**GROUP NO: 5**

**REPORT ON HOW TO SETUP CENTOS 7 CLUSTER AND INSTALLING MPI**

1. **Nodes Setup**

Two VM (Virtual Machine) were created on Vultr.com (VPS provider)

**Specification of the VM’s**

**1st VM**

* Operating System - Cents OS 7 x64
* **1** CPU
* **1024MB** Memory
* **1000GB** Bandwidth
* Hostname **masternode**
* Ip Address **95.179.231.153**

**2nd VM**

* Operating System Cents OS 7 x64
* **1** CPU
* **1024MB** Memory
* **1000GB** Bandwidth
* Hostname **computingnode1**
* Ip Address **45.63.68.100**

**Software required**

1. Putty (software proving terminal for SSH, Telnet, SCP etc.)

**Initial Installation Steps**

Commands

1. ***yum update***

=>Yum Package Manager for Cent OS

=>To Update the Package Manager

***2. yum install sudo***

=>To Have Privilege

3. ***sudo yum groupinstall "Development Tools"***

Sudo yum install kernel-devel

=>For Installing Development tools for our Cent OS VM

***4. yum install net-tools***

=> To install network tools

1. **Installation of MPI on both VM’s**

First, we’ll need a library called **wget**. Wget will allow us the ability to download links to the machine. We will download wget on masternode and computingnode1 (both VM’s).

**yum install wget -y**

We will now download the source of **mpich**, which is an implementation of **MPI**. We download inside the shared folder.

**cd /nfs**

**wget http://www.mpich.org/static/downloads/3.1.4/mpich-3.1.4.tar.gz**

After we have downloaded mpich, we will install the C compilers, **Fortran** compiler, and kernel build tools on both VM’s.

**yum install gcc gcc-c++ gcc-fortran kernel-devel -y (on both machines)**

**vi ~/.bashrc**

To extract the mpich downloaded tar.gz compressed file, we use the following command. The command will create a mpich-3.1.4 folder with all the contents of the extracted compressed file.

**tar -xvf mpich-3.1.4.tar.gz**

We will make a directory where all the compiled binaries and libraries of mpich will go.

**mkdir /nfs/mpich3**

Now, we will configure the settings of mpich for installation.

**cd /nfs/mpich-3.1.4**

**./configure --prefix=/nfs/mpich3**

Afterwards, we run a couple of commands for finishing the compiling and installation of mpich.

**make**

**make install**

If we cd **/nfs/mpich3**, we will see folders containing the binaries and libraries of mpich. If we cd **/nfs/mpich3/bin**, we can see **mpi** binaries like **mpirun**.

Currently, we won’t be able to use **mpirun** from anywhere on the machine. We need to change the **~/.bashrc** file on **masternode** and **computingnode1** to globalize the **mpi** commands.

On both machines:

**vi ~/.bashrc**

At the bottom of ~/.bashrc, add the following two lines:

**export PATH=/nfs/mpich3/bin:$PATH**

**export LD\_LIBRARY\_PATH="/nfs/mpich3/lib:$LD\_LIBRARY\_PATH"**

PATH is used for bin folders, and LD\_LIBRARY\_PATH is used for lib folders. To reload the ~/.bashrc, type the following command on both machines:

**source ~/.bashrc**

1. **Using MPI binaries: Running MPI**

Make the folder called “projects” on “nfs” folder.

**mkdir /nfs/projects**

**cd /nfs/projects**

We need to create a hosts file that contains the IP addresses of all the IP’s that we want MPI to run.

**nano hosts**

**45.63.68.100**

**95.179.231.153**

MPI relies on ports for TCP and UDP packet communication. We will need to stop the firewalld for the process to hop between machines.

**systemctl stop firewalld**

We can test **mpirun** on a Linux command. The -f flag selects the host file that determines the IP addresses where MPI will decide to run the program. The -n flag determines the number of CPU cores that you want the program to run on..

**mpirun -f hosts -n 4 echo "hello world"**

The **mpirun** command prints the following if you have the machines in each other’s known\_hosts records.

**hello world**

**hello world**

**hello world**

**hello world**

**References**

<https://vultr.com>

<https://www.slothparadise.com/how-to-setup-mpi-on-centos-7-2-connected-virtual-machines-part-2/>