



ILLINOIS INSTITUTE OF TECHNOLOGY

ADVANCED OPERATING SYSTEMS
CS 550

System Manual

Jose Toledo
A20414915

Venkata Krishna Chaitanya
Polavarapu
A20378279

John Dugger
A20406586

Keith Bateman
A20347343

Viswatej Kasapu
A20405357

May 5, 2018

Revision History

Revision	Author	Date	Status and Description
1	Venkata	2018-05-01	Initial version
2	Viswatej	2018-05-01	Added Hardware Section
3	John	2018-05-02	Added Cable management
4	Kieth	2018-05-02	Updated Cable management
5	Jose	2018-05-03	Added Software Installation, Repo Setup
7	Venkata	2018-05-04	Added FAQ
6	Jose	2018-05-04	Minor changes

Contents

1	Introduction	4
2	Hardware	5
2.1	Testing Hardware	5
2.2	Setting up Hardware	6
2.3	Cable Management	6
2.3.1	Cable Grouping	6
2.3.2	Cable Routing	6
2.3.3	Port Mapping Rules	7
3	Software	9
3.1	Setup Disk Array	9
3.2	Setup Master Node	9
3.2.1	RAID Array	9
3.2.2	Install OS	9
3.2.3	Install Ethernet drivers	10
3.2.4	Setup InfiniBand	13
3.2.4.1	Connection setup	13
3.2.4.2	Subnet Manager configuration	13
3.2.5	Install xCAT	15
3.2.6	Firewall Setup	15
3.2.7	NFS Setup	17
3.3	Pre-installation Network Setup	17
3.3.1	Network design	17
3.3.2	Modify the IPMI address of compute nodes	18
3.4	Install OS on Compute Nodes	20
3.4.1	Setup xCAT	20
3.4.2	Consistent Interface naming	23
3.4.3	Enable serial communication	25
3.5	Local repositories configuration	25
3.6	Post-installation network setup	27
3.7	Software Installation	27
3.7.1	OpenHPC base installation	27
3.7.2	SLURM installation	27
3.7.3	OpenHPC NFS configuration	28
3.7.4	Development tools installation	28
3.7.5	MPI stacks installation	29
3.8	Software Testing	29
3.8.1	IPoIB	29
3.8.2	SLURM	30

4 **FAQ**

32

1 Introduction

From the existing cluster of 64 nodes, we needed to setup two clusters: one storage cluster with 32 nodes and one high performance computing cluster with an InfiniBand network, 15 compute nodes and a master node.

Existing Hardware: The list are total count of RAM's, Hard disks, SSD's, Infiniband network cables from existing nodes

- RAM: 274
- Hard disks: 70
- SSD: 49
- InfiniBand Network adapter: 16

New Hardware Configuration:

- Storage Cluster Nodes:
 - (1) 256GB SATA SSD (SAMSUNG MZ-76P256BW) installed at SATA0
 - (1) 1TB SATA HDD (Seagate ST1000NM0008) installed at SATA1
 - (1) 40Gbps Ethernet PCI Network Interface Card (NIC)
 - (16) 2GB DDR2 PC2-5300 667MHz Buffered RAM DIMMs (NOT fully buffered)
 - note: at the time of writing this manual, only (4) RAM DIMMs are installed in each storage node due to incorrectly ordering fully buffered RAM
- High Performance Computing (HPC) InfiniBand Cluster Nodes:
 - (1) 1TB SATA HDD (Seagate ST1000NM0008) installed at SATA0
 - (1) 100GB PCI OCZ RevoDrive X2 (OCZSSDPX-1RVDX0100)
 - (1) Mellanox Technologies MT25208 InfiniHost III Ex (rev a0) PCI NIC
 - (4) 2GB DDR2 PC2-5300 667MHz Buffered RAM DIMMs (NOT fully buffered)

The hardware installation can be broken down in to 3 steps

1. Testing hardware
2. Setting up hardware
3. Cable management

The software installation can be broken down in to 8 steps

1. Setup Disk array

2. Setup Master Node
3. Pre-installation network setup
4. Install OS on the compute nodes
5. Configure local repositories for the compute nodes
6. Post-installation network setup
7. Install additional software
8. Test basic cluster functions

2 Hardware

2.1 Testing Hardware

The basic testing of existing hardware included the following tests:

1. Test whether each existing node is booting up or not. A boot test is just a basic test to give overview of how many nodes are working and if one is not working, we can check which part of that node is causing trouble.
2. Test all RAM DIMMs in known good nodes. We initially used MEMTEST86+ to test existing DIMMs, but transitioned to MEMTEST86 since it has been maintained more recently. We used Ultimate Boot CD to load this tool into a bootable USB. To test DIMMs, place them in a working node, and boot into MEMTEST86 by booting from the USB drive. Each node has 16 memory slots; therefore, 16 DIMMs can be tested at the same time. A single pass of MEMTEST86 for 16 2GB DIMMs typically takes 4 hours and two passes is considered the standard to verify that the DIMMs are free of errors. MEMTEST86 automatically starts a second pass when the first pass finishes. Although we didn't encounter any errors during our testing, if an error was identified, MEMTEST86 cannot tell you which DIMM had the error. Isolating the failed DIMM(s) requires splitting the 16 DIMMs into sequentially smaller batches until the failed DIMM(s) is(are) identified.
3. The existing SSDs are considered legacy hardware by OCZ and are no longer supported by any OCZ software to verify drive integrity. Seagate SeaTools is able to test them; however, the tests offered by SeaTools only provide pass/fail results. Since we only intended to use a subset of the existing SSDs, we wanted to quantify the remaining drive life of each SSD. So, we used Smart-Monitoring Analysis and Reporting Technology (SMART) built into each drive to read the recorded SMART data for each of the SSDs. Specifically, we checked the *reallocated_sector_count* since it was strongly correlated to a higher failure rate in a research paper published by Google in 2007 (<https://research.google.com/pubs/>

pub32774.html). The reallocated sector count is incremented each time a drive remaps a faulty sector to a new physical sector drawn from a pool of spares, which occurs when the drive's logic believes that a sector is damaged, typically as a result of recurring soft errors or a hard error. The lower this value is, the fewer bad sectors exist in the SSD.

2.2 Setting up Hardware

After testing existing hardware, we purchased and tested the additional HDDs and SSDs we needed to complete the node configurations.

1. Since the hard disks are from Seagate and Seagate offers a bootable drive analysis tool, we used Seagate SeaTools for DOS to test the new HDDs and SSDs. (<https://www.seagate.com/support/downloads/seatools/>). Similarly to MEMTEST86, we installed it on a bootable USB and used the Short Drive Self Test which provided a PASS/FAIL result.
2. Once we configured each node, we performed a boot test by booting into the BIOS to verify that the BIOS was properly recognizing the installed hardware.
3. We then placed all 32 nodes with 40 Gbps Network card in one rack as the storage cluster and other 15 nodes with InfiniBand network card in another rack as the InfiniBand (IB) cluster. Nodes should always be installed from the bottom of the rack working up to prevent the server rack from becoming top-heavy which could result in it tipping over.

2.3 Cable Management

An important part of cable management is to make sure each cable such as the power cable or Ethernet cable will all follow a mapping rule to connect to make it easy to identify which cable is connected to which node. Additionally, we utilized the space between the node rails and the side of the server rack place excess cable slack so that the cabinet won't be messy.

2.3.1 Cable Grouping

In general, we divided nodes into groups of **six** nodes starting from bottom node in rack. We used Velcro to group cables of these nodes into one and we inserted them to the sides of cabinet.

2.3.2 Cable Routing

We routed the power cables to the outlets built into the sides of the rack. These outlets are grouped into **seven** outlets together. Similarly to previous step, **six** nodes from bottom node are connected to **six** power outlets which are at lower part of cabinet. The next six nodes are

connected to power sockets in next group of sockets. We left the bottom power outlet of each group unused to maintain a better mapping between power outlets and node groups. Bottom to top in the rack outlet groups corresponds to bottom to top for nodes.

