A Complete Guide to Building a Beowulf Cluster

Luis Pimentel

Version 1.0, June 2016, Rockdale Career Academy

Contents

1 Introduction

In the fall of 2015 I was approached by my computer science teacher with a new project that he had in mind. After reading on the Internet about computer clusters with small Raspberry Pi's, he decided that we should attempt to build our own parallel processing computer in our Cisco lab using many of the spare computers we had available. This would result in our very own "super computer". Using this computer cluster we could then explore the many different applications of parallel programming.

He briefly explained how computer clusters worked, and a few of the aspects of the project. I learned that this project would encompass a range of subjects including computer science, networking and the use of Linux operating systems. I began this project eager to learn new technologies and put my my skills with computers and networking up the test.

I began this project with very minimal knowledge and experience using Linux, and even more so with parallel processing. As you do with anything new, I had to do lots of research to understand many of the concepts. I found that doing this research was no easy task, as no matter how much I searched, I failed to find a "complete guide" that not only offered steps to building a cluster, but an explanation of what I was doing. I ended up having to use many different articles, guides and the great wisdom that can only come from Linux forums.

Throughout the project I incorporated various methods that I came across during my research. Through a strenuous process of trial and error I took what worked best, from deciding which operating system to use, to deciding what kind of file-system structure I wanted to have, and I implemented it into my cluster. I have decided to use this guide as documentation for my project, as serve as a standalone reference for others to build Beowulf Clusters and delve in to the applications of parallel processing.

In the next months I will be working alongside a group of students to write documentation on building what will be a custom computer cluster tower made from the 8+ computers of the original cluster. Not only will the cluster look more like a "supercomputer" and less like a shelf of old computers, but it will provide more mobility and be designed so that it can be expanded.

As for the software side of things I am excited to be able to use our cluster to learn more about parallel processing and parallel programming applications. We are currently looking into implementing the cluster into pen-testing security applications. We are also looking into using already developed programs encompassing different physics simulations to learn more. Another interest of ours is seeing how we can incorporate our cluster into online projects that use networks of computers to solve problems. Projects like these include the SETI@home and FOLDING@home projects.

2 Fundamentals

The purpose for this section is to provide an overview of the different technologies that this project entails. This will begin with an explanation of High Performance Computing, computer clusters, and parallel processing. Afterwards is an explanation of what exactly a Beowulf Cluster is and why we chose to build this type of cluster.

The purpose of this section is not to go completely in depth with an explanation of these technologies, but to provide a basic preview that can be useful for beginners. With that being said I highly encourage further reading on each of these topics and other related topics.

2.1 High Performance Computing, Computer Clusters and Parallel Processing

In simple terms, High Performance Computing (HPC) is the use of parallel processing to run advanced programs. These programs are run on HPC systems so that they can be processed much faster and efficiently. HPC systems are often used by advanced researchers in the scientific field and in government. Typically any organization that is need of powerful computing is likely to have an HPC system, whether it is made up of special custom components or everyday computer hardware.

One way to take advantage of HPC is through the use of computer clusters. A computer cluster is typically a goup of computers, in which each computer in the cluster is called a *node*. Each node has its own individual resources, such as a CPU, memory, and an Operating System. These nodes are typically connected through a Local Area Network in order to communicate with each other. Clusters have become relatively easier to build over time, especially through the use of off-the-shelve computers and Linux operating systems. They can be very low-budget allowing practically anyone with spare computer hardware to build them.

These clusters use parallel processing to behave like ”super computers”. Parallel processing splits a task into multiple parts, giving each individual node a part of the task. This makes use of multiple CPUs at the same time, causing the program to be processed in less time than on a typical computer.

Through computer clusters and their use of parallel processing, we essentially create a ”super computer”. There are many different applications for these type of computers. Such applications have included protein folding simulation, climate predication and graphics rendering. As said before, practically every industry in need of the processing power that computer clusters have to offer, is likely to own or use one of these machines.

There are many different types of clusters out there, classified by the computer architecture used (meaning how it is built and what hardware/software is being used). We decided to build a Beowulf Cluster.

