Crane Cluster

Manual

**University of the Witwatersrand**

**2017**

Written by

**Or Hanoch**

**Joshua Bruton**

**Meir Rosendorff**

Nathan Michlo (Formatting)

Group Instructor

**Atif Muhammad**

About the Crane Cluster 3

Head Node 3

Nodes 3

NFS - Network File System 4

Installation 4

Enable and Start 4

Server Configuration 4

Client Configuration 5

Mounting 5

Firewall 5

MPI using NFS (v1) 6

Installation 6

Adding PATHs 6

Configuration and Modules 6

Testing 7

MPI using NFS (v2) 8

Download 8

Installation 8

Adding PATHs 8

Configuration and Modules 9

Testing 9

DNS Server (Domain Naming System) 11

Installation 11

Configuration 11

Forward Zone 12

Reversing Zone 12

Enabling 13

Firewall 13

Permissions 14

Checking 14

Testing 15

Quota 16

Pre-Setup 16

Setting a Quota 16

Torque 18

Head Node Installation 18

Nodes Installation 20

Installation on Clients 21

Enable Torque as a Service 22

Commands 22

Testing 25

LDAP (Lightweight Directory Access Protocol) 27

Installation [1] 27

User Account Migration 32

Firewall Configuration 32

LDAP Client Configuration [2] 33

NFS Server Configuration 34

Automounter Client configuration 34

Create New User 34

Create New Group 34

Wake On LAN 35

Configuration 35

Execution 35

Testing 35

Format and Mounting HDD 36

Format 36

Remove Partitions 36

Mount 36

Additional Scripts 38

network\_config 38

perform 38

ping\_all 38

wits\_proxy 38

wake\_cranes 39

LDAP Scripts 39

Ganglia [1] [2] 40

Installation 40

Configuration 40

Cloning Nodes 42

Create bootable flash drive 42

Creating Image [1] 42

Restoring an image [4] 42

New stuff 44

Set hostname 44

Networking 44

Create SSH-Key 44

Change Date on Machines 44

On Addition of Nodes: 45

References 46

# About the Crane Cluster

The Crane Cluster is a computer cluster made up of one head node and many separate nodes it uses. As of the writing of this manual, the Crane cluster consists of the head node and 19 compute nodes, located in MSL building, second floor, research lab 2.

## Head Node

The head node has 1 physical Ethernet card split logically into two cards - one for the internal cluster LAN and one for internet connectivity through the wits network. The external internet connection goes through the Wits proxy and as such gets its IP using the Wits DHCP. The internal LAN connection uses a static IP.

Hostname crane

LAN IP 10.0.0.101

Internet IP (as of writing): 10.10.187.31

## Compute Nodes

Hostname crane# (Where # is the number of the node)

LAN IP 10.0.0.# (Where # is the number of the node)

Internet IP (as of writing) n/a (Nodes only have a static IP for the internal LAN,

they do not have access to the internet directly.)

# NFS - Network File System

We use an NFS server to have a shared folder for head node and all the other nodes.

The NFS is setup with the folder on the server being /server/folder/ and the folder on the clients being /client/folder/

In our case the NFS folder is located on head node as the folder /craneNest. Physically this folder is located on a separate 1TB HDD mounted to head node from sdb (see Format and Mounting HDD).

## Installation

Run the following command on both the server and the clients:

$ yum install nfs-utils

## Enable and Start

Run the following commands on both the server and the clients:

$ systemctl enable rpcbind

$ systemctl enable nfs-server

$ systemctl enable nfs-lock

$ systemctl enable nfs-idmap

$ systemctl start rpcbind

$ systemctl start nfs-server

$ systemctl start nfs-lock

$ systemctl start nfs-idmap

## Server Configuration

Create the public folder that will be shared and give it full permissions:

$ mkdir /server/folder

$ chmod 777 /server/folder

Add the following to /etc/exports to define who has permissions to access the public folder and what their permissions are. (<client IP> must be replaced by the range of client IPs, the part in the brackets defines the types of permissions. NB the only space is between the folder path and the client IP.)

/server/folder/ <client IP>(rw,sync,no\_root\_squash,no\_all\_squash)

Restart NFS with:

systemctl restart nfs-server

## Client Configuration

Create the folder in which the data inside /server/folder will be displayed:

$ mkdir /client/folder

## Mounting

Use the following to mount the /server/folder inside the /client/folder replacing <server IP> with the server’s IP.

$ mount -t nfs <server IP>:/server/folder /client/folder

NOTE: If the previous command times-out, see firewall section and then return to this section and try again.

In order to set it to mount on boot add the following to the /etc/fstab file replacing <server IP> with the server’s IP:

<server IP>:/server/folder/ /client/folder/ nfs rw,sync,hard,intr 0 0

In order to confirm correct mount enter the df command and you should see your folder on the list.

## Firewall

Uncomment the following in /etc/sysconfig/nfs order to set default ports

MOUNTD\_PORT=port

STATD\_PORT=port

LOCKD\_TCPPORT=port

LOCKD\_UDPPORT=port

Use the following command for all the ports you just uncommented in /etc/sysconfig/nfs as well as ports 111 and 2049 in order to allow nfs through the firewall where <port> is the port number)

$ firewall-cmd --permanent --add-port=<port>/tcp

After allowing all the ports through the firewall use the following command to restart the firewall:

$ firewall-cmd –reload

# MPI using NFS (v1)

MPI is a protocol used to pass messages, over a network, between programs running in parallel across a host or cluster.

Open MPI is an MPI implementation used to handle and run these such jobs and programs.

Below, it will be installed and setup in the shared NFS folder /craneNest/ in the apps/ directory.

## Installation

Create the folder for Open MPI in apps using:

$ mkdir /craneNest/apps/openmpi

Go to the download location of Open MPI and run the following:

$ ./configure -prefix = craneNest/apps/openmpi

$ make

$ make install

## Adding PATHs

You need to add craneNest/apps/openmpi/bin and craneNest/apps/openmpi/lib to the $PATH of all the nodes on the server.

