Raspberry Pi Cluster Setup Guide

Acknowledgements

Project lead: **Devin Smoot**, *SWOSU, Weatherford, OK*
Contributor/Implementation tester: **Madison Matli**, *SWOSU, Weatherford, OK*

To do list

- Add new sections:
  - Table of contents
  - What this guide covers
  - What you need for this guide (Hardware/software)
  - Who this guide is directed at (audience)
  - Conventions
  - Errata (Include known issues)
  - Fleshed out troubleshooting guide
- Add NFS section
- Multi-user setup section
- Script section for ease of cluster management
- Look at IPv4 forwarding for better solution
  - Router type setup with computer nodes on LAN side

Considerations to Consider before starting

If your SD card size will vary you will want to build the head node using the smallest size of SD card. This will ensure that the image for that SD card will ALWAYS be able to be written to a similar sized SD or larger. If you start with a 64GB SD card you will not be able to write the image to a 16GB SD card.

Head Node

Hardware:

- Raspberry Pi board x 1
- WiPi USB dongle x 1
- SD Card 16GB+ x 1
- Ethernet cable x 1
- HDMI cable x 1
- Power cable mini-USB x 1

Compute nodes

Hardware:

- Raspberry Pi board x 7
- SD Card 16GB+ x 1
- Ethernet cable x 1
- Pow er cable mini-USB x 1

Additional Hardware

- 10 Port USB hub
- 16 Port gigabit sw itch

Setup, Installation, and Testing

> ### Step 1 - Install operating systems

Install Raspbian Lite on SD card for head unit(s) and each compute node

Raspbian Lite

Raspbian Install Guides

> ### Step 2 - Initial Head Node Setup

Setup the locale settings to make sure the correct keyboard, language, timezone, etc are set. This w ill ensure w e are able to enter the correct symbols w hile w orking on the command line.

Configure Locale:

Log in w ith username: **pi** and passw ord **raspberry**

Start the Raspberry Pi configuration tool:

```
# raspi-config
```

> Setup Advanced Options:

Select **7 Advanced Options**

- Select **A3 Memory Split**
  - Enter **16**
  - Press **Enter**

> Setup Localisation Options:

Select **4 Localisation Options**

- Select Locale **I1 Change Locale**
  - Unselect **en_GB.UTF-8 UTF-8**
  - Select **en_US ISO-8859-1**
  - Press **Enter**
  - Select **en_US**

Select **4 Localisation Options**

- Select **I2 Change Timezone**
  - Select **US** (or appropriate country)
  - Select **Central** (or appropriate local timezone)

Select **4 Localisation Options**

- Select **I3 Change Keyboard Layout**
  - Use the default selected Keyboard
  - Press **Enter**
  - Select **Other**
  - Select **English (US)**
  - Select **English (US)**
  - Select **The default for the keyboard layout**
  - Select **No compose key**
  - Press **Enter**

Select **4 Localisation Options**

- Select **I4 Change Wi-fi Country**
  - Select **US United States**
  - Select **Ok**

> Setup Network Options:

Select **2 Network Options**

- Select **N1 Hostname**
  - Select **Ok**
  - Enter **head** for the hostname
  - Press **Enter**

Select **2 Network Options**

- Select **N2 Wi-fi**
  - Enter wi-fi SSID
  - Press **Enter**
  - Enter wi-fi passphrase
  - Press **Enter**

> Setup Interfacing Options:

Select **5 Interfacing Options**

- Select **P2 SSH**
  - Select **Yes**
  - Select **Ok**
  - Press **Enter**

Tab to **Finish**

Select **Yes** to reboot

Step 3 - Configure Network Settings

Edit */etc/network/interfaces*:

```
# nano /etc/network/interfaces
```

Add the follow ing to the end of the file:

```
auto lo
iface lo inet loopback

iface eth0 inet manual

allow-hotplug wlan0
iface wlan0 inet manual

wpa-conf /etc/wpa_supplicant/wpa_supplicant.conf
```

Setup *eth0* static ip address:

Edit */etc/dhcpcd.conf*:

```
# nano /etc/dhcpcd.conf
```

Add to the end of the file:

```
interface eth0
static ip_address=192.168.10.5
static domain_name_servers=8.8.8.8
```

Save and exit

Reboot:

```
# reboot
```

Step 4 - Update the system

```
# apt update && sudo apt upgrade -y
```

Reboot:

```
# reboot
```

Step 5 - IP forwarding for nodes to access internet

Setup IP forw arding so that all compute nodes w ill have access to the internet for package installation and to dow nload any needed materials on later use.