2.2 Beowulf Clusters

Beowulf Clusters differ from other clusters in that any old computer capable of running Linux and capable of networking can be used to create a cluster. Furthermore each individual computer can be different from the other. You can use different computers to make up your cluster, making it an inexpensive choice for High Performance Computing. A typical Beowulf Cluster consists of one head node and several other slave nodes. These slave nodes can be added as needed, increasing the power of the cluster. The head node often acts as a central server, controlling all of the slave nodes. Other advantages to this type of cluster is that standards and tools have been developed, making it easier to develop programs for these machines.

For its inexpensiveness and relative easiness, we decided that building a Beowulf Cluster would be the best approach to building our very own ”super computer”. I used many of the old and new spare computers that we had available in our lab, and even build up a few of them.

3 Getting Started

3.1 Hardware

To build the Beowulf Cluster you are going to need a few computers. Technically you only need a minimum of 2 computers: one to act as the head node and another to act as the slave node. In our computer science room we used one head node and 8 slave computers. Our head node was a and our slave nodes consisted of .

When thinking about the number of nodes you are going to have, it is important to consider various factors including how much computing power you need, how much you are willing to spend and whether or not you have the infrastructure necessary to build a larger cluster. While relatively low compared to older "super computers", computer clusters still typically consume a lot of power. Your ability to store and network these computers should also be considered.

Each node in the cluster has a few hardware requirements. Each node will need a CPU, RAM, graphics, networking and storage. While you do not need the most cutting edge hardware, the better your processor is, the more powerful your computer cluster will be. As said before each of the individual nodes do not have to be the same. You can mix and match different motherboards, processors, etc. The only requirements are a basic Input/Output system and the ability to run Linux.

Because computer clusters communicate with each other through the use of a Local Area Network, basic networking equipment will be needed. The only networking requirements consist of a networking switch with enough ports to support the amount of nodes. Enough Ethernet CAT cable for each node is also required.

An Internet connection will be needed to download key software for this project.

3.2 Software

For this cluster we used Ubuntu Server as our operating system. While some experience with Linux is recommended, it is not fully required. This document will include basic tutorials for the necessary Linux commands.

You can download the latest version of Ubuntu Server here

Furthermore for our project I chose to not use a Graphics User Interface (GUI), meaning I used the default Command Line Interface (CLI) that comes after installing Ubuntu Server. I did this so that I could better learn Linux. Overtime I also found it is much more efficient. While you can opt to use a GUI rather than just the CLI, many of the commands necessary to run Linux are run in a terminal. .

The rest of the software used in this project consists of packages that will be installed and explained further along the project.

While this project will provide a brief explanation of many of the commands used in this project, it is recommended that you do further reading to Linux operating systems.

Basic networking knowledge is also needed for this project. This document will not go into detail with some of these networking topics, so further reading into networking topics such as IP addressing is advised.

3.3 How to Use This Guide

I designed this guide to be used on chronological order. You should complete the steps as they are laid out.

I began with instructions for configuring the head node and then move on to configuring the slave nodes. However, as you progress through the guide, you will have to switch between working on the head node and working on the slave nodes. It is of extreme importance that you know which node you are supposed to be working on, as some commands are only supposed to be run on either machine.

As said before this guide provides not only instructions, but explanations for the different technologies you are using and how they are essential to the design of this cluster.

4 Building the Cluster

The following are general instructions for setting up you computer cluster hardware. This part can vary depending on the setup that you want.

Instructions

a. Begin by setting up you Head node. This should be your most powerful computer available. It is recommended to place this node separate from the rest of the nodes in the cluster. You should be able to easily distinguish this node from the others. This node should also have its very own Monitor, keyboard and mouse setup.

b. Begin setting up the Slave nodes. Typically these nodes should be grouped together and labeled. The design of the cluster is completely up to you. You can have the computers stacked up on each other, or placed side by side on a rack.

   Things to consider in this step include distance to the central networking switch and distance to a power source. Use power strips as necessary. Be careful with power however, as you do not want to overload one power outlet with too many computers.

c. Proceed by setting up a central networking switch. When choosing the location of the switch, keep in mind the distance of the nodes.

d. Begin networking the cluster. Connect an Ethernet CAT cable to each of the nodes, and connect each cable to a port on the central networking switch.

