HIGH PERFORMANCE COMPUTING - CASE STUDY

AIM : To build a two node Disk-less HPC-Cluster using XCAT, with Slurm, Nagios and LDAP

Table of contents

Group Members:

Sr No.	Name	PRN
01	Shubham Nimkar	230340127009
02	Vikas Rai	230340127013
03	Subhashit Sathe	230340127011
04	Manjiri Khedekar	230340127039

Prerequisites:

1. Three VM's. One VM will be in master mode with NAT and Host-only configurations
   Another two will only be in Host - only configuration.
2. Master node to be booted with Centos-7 with above mentioned network adapters.

- ## xCAT (Extreme Cloud Administration Toolkit)

  The xCAT is a collection of tools, most of which are script-based, that you can use to build, configure, administer, and maintain Linux clusters. You can use xCAT on any cluster, but it works particularly well on high-performance clusters, horizontal-scaling clusters, and administrative clusters.

  - Here we are using xCAT as a provisioning tool for our cluster. Which will help us to create two diskless nodes.

- **Installation on master node:**

- **Step 1: Disable firewalld and SELINUX**

  [root@master~]# systemctl stop firewalld

  [root@master~]# systemctl disable firewalld

  [root@master~]# systemctl status firewalld

  [root@master~]# setenforce 0

- **Step 2 : Get repos and install xcat**

  [root@master ~]# yum install yum-utils

  root@master ~]# wget -P /etc/yum.repos.d https://xcat.org/files/xcat/repos/yum/latest/xcat-core/xcat-core.repo --no-check-certificate

[root@master~]# wget -P /etc/yum.repos.d https://xcat.org/files/xcat/repos/yum/xcat-dep/rh7/x86_64/xcat-dep.repo --no-check-certificate

```
[root@master ~]# wget -P /etc/yum.repos.d https://xcat.org/files/xcat/repos/yum/latest/xcat-core/xcat-core.repo --no-check-certificate
--2023-07-22 09:40:27--  https://xcat.org/files/xcat/repos/yum/latest/xcat-core/xcat-core.repo
Resolving xcat.org (xcat.org)... 166.70.135.166
Connecting to xcat.org (xcat.org)|166.70.135.166|:443... connected.
WARNING: cannot verify xcat.org's certificate, issued by '/C=US/O=Let's Encrypt/CN=R3':
  Issued certificate has expired.
HTTP request sent, awaiting response... 200 OK
Length: 206
Saving to: '/etc/yum.repos.d/xcat-core.repo'

100%[===================================================================>] 206         --.-K/s   in 0s

2023-07-22 09:40:30 (12.4 MB/s) - '/etc/yum.repos.d/xcat-core.repo' saved [206/206]

[root@master ~]# wget -P /etc/yum.repos.d https://xcat.org/files/xcat/repos/yum/xcat-dep/rh7/x86_64/xcat-dep.repo --no-check-certificate
--2023-07-22 09:40:42--  https://xcat.org/files/xcat/repos/yum/xcat-dep/rh7/x86_64/xcat-dep.repo
Resolving xcat.org (xcat.org)... 166.70.135.166
Connecting to xcat.org (xcat.org)|166.70.135.166|:443... connected.
WARNING: cannot verify xcat.org's certificate, issued by '/C=US/O=Let's Encrypt/CN=R3':
  Issued certificate has expired.
HTTP request sent, awaiting response... 200 OK
Length: 209
Saving to: '/etc/yum.repos.d/xcat-dep.repo'

100%[===================================================================>] 209         --.-K/s   in 0s

2023-07-22 09:40:43 (26.1 MB/s) - '/etc/yum.repos.d/xcat-dep.repo' saved [209/209]
```

[root@master~]# yum install xCAT -y

Step 3 : Enable xCAT

[root@master ~]# . /etc/profile.d/xcat.sh

```
[root@master ~]# . /etc/profile.d/xcat.sh
```

Step 4 : Make interfaces and copy centos7.iso and we get images that can be seen below

[root@master ~]# lsdef -t osimage

centos7.9-x86_64-install-compute (osimage)
centos7.9-x86_64-netboot-compute (osimage)
centos7.9-x86_64-statelite-compute (osimage)

```
[root@master ~]# lsdef -t osimage
centos7.9-x86_64-install-compute  (osimage)
centos7.9-x86_64-netboot-compute  (osimage)
centos7.9-x86_64-statelite-compute  (osimage)
```

Step 5 : Select required image as we are doing diskless installation we are taking centos7.9-x86_64-install-compute (osimage)

[root@master ~]# packimage centos7.9-x86_64-netboot-compute

```
[root@master ~]# packimage centos7.9-x86_64-netboot-compute
Packing contents of /install/netboot/centos7.9/x86_64/compute/rootimg
archive method:cpio
compress method:gzip
```

Step 6 : Make nodes, means add nodes where we need to install OS we are having 2 nodes, enter name for nodes , IP addresses , MAC addresses.

[root@master ~]# mkdef -t node node1 groups=compute,all ip=10.10.10.168 mac=00:0C:29:39:97:E3 netboot=xnba

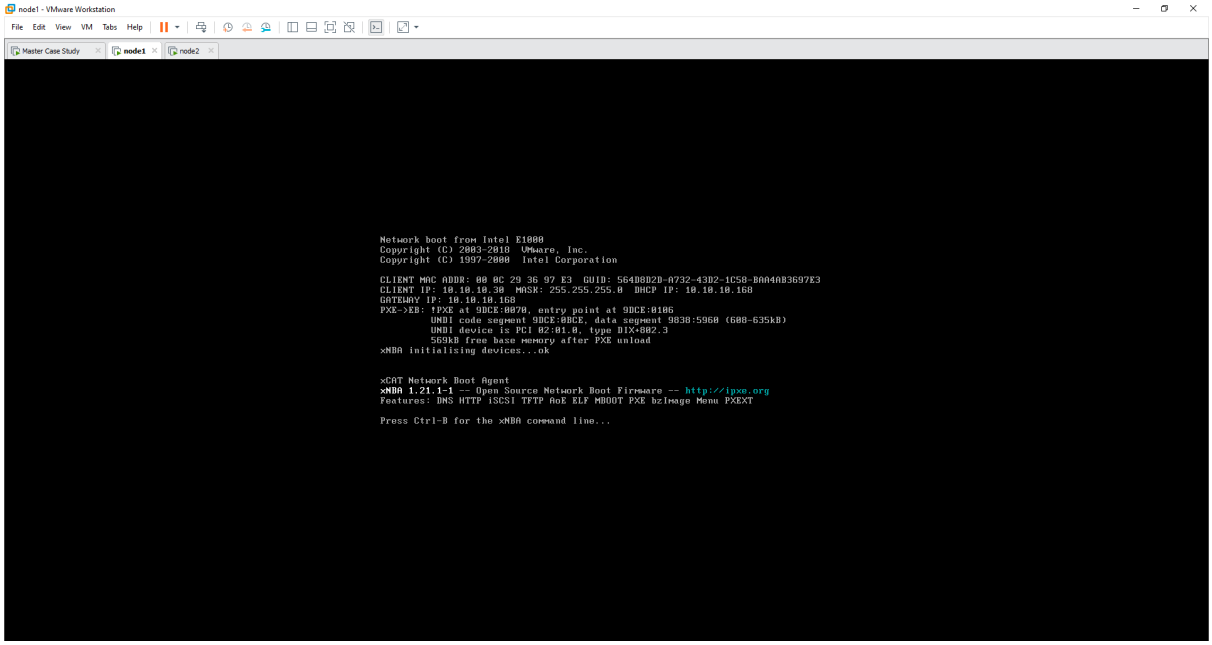
[root@master ~]# mkdef -t node node2 groups=compute,all ip=10.10.10.169 mac=00:0C:29:EB:C4:3D netboot=xnba