You can do this without a restart by typing the following commands:

$ export PATH=$PATH:/craneNest/apps/openmpi/bin

$ export PATH=$PATH:/craneNest/apps/openmpi/lib

However, to make the change permanent you must add the above commands to /home/<user\_name>/.bashrc and /etc/profile. (in our case the users are crane and root)

Adding to profile will make it activate when a user logs-in and adding it to: .bashrc will run it when the computer boots.

## Configuration and Modules

Enter the command

$ which mpirun

If it does not return a path perform the following or run the following commands:

$ module avail

$ add module <path> where <path> is the path seen at the end of the previous output.

Enter the command $ which mpirun to confirm it outputs a path.

Enter the command $ mpiexec --version and it should return the details of your Open MPI version if the paths are configured correctly.

## Testing

To test Open MPI, a host file must first be created on which is simply a list of IP addresses or domain names that you want to run Open MPI on.

Run Open MPI with the following command:

$ mpirun -hostfile <name of hostfile> -np 1 echo Hello

If Open MPI was configured correctly this should output Hello in the terminal of the computers specified in the hostfile.

Test this on several the nodes to confirm the $PATH's are configured correctly everywhere.

To test if the processes are being divided amongst the nodes correctly run the following:

(The sleep command will put the running user to sleep for the number specified in seconds. The –np flags specifies the number of processors to be used)

$ mpirun -hostfile <name of hostfile> -np 1 echo sleep 30

Before the process is completed enter the following commands into any of the nodes specified in the host file:

$ top | grep <name of user>

This should produce at least one sleep process (depending on the number of processors specified and one orted process (orted is the method MPI uses to run.)

NOTE

If this works then Open MPI is running correctly.

# MPI using NFS (v2)

MPI is a protocol used to pass messages, over a network, between programs running in parallel across a host or cluster.

Open MPI is an MPI implementation used to handle and run these such jobs and programs.

Below, it will be installed and setup in the shared NFS folder /craneNest/ in the apps/ directory.

## Download

Download onto a USB stick the latest release version of Open MPI and copy it to the head node.

## Installation

Create the folder for Open MPI in /apps using:

$ mkdir /craneNest/apps/openmpi

Go to the download location of Open MPI and run the following:

$ gunzip -c openmpi-2.0.2.tar.gz | tar xf -

$ cd openmpi-2.0.2

$ ./configure --prefix=/usr/local

$ make all install

You might need to update automake if make all install does not work:

$ wget http://ftp.gnu.org/gnu/automake/automake-1.15.tar.gz

$ tar xvzf automake-1.15.tar.gz

$ cd automake-1.14

$ ./configure

$ make

$ sudo make install

## Adding PATHs

You need to add craneNest/apps/openmpi/bin and craneNest/apps/openmpi/lib to the $PATH of all the nodes on the server.

You can do this without a restart by typing the following commands:

$ export PATH=$PATH:/craneNest/apps/openmpi/bin

$ export PATH=$PATH:/craneNest/apps/openmpi/lib

However, to make the change permanent you must add the above commands to /home/<user\_name>/.bashrc. (in our case the users are crane and root)

Adding to profile will make it activate when a user logs-in and adding it to: .bashrc will run it when the computer boots.

## Configuration and Modules

Enter the command

$ which mpirun

If it does not return a path perform the following:

$ module avail

$ add module <path> where <path> is the path seen at the end of the previous output.

Enter the command which mpirun to confirm it outputs a path.

Enter the command $ mpiexec --version and it should return the details of your Open MPI version if the paths are configured correctly.

## Testing

(If you run into problems with the firewall you might need to turn it off until we figure out how to make exceptions that work “systemctl stop firewalld”, “systemctl disable firewalld”)

To test Open MPI, a host file must first be created on which is simply a list of IP addresses or domain names that you want to run MPI on.

Run MPI with the following command:

$ mpirun -hostfile <nameOfHostfile> -np 1 echo Hello

If Open MPI was configured correctly this should output “Hello” in the terminal of the computers specified in the hostfile.

Test this on several nodes to confirm the $PATH's are configured correctly everywhere.

To test if the processes are being divided amongst the nodes correctly run the following:

(The hostname command will output the running nodes hostname. The –np flags specifies the number of processors to be used)

$ mpirun -hostfile nameOfHostfile -np 1 sleep hostname

NOTE

If this works then Open MPI is running correctly.

# DNS Server (Domain Naming System)

A DNS Server is a server that pairs numbered IP addresses to named network addresses, and creates a domain for a network to work on. We used the head node as our DNS server.

## Installation

Install bind by running:

$ yum install bind bind-utils -y

## Configuration

Open /etc/named.conf and once there comment out listen-on-v6 port 53 { ::1; }; and add your DNS server IP to listen-on port 53 {<ip>;};

In allow-query add the IP range of your slaves. e.g. localhost; 10.0.0.0/24; The allow-transfer command can be used to configure a secondary backup DNS, if you don't have one, comment it out.

At the bottom of your file, under the already existing zone, instantiate your forward and reverse DNS lookup zones. For example, using zonename as the zone’s name.

zone “zonename" IN {

type master;

file "forward.zonename";

allow-update { none; };

};

zone "x.x.x.in-addr.arpa" IN { // 'x.x.x' is the first 3 octets of your DNS IP in reverse e.g. '0.0.10'.

type master;

file "reverse.zonename";

allow-update { none; };

};

Now you must configure and create the aforementioned zones.

Save and exit named.conf

Open /var/named/ and create forward.zonename and reverse.zonename in the folder.

## Forward Zone

Add the following to the forward.zonename file using the number of nodes required and replacing nameofDNShost, root\_zonename, IPofDNS and IPofNode with the relevant information.

NOTE: Make sure to include all your nodes in this file.

$TTL 86400

@ IN SOA nameofDNShost. root.zonename. (

2011071001 ;Serial

3600 ;Refresh

1800 ;Retry

604800 ;Expire

86400 ;Minimum TTL

)

@ IN NS nameofDNShost.