2.3.3 Port Mapping Rules

Network cables consist of three types: standard Ethernet cables, InfiniBand cables, and 40 Gbps Ethernet. In each cluster, we installed the switches on top of the nodes. Bottom to top in the cluster generally corresponds to left to right in a switch.

On the master node, there are 4 ports. From left, 1st port is unused. 2nd port is for internet, 3rd is used for network management and 4th is for IPMI

- Storage Cluster

- Each node has one standard Ethernet cable connected to the Ethernet switch. We used a mapping rule for each cable from node to respective switch port. The mapping rule is **starting from bottom node, node 1 (which is bottom most node) cable is connected to left most port in switch. Node 2 (from bottom) cables are connected to second port (from left) in the switch.** Here, in switch, we are using bottom rows of ports (since each switch has two rows of ports) because bottom row is used for Network Management.
- Each node also has one 40Gbps Ethernet cable connected to the 40Gbps Ethernet switch. Port mapping for the 40Gbps cables is the same as for the standard Ethernet cables.
- Each node has a power cable connected to the rack on the left-hand side (looking from the rear). The sequence of groups for power cables in the storage cluster is special, and **consists of groups of 6, 7, 6, 7, and 6 nodes ordered from bottom to top.** The top group of outlets in the rack is used for switches.

- InfiniBand Cluster

- Each node has two standard Ethernet cables each connected to a different VLAN in the Ethernet switch. The top port of each node is connected to the top port of the switch for IPMI. The bottom port of each node is connected to the bottom port of the switch for cluster data. Since each switch has two rows of ports and each node as two ports. The bottom port of each node is connected to bottom row of switch (for Network Management) and upper port of each node is connected to upper row of switch (for IPMI) See section [3.3.1](#), Network Design, below.
- Each node also has one InfiniBand cable connected to the InfiniBand switch using the 1:1 mapping sequence (Each port has an associated number in physical space written on the switch. The nth port is associated with the nth node, bottom to top). For the InfiniBand switch, this amounts to nodes 1-12 from bottom to top

being connected to the bottom ports from left to right, and the remaining nodes from bottom to top being connected to the top ports from left to right. There are 4 cables in the last group instead of the usual 6. The head node is not grouped with other cables and connected to port 24 (the upper-rightmost port).

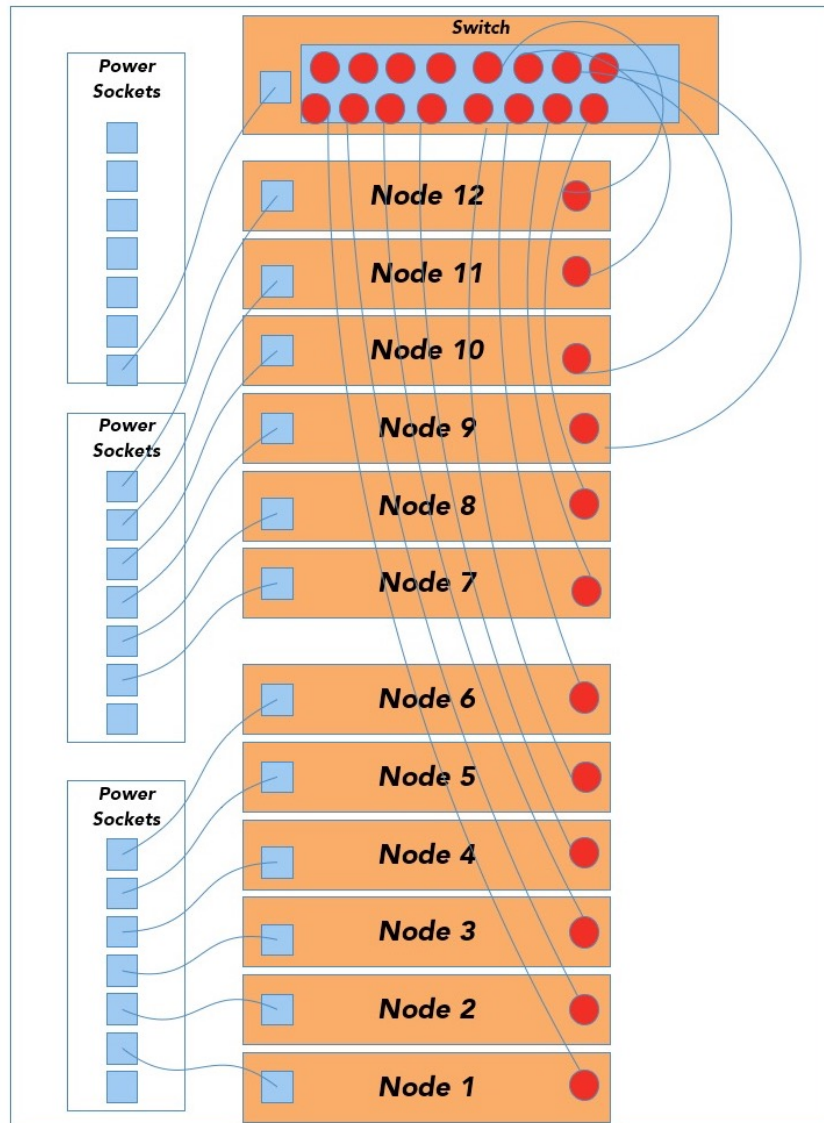


Figure 1: Cable Management

The below figure gives the sample mapping of 12 nodes and a switch along with power cable connections. Ethernet port of node 1 is connected to left most port in switch. And Ethernet port of node 2 is connected to port 2 from left in switch and so on. Here, blue rectangles are power sockets and red circles are Ethernet ports.

3 Software

At the time that this manual was generated, the only cluster we were able to install software on was the InfiniBand cluster due to the storage cluster being dependent upon the new Lenovo cluster being installed and configured. All discussions in the software section refer to the setup of the InfiniBand cluster; however, many of the same principles remain the same. The inspiration for this project comes from OpenHPC stateful recipe from [here](#).

3.1 Setup Disk Array

Take necessary backups of all the files. Since no new hard disks, nothing was modified for the disk array as part of this project.

3.2 Setup Master Node

3.2.1 RAID Array

There were **four 2.5” HDD** (ST914602SSUN146G) configured in **RAID-1**. Replace them with **two 2.5” SSD** (Samsung 860 Pro 256GB) configured in **RAID-1**. The RAID setup can be modified at the time of the boot. Using Ctrl-A during the boot to enter in to the RAID controller setup which would allow you choose the disk slots and RAID mode. Once this is complete the disks are ready for the OS to be installed.