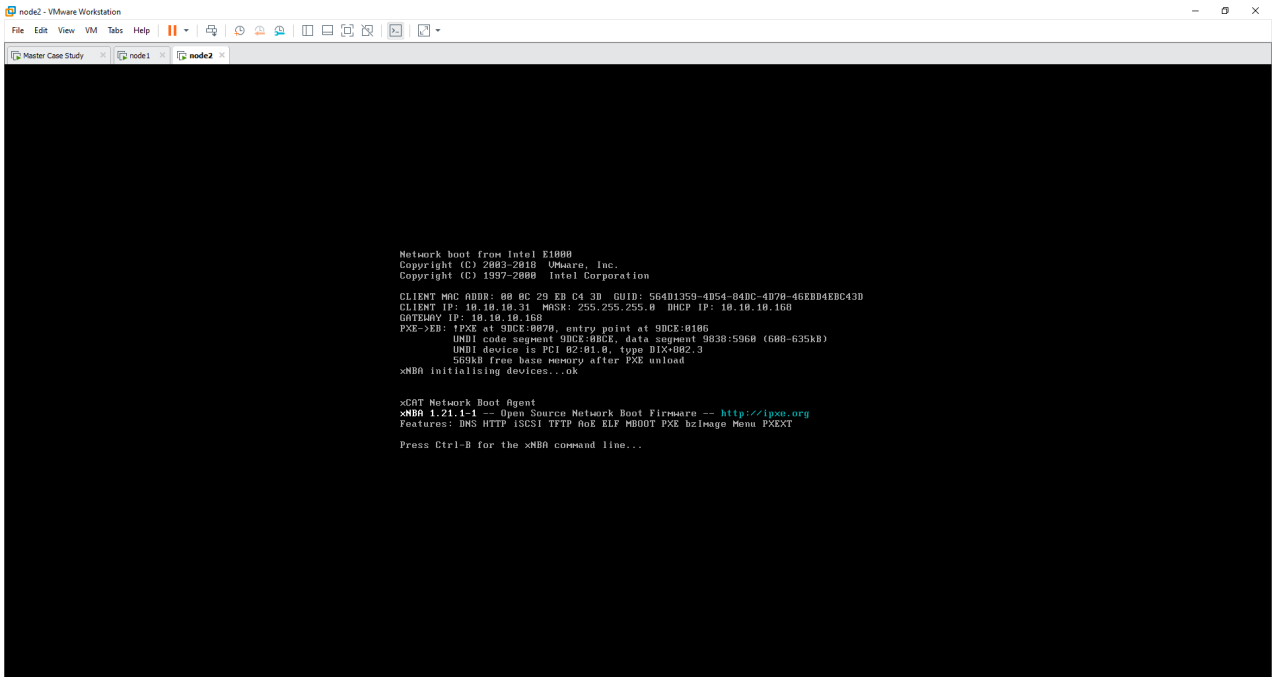
```
[root@master ~]# mkdef -f -t node node1 groups=compute,all ip=10.10.10.168 mac=00:0C:29:36:97:E3 netboot=xnba
1 object definitions have been created or modified.
[root@master ~]# mkdef -f -t node node2 groups=compute,all ip=10.10.10.168 mac=00:0C:29:EB:C4:3D netboot=xnba
1 object definitions have been created or modified.
```

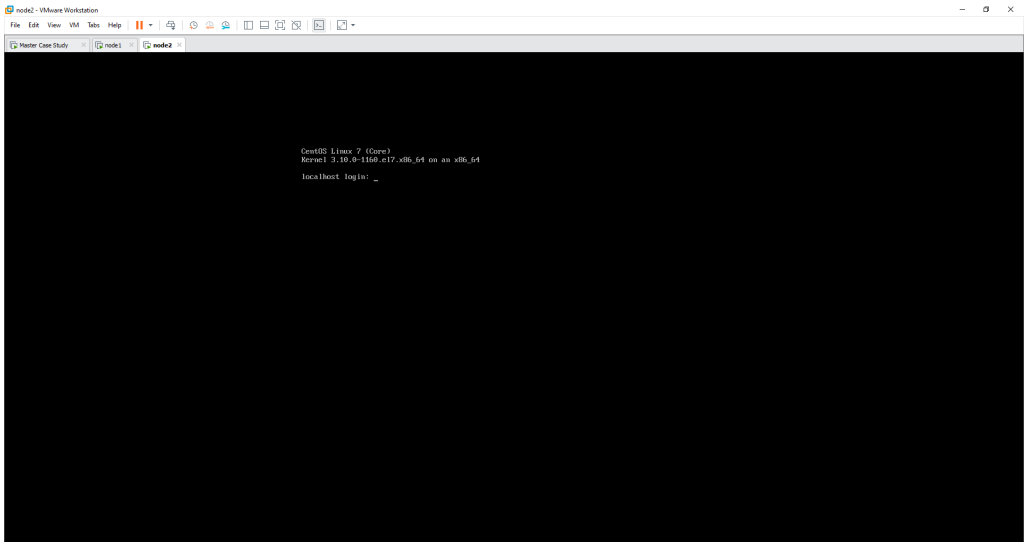
[root@master ~]# nodeset compute osimage=centos7.9-x86_64-netboot-compute

