installing and Configuring Openfiler

Installing Openfiler is done via CD and it is a fairly straightforward Linux installation procedure which will walk us through setting up the RAID 5 partition. After this installation is complete, we must then prepare Openfiler to function properly with the frontend and the rest of the cluster. The NAS needs direct communication with the frontend which, acting as a DHCP server, provides it with an IP address. To properly configure Openfiler, we will have to edit two MySQL databases containing the required network data. Because Openfiler is a different distribution than Rocks we cannot automate this process.

I. Openfiler's hostname must be configured for the rest of the cluster to refer to it as when making requests for files and jobs. We do this by editing `/etc/sysconfig/networks` . This file contains a variable called HOSTNAME which we need to change to something appropriate. For example, in our installation we use nasraid1.local.

II. Next, we address communciation between the frontend and nasraid1. In Rocks the frontend controls the IP addresses and network information for the rest of the machines on the network. So our first order of business is giving nasraid1 its appropriate configuration information. The two MySQL databases that must be edited are the networks and nodes databases on the frontend. When we go to localhost on the frontend's web-browser, we can connect to the Database we need, located under http://localhost/admin/phpMyAdmin/index.php. Both need a combination of the following data:

```
ID - This ID is used by the database to connect the databases together
Name - This is the hostname we added to /etc/sysconfig/networks
Membership - in our case we want membership 2 which is for storage
CPUs - this needs to be set to 1 with our hardware
Rack - NULL for the NAS
Rank - NULL as well
Node - this will simply be the next available number
MAC - the MAC address to the network card on the NAS
Netmask - this is the netmask found when ifconfig is run on the frontend
Gateway - will be set to NULL we don't need it.
Device - this is the device used on the frontend (i.e. eth0)
Module - simply use 8139too (module used by the kernel to recognize the net-
work card)
```

After this information has been added to the database, we need to run the following command on the frontend to ensure the changes take effect.

```
# insert-ethers --update
```

III. A connection needs to be established with the NAS. The frontend is ready to accept connections from the NAS. Run the following command as root on the NAS to establish a network connection with the frontend:

```
# /etc/init.d/network restart
```

Verify that there is a connection with:

```
# ping uscms1
```

IV. Mounting the RAID on the nodes and the frontend requires a bit of work. The first step is to set up the partitions to be accessible to all the other machines on the network. This is done through `/etc/exports` file which we must edit by adding a new line such as the following:

```
/RAID 10.0.0.0/255.0.0.0(rwx)
```

Where RAID is the directory where the RAID partition is mounted.

This is where things get a bit repetitive. We need to edit the `/etc/fstab` file on the frontend and on each node by adding the following line to the bottom:

```
nasraid1:/RAID /mnt/nasraid1 nfs rwx 0 0
```

The best way of doing this is to run:

```
# cluster-fork echo "nasraid1:/RAID /mnt/nasraid1 nfs rwx 0 0" >> /etc/fstab
```

and now you must repeatedly type the password. The cluster-fork command is a built in Rocks program which allows all the compute nodes to receive a particular command at the same time. This is best done once all the nodes have been initially installed. If a new node is installed or you need to shoot a node, you then must input this line manually as if you run the cluster-fork command it may disable the rest of the nodes.

V. From now on all the files can be accessed through the `/RAID` directory. One major drawback of mounting the NAS in the way stated above is that authentication is not allowed. All files are created without owner or group information which is a problem for protecting users data. We are working on a solution to this by configuring the frontend as an NIS server and sharing the authentication files with OpenFiler.

BIBLIOGRAPHY

[1] http://www.top500.org

[2] http://www.rocksclusters.org/wordpress/?page_id=4

[3] http://www.cs.wisc.edu/condor/manual.v6.8.2/index.html

[4] www.opensciencegrid.org

[5] http://www.rocksclusters.org/rocks-documentation/4.2.1/

[6] http://www.openfiler.com