Log in w ith username: **pi** and passw ord **raspberry**

Enable IPv4 Forw arding and Disable IPv6:

```
# nano /etc/sysctl.conf
```

Add the follow ing lines to the end of the file (this includes the IP forw arding rule from above):

```
# Enable IPv4 forwarding
net.ipv4.ip_forward = 1

# Disable IPv6
net.ipv6.conf.all.disable_ipv6 = 1
net.ipv6.conf.default.disable_ipv6 = 1
net.ipv6.conf.lo.disable_ipv6 = 1
```

Save and exit

Update the configuration files:

```
# sysctl -p
```

Edit and Save the iptables:

```
# iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
# iptables -t nat -A POSTROUTING -o wlan0 -j MASQUERADE

# bash -c "iptables-save > /etc/iptables.rules"
```

Add settings to */etc/network/interfaces* file:

```
# nano /etc/network/interfaces
```

Add the follow ing line at the end of the w lan0 section under w pa-conf line to make the changes persistent:

```
pre-up iptables-restore < /etc/iptables.rules
```

Save and exit

Update */etc/hosts* file:

Add the follow ing to the end of the file:

Note: At this point you w ant to assign and name all of your nodes that **WILL** be in your cluster and enter them in the hosts file. Below is an example of a 6 node cluster including the head node as one of the six. This file w ill be copied w ith the image to the compute nodes and w ill save you a step of developing and deploying the hosts file later.

Edit */etc/hosts* file:

```
# nano /etc/hosts
```

Modify or add the follow ing lines to the file:

```
127.0.1.1    head

192.168.10.3        nodeX
192.168.10.5        head
192.168.10.100      node0
192.168.10.101      node1
192.168.10.102      node2
192.168.10.103      node3
192.168.10.104      node4
192.168.10.105         node5
192.168.10.106         node6
```

Reboot:

```
# reboot
```

Install MPICH

Install prerequisite *Fortran* w hich w ill be required for compiling MPICH. All other dependencies are already installed.

> ### Step 1 - Install Fortran

```
# apt install gfortran
```

> ### Step 2 - Install and Setup MPICH3

Create hpc group:

```
# groupadd hpc
```

Add pi user to hpc group:

```
# usermod -aG hpc pi
```

Create hpc directory in root:

```
# mkdir -p /software/lib

$ cd /software/lib
```

Take ow nership of /softw are:

```
# chown -R pi:hpc /software
```

Create build and install directory inside mpich3 directory:

```
$ mkdir mpich_3

$ cd mpich_3

$ mkdir build install
```

Dow nload mpich3 and untar:

```
$ wget http://www.mpich.org/static/downloads/3.3/mpich-3.3.tar.gz

$ tar xvfz mpich-3.3.tar.gz
```

Compile and install mpich3:

```
$ cd build

$ /software/lib/mpich_3/mpich-3.3/configure --prefix=/software/lib/mpich_3/install

$ make

$ make install
```

Activate environment variable:

```
$ export PATH=/software/lib/mpich_3/install/bin:$PATH
```

Add path to environment variables for persistance:

```
# nano ~/.bashrc
```

Add the follow ing to the end of the file:

```
# MPICH
export PATH="/software/lib/mpich_3/install/bin:$PATH"
```

Step 3 - Create list of nodes for MPI:

This list of nodes will need to be updated as you add nodes later. Initially you will only have the head node.

Create node list:

```
$ cd ~
# nano nodelist
```

Add the head node ip address to the list:

```
192.168.10.5
```

Note: Anytime you need to add a node to the cluster make sure to add it here as well as */etc/hosts* file.

Step 4 - Test MPI

Test 1 - Hostname Test

Enter on command line:

```
$ cd ~

$ mpiexec -f nodelist hostname
```

Output:

```
head
```

Test 2 - Calculate Pi

Enter on command line:

```
$ mpiexec -f nodelist -n 2 /software/lib/mpich_3/build/examples/cpi
```

Output:

```
Process 0 of 2 is on head
Process 1 of 2 is on head
pi is approximately 3.1415926544231318, Error is 0.0000000008333387
wall clock time = 0.003250
```

> ### Step 5 - Setup SSH keys

Note: *Must be executed from head node as pi user*

Generate SSH key:

```
$ cd ~

$ ssh-keygen -t rsa -P "" -f ~/.ssh/id_rsa -q
```

Transfer the key to the authorized_keys file:

```
$ cat ~/.ssh/id_rsa.pub > ~/.ssh/authorized_keys
```

Prepare for cloning

Shutdow n the head node:

```
# shutdown -h now
```

Save SD Image

At this point you w ill w ant to save an image of the head node. This w ill give you a fall back point if you make mistakes moving forw ard. You w ill also use this image to begin your node image.

