

A STEP BY STEP TO BUILT A CLUSTER FOR PARALLEL COMPUTING

B. BENNECER*, K. ZANAT, A. BOUFELFEL AND F. KALARASSE
Guelma Physics Laboratory (GPL)
Exact Sciences Department
Faculty of Science and Engineering
Guelma University
BP 401 24000

* Corresponding author: bennecer@univ-guelma.dz or b_bennacer@hotmail.com

ABSTRACT:

In this study we present a simple way of clustering a set of machines for parallel computing. The installation of Redhat Linux 7.3 on each node (master and slave nodes), the configuration of the network and NFS server are given. Finally the installation and configuration of Message Passing Library (MPI) and the compiler on the master node are discussed.

Key words: cluster, REDHAT LINUX, PARALLEL COMPUTING, MPI, INTEL FORTRAN

RESUME:

Dans ce travail nous présentons une méthode simple pour grouper un ensemble de machines pour le calcul parallèle. L'installation de Linux 7.3 sur chaque nœud (maître et esclaves), la configuration du réseau et le serveur NFS sont donnés. Finalement, l'installation et la configuration du MPI (Message passing interface) et le compilateur sur le nœud maître sont donnés.

Mots-clés : calcul parallèle, Redhat Linux, MPI, Intel Fortran

Introduction:

The Physics Laboratory at Guelma University (Guelma Physics Laboratory; www.univ-guelma.dz/recherche) is composed of four research groups. One of these groups is the computational group; its main interest is first principles (ab-initio) studies of semiconductors properties (structural, electronic and optical ...) and magnetic properties of multilayer systems. Computational difficulties, were encountered when we started increasing the size of the supercell (the number of atoms) while studying the properties of semiconductors alloys using the supercell approach. It must be noted that for binary compound a single pc can do the job. To overcome these difficulties and satisfy our needs in computational facilities we decided to build a Beowulf cluster using the Pc's (PIII) available in our laboratory. The idea of building a super computer is an old one. We started thinking about it since our graduate students started working on their projects of Magistère in the laboratory in 2001.

After a detailed search in the internet and a few tries, we managed to build our first local cluster in December 2003. Now, the local cluster which is running our production codes (Wien_2k and the LMTO codes) in our laboratory is composed of 09 computers (one master and 08 slaves).

A cluster is a collection of local memory machines connected through a switch to form a network, i.e. the only way for machine A to communicate to machine B is through the network. The message passing libraries (PVM or MPI) passe messages between the nodes. In our first clustering efforts PVM was used, and then we settled for MPI.

The following sections cover the cluster architecture, its design, construction and configuration.

I. Designing the cluster

Although the purpose of this paper is to explain the way followed to build and to configure the cluster (we are not experts in clustering, we are physicists), it is of great importance to discuss some design aspect:

The computers we used are PIII with the following characteristics:

Processor: Intel 1GH.

Hard Disk: 40 GH.

Random Access Memory (RAM): 256 MB i.e. (2*128 MB).

Ethernet Card: 100 MB/s.

At the beginning we connected the monitors and the keyboards to all machines and they have been removed after the final configuration tests, leaving only one monitor and one keyboard connected to the master node.

First we choose one PC to be the master node and the others to be the slaves. Our cluster is local, therefore it is not seen by the outside world (even the master has one Ethernet card). Each Ethernet card is connected to the switch (CNet CNSH 1600 24 ports) through an Ethernet cable (see figure 1). We choose the standard NFS configuration for disk storage among the other methods known to us (disk-less clients, fully local install, distributed file system), this configuration allows /home and /usr to be mounted off the master node since in our set up each node has its own disk and operating system and swap locally. The NFS set up will be discussed latter.

