Building a Raspberry Pi HPC

Step 0: The Hardware

Parts list
- 3x Raspberry Pi 4 Model B — for the compute nodes
- 1x Raspberry Pi 4 Model B — for the master/login node
- 4x MicroSD Cards
- 4x USB-C power cables
- 1x 5-port 10/100/1000 network switch
- 1x 6-port USB power-supply (optional)
- 1x 128GB SSD (optional)

Step 1: Prepare Raspberry Pis

Download latest version of Raspbian Lite OS by using your MacOS/Linux terminal and type

```
wget https://downloads.raspberrypi.orgraspbian_lite_latest -O
raspbian_lite_latest.zip
```

Now extract the zipped file

```
unzip raspbian_lite_latest.zip
```

Check the directory for the contents from the extraction and find the name of the disk image file with extension .img (e.g. 2020-02-13-raspbian-buster-lite.img)

Now insert a SD/Micro-SD card inside your laptop and check for the attached devices/mount point of the card

For MacOs:

```
diskutil list
```

For Linux:

```
lsblk
```

Let's say it is attached to dev/disk2. First unmount the disk,

```
diskutil unmountDisk /dev/disk2
```

Then flash the image to memory card

```
sudo dd if=2020-02-13-raspbian-buster-lite.img of=/dev/disk2
```
If successful, a drive will be mounted under the name boot. Raspberry Pis usually comes with disabled SSH configuration. We don't want that. To enable it create an empty file inside the boot directory.

For MacOS, you can find it under /Volume/boot
Now, type

```
touch ssh
```

Now, we have successfully configured a Raspbian Lite OS having ssh enabled. Let's eject the card from the Mac

```
diskutil unmountDisk /dev/disk2
```

Repeat this process for all four memory cards. Now insert the cards to your Raspberry Pi s.

Now mark the master node to separate it from others.

Step 2: Network Setup

There are two different ways to do this. One with wireless/wired router with LAN ports and another way by using internet sharing facility from Mac or Windows. I'll explain the Mac way here.

First power up your 1x 5-port 10/100/1000 network switch and connect the master node (Raspberry Pi) through network cable.

MAC OS:
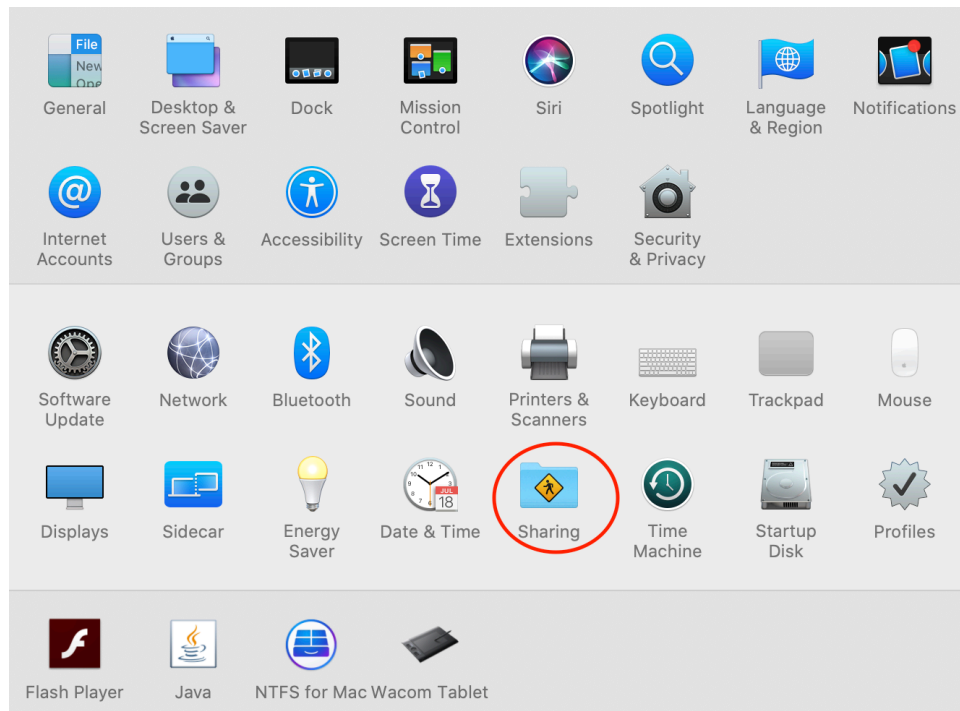
Assuming your laptop is connected to internet via wifi,

- Go to System Preferences
- Choose Sharing
- Click on Internet Sharing and choose share from Wifi to Ethernet

It will start a service which allows you to share the connection from the WiFi across the Ethernet connection (through the switch) to the Raspberry Pis.