   Consider cable management in this step as you do not want to create a tangle of wires. If possible, try to color code these cables or label them to easily identify them when troubleshooting networking issues.

e. Go ahead and plug power strips to each of the nodes. However you should not turn on any of the nodes yet, as they do not have an operating system installed.

f. At this point you should setup a SOHO router that will provide Internet to the cluster. You will need to turn off DHCP and set your router to work with the 10.0.0.0 /24 network. Reference your router's manual for help on how to do this.

After completing these steps you should have successfully built the infrastructure necessary to build a Beowulf Cluster. We now proceed to configuring each of the nodes with the necessary software.

5 Head Node Configuration

Configuring the Head node is the first step in building the Beowulf cluster. This is the most important node in the cluster as we will run all of the jobs. This node will also act as a central server, providing many of the services described later in this document.

As said before this should be your most powerful machine, it should be in a distinguished area, with its own monitor, keyboard and mouse.

Many of the naming conventions that we used in this project can be changed depending on your preference. This includes usernames, hostnames, IP addresses etc. Feel free to change this for your own project. However make sure to document which things you are changing so that you do not cause confusion in future steps. The only requirement is that each node must have the same username and be on the same network.

Instructions

a. Download the latest version of Ubuntu Server and install on Head node.

   For more information on how to install Ubuntu reference other "Other References" section of this document.

b. Configure the IP address and subnet mask of this node (10.0.0.100 and 255.255.255.0).

c. Configure the gateway address and DNS name-server (10.0.0.1 and 8.8.8.8).

d. You should set the hostname of this computer to headwolf.

e. You will need to create a user with the name of mpiuser. Set a password for this user.

5.1 Configuring Hosts File

The hosts file is used so that nodes can communicate with each other without the need to remember and type out each individual node's IP address of which you are trying to communicate with. It maps out each nodes IP address to each node's hostname.

You will manually have to configure the hosts file on every machine.

Instructions

a. Open up the hosts file within the /etc directory using nano text editor.

```
sudo nano /etc/hosts
```

b. Configure it to look like the following

```
127.0.0.1        localhost

10.0.0.100       headwolf

10.0.0.101       wolf01

10.0.0.102       wolf02

10.0.0.103       wolf03
```

c. Add as many nodes as you have set up.

d. Save file and exit out.

The 127.0.0.1 to localhost line is linking localhost to the computers loopback address. This is necessary for network troubleshooting and does not pertain to building the Beowulf Cluster.

5.2 Configuring IP Addresses

To setup our cluster's network we need to configure the IP addresses of each node. We need to establish static IP addresses on each node that match the IP addresses in the hosts file.

The IP address, gateway and DNS server should have been configured as Ubuntu Server was installed. However you can change these configurations by editing the /etc/network/interfaces file.

The configuration of this file will be different depending on your setup.

5.2.1 Configuring IP Addresses

Instructions

a. Identify your current IP configuration.

```
ifconfig
```

b. Your output should match the previous information. Make note of your Network Interface Card's name (i.e. eth0).

c. If your output doesn't match the correct information, or you wish to change the configuration, do the following.

d. Open and edit the following file

```
sudo nano /etc/network/interfaces
```

To look like the following.

```
auto eth0
iface eth0 inet static
address 10.0.0.100
gateway 10.0.0.1
netmask 255.255.255.0
network 10.0.0.0
broadcast 10.0.0.255
dns-nameserver 8.8.8.8
```

e. Restart your networking service for these settings to take place

```
sudo /etc/init.d/networking restart
```

**** Your gateway address should be the address of your router

**** Replace eth0 with the name of your NIC found in step 1. If it is the same do not replace.

6 Slave Node Configuration

Instructions

a. Install Ubuntu Server on each of the slave nodes.

   For more information on how to install Ubuntu reference other "Other References" section of this document.

b. As you install Ubuntu Server, configure the network information of each node following the format shown in Section 6.2

c. You should set the hostname of this computer to headwolf.

d. You will need to create a user with the name of mpiuser. Set a password for this user.

6.1 Hosts File

To set up the hosts file for the slave nodes, use the same steps used to set up the hosts file on the Head node. (Section 5.1)

Repeat these steps for each Slave nodes in your cluster.

6.2 IP Addresses

If the IP address and network information was not set during installation, follow the following instructions.