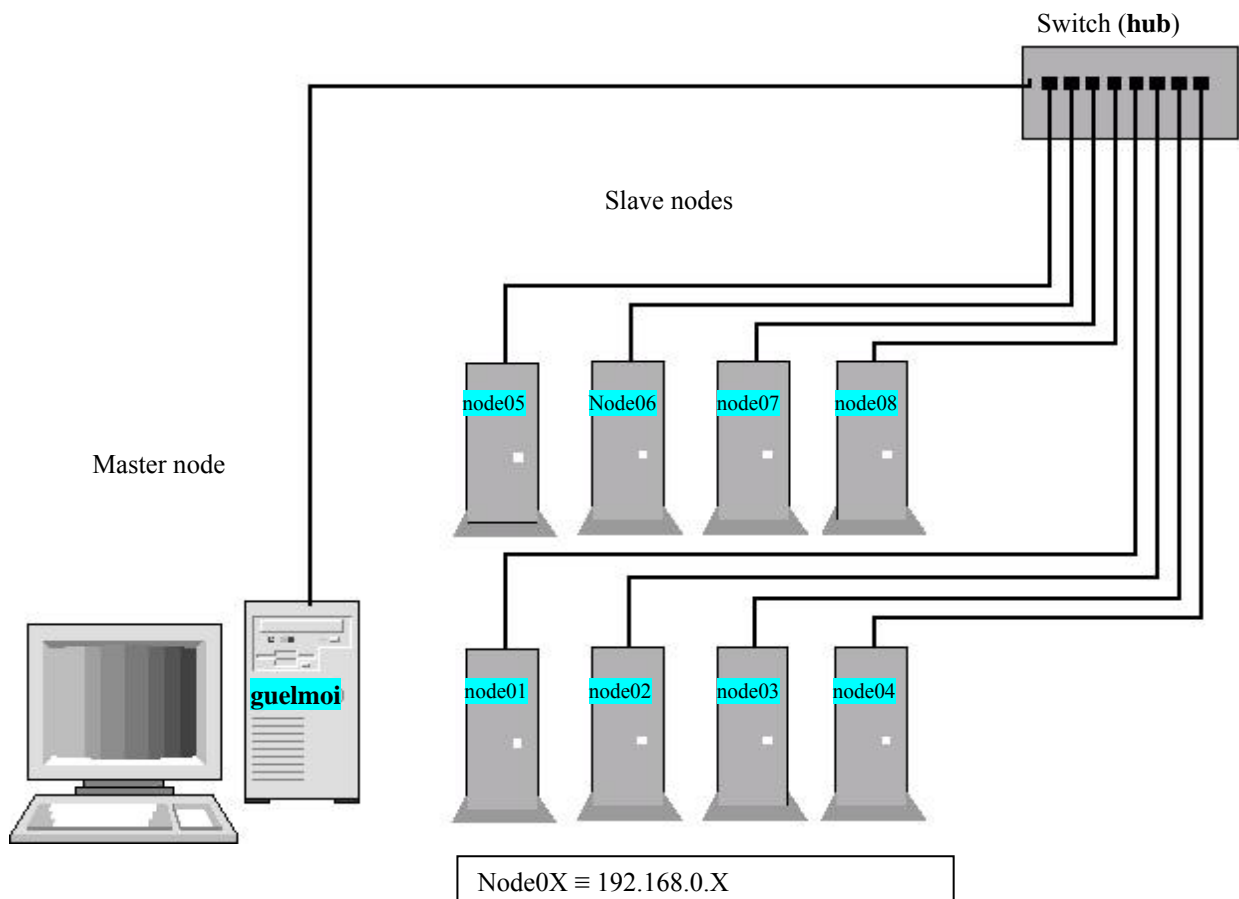


Figure 1: Guelma Physics Laboratory (GPL) cluster (Guelmoi)

II. Linux installation and network configuration:

Theoretically one can use any Linux distribution. We selected Redhat 7.3 which can be downloaded from (www.redhat.com : in our first efforts of clustering we used Redhat 9.0). First we installed the operating system on the master node then on each slave node one by one from a CD-ROM (make sure that the computer has its initial setup (Bios mode), which allows the boot from the CD-ROM, and if only one CD-ROM drive is available for the whole system one has to move it from node to node after each installation as in our case, without forgetting to turn off the machine before connecting and disconnecting the CD-Rom drive). We are not going to discuss in detail all steps followed during the installation, but only those of great importance.

After inserting the RedHat Linux 7.3 CD#1 into the disk drive and rebooting (restarting) the PC, the welcome message will prompt: we typed expert mode (for both master and slave nodes) and we choose the custom type of installing Redhat Linux (for both master and slave nodes).

1-Disk partition sizes:

A very important part of the installation processes is choosing the partition sizes. For the master node, we choose the redhat's disk partition tool and with 30 GB HDD we used the following sizes:

/boot	ext2	100Mo
/home	ext2	25000Mo
/	ext2	6000Mo
/swap	swap	580Mo
/usr	ext2	the size left

The drives “/home” and “/usr” will be mounted on the other slave nodes via NFS. The /usr/ drive will be used to install MPICH and the compiler.

For the slave nodes we used automatic partitioning:

/boot	ext2	100Mo
/swap	swap	510Mo
/	ext	the size left.

2- Configuration of the Ethernet card:

Redhat Linux will automatically detect the Ethernet card, named “ehh0”. We selected activate on boot for master and slave nodes and input the internal IP address and hostname as follows:

For master node

IP address: 192.168.0.1

Hostname: guelmoi(one can put any name)

For slave node:

IP address: 192.168.0.X (where X=2 for node02, 3 for node03 etc.....)