Using the same guide as described in the beginning you w ill w ant to reverse the process of w riting an image to the SD and *read* an image from the SD and save that image to your PC. Now you have saved your SD like a checkpoint.

Sample name for SD image:

`compute_node_mpi_stage_2017_01_03`

Create Node image

The overview of this process:

1. Save image of *head node*.
2. On a new SD card w rite the *head node* image you just saved.
3. Boot the second SD you just created from the head node and make the follow ing changes for "Creating a Generic Node Image".
4. Save image of new ly created *generic compute node*.

At this point you have a copy of both the *head node* and *generic compute node* at the MPI stage. This is a checkpoint that you can fall back to if there are errors after this point.

Create Generic Node image

Completing this step w ill give you a node image that can be quickly w ritten to an SD card and distributed to expand your cluster. This w ill be a repeatable process w hen completed. You w ill setup an initial *compute node* image using your saved *head node* image. You w ill go in and change the hostname, hosts file (to match the hostname), and ip address (to a generic, alw ays easy to find ip address for w hen you need to configure the node after deployment) to *generic settings*. Doing this w ill allow you to alw ays access your *generic compute node* image at the same IP address and hostname. You w ill then be able to set up the compute node image to a specific IP address and hostname. Follow ing this process w ill allow for prompt and efficient deployment of a cluster.

Step 1 - Boot image and login

Log in w ith username: **pi** and passw ord **raspberry**

Step 2 - Enter a generic ip address

```
# nano /etc/dhcpcd.conf
```

Change the *eth0* ip address from:

```
static ip_address=192.168.10.5
```

To:

```
static ip_address=192.168.10.3
```

Also add to the end of the file:

```
static routers=192.168.10.5
```

Save and exit

Step 3 - Enter a generic hostname

```
# nano /etc/hostname
```

Change:

```
head
```

To:

```
nodeX
```

Save and exit

Step 4 - Edit hosts file

```
# nano /etc/hosts
```

Change:

```
127.0.1.1               head
```

To:

```
127.0.1.1               nodeX
```

Save and exit

Step 5 - Remove wireless connection information

Edit *interfaces* file:

```
# nano /etc/network/interfaces
```

Remove:

```
allow-hotplug wlan0
iface wlan0 inet manual

wpa-conf /etc/wpa_supplicant/wpa_supplicant.conf

pre-up iptables-restore < /etc/iptables.rules
```

Edit *wpa_supplicant.conf*:

```
# nano /etc/wpa_supplicant/wpa_supplicant.conf
```

If you are using a secure netw ork remove this section:

```
network={
ssid="<network name>"
psk="<network password>"
}
```

If you are using an unsecure netw ork remove this section:

```
network={
ssid="<network name>"
key_mgmt=NONE
}
```

> ### Step 6 - Shutdown and create a new image of the SD

```
# shutdown -h now
```

Now you w ill go back to WinDiskImager32 and save the image as a node image. This is a generic node image that you can quickly deploy and use to set up your cluster w ith.

Sample name for SD image:

`compute_node_mpi_stage_2017_01_03`

Setup Generic Node image

In this step you w ill configure a new ly deployed node from your generic node image you created. With this you w ill ssh in to the generic node's ip address and configure the hostname, hosts file, and ip address. Each of these w ill be set to the node's new permanent settings w ithin the cluster. When completing this step if you are deploying multiple new nodes you w ill need to pow er them up one at a time and configure them one at a time. This is due to all of the nodes using the same ip address. This allow s for all nodes to use a single point of reference to quickly find and deploy them.

Raspbian Install Guides

> ### Step 1 - Copy generic node image created earlier to an SD card using WinDiskImager32.
>
> ### Step 2 - Boot and login to your system

Log in w ith username: **pi** and passw ord **raspberry**

SSH into the new node:

```
$ ssh pi@nodeX
```

Enter **yes** to accept the key

Verify you are logged in:

Command prompt should read `pi@nodeX:~ $`

Step 3 - Adjust */etc/hostname* file

```
# nano /etc/hostname
```

Change:

```
nodeX
```

To:

```
node0
```

Save and exit

Note: This number w ill increment by one each time you add a node and must be unique on your cluster.