3.2.2 Install OS

We used Live USB of CentOS, more specifically **CentOS-7-x86_64-DVD-1708.iso**, to install OS to the Master node. Live USB was created using the **dd** tool, we recommend using at least 8GB of pen drive as the iso itself is around 4 GB. Assuming the pen drive is in **/dev/sdb** we use the following command

```
# Write iso to /dev/sdb to create Live USB
dd if=CentOS-7-x86_64-Minimal-1708.iso of=/dev/sdb
```

Plug the Live USB in to the master node and follow the standard GUI installation. **Note that BIOS has issues detecting USB on the front of the Master node.** The following partitioning is used on the master node. All volumes use **xfs** file system, with no encryption


New CentOS 7 Installation		
SYSTEM		
/boot sda1		488 MiB
/ centos-root		230.44 GiB >
swap centos-swap		7629.39 MiB

Figure 2: Master Node partition

3.2.3 Install Ethernet drivers

In CentOS 7.4, the necessary drivers for Nvidia MCP55 ethernet controllers, which the master node has, are not available by default. Now we have Chicken and Egg problem. Driver download needs Internet, but drivers are need for Internet. To solve this we used a USB to Ethernet cable. The USB to Ethernet adapter was a plug and play device. Download the drivers for the Nvidia



Figure 3: Ethernet to USB adapter

MCP55 from [elrepo](http://elrepo.org) and install them using rpm.

```
# Update the software on the existing installation
[hec]$ yum update
# Install wget
[hec]$ yum install wget
# Download the driver
[hec]$ wget "http://elrepo.org/linux/elrepo/el7/x86_64/RPMS/
           kmod-forcedeth-0.64-3.el7.elrepo.x86_64.rpm"
# Install the drivers
[hec]$ rpm -Uvh kmod-forcedeth-0.64-3.el7.elrepo.x86_64.rpm
```

Unplug the USB to Ethernet device and restart the master node. You should be able to see all the available interfaces. You can check it with `ip address` or `ip link`. Now configure the interfaces via `nmtui`. [Reference](#) here for more details. `nmtui` provides a text interface to

configure the interfaces. Select *Edit a connection* to change the configuration of interface. Once configured select *Activate a connection* to bring up the connection. Some screenshots below

We strongly recommend you not to connect to the Internet until the firewall is setup. If you do need the Internet please enable the firewall before you connect.

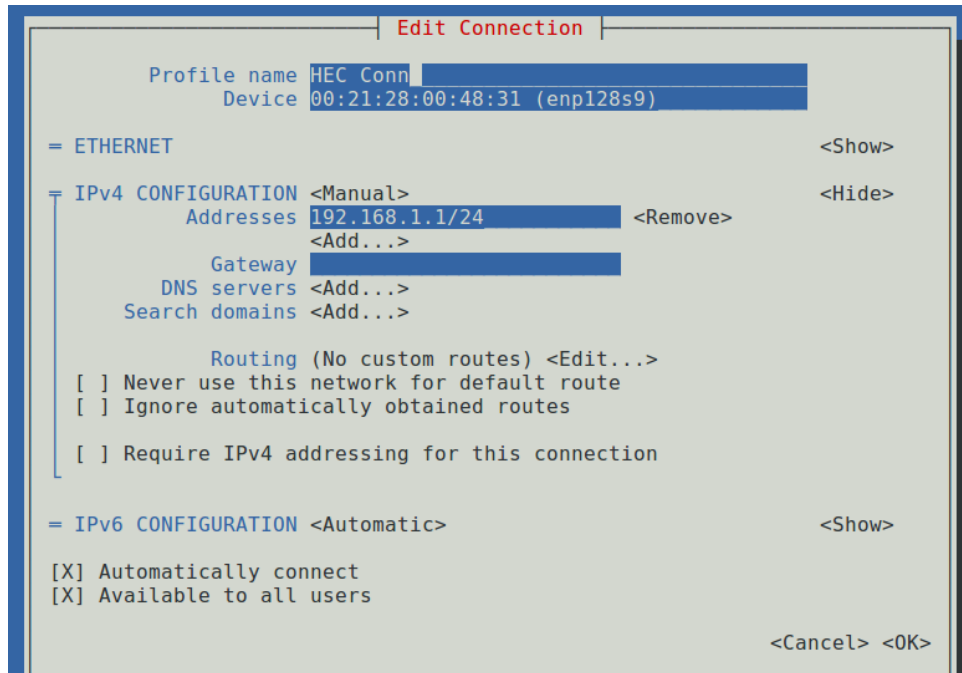


Figure 4: Connection details of LAN connecting to compute nodes

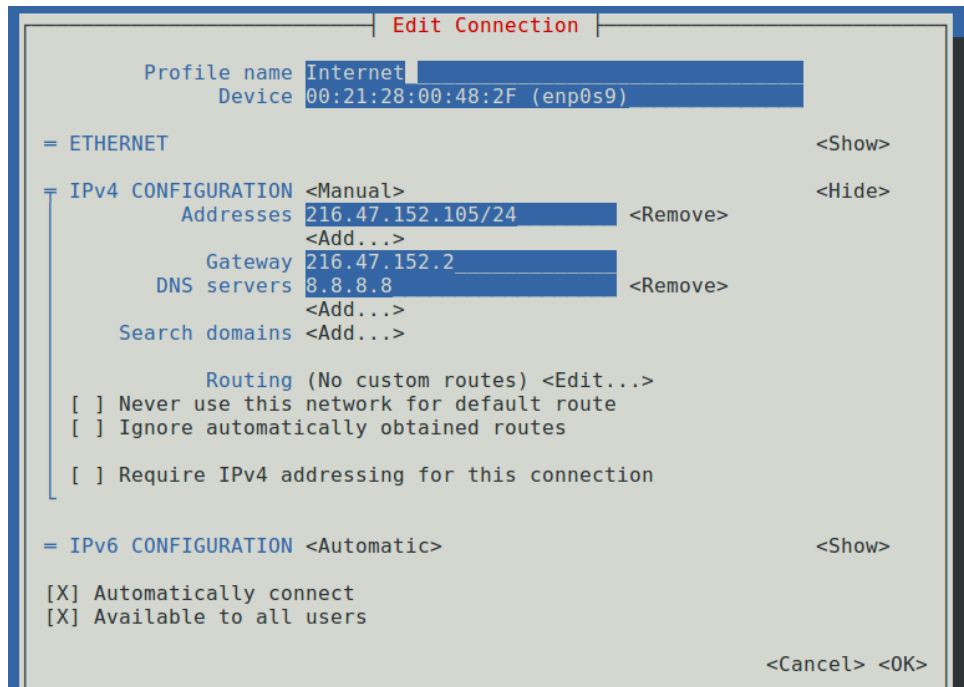


Figure 5: Connection details of Internet

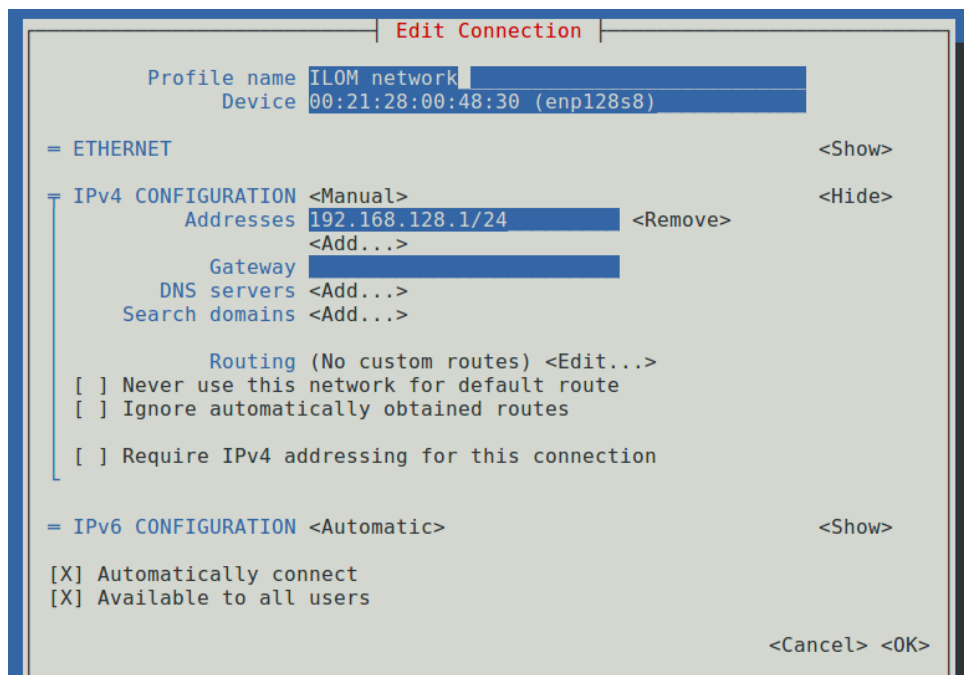


Figure 6: Connection details of ELOM

3.2.4 Setup InfiniBand

3.2.4.1 Connection setup

Install necessary drivers for the InfiniBand using the following command.

```
# Install all packages necessary for infiniband.
[hec]$ yum -y groupinstall "InfiniBand Support"
```

Use `nmtui` to set up the IP address of the InfiniBand to enable IPoIB. IPoIB is needed for Lustre over Infiniband.