Now power up the master node and wait for couple of minutes. The network sharing service will automatically assign some ip to the master node. To access master node via SSH, we need the ip assigned to the master node. If we run "ifconfig" command on a terminal, the shared network connection showed up as an adapter called Bridge100.

```
bridge100:
flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu
1500
     options=3<RXCSUM,TXCSUM>
     ether a3:4e:50:8e:1f:63
     inet 192.168.2.1 netmask 0xffffff00 broadcast
192.168.2.255
     inet6 fe80::a45e:60ff:fe7e:1d64%bridge100 prefixlen 64
scopeid 0xe
     Configuration:
```

```
            id 0:0:0:0:0:0 priority 0 hellotime 0 fwddelay 0
            maxage 0 holdcnt 0 proto stp maxaddr 100 timeout
1200
            root id 0:0:0:0:0:0 priority 0 ifcost 0 port 0
            ipfilter disabled flags 0x2
        member: en4 flags=3<LEARNING,DISCOVER>
              ifmaxaddr 0 port 10 priority 0 path cost 0
        Address cache:
        nd6 options=1<PERFORMNUD>
        media: <unknown type >
        status: inactive
```

Now use the following command to know ip assigned to the master node

```
arp -i bridge100 -a
```

Output:

```
? (192.168.2.2) at 3b:04:4e:3e:f7:b2 on bridge100 ifscope
permanent [ethernet]
```

It is most likely that it would assign ip starting from 192.168.2.2 to 192.168.2.255. Let's say it is 192.168.2.2

Step 3: Setting Up the Master Node

Now, log in to your master node using

```
ssh pi@192.168.2.2
```

Upon connection use password **raspberry**. (Note: it is the default password)

Now, we need to configure the node before starting to use.

```
pi@raspberrypi~$ sudo raspi-config
```

It opens up the config utility. You can change the default password if you want (highly recommended). Next you should set the locale, timezone, and wifi country. Then, select finish and press enter to exit the utility.

A snapshot of the utility screen is provided below.

++++++++++++++++ snapshot of rasps-config ++++++++++++++++++

Next,

SYSTEM UPDATE AND UPGRADE

```
pi@raspberrypi ~> sudo apt-get update && sudo apt-get upgrade
```

Now, we need to decide the hostnames for master node as well as cluster nodes. I would recommend to stick with the usual ones. Set "**master**" for master node and for cluster nodes starting from "**node01**" to "**node03**" (for 3 node cluster). Use the following command to set it up on master node.

```
pi@raspberrypi ~> sudo hostname master    # choice of yours
```

Change "**raspberrypi**" to "**master**" by editing the hostname file.

```
pi@raspberrypi ~> sudo nano /etc/hostname
```

Now edit the hosts file

```
pi@raspberrypi ~> sudo nano /etc/hosts
```

Add the following

++++++++++++++++ snapshot of hosts file ++++++++++++++++++

NETWORK TIME:

Now, since we are planning for a HPC system that uses a SLURM scheduler and the Munge authentication, we need to make sure that the system time is accurate. For that purpose we can install ntpdate package to periodically sync the system time in the background.

```
pi@raspberrypi ~> sudo apt install ntpdate -y
```

To apply the effect of changes that have been made so far reboot the system using the following command

```
sudo reboot
```

After, successful reboot, login to the master node again using ssh.

NEXT STOP, SHARED STORAGE:

The concept of cluster is based on idea of working together. In order to do so, they need to have access to the same files. We can arrange this mounting an external SSD drive (not necessary but convenient and faster) and exporting that storage as a network file system (NFS). It would allow us to access the files from all nodes.

First, insert the external storage into your master node. Now login to your master node using ssh and use the following command to see the dev location and mount point of your storage.

```
pi@master ~> lsblk
NAME          MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
mmcblk0       105:0    0   7.4G  0 disk
 ├─mmcblk0p1 105:1    0  43.8M  0 part /boot
 └─mmcblk0p2 105:2    0   7.4G  0 part /
sda           3:16     0  59.2G  0 disk
 └─sda1       3:17     0  59.2G  0 part
```

In our case, the main partition of the external storage is mounted at /dev/sda1 Before, using it as a NFS drive, we need to format it properly in ext4 file system. Use the following command to do that.

```
sudo mkfs.ext4 /dev/sda1
```

Now, we need to create a directory where the storage will be mounted. We can choose any name for that. But, lets choose something that is easy to remember "**nfsdrive**",

```
sudo mkdir /nfsdrive
sudo chown nobody.nogroup -R /nfsdrive
sudo chmod 777 -R /nfsdrive
```

Now, we need to configure to mount the storage during boot. For, that we need the the UUID of the storage. We can find that using the following command,

```
pi@master ~> blkid
```

Now, copy the number from /dev/sda1. It'll look like

```
UUID="78543e7a-4hy6-7yea-1274-01e0ff974531"
```

Now, open and edit fstab to mount the drive on boot.

```
pi@master ~> sudo nano /etc/fstab
```

```
Add the following line:
```

```
UUID=78543e7a-4hy6-7yea-1274-01e0ff974531 /nfsdrive ext4
defaults 0 2
```

All done, now we can mount the drive using the following command,

```
pi@master ~> sudo mount -a
```

If it fails for some reason, reboot the master node and try again. If it is still not working double check the process and look for typo.