Step 4 - Adjust */etc/dhcpcd.conf*

```
# nano /etc/dhcpcd.conf
```

Change the *eth0* ip address from:

```
static ip_address=192.168.10.3
```

To:

```
static ip_address=192.168.10.100
```

Save and exit

Step 5 - Edit hosts file

```
# nano /etc/hosts
```

Change:

```
127.0.1.1                nodeX
```

To:

```
127.0.1.1               node0
```

Save and exit

> ### Step 6 - Expand Filesystem

Open configuration tool:

```
# raspi-config
```

Select **7 Advanced Options**

Select **A1 Expand Filesystem**

Select **Ok**

Tab to Select **Finish**

Select **Yes**

All settings should take effect on reboot

Deploy Head Node SSH Key

Issue the follow ing command from the head node for each node in the cluster:

Only run this command once the node is restarted w ith a node number.

```
# rsync -a --rsync-path="sudo rsync" ~/.ssh/authorized_keys pi@node0:~/.ssh/authorized_keys
```

Note: At this point you w ill just do this once to develop a compute node image w ith Slurm installed. After that is complete you w ill create a new generic image of the compute node. Once that is complete you can use that image to finish deploying your compute nodes for the rest of your cluster.

SSH in to the new node:

```
$ ssh pi@nodeX
```

Reboot the node:

```
# reboot
```

Install NTP

NTP is used to keep the cluster time close together using outside NTP servers to sync w ith the head node. All computer nodes w ill sync w ith the head node.

Reference:
http://raspberrypi.tomasgreno.cz/ntp-client-and-server.html
http://w w w .pool.ntp.org/zone/north-america

> **Head Node**
>
> Install NTP:

```
# apt install ntp
```

Edit the */etc/ntp.conf*:

```
# nano /etc/ntp.conf
```

Change:

```
pool 0.debian.pool.ntp.org iburst
pool 1.debian.pool.ntp.org iburst
pool 2.debian.pool.ntp.org iburst
pool 3.debian.pool.ntp.org iburst
```

To:

```
server 0.north-america.pool.ntp.org
server 1.north-america.pool.ntp.org
server 2.north-america.pool.ntp.org
server 3.north-america.pool.ntp.org
```

Restart NTP:

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

> **Compute Node**

SSH in to compute node:

```
$ ssh pi@node0
```

Install NTP on comput node:

```
# apt install ntp
```

Set Head Node as NTP server.

Edit */etc/ntp.conf*:

```
# nano /etc/ntp.conf
```

Under `restrict ::1` add:

```
restrict 192.168.10.0 mask 255.255.255.0
```

Change:

```
#broadcast 192.168.123.255
```

To:

```
broadcast 192.168.10.255
```

Save and exit

Restart NTP service:

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

Exit to head node:

```
$ exit
```

Install Slurm on Head Node

Slurm is the scheduler that organizes jobs to be run on the cluster. This interfaces with MPI and finds the most efficient ways to run jobs according to available resources. This must be installed after creating the base generic image as there is a Slurm controller that must be run on the head node. This is different from the Slurm client that is run on compute nodes.

[Slurm scontrol command]https://slurm.schedmd.com/scontrol.html
[Slurm configuration information]https://wiki.fysik.dtu.dk/niflheim/Slurm_configuration

> ### Step 1 - Install Slurm

```
# apt install slurm-wlm slurmctld
```

Step 2 - Add configuration file

Create the new Slurm configuration file */etc/slurm-llnl/slurm.conf*:

```
# nano /etc/slurm-llnl/slurm.conf
```

Add the follow ing to the file and save:

```
# slurm.conf file generated by configurator easy.html.
# Put this file on all nodes of your cluster.
# See the slurm.conf man page for more information.
#
ControlMachine=head
ControlAddr=192.168.10.5
#
#MailProg=/bin/mail
MpiDefault=none
#MpiParams=ports=#-#
ProctrackType=proctrack/pgid
ReturnToService=2
SlurmctldPidFile=/var/run/slurm-llnl/slurmctld.pid
#SlurmctldPort=6817
SlurmdPidFile=/var/run/slurm-llnl/slurmd.pid
#SlurmdPort=6818
SlurmdSpoolDir=/var/lib/slurm/slurmd
SlurmUser=slurm
#SlurmdUser=root
StateSaveLocation=/var/lib/slurm/slurmctld
SwitchType=switch/none
TaskPlugin=task/none
#
#
# TIMERS
#KillWait=30
#MinJobAge=300
#SlurmctldTimeout=120
#SlurmdTimeout=300
#
#
# SCHEDULING
FastSchedule=1
SchedulerType=sched/backfill
#SchedulerPort=7321
SelectType=select/linear
#
#
# LOGGING AND ACCOUNTING
AccountingStorageType=accounting_storage/none
ClusterName=raspi3
#JobAcctGatherFrequency=30
JobAcctGatherType=jobacct_gather/none
#SlurmctldDebug=3
SlurmctldLogFile=/var/log/slurm/slurmctld.log
#SlurmdDebug=3
SlurmdLogFile=/var/log/slurm/slurmd.log
#
#
# COMPUTE NODES
NodeName=node[0-6] Procs=1 RealMemory=768 State=UNKNOWN
```

```
PartitionName=raspi3 Default=YES  Nodes=node[0-6] State=UP MaxTime=INFINITE
```

Check if Slurm controller is running:

```
# scontrol show daemons
```

Output:

```
slurmctld
```

Step 3 - Create Munge key

```
# /usr/sbin/create-munge-key
```

Agree to overwrite.