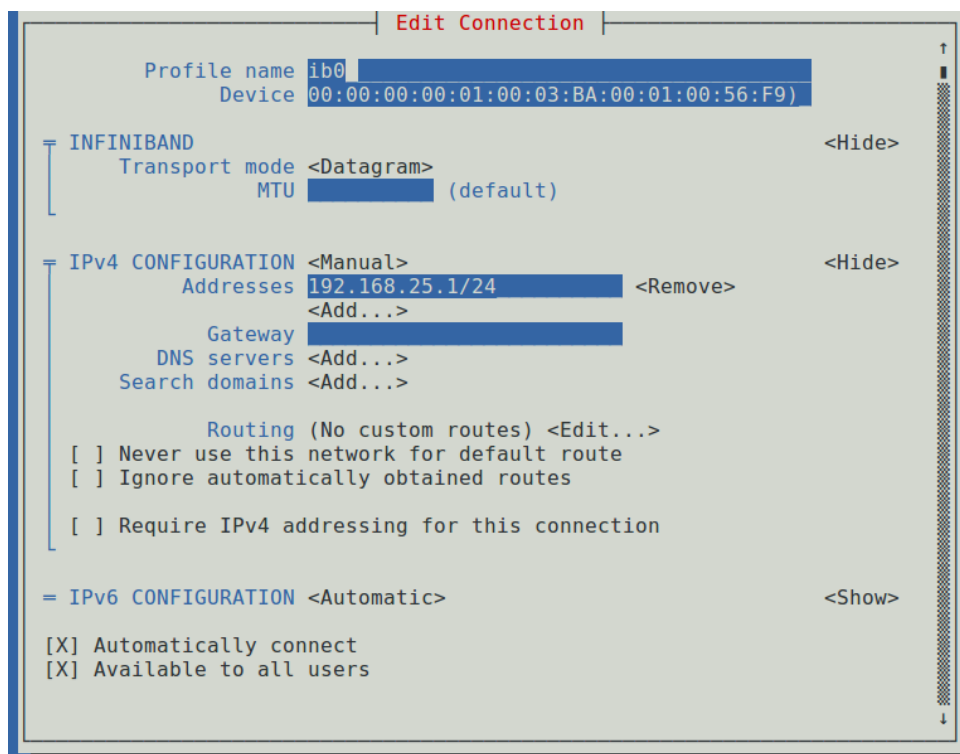


Figure 7: Master node InfiniBand connection details

3.2.4.2 Subnet Manager configuration

Subnet manager(SM) is an essential component of InfiniBand fabric. It assigns LID (local identifier) to each port of the network. The Silverstone InfiniBand switch comes with hardware based subnet manager, however the login credentials are unknown and the serial port (RJ-11) didn't work. So we decided to run the subnet manager on the master node, for which we chose `opensm`, a software implementation of the SM.

```
# Install opensm  
[hec]$ yum install opensm
```

The next step would be to tell on which port should opensm be running and the priority of the SM. This can be modified in `/etc/sysconfig/opensm`. Each port can uniquely identified by **GUID** (Global unique identifier) which is available in the output of `ibstat`

```
[root@hec hec_admin]# ibstat  
CA 'mthca0'  
  CA type: MT25208 (MT23108 compat mode)  
  Number of ports: 2  
  Firmware version: 4.7.400  
  Hardware version: a0  
  Node GUID: 0x0003ba00010056f8  
  System image GUID: 0x0003ba00010056fb  
  Port 1:  
    State: Active  
    Physical state: LinkUp  
    Rate: 10  
    Base lid: 18  
    LMC: 0  
    SM lid: 18  
    Capability mask: 0x02590a6a  
    Port GUID: 0x0003ba00010056f9  
    Link layer: InfiniBand  
  Port 2:  
    State: Down  
    Physical state: Polling  
    Rate: 10  
    Base lid: 0  
    LMC: 0  
    SM lid: 0  
    Capability mask: 0x02590a6a  
    Port GUID: 0x0003ba00010056fa  
    Link layer: InfiniBand
```

Figure 8: `ibstat` output on master node

```
# Ignoring the comments in the output for brevity  
[hec]$ cat /etc/sysconfig/opensm  
GUIDS="0x0003ba00010056fa 0x0003ba00010056f9"  
PRIORITY=0
```

3.2.5 Install xCAT

xCAT is the management tool used for the cluster. More installation details [here](#). Steps for diskful OS install can be found [here](#). The working mechanics of xCAT can be found in detail [here](#). xCAT provides additional tools like [psh](#), [xdcp](#) which helps with parallel execution of commands across nodes and synchronize files among nodes. First install it on the master node, setup the necessary options and use it to install, manage all compute nodes in the cluster.

```
[root@hec]$ yum -y install yum-utils

# Add xCAT repo
[root@hec]$ wget -P /etc/yum.repos.d \
https://xcat.org/files/xcat/repos/yum/latest/xcat-core/xCAT-core.repo

# Add xCAT dependencies repo
[root@hec]$ wget -P /etc/yum.repos.d \
https://xcat.org/files/xcat/repos/yum/xcat-dep/rh7/x86_64/xCAT-dep.repo

# Install xCAT
[root@hec]$ yum -y install xCAT
```

3.2.6 Firewall Setup

It is essential we setup the firewall as we connect to the Internet. The port on the master node which is connected to the Internet is put in **public zone**, while all other ports are in **trusted zone**. Public zone allows ssh and dhcp6-client traffic. Trusted zone allows all traffic.

```
# Enable firewall
systemctl enable firewalld
reboot

# Check the firewall status
firewall-cmd --state

# Change the internal ethernet port to trusted zone
firewall-cmd --zone=trusted --permanent --change-interface=enp128s8
firewall-cmd --zone=trusted --permanent --change-interface=enp0s8
firewall-cmd --zone=trusted --permanent --change-interface=enp128s9

# Add the internet port to the public zone
firewall-cmd --zone=public --permanent --change-interface=enp0s9
```



```
# See which interfaces are on which zones
firewall-cmd --get-active-zones
# List the property of the zone
firewall-cmd --permanent --list-all --zone=trusted
```

In order to add an extra layer of protection to the SSH service we edit `/etc/ssh/sshd_config` to change the listening port to 9988 instead of the default.

```
Port 9988
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::
```

The firewall configuration also needs to be updated to reflect this change.

```
# Add rule for new SSH port
firewall-cmd --permanent --zone=public --add-port=9988/tcp
# List the property of the zone
firewall-cmd --permanent --remove-port=22/tcp
```

Finally, we install the `fail2ban` package, which will automatically include firewall rules to reject traffic from hosts who do multiple failed SSH login attempts. By default fail2ban will ban IP address after 5 failed attempts.

```
[hec]$ yum install fail2ban
```

We create a configuration file under `/etc/fail2ban/jail.local` that enables blocking of failed SSH attempts on the previously configured port.

```
[DEFAULT]
# ban hosts for 100*one hour:
bantime = 360000

# Override /etc/fail2ban/jail.d/00-firewalld.conf:
banaction = iptables-multiport

[sshd]
enabled = true
port = 9988
```

3.2.7 NFS Setup

The cluster's users' home directories are stored on the external disk array. The master node connects to the external disk array using Fibre Channel and exposes its file system to the cluster using NFS.