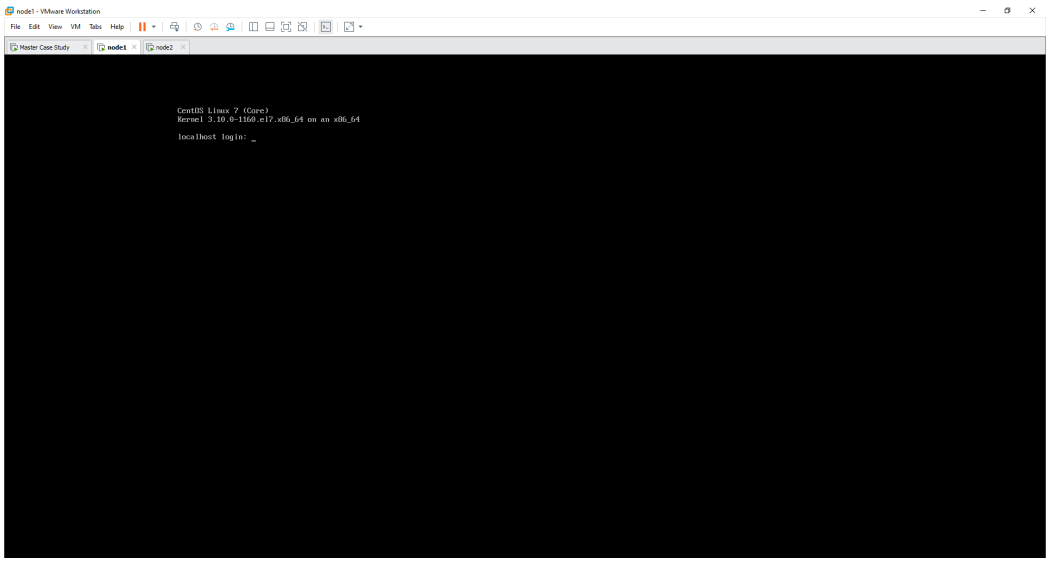
```
[root@centos7 ~]# nodeset compute osimage=centos7.9-x86_64-netboot-compute
xcat-node1: netboot centos7.9-x86_64-compute
xcat-node2: netboot centos7.9-x86_64-compute
```

Here in images below you can see our nodes are booted with the image we've selected and they are up and running.









In Above manner we can provision our cluster using xCAT tool.

- ## LDAP (Lightweight Directory Access Protocol)

Lightweight directory access protocol (LDAP) is a protocol that helps users find data about organizations, persons, and more. LDAP has two main goals: to store data in the LDAP directory and authenticate users to access the directory.

- **Installation**

- **Step 1 : On master :**

```
yum install -y openldap-servers openldap-clients
cp /usr/share/openldap-servers/DB_CONFIG.example  /var/lib/ldap/DB_CONFIG
```

- **Change ownership :**

```
chown ldap. /var/lib/ldap/DB_CONFIG
```

- **Start slapd service :**

```
systemctl start slapd
systemctl enable slapd
systemctl status slapd
```

- **Add Schema and users which you want to configure**

```
[root@master ~]# cat chdomain.ldif
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=cdac,dc=in" read by * none

dn: olcDatabase={2}hdb,cn=config
changetype: modify
replace: olcSuffix
olcSuffix: dc=cdac,dc=in

dn: olcDatabase={2}hdb,cn=config
changetype: modify
```

```
replace: olcRootDN
olcRootDN: cn=Manager,dc=cdac,dc=in

dn: olcDatabase={2}hdb,cn=config
changetype: modify
add: olcRootPW
olcRootPW: {SSHA}Xw4kWMRG1Su3XeqSF/bMah5RCgDH54o+

dn: olcDatabase={2}hdb,cn=config
changetype: modify
add: olcAccess
olcAccess: {0}to attrs=userPassword,shadowLastChange by
  dn="cn=Manager,dc=cdac,dc=in" write by anonymous auth by self write by *
      none
olcAccess: {1}to dn.base="" by * read
olcAccess: {2}to * by dn="cn=Manager,dc=cdac,dc=in" write by * read
```

[root@master ~]# ldapmodify -Y EXTERNAL -H ldapi:/// -f chdomain.ldif

```
SASL/EXTERNAL authentication started
SASL username: gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth
SASL SSF: 0
modifying entry "olcDatabase={1}monitor,cn=config"

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

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

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

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

[root@master ~]# vi basedomain.ldif
[root@master ~]# cat basedomain.ldif

```
dn: dc=cdac,dc=in
objectClass: top
objectClass: dcObject
objectclass: organization
o: cdac in
dc: cdac
```

```
dn: cn=Manager,dc=cdac,dc=in
objectClass: organizationalRole
cn: Manager
description: Directory Manager

dn: ou=People,dc=cdac,dc=in
objectClass: organizationalUnit
ou: People

dn: ou=Group,dc=cdac,dc=in
objectClass: organizationalUnit
ou: Group
```

[root@master ~]# ldapadd -x -D cn=Manager,dc=cdac,dc=in -W -f basedomain.ldif

```
Enter LDAP Password:
adding new entry "dc=cdac,dc=in"
adding new entry "cn=Manager,dc=cdac,dc=in"
adding new entry "ou=People,dc=cdac,dc=in"
adding new entry "ou=Group,dc=cdac,dc=in"
```

[root@master ~]# vim ldapuser.ldif

[root@master ~]# ldapadd -x -D cn=Manager,dc=cdac,dc=in -W -f ldapuser.ldif

```
Enter LDAP Password:
adding new entry "uid=user1,ou=People,dc=cdac,dc=in"

adding new entry "cn=user1,ou=Group,dc=cdac,dc=in"
```

[root@master ~]# cat ldapuser.ldif