Step 4 - Create log folder and take ownership

```
# mkdir /var/log/slurm

# chown -R slurm:slurm /var/log/slurm/
```

Step 5 - Finish installs and start services

```
# systemctl enable slurmctld.service
# ln -s /var/lib/slurm-llnl /var/lib/slurm
# systemctl start slurmctld.service
# systemctl enable munge.service
# systemctl restart munge.service
```

Verify Slurm controller is running:

```
# systemctl status slurmctld.service
```

Will return feedback to the screen. Verify *Active* line states: ***active (running)***.

Verify Munge is running:

```
# systemctl status munge.service
```

Will return feedback to the screen. Verify *Active* line states: ***active (running)***.

Step 6 - Add user to Slurm group

```
# adduser pi slurm
```

Check Slurm status:

```
$ sinfo
```

Output:

```
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
raspi3*      up   infinite      7   idle node[0-6]
```

Install Slurm on Compute Node

Step 1 - Copy Slurm configuration and Munge files from *Head Node*

On *head node*:

Give rsync proper permission to run:

```
# visudo
```

Add the follow ing to the end of the file:

```
<username> ALL=NOPASSWD: /usr/bin/rsync *
```

On *compute node*:

SSH in to compute node:

```
$ ssh pi@node0
```

Give rsync proper permission to run:

```
# visudo
```

Add the follow ing to the end of the file:

```
<username> ALL=NOPASSWD: /usr/bin/rsync *
```

Exit back to head node:

```
$ exit
```

Step 2 - Install Slurm daemon

Execute on *node0*:

SSH in to *node0*:

```
$ ssh pi@node0
```

Install Slurm daemon and Slurm client:

```
# apt install slurmd slurm-client
# ln -s /var/lib/slurm-llnl /var/lib/slurm
```

Create log folders and take ow nership:

```
# mkdir -p /var/log/slurm

# chown -R slurm:slurm /var/log/slurm
```

Take ow nership of Slurm run folder:

```
# chown -R slurm:slurm /var/run/slurm-llnl
```

Exit to head node:

```
$ exit
```

On *head node*:

Copy Munge and Slurm configuration files from head node to compute node:

See "Generate and distribute Munge key script" in the Troubleshooting section

```
# rsync -a --rsync-path="sudo rsync" /etc/slurm-llnl/slurm.conf pi@node0:/etc/slurm-llnl/slurm.conf
```

On *compute node*:

SSH in to *node0*:

```
$ ssh pi@node0
```

Take ow nership of *munge.key* file:

```
# chown munge:munge /etc/munge/munge.key
```

Finish install and start Slurm and Munge:

```
# systemctl enable slurmd.service
# systemctl restart slurmd.service
# systemctl enable munge.service
# systemctl restart munge.service
```

Verify Slurm daemon is running:

```
# systemctl status slurmd.service
```

Will return feedback to the screen. Verify *Active* line states: ***active (running)***.

Verify Munge is running:

```
# systemctl status munge.service
```

Will return feedback to the screen. Verify *Active* line states: ***active (running)***.

Step 3 - Add user to Slurm group

```
# adduser pi slurm
```

Execute on *head node*:

```
# scontrol reconfigure
```

Check node status:

```
$ sinfo
```

This should show all nodes in an idle state.

Deploying the Rest of the Cluster

By now you have developed a head node image that contains both MPI and Slurm. You have also developed a compute node image that contains both MPI and Slurm as w ell. Now you should go back to the instructions for "Create Node Image" to save both images and then use the compute node image to finish deploying your cluster. Saving these images at each stage gives you different configurations that you can easily deploy in the future and also allow s you to have a checkpoint in case something goes w rong. You can w rite the saved node image to your SD and start from that point rather then starting from the beginning.

You should now have a working Raspberry Pi cluster.