@ IN A IPofDNS

@ IN A IPofNode1

@ IN A IPofNode2

nameofDNShost IN A IPofDNS

nameOfNode1 IN A IPofNode1

nameOfNode2 IN A IPofNode2

## Reversing Zone

Add the following lines to the reverse.zonename file using the number of nodes required and replacing nameofDNS, root\_zonename, IPofDNS and IPofNode with the relevant information. NB make sure to include all your nodes in this file.

$TTL 86400

@ IN SOA nameOfDNShost. root.zonename. (

2011071001 ;Serial

3600 ;Refresh

1800 ;Retry

604800 ;Expire

86400 ;Minimum TTL

)

@ IN NS nameOfDNShost.

@ IN PTR zonename.

nameOfDNShost IN A iPofDNS

nameOfNode1 IN A iPofnode1

nameOfNode2 IN A iPofnode2

lastoctetofDNSIP IN PTR nameOfDNShost

lastocterofNode1IP IN PTR nameOfNode1

lastocterofNode2IP IN PTR nameOfNode2

## Enabling

You can now turn on the DNS by running the following

$ systemctl enable named (Will start named on boot)

$ systemctl start named (Will start named now)

## Firewall

Configure the firewall to allow DNS on port 53 with the following:

$ firewall-cmd --permanent --add-port=53/tcp

$ firewall-cmd --permanent --add-port=53/udp

Reload the firewall for the new settings to take effect with:

$ firewall-cmd --reload

## Permissions

Permissions, ownership and SELinux configuration is done with the following:

$ chgrp named -R /var/named

$ chown -v root:named /etc/named.conf

$ restorecon -rv /var/named

$ restorecon /etc/named.conf

## Checking

Run the following if there is no output continue.

$ named-checkconf /etc/named.conf

Zones checks replace zonename with your zone’s name, if it returns OK, they are okay.

$ named-checkzone zonename /var/named/forward.zonename

$ named-checkzone zonename /var/named/reverse.zonename

Add the DNS IP to your ifcfg file so it knows who to contact. (if you do not use eno1, replace it with your internet interface.)

vim /etc/sysconfig/network-scripts/ifcfg-eno1

~~Add DNS="IPofDNS" to that ifcfg-eno1 and then save and exit.~~

Go to /etc/resolv.conf and add at the beginning:

search zonename

nameserver IPofDNS

Comment out (#) all the existing lines except for the first two nameservers (other than your DNS).

note: You must add this to all the resolv.conf files of the nodes as well, BUT you can only search one zone per node.

Now restart the network and DNS.

$ systemctl restart network

$ reboot

Check your resolv.conf file and make sure your changes are still there. If not, add them again and use the command $ chattr +i /etc/resolv.conf to make the file non-editable. To undo this, use $ chattr -i /etc/resolv.conf.

## Testing

Use: nslookup zonename and you should receive the following:

Server: IPofDNS

Address: IPofDNS#53

Name: nameofZone

Address: IPofDNS

Name: nameofZone

Address: IPofNode1

Name: nameofZone

Address: IPofNode2

.

.

.

Reboot all nodes

Now ping an active node using its name (not its IP-address) and you should get a response.

NOTE

If both tests work the DNS is running.

# Quota

Quota allows you to limit the size of folders for certain users

## Pre-Setup

To setup perform the following:

add “usrquota,grpquota” to the folder you want to add a quota to in /etc/fstab

For example, in our case we add a quota to craneNest by editing /etc/fstab in the following way:

/craneNest/ home ext3 defaults,usrquota,grpquota 1 2

Remount the folder to apply the settings:

$ mount -o remount the folder / reboot

Use the following to confirm your folder has quota enabled

$ mount | grep quota

Now switch quota on for that folder

$ quotaon /craneNest

## Setting a Quota

Now we set the actual quota for the folder

Use the following command to set it for the user replacing USER with the user’s name

$ edquota -u USER

and to set it for groups, replacing GROUP with the group’s name we use

$ eqquota –g GROUP

This will open a window that looks like the following for a user named Jack

Disk Qoutas for user Jack (uid 1001):

Filesystem blocks soft hard inodes soft hard

/dev/mapper/centos-home 0 5500 6000 0 0 0

By editing the first two values underneath hard and soft we can change the quota on the disk. When the user’s usage reaches the soft value, he we will receive a warning and when he reaches the hard value it will no longer allow him to create new files.

The inodes field allows us to limit the number of files the user can make.

$ edquota -t allows you to configure grace period for soft limit after which it becomes hard storage

In order to track the usage, we use the command $ repquota –as which will give a summary of all the different user’s usages and their limits

Automation of the process

This can be done for user creation.

We can automate the process by creating a template user and copying those setting to other users upon their creation. In this case, we use the following command where <user> is the user we are creating and <user prototype> is the name of the prototype user we want to use

$ edquota <user> -p <user prototype>

# Torque

Torque is a program that allows scheduling and distribution of jobs from a head node to separate nodes of a cluster.

## Head Node Installation

### Opening Ports

Torque requires certain ports to be open for essential communication.

- For client and pbs\_mom communication to pbs\_server, the default port is 15001.

- For pbs\_server communication to pbs\_mom, the default port is 15002.

- For pbs\_mom communication to pbs\_mom, the default port is 15003.

$ firewall-cmd --add-port=15001/tcp --permanent

$ firewall-cmd --reload

### Verify Hostname

Make sure that the correct hostname and IP are entered correctly in /etc/hosts (in our case "crane")

To verify that the hostname resolves correctly, make sure that hostname and hostname -f report the correct name for the host.