```
dn: uid=user1,ou=People,dc=cdac,dc=in
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
cn: user1
sn: test
userPassword: {SSHA}Xw4kWMRG1Su3XeqSF/bMah5RCgDH54o+
```

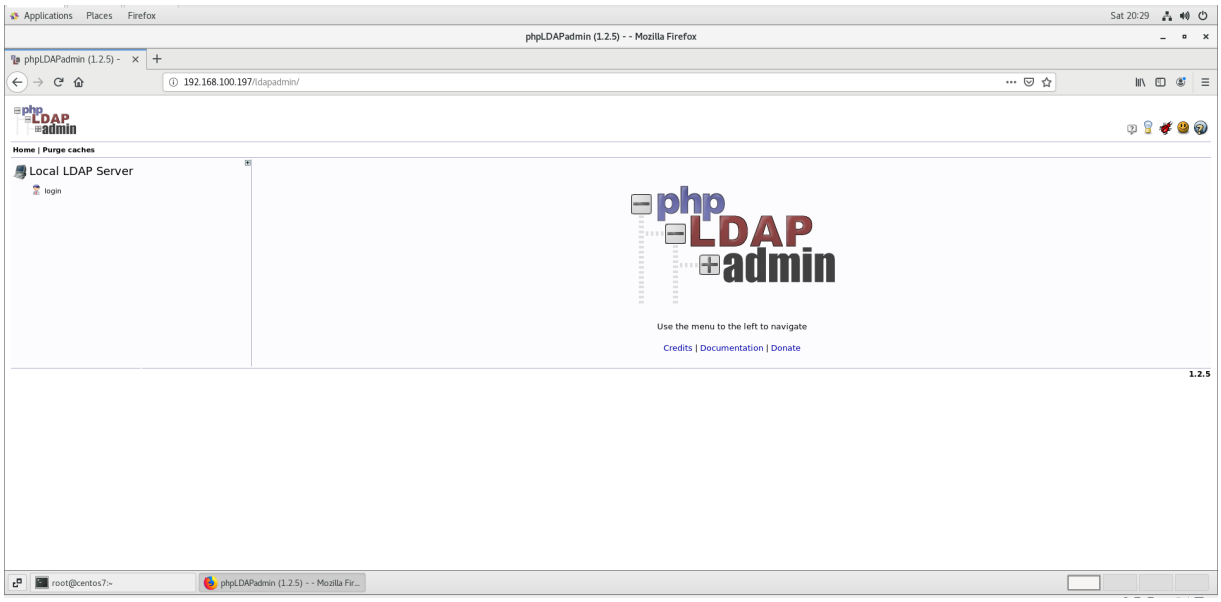
loginShell: /bin/bash
uidNumber: 1001
gidNumber: 1001
homeDirectory: /home/user1

dn: cn=user1,ou=Group,dc=cdac,dc=in
objectClass: posixGroup
cn: user1
gidNumber: 1001

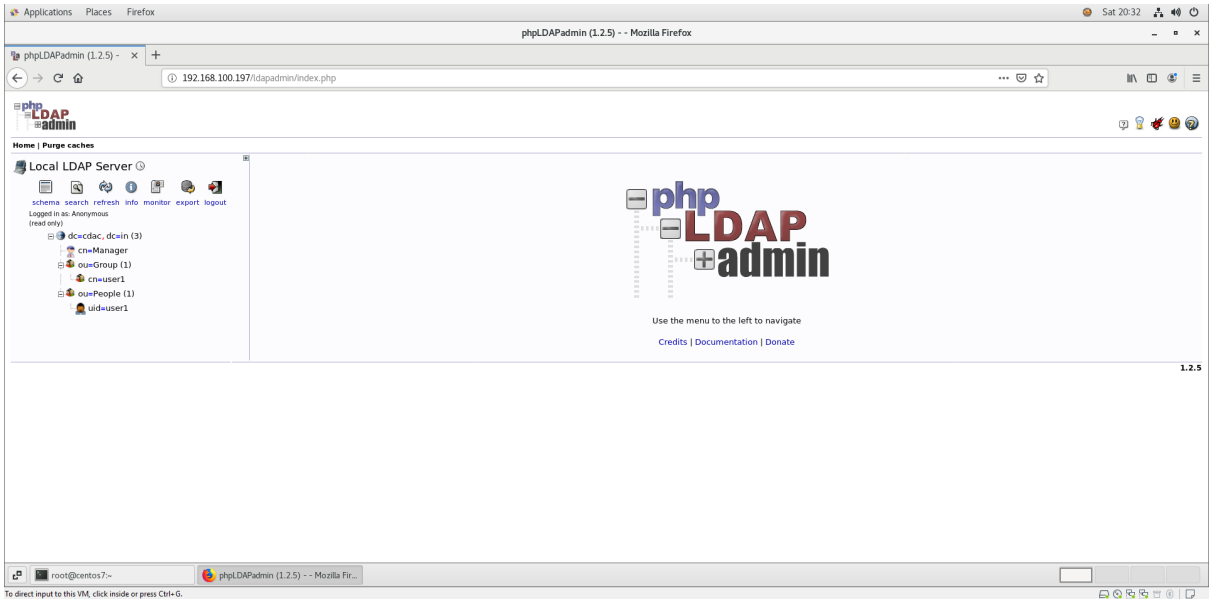
- In similar way install services on client machines too (nslcd)

- **Go to WebUI to view the users and schema we have created , <u>http://localhost:ldapadmin</u>**

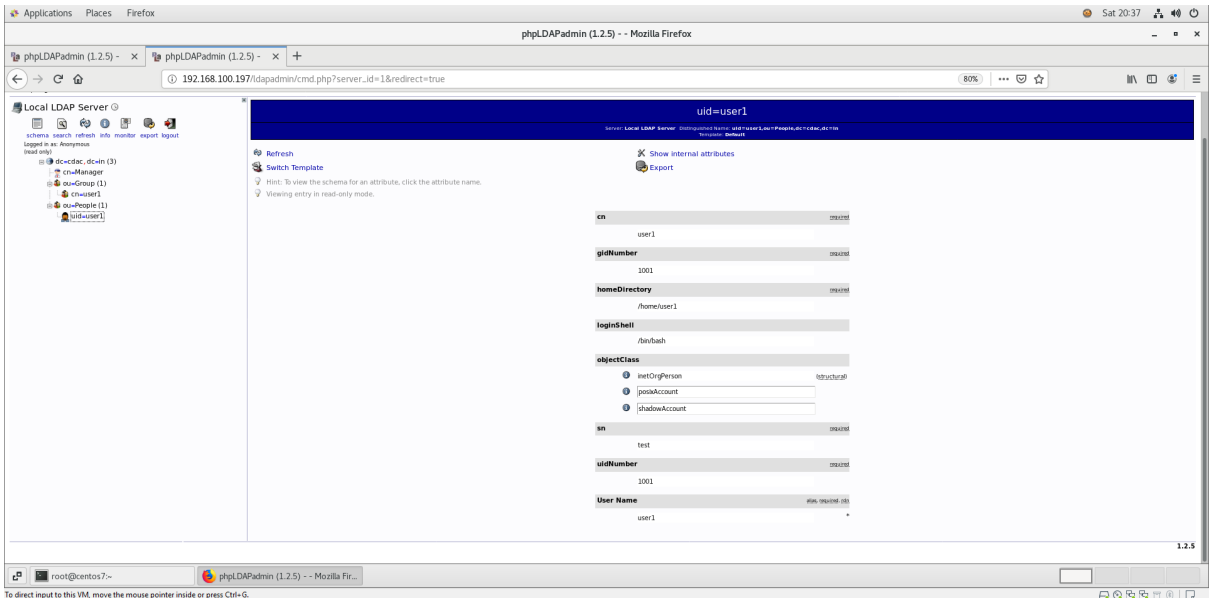
  Below you can see the landing page

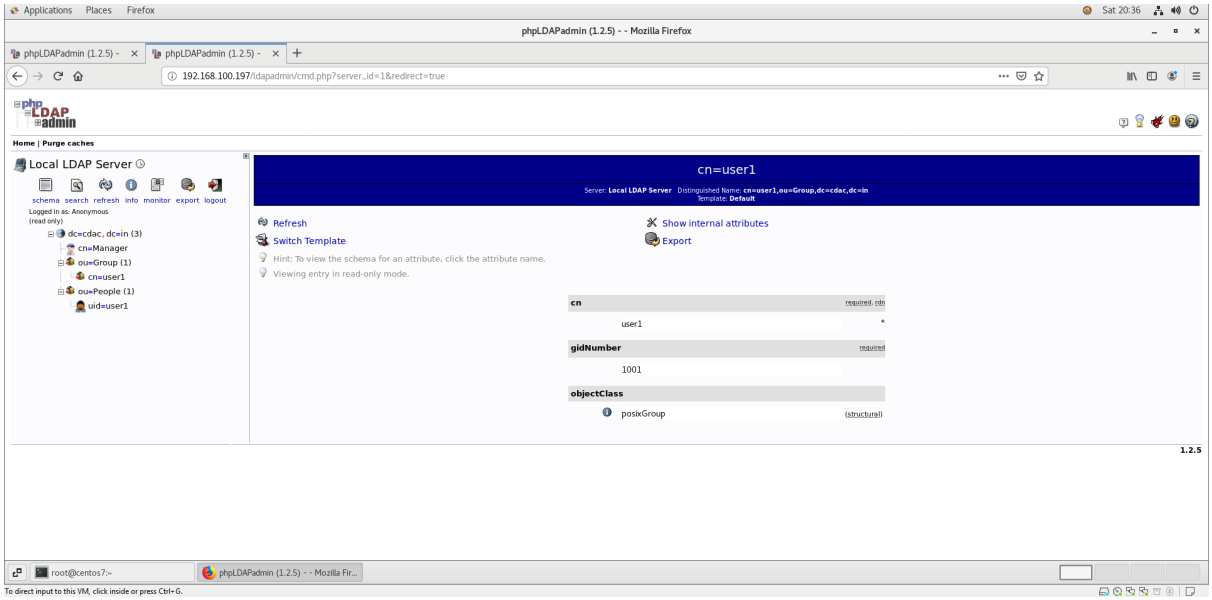


- **Check for our created users**



- **User 1**




Applications Places Firefox
Sat 20:36
phpLDAPadmin (1.2.5) - - Mozilla Firefox
phpLDAPadmin (1.2.5) -
phpLDAPadmin (1.2.5) -
192.168.100.197/ldapadmin/cmd.php?server_id=1&redirect=true
Home | Purge caches
Local LDAP Server
schema search refresh info monitor export logout
Logged in as: Anonymous
(read only)
dc=cdac, dc=in (3)
cn=Manager
ou=Group (1)
cn=user1
ou=People (1)
uid=user1
cn=user1
Server: Local LDAP Server Distinguished Name: cn=user1,ou=Group,dc=cdac,dc=in
Template: Default
Refresh
Show internal attributes
Switch Template
Export
Hint: To view the schema for an attribute, click the attribute name.
Viewing entry in read-only mode.
cn
required, rdn
user1
*
gidNumber
required
1001
objectClass
posixGroup
(structural)
1.2.5
root@centos7:~
phpLDAPadmin (1.2.5) - - Mozilla Fir...
To direct input to this VM, click inside or press Ctrl+G.

- ## NAGIOS

Nagios is an open source IT system monitoring tool. It was designed to run on the Linux operating system and can monitor devices running Linux, Windows and Unix OSes. Nagios software runs periodic checks on critical parameters of application, network and server resources.

- **Installation on master :**

- **Install required services and download nagios tar and make configurations**