Add an ethernet adapter

> ### Add *eth0*:
>
> Edit */etc/network/interfaces* file:

```
# nano /etc/network/interfaces
```

Add below eth0 section:

```
auto eth1
iface eth1 inet manual
```

Change or add iptables rule to end of file:

```
pre-up iptables-restore < /etc/iptables_wired.rules
```

Edit */etc/dhcpcd.conf* file:

Add the follow ing to the end of the file:

```
interface eth1
static ip_address=192.168.1.XXX
static domain_name_servers:8.8.8.8
static routers=192.168.1.1
```

Create iptables rules file:

Flush the iptables in Memory

```
sudo iptables -F
```

Rebuild the rules and file:

```
sudo iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
sudo iptables -t nat -A POSTROUTING -o eth1 -j MASQUERADE
```

Save the iptables_w ired.rules file:

```
sudo bash -c "iptables-save > /etc/iptables_wired.rules"
```

Check for the iptable rules in the */etc/network/interfaces file*:

Make sure that the line below is present and not commented out:

```
pre-up iptables-restore < /etc/iptables_wired.rules
```

Disable *wlan0*:

Edit */etc/wpa_supplicant/wpa_supplicant.conf* file:

```
# nano /etc/wpa_supplicant/wpa_supplicant.conf
```

Comment out the `network={ connection information }` section (all lines)

Now all traffic for the cluster is routed through eth0 and out eth1 to the internet. Any returning traffic or dow nloads come in via eth1 and through eth0 to the cluster unless its meant for the head node.

NFS

Choose a disk:
You can use either a standard external hard drive, or a USB flash drive. If using an external hard drive you w ill w ant one that has its ow n pow er plug. Draw ing pow er from the Raspberry Pi may cause an undervoltage situation.

Formatting the disk:
For this use w e w ill be formatting in FAT32 for ease of use.

Plug in the drive to a USB port.

Install the softw are needed and format the USB drive:

```
# apt-get install dosfstools
# mkfs.vfat /dev/sda1 -n USB
```

Mount the disk to */users* directory:

```
# mkdir /users
# chown -R pi:hpc /users
# mount /dev/sda1 /users -o uid=pi,gid=pi
```

Add automatic mounting on boot:

Add the follow ing to */etc/fstab* file:

```
/dev/sda1 /users auto defaults,user 0 1
```

Install the NFS server on the head node:

```
# apt install nfs-server
```

Add the follow ing to the */etc/exports* file:

```
/users 192.168.10.5/24(rw,sync)
```

Restart RPC services:

```
# update-rc.d rpcbind enable && sudo update-rc.d nfs-common enable
# reboot
```

Mount the disk from another Raspberry Pi node:

Install required softw are:

```
# apt install nfs-common autofs
```

Create the mount point:

```
# mkdir /users
# chown -R pi:hpc /users
```

Add the follow ing to the */etc/auto.master* file:

```
/mnt/nfs /etc/auto.nfs
```

Create the */etc/auto.nfs* file and add the follow ing:

```
pi   192.168.10.5:/users
```

Restart the `autofs` service:

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

Add the follow ing to the end of the */etc/fstab* file:

```
192.168.10.5:/users   /users  nfs     auto    0       0
```

https://medium.com/@aallan/adding-an-external-disk-to-a-raspberry-pi-and-sharing-it-over-the-netw ork-5b321efce86a

https://raspberrypi.stackexchange.com/questions/87057/cannot-automatically-mount-nfs-share-to-raspberry-pi

Scripts

Create a */software/scripts* folder:

```
sudo mkdir /software/scripts
sudo mkdir /software/files
```

Edit *.bashrc* file:

```
sudo nano ~/.bashrc
```

Add to the end of the file:

```
# SCRIPTS
export PATH="/software/scripts:$PATH"
```

Add scripts to the */software/scripts* folder to use as commands system w ide.

Deploy file to all compute nodes

This script w ill deploy files to all nodes to a folder defined by the user. This is a w orkaround for a multi-user environment.

```
#!/bin/bash

if [ "$1" == "-help" ] || [ "$1" == "" ]; then
    echo -e "Command    \tExample"
    echo "-----------------------------------------------------------"
    echo "deploy_file   deploy_file <filename> <destination folder>"
    echo -e "\t\tNote:Default destination folder is '/software/files'"
    echo "Help          deploy -help"
    exit
fi
```

```
if [ "$1" != "" ]; then
  if [ "$2" == "" ]; then
        filelocation=/software/files
  else
        filelocation=$2
  fi
    echo "Transferring file: $1 to node0:$filelocation"
    rsync $1 pi@node0:$filelocation
    echo "Transferring file: $1 to node1:$filelocation"
    rsync $1 pi@node1:$filelocation
    echo "Transferring file: $1 to node2:$filelocation"
    rsync $1 pi@node2:$filelocation
    echo "Transferring file: $1 to node3:$filelocation"
    rsync $1 pi@node3:$filelocation
    echo "Transferring file: $1 to node4:$filelocation"
    rsync $1 pi@node4:$filelocation
    echo "Transferring file: $1 to node5:$filelocation"
    rsync $1 pi@node5:$filelocation
    echo "Transferring file: $1 to node6:$filelocation"
    rsync $1 pi@node6:$filelocation
else
    echo "All nodes are already defined in the script"
    echo "Please enter a filename and destination folder: ie. deploy_file <filename> <destination folder>"
fi
```