Using the `blkid` command we can obtain the UUID of the partitions from the disk array.

```
[root@hec hec_admin]# blkid
/dev/sda1: UUID="4b7f4479-da26-4d12-98ef-c58de1b63f0c" TYPE="xfs"
/dev/sda2: UUID="D7iRQd-IQE1-NtcF-sdU8-QZSV-CnyT-QwjKym" TYPE="LVM2_member"
/dev/sdb1: UUID="uxT0F4-lTnX-UkyW-kIVh-2DDG-6Afu-gqTK1E" TYPE="LVM2_member"
/dev/sdc1: UUID="j1uCqN-qBnc-oFED-1WwG-0JLv-8B70-VlobiM" TYPE="LVM2_member"
/dev/sde1: UUID="uxT0F4-lTnX-UkyW-kIVh-2DDG-6Afu-gqTK1E" TYPE="LVM2_member"
/dev/sdf1: UUID="j1uCqN-qBnc-oFED-1WwG-0JLv-8B70-VlobiM" TYPE="LVM2_member"
/dev/mapper/centos-root: UUID="8e9906b6-3e24-4c5d-97a1-c1ef5aac0c20" TYPE="xfs"
/dev/mapper/centos-swap: UUID="454a710e-359a-400a-9860-b63d9c1d1595" TYPE="swap"
/dev/mapper/dataavg-lvol0: UUID="85298c5f-d7ec-42ce-8a13-63687040365e" TYPE="ext3"
/dev/sdd1: PTTYPE="dos"
/dev/sdq1: PTTYPE="dos"
```

Figure 9: `blkid` output on master node

Now we add an entry to the `/etc/fstab` file to mount the disk array file system automatically.

```
UUID=85298c5f-d7ec-42ce-8a13-63687040365e /mnt/common ext3 \
relatime,errors=remount-ro 0 1
```

Finally, include the following line in the master node's `/etc/exports` configuration file to expose the partition using NFS .

```
/mnt/common 192.168.1.0/255.255.255.0(rw,sync,no_subtree_check)
```

3.3 Pre-installation Network Setup

3.3.1 Network design

The network consists of **two VLANs**. All **even numbered ports** on the switch are on **VLAN 2**, while all **odd numbered ports** are on **VLAN 28**. VLAN 2 consists of all the IPMI traffic and VLAN 28 consists of all intra network traffic between the hosts. The connection layouts for Master and compute nodes are depicted in the picture below.

The **management addresses of switch** are **192.168.1.253** and **192.168.128.253**. We chose two different addresses so that even in the case where one vlan goes down, the switch is

reachable via other. **Telnet** to one of the IP's to login to the switch from any node connected to the switch. The login credentials are in a separate excel sheet.

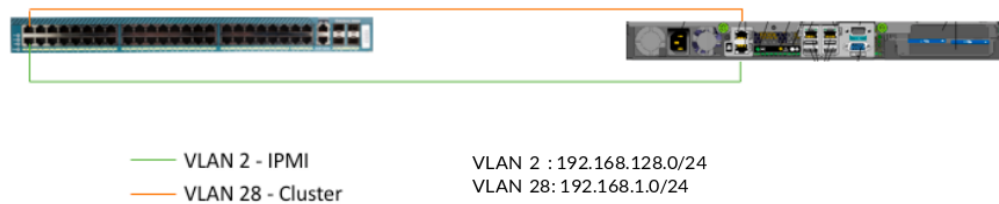


Figure 10: Network diagram of compute nodes

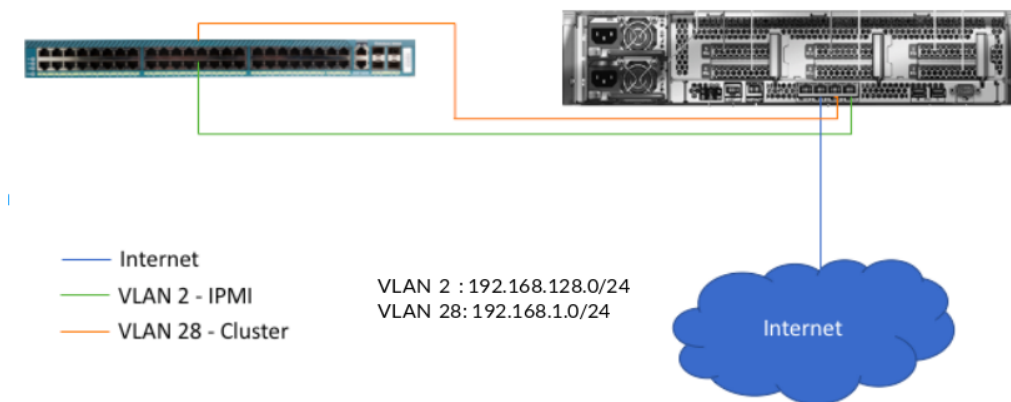


Figure 11: Network Diagram of Master

The IP address assignment for the cluster is shown in the table below.

3.3.2 Modify the IPMI address of compute nodes

The network installation procedure and some other management features require access to the compute node's Embedded Lights Out Manager System (ELOM) which is an Intelligent Platform Management Interface (IPMI) implementation from Sun Microsystems. [IPMI](#) is specification for a sub-system that monitors, manages the server independent of the main CPU. Each vendor has its own implementation of IPMI. ELOM, ILOM, ALOM are some of the implementations by Sun Microsystems. On the master node we have Integrated Lights Out Manager (ILOM) and on the compute nodes we have ELOM.

ELOM can be accessible through `ipmitool`, but it can also be accessed through SSH or a Web GUI (Note: The web GUI uses RC4 encryption which is not supported by modern browsers. Chrome version 47 or older is required to access the Web GUI) via the IPMI IPv4 configured. Sending IPMI commands to the ELOM interface, one can perform functions such as hardware

Old HEC Names	New HEC Names	MAC address Ether	New IPv4 Ether	New Infiniband Address	New IPv4 IPMI	Old IPv4 IPMI
HEC-13	hec-1	00:23:8b:97:97:89	192.168.1.2	192.168.25.2	192.168.128.2	192.168.128.34
HEC-14	hec-2	00:1e:68:9b:30:67	192.168.1.3	192.168.25.3	192.168.128.3	192.168.128.35
HEC-16	hec-3	00:23:8b:97:e1:fd	192.168.1.4	192.168.25.4	192.168.128.4	192.168.128.37
HEC-19	hec-4	00:23:8b:97:e0:e9	192.168.1.5	192.168.25.5	192.168.128.5	192.168.128.40
HEC-20	hec-5	00:23:8b:97:e1:ad	192.168.1.6	192.168.25.6	192.168.128.6	192.168.128.41
HEC-21	hec-6	00:23:8B:97:E1:D1	192.168.1.7	192.168.25.7	192.168.128.7	192.168.128.42
HEC-23	hec-7	00:23:8B:98:2A:FC	192.168.1.8	192.168.25.8	192.168.128.8	192.168.128.44
HEC-17	hec-8	00:23:8b:a9:9d:f5	192.168.1.9	192.168.25.9	192.168.128.9	192.168.128.38
HEC-18	hec-9	00:23:8b:98:2b:80	192.168.1.10	192.168.25.10	192.168.128.10	192.168.128.39
HEC-26	hec-10	00:23:8b:97:df:a5	192.168.1.11	192.168.25.11	192.168.128.11	192.168.128.47
HEC-25	hec-11	00:1E:68:57:8C:58	192.168.1.12	192.168.25.12	192.168.128.12	192.168.128.46
HEC-24	hec-12	00:23:8B:98:2F:1C	192.168.1.13	192.168.25.13	192.168.128.13	192.168.128.45
HEC-22	hec-13	00:23:8B:98:2A:E4	192.168.1.14	192.168.25.14	192.168.128.14	192.168.128.43
HEC-28	hec-14	00:23:8B:98:2E:FC	192.168.1.15	192.168.25.15	192.168.128.15	192.168.128.49
HEC-7	hec-15	00:23:8b:9a:2b:d8	192.168.1.16	192.168.25.16	192.168.128.16	192.168.128.28
HEC-27	hec-16	00:23:8B:98:2C:34	192.168.1.17	192.168.25.17	192.168.128.17	192.168.128.48

Figure 12: IP address assignment

resets, change boot parameters, power on the server or connect to the system's console.