```
yum install -y gcc glibc glibc-common wget unzip httpd php gd gd-devel perl postfix
cd /tmp

wget -O nagioscore.tar.gz https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.4.3.tar.gz

cd /tmp/nagioscore-nagios-4.4.3/
./configure

make all

make install-groups-users

usermod -a -G nagios apache

make install
make install-daemoninit

systemctl enable httpd.service
make install-commandmode
make install-config
make install-webconf
```

- **Allow nagios services through firewall**

```
firewall-cmd --zone=public --add-port=80/tcp
firewall-cmd --zone=public --add-port=80/tcp --permanent
```

htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin

- **Installing Nagios Plug-in**

yum install -y make gettext automake autoconf wget openssl-devel net-snmp net-snmp-utils epel-release
yum install -y perl-Net-SNMP
cd /tmp
wget --no-check-certificate -O nagios-plugins.tar.gz https://github.com/nagios-plugins/nagios-plugins/archive/release-2.2.1.tar.gz
tar zxf nagios-plugins.tar.gz
cd /tmp/nagios-plugins-release-2.2.1/
./tools/setup
./configure
make
make install

- **Install Linux Host**

- **Configure EPEL repository and install nrpe plugins and configure it**

yum install epel-release -y
yum install nrpe nagios-plugins-all

vi /etc/nagios/nrpe.cfg

allowed_hosts=127.0.0.1,nagios_server_ip

vi /etc/nagios/nrpe.cfg

command[check_users]=/usr/lib64/nagios/plugins/check_users -w 5 -c 10
command[check_load]=/usr/lib64/nagios/plugins/check_load -w 15,10,5 -c 30,25,20
command[check_root]=/usr/lib64/nagios/plugins/check_disk -w 20% -c 10% -p /dev/mapper/centos-root
command[check_swap]=/usr/lib64/nagios/plugins/check_swap -w 20% -c 10%
command[check_total_procs]=/usr/lib64/nagios/plugins/check_procs -w 150 -c 200

```
systemctl start nrpe
systemctl enable nrpe
```

- **Configure nagios server**

```
yum -y install nagios-plugins-nrpe

vi /usr/local/nagios/etc/nagios.cfg   ..... add...
cfg_dir=/usr/local/nagios/etc/servers

mkdir /usr/local/nagios/etc/servers

vi /usr/local/nagios/etc/objects/commands.cfg
# .check_nrpe. command definition
define command{
command_name check_nrpe
command_line /usr/lib64/nagios/plugins/check_nrpe -H $HOSTADDRESS$ -t 30
        -c $ARG1$
}

vi /usr/local/nagios/etc/servers/node1.cfg

define host{

        use                 linux-server

        host_name           node1

        alias               node1

        address             client_ip

}
```

- verify nagios for errors

```
        /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
```

```
systemctl restart nagios

firewall-cmd --permanent --add-port=5666/tcp
firewall-cmd --reload
```

- **Running bash script as nrpe command on linux host**

- **create script /usr/lib64/nagios/plugins/disk_check.sh**