Hostname: node0X (i.e: node02, node03,.....etc)

3- Firewall configuration: we choose medium level and selected “ssh” for the master and slaves nodes.

4- Zone configuration: we choose Africa/Algiers

5- We input the root user password and confirm it.

6- Selection of the packages: We chose KDE and network support for the master while we chose the later only for the slave nodes and we activated the checkbox “selection of individual packages”.

7- Individual selection of packages: for the master and slave nodes in our cluster we chose the following packages: mc, bc, openssh, open ssh-server, open ssh-client, rsh, rsh-server and xinetd

8- We didn't create a bootable floppy disk.

9- Once the installation of the Redhat Linux on each node is completed, we rebooted the nodes. we logged on as root user (in the master and the other nodes).

11- We edited the /etc/hosts file (using vi or emacs editor). Our /etc/hosts for the master looks like:

```
# /etc/hosts master
```

```

192.168.0.1      guelmoi      node01
127.0.0.1      localhost
192.168.0.2      node02
192.168.0.3      node03

```

```

.
.
.

```

For the slave node02 for example the /etc/hosts looks like

```

# /etc/hosts node02
192.168.0.2      node02      node02
127.0.0.1      localhost
192.168.0.1      guelmoi
192.168.0.3      node03

```

```

.
.
.

```

After we finished the edition of the /etc/hosts file in all the nodes, In a console terminal (all nodes) we restarted the system by tyoing the command ./network restart in the directory /etc/init.d. Then we tested the network configuration by using the command: ping < hostname<, for example: ping node02, we got on the screen of the master:

```

PING Node02 (192.168.0.2) from 192.168.0.1 : 56 (84) bytes of data.
64 bytes from node02 (192.168.0.2) : icmp_seq=1 ttl=255 time=0.307 ms
64 bytes from node02 (192.168.0.2) : icmp_seq=2 ttl=255 time=0.237 ms
64 bytes from node02 (192.168.0.2) : icmp_seq=3 ttl=255 time=0.267 ms
64 bytes from node02 (192.168.0.2) : icmp_seq=4 ttl=255 time=0.238 ms
64 bytes from node02 (192.168.0.2) : icmp_seq=5 ttl=255 time=0.287 ms
64 bytes from node02 (192.168.0.2) : icmp_seq=5 ttl=255 time=0.224 ms
--- node02 ping statistic---
6 packets transmitted, 6 received, 0% loss, time 5028 ms
rtt min/avg/max/mdev=0.224/0.259/0.307/0.035 ms

```

III. THE NFS SERVER CONFIGURATION

After configuring the network of our cluster we setup the NFS server on the nodes.

1- On the master

- We logged on as root, enter the command setup, we chose the “systeme services” to activate the following doemons: network, nfs, nfslock, portmaps, rsh, rlogin, ssh and xinetd.
- We edited the file /etc/exports and it looks like

```

# /etc/exports master
/home 192.168.0.1/9(rw,no_root_squash)

```
- We added a user on the master node “guelmoi” and it is named **lpg**.

2- On each node:

- We activated the same doemons as in the master node in “system services” by using the command setup.
- We created on each node the directory home by using the command: mkdir -p /home
- We edited the file /etc/fstab and added the lines:

```

# /etc/fstab
guelmoi:/home /home nfs noac 0 0
guelmoi:/usr /usr nfs noac 0 0

```

3- Then on the master node we entered the commands:

```

/usr/sbin/exports -a
/usr/service nfs reload

```

the first one is to check that every thing is alright for the /etc/exports file and the second is to restart the NFS service

4- We added the same user lpg on slave nodes by using the command: adduser lpg

(Whenever one wants to change the shell for the user the command is: chsh user name -s /bin/name of the shell)

5- We rebooted all slave nodes.

6- RHE configuration of rsh on each node in our cluster: rsh permits the connection from one node to another.

- We created .rhosts file in the user and root directories in the master and slave nodes.

Our .rhosts file looks like:

For user (lpg)

```
# /home/lpg/.rhosts
guelmoi      lpg
nodes02      lpg
node03       lpg
.            .
.
```

For root users

```
# /root/.rhosts
guelmoi      root
node02       root
node03       root
.            .
.
```

7- We added the file /etc/hosts.allow in master and slave nodes and added the line ALL

```
# /etc/hosts.allow
ALL
```

8- We edited the file /etc/securetty on the master and slave nodes and added the lines:

```
rsh
rlogin
rexec
pts/0
pts/1
```

To allow root users use rsh.