This section describes how to change the IP addresses of the IPMI interfaces of the compute nodes (for which we use IPMI itself) and how to set the compute nodes to boot from the network in preparation for the network OS installation.

Run the following script on the master node to change the IPMI addresses of all the compute nodes. `ipmitool` is a program that allows us to interact with any IPMI compatible device. The variable `orig_ipmi` is an array of old IPv4 IPMI address of compute nodes and `new_ipmi` is the array corresponding new IPv4 IPMI addresses. The password is in the excel sheet.

```
#!/bin/bash
for i in $(seq 0 15)
do
# Change the IPMI ip address
ipmitool -U root -P <passwd> -H ${orig_ipmi[i]} lan set 1 ipaddr ${new_ipmi[i]}
# Change the default gateway for IPMI of compute nodes,
#       which would point to the master
ipmitool -U root -P <passwd> -H ${new_ipmi[i]} lan set 1 defgw ipaddr 192.168.128.1

# Change the boot mode to pxe, this temporary and works for only the next restart
ipmitool -U root -P <passwd> -H ${new_ipmi[i]} chassis bootdev pxe
done
```

3.4 Install OS on Compute Nodes

xCAT is the chosen administrative tool for the cluster setup. xCAT needs that the master node can talk to the IPMI of all the other nodes. i.e., there is underlying L3 connectivity between Master and other nodes.

3.4.1 Setup xCAT

1. Choose and tell xCAT the OS to use for installation

```
# create the image from the iso file
$ copycds CentOS-7-x86_64-Minimal-1708.iso
# Check the available images
$ lsdef -t osimage
centos7.4-x86_64-install-compute (osimage) # For stateful install
centos7.4-x86_64-netboot-compute (osimage) # For diskless install
centos7.4-x86_64-statelite-compute (osimage) # For statelite install
```

2. Tell xCAT about the nodes. Define the nodes along with their properties.

```
bmc_username=root
bmc_password=<passwd> # Refer excel for the actual password
for ((i=0; i<$num_computes; i++))
do
mkdef -t node ${c_name[$i]} groups=hec,all ip=${c_ip[$i]} \
mac=${c_mac[$i]} netboot=xnba \
arch=x86_64 bmc=${c_bmc[$i]} bmcusername=${bmc_username} \
bmcpassword=${bmc_password} \
mgt=ipmi
done
```

3. Tell xCAT about the root password to use on the compute nodes. The <passwd> here is a NEW password that you would want to give to the compute nodes. Assign a domain name for the cluster.

```
# Set the root password for the compute node
chtab key=system passwd.username=root passwd.password=<passwd>
# Set up domain name to use for the compute nodes
chdef -t site domain=cs.iit.edu
```

c_name	c_mac	c_ip	c_bmc
hec-1	00:23:8b:97:97:89	192.168.1.2	192.168.128.2
hec-2	00:1e:68:9b:30:67	192.168.1.3	192.168.128.3
hec-3	00:23:8b:97:e1:fd	192.168.1.4	192.168.128.4
hec-4	00:23:8b:97:e0:e9	192.168.1.5	192.168.128.5
hec-5	00:23:8b:97:e1:ad	192.168.1.6	192.168.128.6
hec-6	00:23:8B:97:E1:D1	192.168.1.7	192.168.128.7
hec-7	00:23:8B:98:2A:FC	192.168.1.8	192.168.128.8
hec-8	00:23:8b:a9:9d:f5	192.168.1.9	192.168.128.9
hec-9	00:23:8b:98:2b:80	192.168.1.10	192.168.128.10
hec-10	00:23:8b:97:df:a5	192.168.1.11	192.168.128.11
hec-11	00:1E:68:57:8C:58	192.168.1.12	192.168.128.12
hec-12	00:23:8B:98:2F:1C	192.168.1.13	192.168.128.13
hec-13	00:23:8B:98:2A:E4	192.168.1.14	192.168.128.14
hec-14	00:23:8B:98:2E:FC	192.168.1.15	192.168.128.15
hec-15	00:23:8b:9a:2b:d8	192.168.1.16	192.168.128.16
hec-16	00:23:8B:98:2C:34	192.168.1.17	192.168.128.17

Figure 13: Variable definitions

4. Tell xCAT how to partition the disks on the compute nodes. `custom_partition_file` is text file that defines the layout of the disk. The content of the file is based on the installation distribution. For CentOS, it called the “kickstarter file”. `system-config-kickstart` package is a gui tool that allows you to generate the kickstart file. Once the file is generated, only keep the section relevant to the partitioning. An online tutorial is [here](#) and [here](#).

```

sde                8:64    0 931.5G  0 disk
├─sde1             8:65    0    1G    0 part /boot
└─sde2             8:66    0   630G  0 part
   ├─rootvg-root    253:0    0   120G  0 lvm  /
   ├─rootvg-swap    253:1    0    10G  0 lvm  [SWAP]
   ├─rootvg-home    253:3    0   300G  0 lvm  /home
   └─rootvg-var     253:4    0   200G  0 lvm  /var

```

Figure 14: The partition layout chosen for compute nodes

```
$ cat partition_file
# Disk partitioning information
part pv.390 --fstype="lvmpv" \
--ondisk=/dev/disk/by-path/pci-0000:00:05.0-ata-1.0 --size=645124
part /boot --fstype="xfs" \
--ondisk=/dev/disk/by-path/pci-0000:00:05.0-ata-1.0 --size=1024
volgroup rootvg --pesize=4096 pv.390
logvol /var --fstype="xfs" --size=204800 --name=var --vgname=rootvg
logvol /home --fstype="xfs" --size=307200 --name=home --vgname=rootvg
logvol / --fstype="xfs" --size=122880 --name=root --vgname=rootvg
logvol swap --fstype="swap" --size=10240 --name=swap --vgname=rootvg

# Give the partition layout to xCAT
chdef -t osimage centos7.4-x86_64-install-compute \
partitionfile=custom_partition_file
```

5. Tell xCAT the IP address we use for InfiniBand (IPoIB)

```
# Define subnet mask for the infiniband interface
chdef -t network -o ib0 mask=255.255.255.0 net=${c_ipoib[0]}

# Define the ip address for compute nodes IPoIB
for ((i=0; i<$num_computes; i++)) ; do
chdef ${c_name[i]} nicips.ib0=${c_ipoib[i]} nictypes.ib0="InfiniBand" \
nicnetworks.ib0=ib0
done

# Ask xCAT to run scripts to configure the Infiniband
chdef compute -p postbootscripts=confignics
```

6. Finalize the network configuration for compute nodes on xCAT

```
# Creates /etc/hosts file based on the nodes
$ makehosts
# Gather cluster network information and add it to the xCAT database.
$ makenetworks
# Creates necessary config files for dhcpd on the master
$ makedhcp -n
# Configure dns server on the master node based on /etc/hosts
$ makedns -n
# Associate OS to use with the compute nodes
$ nodeset hec osimage=centos7.4-x86_64-install-compute
```


7. Reboot all the compute nodes. Installation starts and no manual intervention is needed.

```
# Reboot the nodes that are powered on
$ rpower hec reset

# Power on the nodes that are switched off
$ rpower hec on

# Check if the nodes are up
$ psh hec uptime
```

3.4.2 Consistent Interface naming

DO NOT SKIP THIS STEP

CentOS 7.4 tends to name the Ethernet interface based on the PCI slot id detected during the boot. Although we expect this to be consistent across reboots, it's actually not.