```
#!/bin/bash
used_space=`df -h / | grep -v Filesystem | awk '{print $5}' | sed 's/%//g'`
case $used_space in
[1-84]*)
echo "OK - $used_space% of disk space used."
exit 0
;;
[85]*)
echo "WARNING - $used_space% of disk space used."
exit 1
;;
[86-100]*)
echo "CRITICAL - $used_space% of disk space used."
exit 2
;;
*)
echo "UNKNOWN - $used_space% of disk space used."
exit 3
;;
Esac
```

```
chmod +x /usr/lib64/nagios/plugins/disk_check.sh
```

- **Make changes in nrpe.cfg**

```
vi /etc/nagios/nrpe.cfg
```

command[diskcheck_script]=/usr/lib64/nagios/plugins/disk_check.sh

restart the nrpe service on client

- server side configuration

```
vi /etc/nagios/objects/commands.cfg
define command{
        command_name    diskcheck_script
          command_line      $USER1$/check_nrpe -H $HOSTADDRESS$ -c diskcheck_script
        }
save the file
```

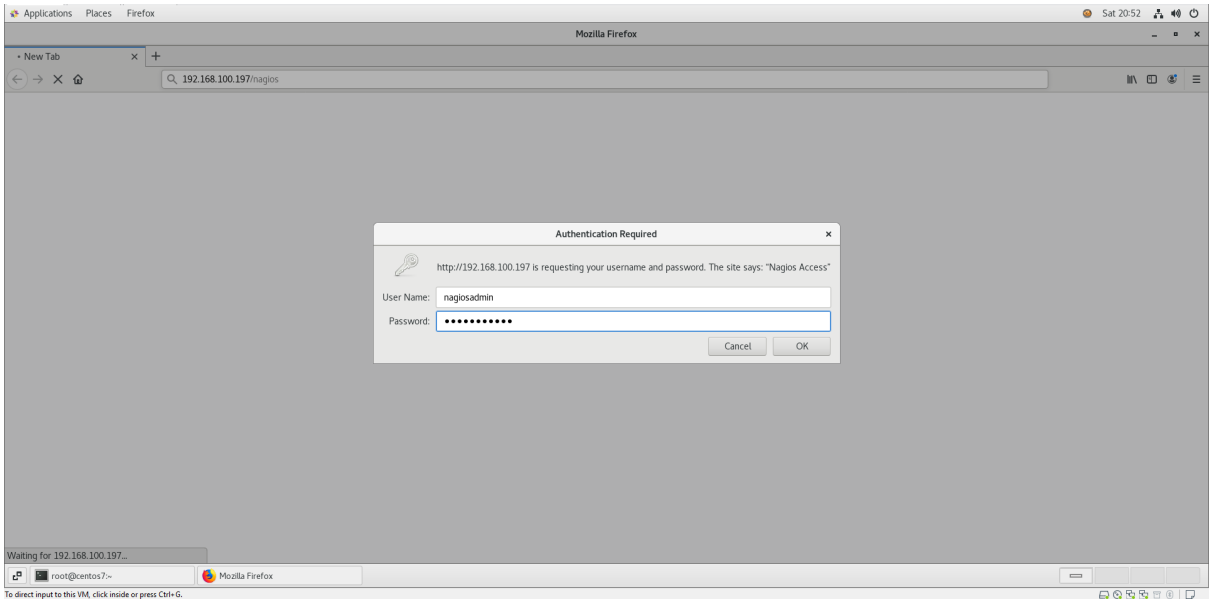
- vi /etc/nagios/servers/client.cfg (the cfg file for client where script is created) and add following.