To set up the IP addresess of the Slave nodes in your cluster, use the same steps used for the Head node. (Section 5.2)

The Static IP set in the /etc/network/interfaces configuration file should follow the following format:

```
auto eth0
iface eth0 inet static
address 10.0.0.XXX
gateway 10.0.0.1
netmask 255.255.255.0
network 10.0.0.0
broadcast 10.0.0.255
dns-nameserver 8.8.8.8
```

For example if you were configuring the 6th slave node in your cluster:

```
auto eth0
iface eth0 inet static
address 10.0.0.106
gateway 10.0.0.1
netmask 255.255.255.0
network 10.0.0.0
broadcast 10.0.0.255
dns-nameserver 8.8.8.8
```

Always remember to restart your networking settings.

```
sudo /etc/init.d/networking restart
```

7 Setting Up the Network File System

Network File System is a computer protocol that allows networked computers to access each other's file systems as if they were stored locally. We will use NFS to share the Head node's /home/mpiuser directory with the other nodes. Because every node has the username **mpiuser** , each node already has this directory. Therefore the /home/mpiuser directory is shared by the whole cluster. This is necessary because the nodes need to be able to access the jobs that we are running. Having the nodes virtually share the same directory also comes in handy as we install SSH for communication later on. We will also need to share the /usr/local/bin directory of the Head node. This directory will hold the MPICH programs needed to run the cluster. By sharing it, the Slave nodes have access to these programs, and therefore we won't need to install MPICH on each node.

7.1 Installing NFS on Head Node

NFS must first be installed on the Head node from which we will share this directory.

To install NFS service on the Head node, run the following command:

```
sudo apt-get install nfs-kernel-server
```

Next export the /home/mpiuser directory so that the other nodes can have access to it. Do this by editing the /etc/exports file:

```
sudo nano /etc/exports
```

And add the following line to the end of the file:

```
/home/mpiuser *(rw,sync,no_subtree_check)
/usr/local/bin *(rw,sync,no_subtree_check)
```

Restart the NFS service:

```
sudo service nfs-kernel-server restart
```

For NFS to work with the rest of the nodes, the Head node must always boot up before the slave nodes. If you encounter problems with a node not having access to the shared /home/mpiuser directory, restart the slave nodes so that the Head node is already on.

Ubuntu's firewall can also cause the nodes to have problems accessing the shared directory. To fix firewall problems we can set it so that it allows access from our network. Enter the following on the master node to do this:

```
sudo ufw allow from 10.0.0.0/24
```

7.2 Installing and Mounting NFS on Slave Nodes

For the slave nodes to be able to access and mount the shared directory, the following must be installed on each slave nodes:

```
sudo apt-get install nfs-common
```

Next we need to mount the directory to the slave nodes and ensure that the directory is mounted automatically every time we turn on our nodes. Do this by editing the /etc/fstab file:

```
sudo nano /etc/fstab
```

And add the following line:

```
headwolf:/home/mpiuser /home/mpiuser nfs
headwolf:/etc/local/bin /etc/local/bin nfs
```

NFS should now be setup and each node should be sharing the same directory. Now we can go on the next step and install SSH for communication.

8 Setting Up SSH Password-less Communication

The cluster uses the Secure Shell (SSH) protocol to allow the Head node to control the rest of the nodes. Basically the head node can "login" to any of the nodes and run commands within that node. You can also manually take control of any node from any other node using SSH. Before you login SSH asks you for a pre-assigned password, meaning everytime you would run a job on the cluster, it would ask you to enter the password. We will setup passwordless SSH for more efficiency.

First install the SSH service into every node in the cluster by entering the following in every node:

```
sudo apt-get install openssh-server
```

On the Head node, generate an SSH key:

```
ssh-keygen -t rsa
```

Leave the field empty when asked for a password.

Next add this to the computer's list of authorized keys:

```
cd .ssh
cat id_rsa.pub >> authorized_keys
```

Typically for the other nodes to have access, this key would need to be copied to each of the node's list of authorized keys. But since every node is sharing the same directory through NFS, the key is already in every node.

To test that SSH is working properly try logging into one node from a different node. For example on the Head node try to login to the fist node

```
ssh wolf01
```

You should now be logged into wolf01 from headwolf. Try shutting down wolf01 to verify functionality:

```
sudo poweroff
```

This should have shutdown the wolf01 node and logged you back into headwolf on the Head node.