Mutli-user setup script

This w ill create and configure users from the head node. It w ill also remove users w ith options.

```
#!/bin/bash

# Creation script for user creation on SWOSU Raspberry Pi 3 cluster

# Usage display

# Local user creation

# Remote user creation on all nodes

# Set required environment variables

# Help display
```

Troubleshooting Section

Generate and distribute new SSH key script

Generate and distribute new Munge key script

Create a new file in pi user home directory called munge-key-gen.sh:

```
nano ~/munge-key-gen.sh

#!/bin/bash

# Create a new munge key on head node
sudo /usr/sbin/create-munge-key
```

```
for i in {0..6}
do
        # Copy to compute nodes
        sudo cat /etc/munge/munge.key | ssh pi@node$i "sudo cat >> ~/munge.key"

        # Remove old munge.key file
        ssh pi@node$i "sudo rm /etc/munge/munge.key"

        # Move from /home/pi/ to /etc/munge/
        ssh pi@node$i "sudo mv ~/munge.key /etc/munge/munge.key"

        # Take ownership of munge.key by munge user
        ssh pi@node$i "sudo chown munge /etc/munge/munge.key"

        # Change permissions of group and world
        ssh pi@node$i "sudo chmod g-rw,o-rw /etc/munge/munge.key"

        # Restart munge service
        ssh pi@node$i "sudo service munge restart"
done
```

> ### Received SIGHUP or SIGTERM from Nano
>
> Enter the command:
>
> ```
> bash
> ```
>
> ### NETWORK UNREACHABLE:
>
> When experiencing netw ork connectivity problems w ith compute nodes:

1. Flush the iptables in Memory

   sudo iptables -F

2. Rebuild the rules and file
   Repeat the IP tables section of the guide, starting w ith the commands:

   sudo iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
   sudo iptables -t nat -A POSTROUTING -o eth1 -j MASQUERADE

3. Save the iptables_w ired.rules file:

   sudo bash -c "iptables-save > /etc/iptables_w ired.rules"

4. Check for the iptable rules in the */etc/network/interfaces file*:

Make sure that the line below is present and not commented out:

```
pre-up iptables-restore < /etc/iptables_wired.rules
```

5. Reboot:

sudo reboot

If it is missing then add it to the end of the file. Save and exit.

RSYNC ISSUES:

If having trouble w ith using rsync commands:

Setup Rsync:

On *Both nodes*

Edit the /etc/sudoers file:

```
sudo visudo
```

Add this line to the end of the file:

```
<username> ALL=NOPASSWD: /usr/bin/rsync *
```

MPI ISSUES

If mpiexec command fails to execute, stalls, or displays an error message about an unreadable path file:

- Mpich3 could be in the w rong directory
- Make sure the export path correlates to the actual install path for MPICH3
- Reinstalling MPICH3 and setting up the proper environment variables can fix many problems, re-evaluate the MPICH3 install instructions and verify all settings before attempting a reinstall.

SSH ISSUES

If the Pi is displaying SSH errors w hen running the mpiexec command:
Check the problematic node's authorized_keys file, and compare it w ith the head node's authorized_keys file.

Check the file by going to the SSH directory:

```
cd ~/.ssh
```

Now check the file information for *authorized_keys* file:

```
ls -ls
```

The filesize is listed after the ow ner and group names.

These file should be identical in length, if not redistribute the head node's authorized_keys file to the compute node using the follow ing command:

```
rsync -a --rsync-path="sudo rsync" ~/.ssh/authorized_keys pi@nodeX:~/.ssh/authorized_keys
```

COMMANDS TO CHECK SERVICE STATUSES

These commands do the same thing, just with a different syntax:

```
sudo systemctl [start,stop,restart,status] <service name>

sudo service <service name> [start,stop,restart,status]

sudo /etc/init.d/<service name> [start,stop,restart,status]
```

ENABLING/DISABLING NETWORK INTERFACE CONNECTIONS

This is a quick way to bring down and bring back up network interfaces without restarting.

Disable the specified connection

```
sudo ifdown <connection name>
```

Enable the specified connection

```
sudo ifup <connection name>
```

SLURM ISSUES

Make sure the slurm.conf file is identical across all nodes.

Use `sudo scontrol reconfigure` to distribute the slurm.conf from the head node to all compute nodes responding.

When running the service status command, read the error messages that are displayed: ***these messages are vital in order to troubleshoot current problems***.