```
define service {
        use                     generic-service
        host_name               node1
        service_description     Custom Disk Checker Bash Script
        check_command           diskcheck_script
        }
```

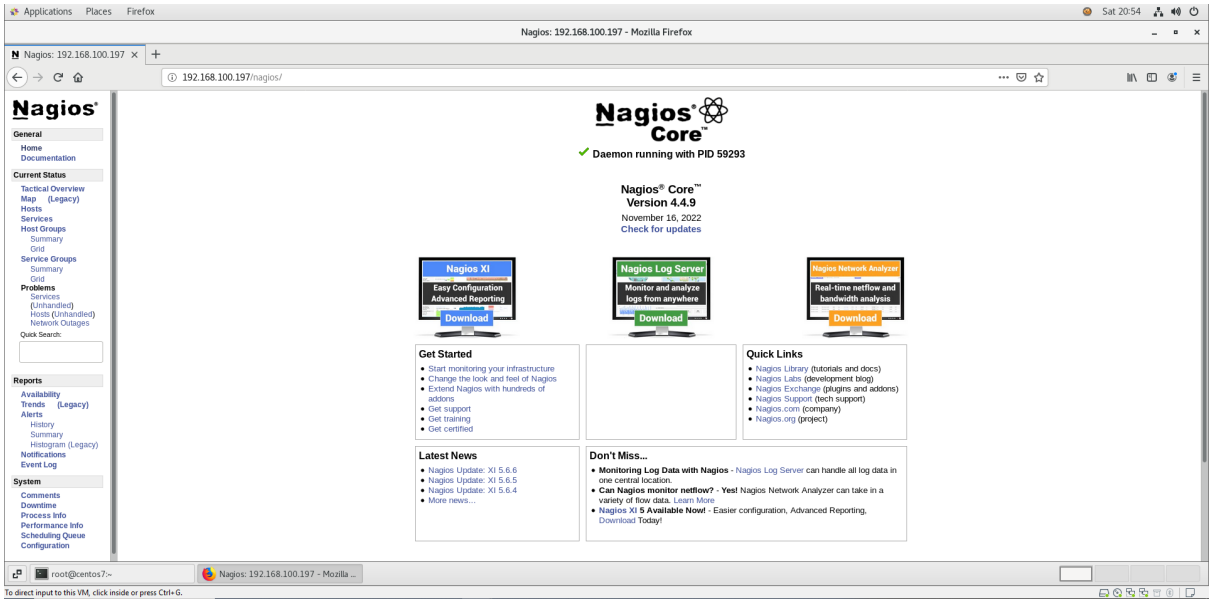
save the file.

Restart the nagios service.

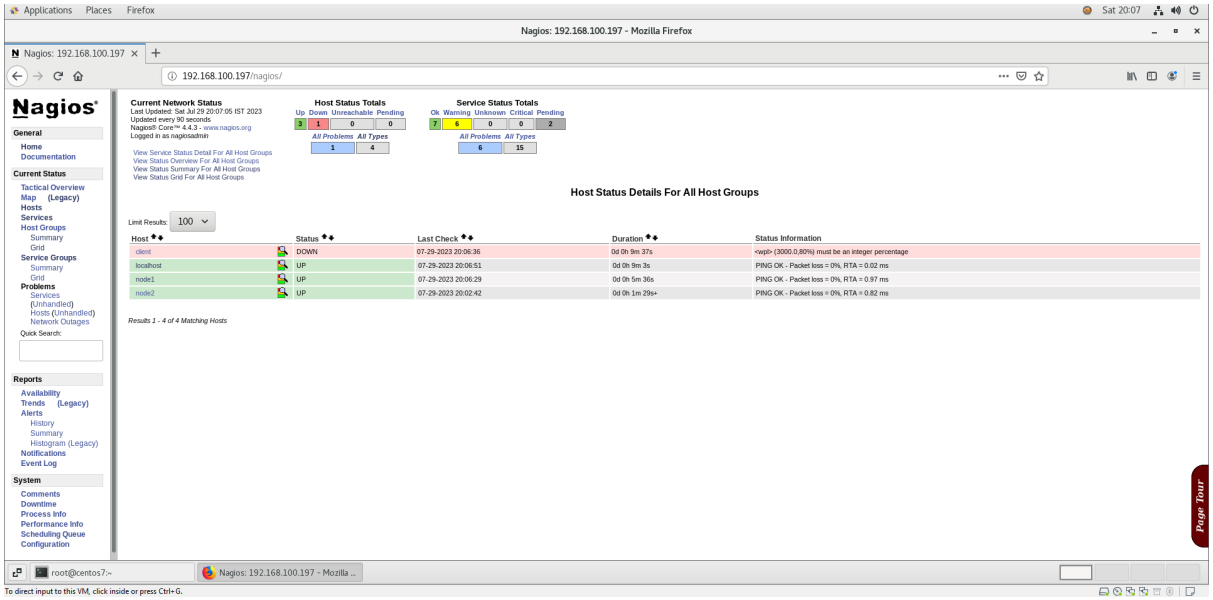
- **Nagios WebUI: http://localhost/nagios**

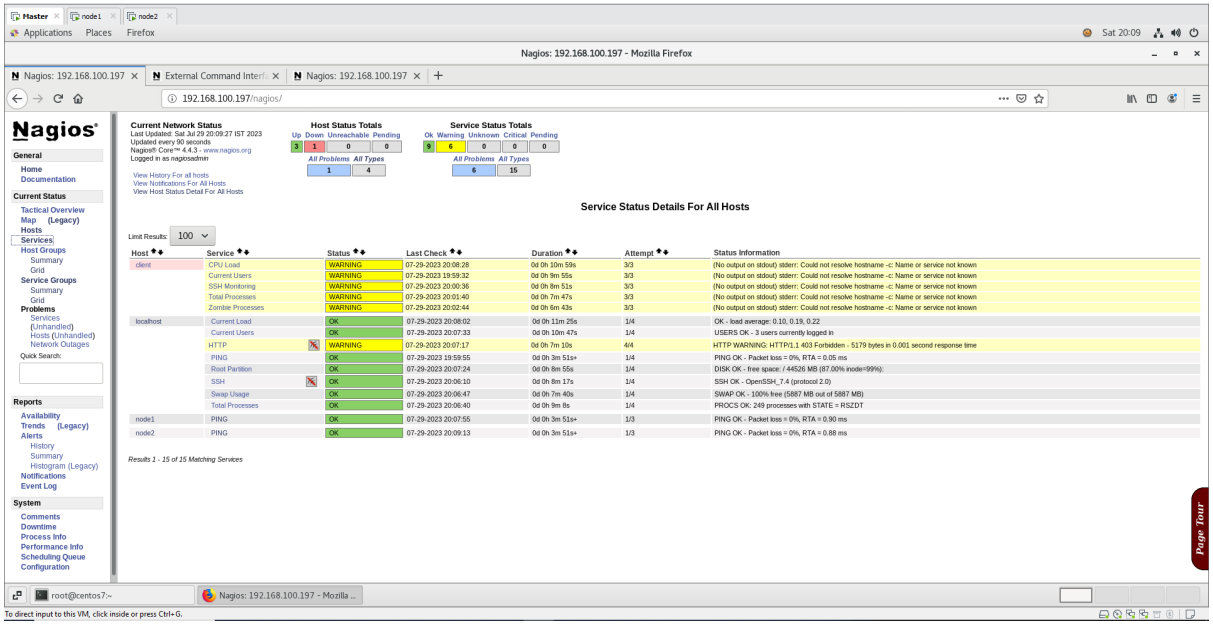


- Landing Page



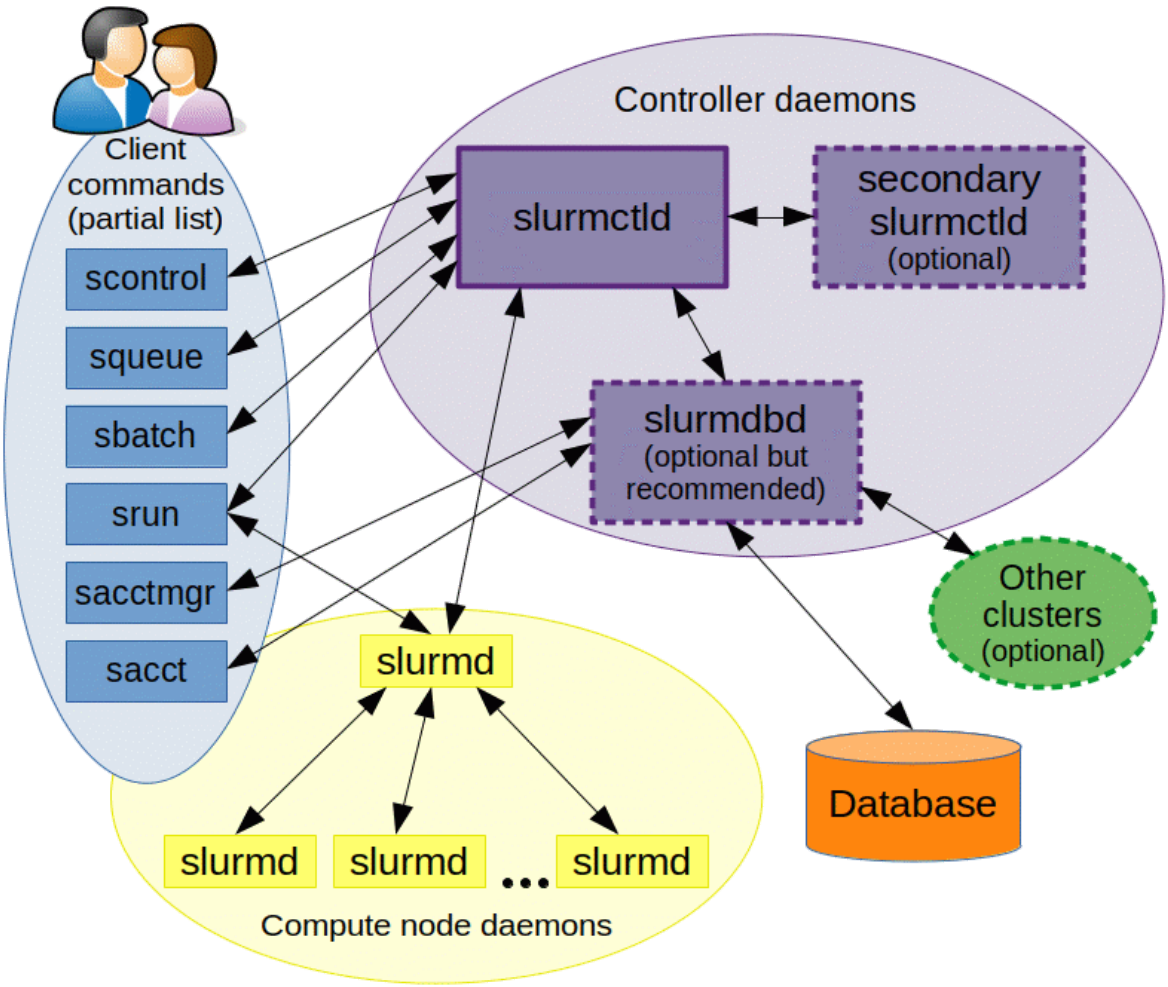
- **Here we can see added nodes and parameters of the same**





- ## SLURM (Simple Linux Utility for Resource Management)

The Slurm Workload Manager, formerly known as Simple Linux Utility for Resource Management (SLURM), or simply Slurm, is a free and open-source job scheduler for Linux and Unix-like kernels, used by many of the world's supercomputers and computer clusters.



- **Installation on master**

- **Configure NFS Server Master**

yum install -y nfs-utils
Once the packages are installed, enable and start NFS services.
systemctl start nfs-server rpcbind
systemctl enable nfs-server rpcbind
Create NFS Share

- **Now, let's create a directory to share with the NFS client. Here I will be creating a new directory named home in the / partition.**

  mkdir /home

- **Allow NFS client to read and write to the created directory.**

  chmod 777 /home/

- **We have to modify /etc/exports file to make an entry of directory /home that you want to share.**

  vi /etc/exports

  and add this

  /home *(rw,sync,no_root_squash)

[/home ip needs to be here where clients are there(rw,sync,no_root_squash)]

- **Export the shared directories using the following command.**

  exportfs -r

- **Install NFS Client:**

  yum install -y nfs-utils

- Check NFS Share
  Before mounting the NFS share, I request you to check the NFS shares available on the NFS server by running the following command on the NFS client.

  showmount -e 192.168.100.197