### Install required packages

$ yum install libtool openssl-devel libxml2-devel boost-devel gcc

gcc-c++

### Download Torque

Go to folder you want to save Torque files at (I chose /root/programs)

Download git and copy folder from git:

$ yum install git

$ git clone

https://github.com/adaptivecomputing/torque.git -b 6.0.1 6.0.1

$ cd 6.0.1

$ ./autogen.sh

### Install Torque

While in the folder 6.0.1

$ ./configure

$ make

$ make install

### Set Torque server name

Verify that the /var/spool/torque/server\_name file exists and contains the correct name of the server (in our case "crane").

can be done using:

$ echo <torque\_server\_hostname> > /var/spool/torque/server\_name

### Boot config

Configure the trqauthd daemon to start automatically at system boot

$ cp contrib/systemd/trqauthd.service /usr/lib/systemd/system/

$ systemctl enable trqauthd.service

$ echo /usr/local/lib > /etc/ld.so.conf.d/torque.conf

$ ldconfig

$ systemctl start trqauthd.service

### PATHs

Make sure /usr/local/bin and /usr/local/sbin are in the $PATH environment variable:

Checking what is in PATH:

$ echo $PATH

Adding to PATH:

$ export PATH=/usr/local/bin/:/usr/local/sbin/:$PATH

### Torque.setup Script

Initialize serverdb by executing the torque.setup script:

in 6.0.1 folder:

$ ./torque.setup root

Set nodes to be used:

Add each node to the /var/spool/torque/server\_priv/nodes on one line in the following format (separated by lines below due to document limitations):

<node\_host\_name> np=<number\_of\_cores\_available>

gpus=<number\_of\_gpus\_available\_(optional)>

<string\_to\_characterize\_node\_(optional)>

Configure pbs\_server to start automatically at system boot, and then start the daemon:

$ qterm

$ cp contrib/systemd/pbs\_server.service /usr/lib/systemd/system/

$ systemctl enable pbs\_server.service

$ systemctl start pbs\_server.service

## Nodes Installation

Opening Ports - done on each node:

On nodes:

$ firewall-cmd --add-port=15002-15003/tcp --permanent

$ firewall-cmd --reload

Create packages to be installed on nodes - done on head node:

from 6.0.1 folder

$ make packages

Building ./torque-package-clients-linux-x86\_64.sh ...

Building ./torque-package-mom-linux-x86\_64.sh ...

Building ./torque-package-server-linux-x86\_64.sh ...

Building ./torque-package-gui-linux-x86\_64.sh ...

Building ./torque-package-devel-linux-x86\_64.sh ...

The package files are self-extracting packages that can be copied and executed on your production machines. Use --help for options.

Copy packages to nodes - done from head node:

from 6.0.1 folder

$ scp torque-package-mom-linux-x86\_64.sh

<mom-node>:<location-on-node>

$ scp torque-package-clients-linux-x86\_64.sh

<mom-node>:<location-on-node>

Copy MOM startup script to nodes - done from head node:

$ scp contrib/systemd/pbs\_mom.service

<mom-node>:/usr/lib/systemd/system/

Install MOM packages on nodes - done on each node:

$ ssh root@<mom-node>

Go to location packages were copied to

$ ./torque-package-mom-linux-x86\_64.sh --install

$ ./torque-package-clients-linux-x86\_64.sh --install

$ ldconfig

Configure pbs\_mom to start at system boot, and then start the daemon. - done on each node:

$ systemctl enable pbs\_mom.service

$ systemctl start pbs\_mom.service

NOTICE

Make sure that /etc/hosts /etc/hostname /etc/sysconfig/network /var/spool/torque/server\_name /var/spool/torque/server\_name.new ALL have the same hostname formats in both nodes and head node!

If you change head nodes hostname you need to correct /var/spool/torque/server\_name as well.

## Installation on Clients

If you want Torque client commands installed on hosts other than the Torque Server Host (head node)

Copy packages to client - done from head node:

Go to 6.0.1 folder

$ scp torque-package-clients-linux-x86\_64.sh

<torque-client-host>:<locatio\_on\_client>

Copy the trqauthd startup script to each Torque Client Host (nodes) - done from head node:

$ scp contrib/systemd/trqauthd.service

<torque-client-host>:/usr/lib/systemd/system/

Install packages on Clients (nodes) - done on each client (node):

$ ./torque-package-clients-linux-x86\_64.sh --install

$ echo /usr/local/lib > /etc/ld.so.conf.d/torque.conf

$ ldconfig

Enable and start the trqauthd service - done on each client (node):

$ systemctl enable trqauthd.service

$ systemctl start trqauthd.service

## Enable Torque as a Service

To have it run in background waiting for commands

from 6.0.1 folder:

On nodes:

$ cp contrib/init.d/pbs\_mom /etc/init.d/pbs\_mom

$ chkconfig --add pbs\_mom

On head node:

$ cp contrib/init.d/pbs\_server /etc/init.d/pbs\_server

$ chkconfig --add pbs\_server

## Commands

|  |  |
| --- | --- |
| Create PBS Server (>= 6.0.1) | $ ./torque.setup <username> |
| Start / Stop trqauthd | service trqauthd <start|stop> |
| Start PBS Server | pbs\_server |
| Stop PBS Server | qterm |
| Start MOM Server | pbs\_mom |
| Start Scheduler | pbs\_sched |
| Display Server Settings | qmgr -c 'p s' |

Give manager and operator permission to user (i.e. to be able to use qrun)

$ qmgr

Qmgr: set server managers += crane@crane

Qmgr: set server operators += crane@crane

|  |  |
| --- | --- |
| Submit Job | qsub <job\_file> <-c enabled (optional chckpt)> |
| Status of Jobs | qstat |
| Run Job | qrun <job\_number> |
| Status of Nodes | pbsnodes |
| Cancel Job | qdel <job\_number> <-m “message” (optional)> |
| Place Job on Hold | qhold <job\_number> |
| Release Job Hold | qrls <job\_number> |
| Pause Running Job | qsig -s STOP <job\_number> |
| Continue Running Job | qsig -s CONT <job\_number> |
| Create Checkpoint | qchkpt <job\_number>vi |

## Job Format

Typically, a submit script is written to hold all the parameters of a job. These parameters could include how long a job should run (walltime), what resources are necessary to run, and what to execute. The following is an example submit file:

#PBS -N localBlast

#PBS -S /bin/sh

#PBS -l nodes=1:ppn=2,walltime=240:00:00

#PBS -M user@my.organization.com

#PBS -m ea

source ~/.bashrc

cd $HOME/work/dir

sh myBlast.sh -i -v

This submit script specifies the name of the job (localBlast), what environment to use (/bin/sh), that it needs both processors on a single node (nodes=1:ppn=2), that it will run for at most 10 days, and that Torque should email "user@my.organization.com" when the job exits or aborts. Additionally, the user specifies where and what to execute.