ENABLE NFS SHARE

Now, we have a storage that can be shared but we need to install NFS server on master node in order to do so.

```
pi@master ~> sudo apt install nfs-kernel-server -y
```

Now, edit /etc/exports and add the following line to export

```
/nfsdrive
192.168.2.0/24(rw,sync,no_root_squash,no_subtree_check)
```

Remember, depending upon the IP address schema used on your local network, the ip will be different for setup. For example, if your master node ip is 10.0.0.1, then you need to replace the ip with 10.0.0.0/24.

Now, we can update the configuration of the NFS kernel with the following command,

```
sudo exportfs -a
```

One of the tasks for NFS to work remains unfinished which we will do in the next section.

Step 4: Setting Up the Worker Nodes

At this point, we already have the ssh access to all the worker nodes. But before configuring anything we need to prepare the worker nodes first. After connecting all the available nodes to the same network (just plug the Ethernet cables into the network switch), If we repeat one of the command in step 2,

```
arp -i bridge100 -a
```

We should be able to see the assigned IPs for the worker nodes like the following,

```
? (192.168.2.2) at 3b:04:4e:3e:f7:b2 on bridge100 ifscope
permanent [ethernet]
? (192.168.2.3) at 1a:54:3e:7b:27:ac on bridge100 ifscope
permanent [ethernet]
? (192.168.2.4) at 4c:b6:3h:66:g3:2a on bridge100 ifscope
permanent [ethernet]
? (192.168.2.5) at 0a:41:3h:bc:2a:k3 on bridge100 ifscope
permanent [ethernet]
```

The bracketed number is the assigned IP and the 12 digit address is our respective MAC address of raspberry pis.

Note: Do not worry if they are not in order and wait for a while if they do not appear at first. Repeat the command until you see all of them.

Now, we will assign some name to each worker node to refer them easily. We decide the following,

192.168.2.2: master (already assigned)

192.168.2.3: node01

192.168.2.3: node02

192.168.2.4: node03

Let's prepare them one by one. Log into node01 by using the following command,

```
ssh pi@192.168.2.3
```

It will ask for a password, use the default one "**raspberry**." It will open up a terminal, the same that you had for the master. Now configure the node using

```
pi@raspberrypi~$ sudo raspi-config
```

It opens up the config utility. Next you should set the locale, timezone, and wifi country. Then, select finish and press enter to exit the utility. Exactly same what you did for the master node, except the password.

Now let's update the hostname.
```
pi@raspberrypi ~> sudo hostname node01
```

Change "**raspberrypi**" to "**node01**" by editing the hostname file.

```
pi@raspberrypi ~> sudo nano /etc/hostname
```

Now edit the hosts file

```
pi@raspberrypi ~> sudo nano /etc/hosts
```

Add the following
++++++++++++++++++ snapshot of hosts file ++++++++++++++++++++

Next,

SYSTEM UPDATE AND UPGRADE

```
pi@raspberrypi ~> sudo apt-get update && sudo apt-get upgrade
```

Now reboot the system to apply the effect of changes that have been made so far.

```
pi@raspberrypi ~> sudo reboot
```

After the reboot, login to the system again.

NFS MOUNT

To access the storage that we shared on master node from individual worker nodes, we need to install and configure NFS services.

```
sudo apt install nfs-common -y
```

Now, we need to create the same directory in order to mount the storage.

```
pi@node01 ~> sudo mkdir /nfsdrive
pi@node01 ~> sudo chown nobody.nogroup /nfsdrive
pi@node01 ~> sudo chmod -R 777 /nfsdrive
```

To allow automatic mounting we need to edit the fstab file for each node. Use the following command to edit,

```
pi@node01 ~> sudo nano /etc/fstab
```

And add the following line below the existing texts

```
192.168.2.2:/nfsdrive    /nfsdrive    nfs    defaults   0 0
```

Now, use the following to finish the mounting

```
pi@node01 ~> sudo mount -a
```

To check whether the shared storage is working. Open a new terminal window and login to your master node. The create a blank file.

```
ssh pi@192.168.2.2

pi@master ~> cd /nfsdrive
pi@master ~> touch nas_test.dat
```

Now go back to the node01 terminal and check the contents of your shared directory

```
pi@node01 ~> cd /nfsdrive
pi@node01 ~> ls
```

If you see nas_test.dat file here, means you have successfully created a Network File System. If you can't see, you may have to reboot the node once.

Now repeat the process for rest of the worker nodes. Remember to replace "**node01**" word with their respective node numbers.

References:

1. https://medium.com/@glmdev/building-a-raspberry-pi-cluster-784f0df9afbd
2. https://medium.com/@glmdev/building-a-raspberry-pi-cluster-aaa8d1f3d2ca
3. https://medium.com/@glmdev/building-a-raspberry-pi-cluster-f5f2446702e8
4. https://epcced.github.io/wee_archlet/
5. https://scw-aberystwyth.github.io/Introduction-to-HPC-with-RaspberryPi/
6. https://github.com/colinsauze/pi_cluster
7. https://magpi.raspberrypi.org/articles/build-a-raspberry-pi-cluster-computer
8. https://www.hydromag.eu/~aa3025/rpi/