9- We modified the file /etc/pam.d/rsh on all nodes. Our /etc/pam.d/rsh looks like:

```
#/etc/pam.d/rsh
##PAM-1.0
# For root login to succeed here with pam_securetty, "rsh" must be
# listed in /etc/securetty.
auth      sufficient /lib/security/pam_nologin.so
auth      optional  /lib/security/pam_securetty.so
auth      sufficient /lib/security/pam_env.so
auth      sufficient /lib/security/pam_rhosts_auth.so
account   sufficient /lib/security/pam_stack.so service=system-auth
session   sufficient
/lib/security/pam_stack.so service=system-auth
```

10- At this stage we tested if rsh works, we logged on the master as lpg user and executed the command: rsh -l lpg node02 reboot (or any command in place of reboot; ls for example)

IV. INSTALLATION OF MPICH AND THE INTEL COMPILER

IV-a. MPICH

We downloaded the MPICH (version 1.2.5.2) from the web site:

<http://www-univ-mcs.ane.gov.mpi/> : 12.4 Mo [one has to read the installation instruction carefully].

We untar mpich.tar.gz in /usr/local using the command:

```
Tar -xvzf mpich.tar.gz
```

Then we go to the directory mpich-1.2.5.2 (it has been created where we untared the mpich)

We executed the MPI shell file: **./configure**

We executed the compile command: **make**

We edited the file /usr/local/mpich-1.2.5.2/util/machines/**machines.Linux** and we added the nodes in our cluster

The file looks like:

Guelmoi

node02

node03

.

.

.

We edited the file **.cshrc** (we are using the tcsh shell) in /home/lpg (lpg is the user) and added the following lines:

```
#!/home/lpg/.cshrc
```

```
setenv MPICH "/usr/local/mpich-1.2.5.2"
```

```
setenv PATH ":{MPICH}/bin:{MPICH}:/util
```

Note: at this stage, by default mpif77 and mpicc are linked to the f77 and gcc compilers of the operating system respectively and the mpif90 compiler can not work.

Test:

- 1- execute the command make as follows:

```
cd /usr/local/mpich-1.2.5.2/examples/basic
```

```
make
```

if one wants to test other examples see README file in the directory basic

- 2- We chose the program cpilog as example, so we copied it to /home/lpg then we executed the command:

```
mpirun -np 8 cpilog
```

```
Process 0 running on guelmoi
```

```
Process 2 running on node03
```

```
Process 1 running on node02
```

```
Process 4 running on node05
```

```
Process 7 running on node08
```

```
Process 3 running on node04
```

```
Process 6 running on node07
```

```
Process 5 running on node06
```

```
pi is approximately 3.1415926535898908, Error is 0.0000000000000977
```

```
wall clock time = 1.741243
```

IV.b. Installation of the FORTRAN compiler (Intel FORTRAN compiler 7.1)

We created /opt/intel/licenses directory, we copied the license file "**intel.lic**" to it by using the command:

```
cp intel.lic /opt/intel/licenses
```

- We created two directories in /home/lpg (working directory)

```
mkdir fc_7.1
```

```
mkdir cc_7.1
```

- We copied the two files **l_fc_p_7.1.008.tar** and **l_cc_p_7.1.006.tar** in the above directories respectively, then - we untared them

```
tar -xvf <filename>
```

in the directory `fc_7.1`, we executed:

```
./install
```

at the end of the conditions of the licence (or type CTRL ^C) we typed **accept**, then we specified the path of the installation directory (not the one given by default), we chose

```
/usr/local/intel/fc_7.1
```

[Now we can delete the directory `fc_7.1` in `/home/lpg`]

- We edited the file **.cshrc** in `/home/lpg` and added the path of the compiler and its libraries:

```
#!/home/lpg/.cshrc
```

```
source /usr/local/intel/fc_7.1/compiler70/ia32/bin/ifcvars.csh
```

For the installation of “icc”, C, compiler we followed the same steps as for the ifc except that we replaced `fc_7.1` by `cc_7.1`.

In order to make the connection between the MPI and Intel Fortran and C compilers we followed the steps bellow:

```
setenv F90 ifc
```

```
setenv FC ifc
```

```
setenv CC icc
```

```
./configure --prefix=/usr/local/mpich-1.2.5.2 --with-common-prefix=/usr/local/mpich-1.2.5.2
```

```
make
```

and the final step is to specify the path of mpich libraries, we edited the file `ifcvars.csh` in `/usr/local/intel/fc_7.1/compiler70/ia32/bin` and added “`:/usr/local/mpich-1.2.5.2/lib`” in the end of the line: `setenv LD_LIBRARY_PATH`

V. CONCLUSION

A cluster of nine nodes (master and eight slaves) has been designed and configured using Redhat 7.3 and the MPI library.