### PBS Options

#PBS -N myJob Assigns a job name. The default is the name of PBS job script.

#PBS -l nodes=4:ppn=2 The number of nodes and processors per node.

#PBS -q queuename Assigns the queue your job will use.

#PBS -o mypath/my.out The path and file name for standard output.

#PBS -e mypath/my.err The path and file name for standard error.

#PBS -j oe Merges the standard error and output stream of the job.

#PBS -m b Sends mail to the user when the job begins.

#PBS -m e Sends mail to the user when the job ends.

#PBS -m a Sends mail to the user when job aborts (with an error).

#PBS -r n Indicates that a job should not rerun if it fails.

#PBS -V Exports all environment variables to the job.

#PBS -S Specifies environment

#PBS -m ba Allows a user to have more than one command with the same flag by grouping the messages together on one line, else only the last command gets executed.

#PBS -l walltime=01:00:00 The maximum wall-clock time during which this job can run.

#PBS -W stagein=file\_list Copies the file onto the execution host before the job starts.

#PBS -W stageout=file\_list Copies file from the execution host after the job completes.

#PBS -M usr@domain.com Specifies e-mail address to send to

## Testing

We keep our jobs at /craneNest/Jobs

Go to Jobs directory: cd /craneNest/Jobs

Check that node status for all nodes is free:

pbsnodes

if one of the nodes has a problem SSH to it and restart the mom service: systemctl restart pbs\_mom.service, if there is problem with SSH - fix it - Torque needs SSH to work properly.

Submit a job: qsub hello.pbs (or other job you have)

Check status of job: qstat

If a job is running it has an R under "S"

If status is Q try to forcefully run it: qrun <job\_number>

If qrun works (you see R under job status using qstat) then there might be a problem with the scheduler. restart scheduler:

systemctl restart pbs\_sched.service

When job ends look at the files in your directory, you should see two new files - test.out test.err. Make you have the correct output in test.out and that test.err doesn't have problematic errors.

Note: you can edit hello.pbs to specify the number of nodes and CPUs on each with PPN.

#PBS -l nodes=14:ppn=1,walltime=199:0:30

Test using different number of nodes and PPN to see that everything is OK.

NOTICE

Make sure that /etc/hosts /etc/hostname /etc/sysconfig/network /var/spool/torque/server\_name /var/spool/torque/server\_name.new ALL have the same hostname formats in both nodes and head node!

If running jobs has problems, check qsub -f

Also check if jobs are stuck in node in /var/spool/torque/undelivered or in /var/spool/torque/spool

Make sure you have permissions to write to output destinations and make sure users can SSH and SCP without a password between node and head node

# LDAP

# (Lightweight Directory Access Protocol)

LDAP is a user management system that allows you to create users and groups that are universal for your domain. The main system is installed on a Domain Controller and client versions are installed on each of the computers in the domain.

In our case the Domain Controller is head node and client versions are installed on each of the compute nodes.

NOTE:  
In our case our domain name is cranezone (we set this when we set the DNS)

## Installation [1]

Install the following packages:

$ yum install -y openldap openldap-clients openldap-servers

migrationtools

Generate a **LDAP** password from a secret key and save an SSHA encryption of it in /etc/openldap/passwd (we also saved the non-encrypted password there)

$ slappasswd -s <admin\_password> -n > /etc/openldap/passwd

Generate a X509 certificate valid for **365** days (enter all blanks except for Common Name):

$ openssl req -new -x509 -nodes -out /etc/openldap/certs/cert.pem \

-keyout /etc/openldap/certs/priv.pem -days 365

Generating a 2048 bit RSA private key

.....+++

..............+++

writing new private key to '/etc/openldap/certs/priv.pem'

-----

You are about to be asked to enter information that will be incorporated

into your certificate request.

What you are about to enter is what is called a Distinguished Name or a DN.

There are quite a few fields but you can leave some blank

For some fields there will be a default value,

If you enter '.', the field will be left blank.

-----

Country Name (2 letter code) [XX]:

State or Province Name (full name) []:

Locality Name (eg, city) [Default City]:

Organization Name (eg, company) [Default Company Ltd]:

Organizational Unit Name (eg, section) []:

Common Name (eg, your name or your server's hostname) []:<serverHostname>.<domain>

Email Address []:

Secure the content of the /etc/openldap/certs directory:

$ cd /etc/openldap/certs

$ chown ldap:ldap \*

$ chmod 600 priv.pem

Prepare the **LDAP** database:

$ cp /usr/share/openldap-servers/DB\_CONFIG.example

/var/lib/ldap/DB\_CONFIG

Generate database files (don’t worry about error messages!):

$ slaptest

53d61aab hdb\_db\_open: database "dc=my-domain,dc=com": db\_open(/var/lib/ldap/id2entry.bdb) failed: No such file or directory (2).