PROBLEMATIC NODES

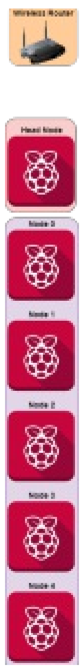
On many occasions, certain nodes fail to work because of a software/hardware malfunction. This can be fixed by removing and reinstalling the software. Hardware problems can be fixed by reformatting the node's SD card, and rewriting it with a functional node image. Also check each Ethernet cable for weaknesses, and verify that each node in the cluster is properly connected.

-For Pi 3 Clusters: The head node is connected via Wi-Fi, and each compute node uses the head node's wireless connection to download files.

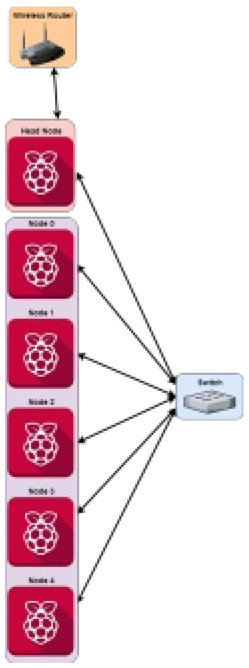
-For Pi 2 Clusters: A Wi-Pi adapter is a tested solution for establishing a wireless connection with a Raspberry Pi model 2. Using other wireless adapters could result in incompatible drivers or other various issues. The head node can also be connected to the Internet via an Ethernet cable.

Network Diagrams

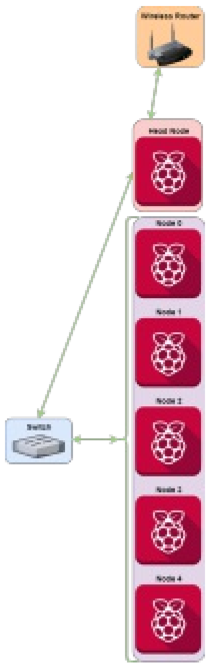
Base Equipment Layer (Pictured Below)



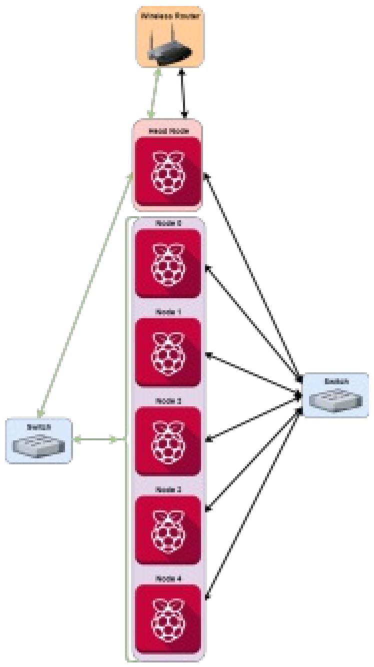
Physical Layer (Pictured Below)



Logical Layer (Pictured Below)



Physical and Logical Layers (Pictured Below)



References

https://www.modmypi.com/blog/how-to-give-your-raspberry-pi-a-static-ip-address-update

https://www.raspberrypi.org/forums/viewtopic.php?f=28&t=44609

http://weworkweplay.com/play/automatically-connect-a-raspberry-pi-to-a-wifi-network/

https://w w w.raspberrypi.org/forums/view topic.php?f=36&t=162096

https://w w w.raspberrypi.org/forums/view topic.php?t=118804&p=808453

https://w w w.raspberrypi.org/forums/view topic.php?f=28&t=37575

http://unix.stackexchange.com/questions/88100/importing-data-from-a-text-file-to-a-bash-script

http://w w w.tldp.org/LDP/abs/html/arrays.html

http://how -to.w ikia.com/w iki/How _to_read_command_line_arguments_in_a_bash_script

http://ryanstutorials.net/bash-scripting-tutorial/bash-variables.php

http://stackoverflow .com/questions/19996089/use-ssh-to-start-a-background-process-on-a-remote-server-and-exit-session

http://stackoverflow .com/questions/29142/getting-ssh-to-execute-a-command-in-the-background-on-target-machine

http://w w w.igeekstudio.com/blog/setting-up-ganglia-and-hadoop-on-raspberry-pi

http://ccm.net/faq/2540-linux-create-your-ow n-command

https://github.com/XavierBerger/RPi-Monitor

https://w w w.raspberrypi.org/blog/visualising-core-load-on-the-pi-2/

https://github.com/davidsblog/rCPU

https://raseshmori.w ordpress.com/2012/10/14/install-hadoop-nextgen-yarn-multi-node-cluster/

https://w w w.packtpub.com/hardw are-and-creative/raspberry-pi-super-cluster