Because of this inconsistency `network.service` fails to find the appropriate configuration

```
[root@hec-1 ~]# ls -la /sys/class/net
total 0
drwxr-xr-x  2 root root 0 Apr 28 15:48 .
drwxr-xr-x 64 root root 0 Apr 28 15:48 ..
lrwxrwxrwx  1 root root 0 Apr 28 15:48 enp7s4f0 -> ../../devices/pci0000:00/0000:00:0d.0/0000:06:00.0/0000:07:04.0/net/enp7s4f0
lrwxrwxrwx  1 root root 0 Apr 28 15:48 enp7s4f1 -> ../../devices/pci0000:00/0000:00:0d.0/0000:06:00.0/0000:07:04.1/net/enp7s4f1
lrwxrwxrwx  1 root root 0 Apr 28 15:49 ib0 -> ../../devices/pci0000:00/0000:00:0f.0/0000:08:00.0/net/ib0
lrwxrwxrwx  1 root root 0 Apr 28 15:49 ib1 -> ../../devices/pci0000:00/0000:00:0f.0/0000:08:00.0/net/ib1
lrwxrwxrwx  1 root root 0 Apr 28 15:48 lo -> ../../devices/virtual/net/lo
```

Figure 15: Association of PCI id with interface names

file for the interface in `/etc/sysconfig/network-scripts`. To avoid this we must add the `HWADDR=<MAC address>` for each interface in the corresponding configuration. We automate this using a script shown below.

The script should be run on all the compute nodes once. If you get **"file not found"** it means that there's already a mismatch between the interface name and configuration file. In this case we would have to manually edit the `/etc/sysconfig/network-scripts/ifcfg-<interface>` to add the `HWADDR=<MAC address>`.


```
#!/bin/bash
# For all interfaces detected by the OS ...
for path in $(find /sys/class/net -maxdepth 1)
do
    iface=$(basename $path)
    # Check it's an ethernet interface
    if ! [[ $iface =~ ^enp ]]
    then
        continue
    fi

    # If the corresponding configuration file exists ...
    if [ -e /etc/sysconfig/network-scripts/ifcfg-$iface ]
    then
        # Check if the config file already has HWADDR ...
        if grep -q HWADDR /etc/sysconfig/network-scripts/ifcfg-$iface
        then
            echo "Address exists"
            continue
        fi

        # Take a backup of the config file
        cp /etc/sysconfig/network-scripts/ifcfg-$iface \
        /etc/sysconfig/network-scripts/backup_ifcfg-$iface

        # Add HWADDR= <MAC Address> to the config file
        echo HWADDR=$(cat $path/address) >> \
        /etc/sysconfig/network-scripts/ifcfg-$iface

        # Print the config file. The last line must start with HWADDR
        cat /etc/sysconfig/network-scripts/ifcfg-$iface
        echo "done"
    else
        echo "file not found"
    fi
done
```

3.4.3 Enable serial communication

The BIOS has serial communication enabled by default. However we need CentOS to redirect the output to console. To do this we add kernel parameters `console=tty0 console=ttyS1,9600` to the grub.

```
# Take a backup
psh hec "cp /etc/grub/default /etc/grub/backup_default"
# Add the kernel parameters
psh hec "sed 's/nodmraid/nodmraid console=tty0 console=ttyS1,9600/g' \
/etc/default/grub > /etc/grub/default"
# Generate the grub configuration file
psh hec "grub2-mkconfig -o /boot/grub2/grub.cfg"
```

In our experience using `ipmitool` for serial communication was very unreliable. So we suggest that you login to the ELOM of the compute node and start a console session.

```
# If you want to serial to hec-1,
#           First login to ELOM of hec-1 from master
$ ssh root@192.168.128.2
# This would take you to the ELOM Shell of hec-1
/SP# start AgentInfo/console
```

3.5 Local repositories configuration

The compute nodes are not connected to the Internet so they use local repositories configured on the master node. On this section we enable local repositories for OpenHPC and EPEL.

1. Configuration of OpenHPC repo on master node

```
# Download openHPC repo, contains rpm's and scripts
[hec]$ wget \
http://build.openhpc.community/dist/1.3.4/OpenHPC-1.3.4.CentOS_7.x86_64.tar
# Unzip the tar to /openhpc
[hec]$ mkdir -p /openhpc
[hec]$ tar xvf OpenHPC-1.3.4.CentOS_7.x86_64.tar -C /openhpc
# Create repo
[hec]$ /openhpc/make_repo.sh
# Expose the repo via http server to the compute nodes
[hec]$ ln -s /openhpc /var/www/html/
```

2. Configuration of EPEL repo on the master node.

```

# Create directory holding EPEL packages
[hec]$ mkdir -p /epel_repo
[hec]$ yum -y install yum-utils createrepo
# Download required EPEL packages
[hec]$ yumdownloader --destdir /epel_repo \
fping qstat libconfuse python34 python34-libs
# Create a repository from the downloaded packages
[hec]$ createrepo /epel_repo
# Expose the repo via http server to the compute nodes
[hec]$ ln -s /epel_repo /var/www/html/
# Give access to full EPEL repo to master node
[hec]$ yum -y install epel-release
# Expose the repo via http server to the compute nodes
[hec]$ ln -s /epel_repo /var/www/html/

```

3. Configuration of compute nodes to access the repositories on the master node

```

# Install yum-utils to have yum-config-manager available
[hec]$ psh hec yum --setopt=*.skip_if_unavailable=1 -y install yum-utils
# Disable OS repositories pointing to outside network
[hec]$ psh hec yum-config-manager --disable CentOS\*

# Add OpenHPC repo mirror hosted on SMS
[hec]$ psh hec yum-config-manager \
--add-repo=http://192.168.1.1/openhpc/OpenHPC.local.repo
# Replace local path with SMS URL
[hec]$ psh hec \
"perl -pi -e 's/file:\\/\\/\\/openhpc/http:\\/\\/192.168.1.1\\/openhpc/s' \
/etc/yum.repos.d/OpenHPC.local.repo"

# Setup access to the epel repo from the compute hosts
[hec]$ psh hec yum-config-manager \
--add-repo=http://192.168.1.1/epel_repo
[hec]$ psh hec \
"echo gpgcheck=0 >> /etc/yum.repos.d/192.168.1.1_epel_repo.repo"

# Prevent compute nodes from installing documentation
[hec]$ psh hec echo -e %_excludedocs 1 \>\> ~/.rpmmacros

```

3.6 Post-installation network setup

Now we enable the InfiniBand interfaces on the compute nodes.

```
[hec]$ psh hec yum -y groupinstall "InfiniBand Support"
[hec]$ psh hec systemctl start rdma
hec]$ updatenode hec -P "confignics"
```

3.7 Software Installation

3.7.1 OpenHPC base installation

We need to install OpenHPC packages on the master and compute nodes.

```
# Install OpenHPC base on master
[hec]$ yum -y install ohpc-base
# Install OpenHPC base on compute nodes
[hec]$ psh hec yum -y install ohpc-base-compute
```

3.7.2 SLURM installation

Now we can install the software packages for the SLURM workload manager.

```
# Install slurm server meta-package
[hec]$ yum -y install ohpc-slurm-server
# Identify resource manager hostname on master host
[hec]$ perl -pi -e \
"s/ControlMachine=\S+/ControlMachine=hec/" /etc/slurm/slurm.conf

# Add Slurm client support meta-package
[hec]$ psh hec yum -y install ohpc-slurm-client
# Copy SLURM configuration to nodes
[hec]$ xdcp compute /etc/slurm/slurm.conf /etc/slurm/slurm.conf
[hec]$ xdcp compute /etc/munge/munge.key /etc/munge/munge.key

# Start munge and slurm controller on master host
[hec]$ systemctl enable munge
[hec]$ systemctl enable slurmctld
[hec]$ systemctl start munge
[hec]$ systemctl start slurmctld
```

```
# Start slurm clients on compute hosts
[hec]$ psh hec systemctl enable munge
[hec]$ psh hec systemctl enable slurmd
[hec]$ psh hec systemctl start munge
[hec]$ psh hec systemctl start slurmd
```