53d61aab backend\_startup\_one (type=hdb, suffix="dc=my-domain,dc=com"): bi\_db\_open failed! (2)

slap\_startup failed (test would succeed using the -u switch)

Change **LDAP** database ownership:

$ chown ldap:ldap /var/lib/ldap/\*

Activate the **slapd** service at boot:

$ systemctl enable slapd

Start the **slapd** service:

$ systemctl start slapd

Check the **LDAP** activity:

# netstat -lt | grep ldap

tcp        0      0 0.0.0.0:ldap            0.0.0.0:\*               LISTEN

tcp6       0      0 [::]:ldap               [::]:\*                  LISTEN

To start the configuration of the **LDAP** server, add the **cosine** &**nis LDAP** schemas:

$ cd /etc/openldap/schema

$ ldapadd -Y EXTERNAL -H ldapi:/// -D "cn=config" -f cosine.ldif

SASL/EXTERNAL authentication started

SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth

SASL SSF: 0

adding new entry "cn=cosine,cn=schema,cn=config"

$ ldapadd -Y EXTERNAL -H ldapi:/// -D "cn=config" -f nis.ldif

SASL/EXTERNAL authentication started

SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth

SASL SSF: 0

adding new entry "cn=nis,cn=schema,cn=config"

Then, create the /etc/openldap/changes.ldif file and paste the following lines:

NOTE:

There can be as little as one dc and as many as needed

For domain name wits.ac.za you must put dc=wits,dc=ac,dc=za

In our example our domain is cranezone so we use only dc=cranezone

You can retrieve the SSHA password from the previously saved file /etc/openldap/passwd

dn: olcDatabase={2}hdb,cn=config

changetype: modify

replace: olcSuffix

olcSuffix: dc=cranezone

dn: olcDatabase={2}hdb,cn=config

changetype: modify

replace: olcRootDN

olcRootDN: cn=Manager,dc=cranezone

dn: olcDatabase={2}hdb,cn=config

changetype: modify

replace: olcRootPW

olcRootPW: {SSHA}l8A+0c+lRcymtWuIFbbc3EJ1PRZz9mGg

dn: cn=config

changetype: modify

replace: olcTLSCertificateFile

olcTLSCertificateFile: /etc/openldap/certs/cert.pem

dn: cn=config

changetype: modify

replace: olcTLSCertificateKeyFile

olcTLSCertificateKeyFile: /etc/openldap/certs/priv.pem

dn: cn=config

changetype: modify

replace: olcLogLevel

olcLogLevel: -1

dn: olcDatabase={1}monitor,cn=config

changetype: modify

replace: olcAccess

olcAccess: {0}to \* by dn.base="gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth" read by dn.base="cn=Manager,dc=cranezone" read by \* none

Send the new configuration to the **slapd** server:

$ ldapmodify -Y EXTERNAL -H ldapi:/// -f /etc/openldap/changes.ldif

SASL/EXTERNAL authentication started

SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth

SASL SSF: 0

modifying entry "olcDatabase={2}hdb,cn=config"

modifying entry "olcDatabase={2}hdb,cn=config"

modifying entry "olcDatabase={2}hdb,cn=config"

modifying entry "cn=config"

modifying entry "cn=config"

modifying entry "cn=config"

modifying entry "olcDatabase={1}monitor,cn=config"

Create the /etc/openldap/base.ldif file and paste the following lines:

dn: dc=cranezone

dc: example

objectClass: top

objectClass: domain

dn: ou=People,dc=cranezone

ou: People

objectClass: top

objectClass: organizationalUnit

dn: ou=Group,dc=cranezone

ou: Group

objectClass: top

objectClass: organizationalUnit

Build the structure of the directory service:

$ ldapadd -x -w <admin\_password> -D cn=Manager,dc=cranezone

–f /etc/openldap/base.ldif

adding new entry "dc=example,dc=com"

adding new entry "ou=People,dc=example,dc=com"

adding new entry "ou=Group,dc=example,dc=com"

Create two users for testing:

$ mkdir /home/guests

$ useradd -d /<home\_directory> ldapuser01 ldapuser01

$ passwd ldapuser01

Changing password for user ldapuser01.

New password:

Retype new password:

passwd: all authentication tokens updated successfully.

$ useradd -d /home/guests/ldapuser02 ldapuser02

$ passwd ldapuser02

Changing password for user ldapuser02.

New password:

Retype new password:

passwd: all authentication tokens updated successfully.

## User Account Migration

Go to the directory for the migration of the user accounts:

$ cd /usr/share/migrationtools

Edit the **migrate\_common.ph** file and replace in the following lines:

$DEFAULT\_MAIL\_DOMAIN = "cranezone";

$DEFAULT\_BASE = "dc=cranezone";

Create the current users in the directory service:

$ grep ":10[0-9][0-9]" /etc/passwd > passwd

$ ./migrate\_passwd.pl passwd users.ldif

$ ldapadd -x -w <admin\_password> -D cn=Manager,dc=cranezone -f users.ldif

adding new entry "uid=ldapuser01,ou=People,dc=example,dc=com"

adding new entry "uid=ldapuser02,ou=People,dc=example,dc=com"

$ grep ":10[0-9][0-9]" /etc/group > group

$ ./migrate\_group.pl group groups.ldif

$ ldapadd -x -w <admin\_password> -D cn=Manager,dc=cranezone -f groups.ldif

adding new entry "cn=ldapuser01,ou=Group,dc=example,dc=com"

adding new entry "cn=ldapuser02,ou=Group,dc=example,dc=com"

Test the configuration with the user called **ldapuser01**:

$ ldapsearch -x cn=ldapuser01 -b dc=cranezone

## Firewall Configuration

Add a new service to the firewall (**ldap**: port **tcp** **389**):

$ firewall-cmd --permanent --add-service=ldap

Reload the firewall configuration:

# firewall-cmd --reload

Edit the **/etc/rsyslog.conf** file and add the following line:

local4.\* /var/log/ldap.log

Restart the **rsyslog** service:

$ systemctl restart rsyslog

## LDAP Client Configuration [2]

As the authconfig-tui is deprecated, to configure the LDAP client side, there are two available options:nslcd and sssd.  
In this tutorial, the nslcd option will be used, see the [authconfig tutorial](https://www.certdepot.net/ldap-client-configuration-authconfig/) for the sssd option.

Install the following packages:

$ yum install -y openldap-clients nss-pam-ldapd

Then, type:

NOTE: our server hostname is crane and our domain name is cranezone

$ authconfig --enableforcelegacy --update

$ authconfig --enableldap –enableldapauth

--ldapserver="crane.cranezone"

--ldapbasedn="dc=cranezone"

--update

NOTE 1: According to your requirements, you can specify the –enablemkhomedir option. This option creates a local user home directory at the first connection if none exists.