When you first login to a node using ssh, it will prompt you for a password. You can just press Enter as we set the password to empty. This should be the only time you are asked for a password. It is important to login once with each node using the password so that ssh will not ask again.

If you are still prompted for a password when trying to login to a node using SSH, you need to setup a keychain by installing Keychain. Keychain is an ssh-agent that can store your key information in the computer's memory. Therefore you will only be prompted for a password once every time you reboot.

To install Keychain:

```
sudo apt-get install keychain
```

Next we want to edit the /.bashrc file and add the following:

```
if type keychain >/dev/null 2>/dev/null; then
        keychain --nogui -q id_rsa
        [ -f ~/.keychain/${HOSTNAME}-sh ] && . ~/.keychain/${HOSTNAME}-sh
        [ -f ~/.keychain/${HOSTNAME}-sh-gpg ] && . ~/.keychain/${HOSTNAME}-sh-gpg
fi
```

/.bashrc file is a shell script that automatically runs every time a new terminal is opened. This allows you to easily configure settings for your environment.

***** **Bolding and lining are formatting errors please ignore.**

To make the changes exit and login once again or do:

```
source ~/.bashrc
```

9 Installing Compilers

Compilers are programs that translate the source code written by developers into machine code used by computer processors. Programing libraries such as MPI use these compilers to run parallel programs on clusters such as this one. However to run the programs, the computers must have the compilers installed. The following are a few common compilers that you should install on your cluster.

C, C++, Objective-C, Fortran etc.

```
sudo apt-get install build-essential
```

JAVA

```
sudo apt-get install sun-java6-jdk
```

PYTHON

```
sudo apt-get install python3
```

10 Installing MPICH2

We will install MPICH on the Head Node.

Instructions

a. Go to the MPICH download website http://www.mpich.org/downloads/ Choose the appropriate release. Hover your mouse over the http link next to the release and make note of its link.

   For example it should look like the following: http://www.mpich.org/static/downloads/3.2/mpich-3.2.tar.gz

   We will download the file from this link using wget on our cluster.

```
wget http://mpich.org/static/downloads/3.2/mpich-3.2.tar.gz
```

b. You should see the file downloaded in your home directory. Uncompress the folder and change into the directory.

```
tar -xvzf mpich***.tar.gz
cd mpich***
```

c. Next configure your installation and install.

```
./configure
sudo make install
```

d. To verify that MPICH was installed run the following. It should output the directory of the program.

```
which mpiexec
```

e. Do the same on each of the Slave nodes to verify that they have access to this program.

   If you have problems, make sure that you installed the compilers correctly.

11 Testing

Instructions

a. From mpiuser's home directory navigate to the examples folder within the mpich-3.2 folder and look through the example MPI programs.

```
cd mpich-3.2/example
ls
```

b. If not already compiled, compile one of the programs so you can run it on the cluster. Programs that are not compiled will end in ".c" We will compile the hellow.c program to test our cluster.

```
make hellow.c
```

c. Navigate back to the home directory. We will now run the hellow program using mpiexec. Follow the following syntax: **refer to the mpich manual for argument descriptions**

```
mpiexec args executable pgmargs [ : args executable pgmargs ... ]
```

d. To run the hellow program

```
mpiexec -f hosts /home/mpiuser/mpich-3.2/examples/hellow
```

e. The cluster should output a "Hello" message from each node.Running this program successfully verifies that everything was installed correctly. Congratulations on building your Beowulf cluster!

f. Important flag descriptions:

   -f — specify the hosts file which tells the clusters which nodes to use

   -n — specify the the number of processes to run on each node

   -hosts node1,node2,node3,etc — used to run on specific nodes; in place of hosts

12 Credits and References

https://help.ubuntu.com/community/MpichCluster

http://www.ducea.com/2006/08/07/how-to-change-the-hostname-of-a-linux-system/

https://help.ubuntu.com/community/NetworkConfigurationCommandLine

http://www.climatemodeling.org/ forrest/osdj-2000-11/

https://www.mpich.org/static/docs/v3.1/www1/mpiexec.html