3.7.3 OpenHPC NFS configuration

The directory `/opt/ohpc/pub` is exposed using NFS.

```
# Export OpenHPC public packages from master server
[hec]$ echo "/opt/ohpc/pub *(ro,no_subtree_check,fsid=11)" >> /etc/exports
[hec]$ exportfs -a
[hec]$ systemctl restart nfs-server
[hec]$ systemctl enable nfs-server
# Create NFS client mounts of /mnt/common and /opt/ohpc/pub on compute hosts
[hec]$ psh hec-1 echo \
"\\"192.168.1.1:/mnt/common
/mnt/common nfs nfsvers=3,nodev,nosuid,noatime 0 0\\" \>\> /etc/fstab
[hec]$ psh hec echo \
"\\"192.168.1.1:/opt/ohpc/pub
/opt/ohpc/pub nfs nfsvers=3,nodev,noatime 0 0\\" \>\> /etc/fstab
[hec]$ psh hec systemctl restart nfs
# Mount NFS shares
[hec]$ psh hec mkdir -p /opt/ohpc/pub
[hec]$ psh hec mount /opt/ohpc/pub
```

3.7.4 Development tools installation

Install openHPC packages for compilers and development tools.

```
# Install autotools meta-package
[hec]$ yum -y install ohpc-autotools
[hec]$ yum -y install EasyBuild-ohpc
[hec]$ yum -y install hwloc-ohpc
[hec]$ yum -y install spack-ohpc
[hec]$ yum -y install valgrind-ohpc
[hec]$ yum -y install gnu7-compilers-ohpc
[hec]$ yum -y install llvm5-compilers-ohpc
```

3.7.5 MPI stacks installation

Install software packages for OpenMPI, MVAPICH and MPICH

```
# Install
[hec]$ yum -y install openmpi3-gnu7-ohpc mpich-gnu7-ohpc mvapich2-gnu7-ohpc
```

3.8 Software Testing

3.8.1 IPoIB

To test IPoIB is working correctly, login to any of the compute nodes and ping all the IPoIB addresses. You should see a ICMP echo reply from all the nodes except hec-7. **The InfiniBand card of hec-7 failed and cannot be detected by the kernel.**

```
#!/bin/bash
# Script to ping all the IB interfaces
for i in $(seq 1 16)
do
    ping 192.168.25.$i -c 1
done
```

```
PING 192.168.25.1 (192.168.25.1) 56(84) bytes of data.
64 bytes from 192.168.25.1: icmp_seq=1 ttl=64 time=0.225 ms

--- 192.168.25.1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.225/0.225/0.225/0.000 ms
PING 192.168.25.2 (192.168.25.2) 56(84) bytes of data.
64 bytes from 192.168.25.2: icmp_seq=1 ttl=64 time=0.052 ms

--- 192.168.25.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.052/0.052/0.052/0.000 ms
PING 192.168.25.3 (192.168.25.3) 56(84) bytes of data.
64 bytes from 192.168.25.3: icmp_seq=1 ttl=64 time=0.680 ms

# ... Output omitted for brevity ...
# Note that hec-7 failed!!
--- 192.168.25.8 ping statistics ---
1 packets transmitted, 0 received, +1 errors, 100% packet loss, time 0ms
```

```

PING 192.168.25.9 (192.168.25.9) 56(84) bytes of data.
64 bytes from 192.168.25.9: icmp_seq=1 ttl=64 time=0.613 ms

# ... Output omitted for brevity ...

--- 192.168.25.15 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.716/0.716/0.716/0.000 ms
PING 192.168.25.16 (192.168.25.16) 56(84) bytes of data.
64 bytes from 192.168.25.16: icmp_seq=1 ttl=64 time=0.143 ms

--- 192.168.25.16 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.143/0.143/0.143/0.000 ms

```

3.8.2 SLURM

We run a simple "hello world" job across 16 nodes of the cluster to test if slurm is working. The result should look as follows. Note the warning, this is ok as the InfiniBand card of hec-7 is not working. Please take a look at the SLURM [cheat-sheet](#), it provides a comprehensive view of all the SLURM commands.

```

# Compile MPI "hello world" example
[hec_admin@hec ~]$ mpicc -O3 /opt/ohpc/pub/examples/mpi/hello.c
# Submit interactive job request and use prun to launch executable
# Run 20 jobs across 16 nodes
[hec_admin@hec ~]$ srun -n 20 -N 16 --pty /bin/bash
[hec_admin@hec ~]$ prun ./a.out
[hec_admin@hec-1 ~]$ prun ./a.out
[prun] Master compute host = hec-1
[prun] Resource manager = slurm
[prun] Launch cmd = mpirun ./a.out (family=openmpi3)

-----

[[13739,1],10]: A high-performance Open MPI point-to-point messaging module
was unable to find any relevant network interfaces:

Module: OpenFabrics (openib)
Host: hec-7

```

Another transport will be used instead, although this may result in lower performance.

NOTE: You can disable this warning by setting the MCA parameter `btl_base_warn_component_unused` to 0.

Hello, world (20 procs total)

```
--> Process # 1 of 20 is alive.  
--> Process # 0 of 20 is alive.  
--> Process # 17 of 20 is alive.  
--> Process # 13 of 20 is alive.  
--> Process # 12 of 20 is alive.  
--> Process # 9 of 20 is alive.  
--> Process # 10 of 20 is alive.  
--> Process # 18 of 20 is alive.  
--> Process # 15 of 20 is alive.  
--> Process # 14 of 20 is alive.  
--> Process # 5 of 20 is alive.  
--> Process # 8 of 20 is alive.  
--> Process # 7 of 20 is alive.  
--> Process # 16 of 20 is alive.  
--> Process # 19 of 20 is alive.  
--> Process # 2 of 20 is alive.  
--> Process # 11 of 20 is alive.  
--> Process # 4 of 20 is alive.  
--> Process # 6 of 20 is alive.  
--> Process # 3 of 20 is alive.
```


4 FAQ

How to add users .?

Answer: First create user on master node, set the password then synchronize `/etc/passwd`, `/etc/group` and `/etc/shadow` with all other nodes. Note that the default home directory is changed in `/etc/default/useradd` to `HOME=/mnt/common`. So there is no need point home directory to the NFS.

```
[root@hec]$ useradd -m <user-name>
[root@hec]$ passwd <user-name>
# Create a sync file for pushing user credentials to the nodes
[root@hec]$ echo "MERGE:" > syncusers
[root@hec]$ echo "/etc/passwd -> /etc/passwd" >> syncusers
[root@hec]$ echo "/etc/group -> /etc/group" >> syncusers
[root@hec]$ echo "/etc/shadow -> /etc/shadow" >> syncusers
# Use xCAT to distribute credentials to nodes
[root@hec]$ xdcp hec -F syncusers
```

How to delete user .?

Answer: First delete the user along with the HOME directory, then use `psh` to delete users across the nodes. **Note that using `xdcp` doesn't delete lines from the destination and thus cannot be used.**

```
# option -r means remove user's home directory
[root@hec]$ userdel -r <user-name>
[root@hec]$ psh hec userdel <user-name>
```

How to block/unblock IP from fail2ban ?

Answer:

```
[root@hec]$ fail2ban-client set sshd unbanip <ip-address>
# Note that the ip is banned for a time configured in
# /etc/fail2ban/jail.local
# This is not a permanent ban.
[root@hec]$ fail2ban-client set sshd banip <ip-address>
```

How to white list an IP address in fail2ban?

Answer: Under the [DEFAULT] section of `/etc/fail2ban/jail.local` add `ignoreip = 127.0.0.1/8 <A.B.C.D> <E.F.G.H>`, then restart the fail2ban service. More details [here](#).