NOTE 2: Type authconfig –help | grep ldap to remember the necessary options.

Put the **LDAP** server certificate into the /etc/openldap/cacerts directory:

$ scp root@crane.cranezone:/etc/openldap/certs/cert.pem

/etc/openldap/cacerts/cert.pem

Apply the correct SELinux context to the certificate:

$ restorecon /etc/openldap/cacerts/cert.pem

Activate the TLS option:

$ authconfig --enableldaptls --update

Test the configuration:

$ getent passwd ldapuser02

ldapuser02:\*:1001:1001:ldapuser02:/home/guests/ldapuser02:/bin/bash

## NFS Server Configuration

To get the home directory mounted, you need to [configure a NFS server](https://www.certdepot.net/rhel7-provide-nfs-network-shares-specific-clients/). The NFS server is called instructor.example.com in the procedure.

NOTE: It’s not required to have the LDAP server and the NFS server on the same machine, it’s only easier.

## Automounter Client configuration

Install the following packages:

$ yum install -y autofs nfs-utils

Create a new indirect /etc/auto.guests map and paste the following line:

\* -rw,nfs4 instructor.example.com:/home/guests/&

Add the following line at the beginning of the /etc/auto.master file:

/home/guests /etc/auto.guests

Start the Automounter daemon and enable it at boot:

$ systemctl enable autofs && systemctl start autofs

Test the configuration:

$ su - ldapuser02

## Create New User

$ cd /usr/share/migrationtools

$ useradd -d /<home\_directory> ldapuser01

$ passwd ldapuser01

$ grep ":10[0-9][0-9]" /etc/passwd > passwd

$ ./migrate\_passwd.pl passwd users.ldif

$ ldapadd -x -w <admin\_password> -D cn=Manager,dc=cranezone -f users.ldif

## Create New Group

$ cd /usr/share/migrationtools

$ groupadd <group\_name>

$ grep ":10[0-9][0-9]" /etc/group > group

$ ./migrate\_group.pl group groups.ldif

$ ldapadd -x -w <admin\_password> -D cn=Manager,dc=cranezone

-f groups.ldif

# Wake On LAN

## Configuration

On machines that will be woken up

vim /etc/sysconfig/network-scripts/ifcfg-<NIC> (eno1 in our case)

Add the line:

ETHTOOL\_OPTS="wol g"

Get MAC address from each computer and write it down:

ifconfig | grep ether

In BIOS of each computer enable Wake-On-Lan:

* Gigabyte:
  + press F2 on boot and enter BIOS
  + under power Power tab Enable Wake On Lan option
* HP
  + Press F10 on boot and enter bios
  + Advanced >> Power-On Options >> Change "Remote Wake up Boot source" to "Remote Server"

## Execution

On Machine that sends wake up signal:

$ ether-wake <MAC\_ADDRESS>

## Testing

Shutdown remote node

send ether-wake command from head node

see that the remote node turns on.

# Format and Mounting HDD

When adding a HDD (Hard Disk Drive) to a machine you first need to format it, then you mount it.

In our case we wanted /craneNest to be on a separate, large, HDD. This is meant for organization and to avoid hard drive memory shortages. /craneNest is located on /dev/sdb1 which is the single partition of a 1TB HDD.

Connect HDD to power and SATA on the computer.

## Format

Check if the computer can see your HDD:

$ cat /proc/partitions

See if there is a new HDD (usually sdb) and see that its size is the same as the one you put in.

Note: sdb# are partitions on the HDD, we will remove them.

## Remove Partitions

$ fdisk /dev/<new HDD>

If the new HDD has been previously used:

make new partition table: o

make new partition: n

choose primary partition: p

choose partition number: 1

choose default settings twice: enter, enter

write the new partition table: w

mkfs.xfs /dev/sdb1

## Mount

Create folder to be mounted to:

$ mkdir <folder\_path\_and\_name>

$ mount /dev/sdb1

Edit fstab so it will be mounted on boot:

$ vim /etc/fstab

Add the following row:

/dev/sdb1 /craneNest xfs defaults 0 1

NOTE

If you are remounting an old broken mount, make sure to unmount the broken mount using the command:

$ umount

# Additional Scripts

We have created some scripts for our own use, they are located in /craneNest/admin/.

/craneNest/admin is added to the $PATH environment variable in the root users .bashrc on head node and as such automatically exists in the $PATH variable for root.

## network\_config

This script sets the network table to create the internal LAN. It splits the network adapter logically so that one logical network card is connected to the normal Wits network through the wits proxy and gets an IP from the wits DHCP and the other logical card works with the internal LAN with static IP settings.

It is called in /etc/profile and runs automatically when a user logs in to head node.

## perform

This script lets you run a command on any nodes you choose, or all the nodes, using one line. The format for it is:

$ ./perform “<command\_between\_quotations>” -on

<node numbers with spaces as separator, head for head node, all

for all not including head node, node1 - node2 for all lists

between node1 and node2>

Example:

$ ./perform “ll | grep hello” -on 1 2 7 - 10 13 head

NOTE: This script needs to be edited when adding nodes.

## ping\_all

Pings all the nodes.

NOTE: This script needs to be edited when adding nodes.

## wits\_proxy

This script sets the wits proxy settings for wits.

It is called in /etc/profile and runs automatically when a user logs in.

## wake\_cranes

This script runs the wake-on-lan command on the nodes specified. The wake-on-lan needs to be enabled in the bios on the nodes for this script to work (all nodes have been enabled as of writing this manual).

The format for it is:

wake\_cranes <node1> <node2> ...

Or

wake\_nodes all

NOTE: This script needs to be edited when adding nodes.

## LDAP Scripts

Inside the folder /craneNest/admin/ldap you will find the following scripts:

* installClient- this script installs and adds the necessary certificate needed for ldap to work on a new client (node). This should be run on any new node.
* newUser <user\_name> - creates a new user in the ldap database (as well as on head node, which is the domain controller) with a user home folder located in /craneNest/admin/userAccounts and sets the size limit on the folder (using Quota)
* newGroup <group\_name> - creates a new group in the LDAP database.
* deleteUser <user\_name> - deletes a user from the ldap database (as well as on head node, which is the domain controller)
* DeleteGroup <group\_name> - deletes a LDAP group

# Ganglia [1] [2]

## Installation

### Head Node

$ yum install epel-release  
$ yum install ganglia rrdtool ganglia-gmetad ganglia-gmond ganglia-web

### Nodes

$ yum install epel-release  
$ yum install ganglia-gmond  
$ yum update && yum

## Configuration

### Head Node

$ cd /etc/ganglia  
$ vim gmetad.conf

find the line “data source” in the file and change it to:

data\_source "<cluster\_name>" 1 <head node\_hostname>

### Nodes

vim /etc/ganglia/gmond.conf

make following sections look like and add “udp\_send\_channel” sections for every node (including head node):

cluster {

name = "bu<cluster\_name (same as in gmetad.conf)>hpc"

owner = "unspecified"

latlong = "unspecified"

url = "unspecified"

}

udp\_send\_channel {

host = <node\_hostname>

port = 8649

ttl = 1

}

udp\_recv\_channel {

port = 8649

retry\_bind = true

}

# Cloning Nodes

When getting a new computer node, it is much easier to copy a HDD from an existing node onto the new nodes HDD, this is called cloning. We used Clonezilla, an open source program, to do this.

## Create bootable flash drive

If you have a windows computer, download YUMI and the clonezilla iso file. Use YUMI to burn the clonezilla iso onto the flash drive.

## Creating Image [1]

Connect the flash drive to an existing node and boot from it onto clonezilla. Connect a second flash drive to save the image on.

Use default settings when clonezilla boots.

Choose language English.

Don’t touch keymap

Start Clonezilla

device-image

local\_dev

choose the flash drive to save image on

choose directory name of flash drive to save in

Beginner mode

Savedisk

choose image name

choose the hdd to copy (usually sda)

skip

Yes, check saved images

Not to encrypt image

make sure everything is correct. “y” to continue “n” to abort

After everything is done, poweroff

## Restoring an image [4]

Connect the flash drive with clonezilla on it and the flash drive with the image on it to the computer. Boot from clonezilla.

Use default settings when clonezilla boots.

Choose language English.

Don’t touch keymap

Start Clonezilla

device\_image

local\_dev

choose flash drive with image

choose directory image is in

Beginner

NOTE: If the size of the HDD in the new node is a different size the the original node you took the image from you may need to go to expert mode (instead of beginner) and mark -icds

restoredisk

select the image

choose target partition to put image on (usually sda1)

Enter

”y” to start writing to disk, “n” to abort”

Poweroff

# New stuff

## Set hostname

check hostname:

$ hostnamectl status

change hostname:

$ hostnamectl set-hostname <new-host-name-here>

## Networking

To enable network card:

$ nmtui

Choose your network card.

Add am X on “automatically turn on”

reboot

~~To split network card (assuming main network card is eno1):  
ifconfig eno1:0 10.0.0.9 netmask 255.255.255.0 up &~~

create a file in /etc/sysconfig/network-scripts name ifcfg-eno1:0 (where eno1 is your interface) and insert in it (changing adevice, hwaddr, ipaddr and gateway appropriately):

DEVICE="eno1:0"

BOOTPROTO=static

TYPE="Ethernet"

HWADDR=38:60:77:d6:03:e6

IPADDR=10.0.0.100

NETMASK=255.255.255.0

ONBOOT=yes

GATEWAY=10.0.0.100

## Create SSH-Key

On computer to that you want to SSH into:

$ ssh-keygen -t rsa

(enter all the way through for defaults)

$ ssh-copy-id <user>@<destination>

## Change Date on Machines

date MMDDHHMMYYYY

$ hwclock –systohc

## On Addition of Nodes:

### Head Node

/var/named/forward.<zonename>

/var/named/reverse.<zonename>

Turn off pbs\_server with $ qterm

→ Add node to /var/spool/torque/server\_priv/nodes

→ Restart pbs\_server with $ systemctl start pbs\_server.service

Reboot

Add udp\_send\_channel in /etc/ganglia/gmond.conf

### Nodes

/etc/hosts  
/etc/hostname  
/etc/sysconfig/network  
/etc/sysconfig/network-scripts/ifcfg-eno1:0

Add udp\_send\_channel in /etc/ganglia/gmond.conf

If network adapter does not appear in $ ifconfig you may need to change the name of the ifcfg file in /etc/sysconfig/network-scripts (in most cases from ifcfg-eno1 to ifcfg-enp0s25)

# References

|  |  |
| --- | --- |
| [1] | "RHEL7: Configure a LDAP directory service for user connection," [Online]. Available: https://www.certdepot.net/rhel7-configure-ldap-directory-service-user-connection/ . |
| [2] | "RHEL7: Configure a system to use an existing LDAP directory service for user and group information," [Online]. Available: https://www.certdepot.net/rhel7-configure-system-use-existing-ldap-directory-service-user-group-information/. |
| [3] | "Setting Up Real-Time Monitoring with ‘Ganglia’ for Grids and Clusters of Linux Servers," [Online]. Available: http://www.tecmint.com/install-configure-ganglia-monitoring-centos-linux/. |
| [4] | "How to Install Ganglia on CentOS 7," [Online]. Available: http://www.slothparadise.com/how-to-install-ganglia-on-centos-7/. |
| [5] | "Save Disk Image," [Online]. Available: http://clonezilla.org/show-live-doc-content.php?topic=clonezilla-live/doc/01\_Save\_disk\_image. |
| [6] | "Restore Disk Image," [Online]. Available: http://clonezilla.org/show-live-doc-content.php?topic=clonezilla-live/doc/02\_Restore\_disk\_image . |
| [7] | "Google Fonts," [Online]. Available: https://fonts.google.com/?selection.family=Arvo|Lato|Nixie+One|Source+Code+Pro. |