```
[root@master ~]# showmount -e 192.168.100.197
Export list for 192.168.100.197:
/shared *
[root@master ~]# mount | grep nfs
sunrpc on /var/lib/nfs/rpc_pipefs type rpc_pipefs (rw,relatime)
nfsd on /proc/fs/nfsd type nfsd (rw,relatime)
[root@master ~]#
```

- Mount NFS Share

Now, create a directory on NFS client to mount the NFS share /home which we have created in the NFS server.

mkdir /mnt/home

Use below command to mount a NFS share /home from NFS server 192.168.100.197 in /mnt/nfsfileshare on NFS client.

mount 192.168.100.197:/home /mnt/home

[root@node1 ~]# mount | grep nfs

Create a file on the mounted directory to verify the read and write access on NFS share.

touch /mnt/home/test
Automount NFS Shares
To mount the shares automatically on every reboot, you would need to modify /etc/fstab file of your NFS client.

```
#
# /etc/fstab
# Created by anaconda on Thu Jul 27 16:18:46 2023
#
# Accessible filesystems, by reference, are maintained under '/dev/disk'
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info
#
/dev/mapper/centos-root /                       xfs     defaults        0 0
UUID=2c31b501-4b46-4cd9-903e-0db4d415aa4a /boot                   xfs     defaults        0 0
/dev/mapper/centos-home /home                   xfs     defaults        0 0
/dev/mapper/centos-swap swap                    swap    defaults        0 0
192.168.100.197:/shared /mnt/shared             nfs     nosuid,rw,sync,hard,intr 0 0
~
```

- **Verify the mounted share on the NFS client using mount command.**

  mount | grep nfs

```
[root@master ~]# showmount -e 192.168.100.197
Export list for 192.168.100.197:
/shared *
[root@master ~]# mount | grep nfs
sunrpc on /var/lib/nfs/rpc_pipefs type rpc_pipefs (rw,relatime)
nfsd on /proc/fs/nfsd type nfsd (rw,relatime)
[root@master ~]#
```

- **Create munge key:**

- master
  [root@master home]# rpm -qa | grep munge
  munge-libs-0.5.11-3.el7.x86_64
  munge-devel-0.5.11-3.el7.x86_64
  munge-0.5.11-3.el7.x86_64

```
[root@master ~]# rpm -qa | grep munge
munge-libs-0.5.11-3.el7.x86_64
munge-devel-0.5.11-3.el7.x86_64
munge-0.5.11-3.el7.x86_64
[root@master ~]#
```

- To check the key
  [root@master home]# ll /etc/munge/
  total 4

-r--------. 1 munge munge 1024 Jul 13 16:55 munge.key

```
[root@master ~]# ll /etc/munge/
total 4
-r--------. 1 munge munge 1024 Jul 27 17:15 munge.key
[root@master ~]#
```

scp /etc/munge/munge.key client1 client2:/etc/munge/
chown munge:munge /etc/munge/

- on all clients
  chown munge:munge /etc/munge/munge.key
  on all nodes

```
[root@node1 ~]# ll /etc/munge/
total 4
-r--------. 1 munge munge 1024 Jul 27 17:17 munge.key
[root@node1 ~]#
```

- **ONMASTER**

- Slurm.conf File :

  [root@master ~]# vi /etc/slurm/slurm.conf

```
#
# Example slurm.conf file. Please run configurator.html
# (in doc/html) to build a configuration file customized
# for your environment.
#
#
# slurm.conf file generated by configurator.html.
#
# See the slurm.conf man page for more information.
#
ClusterName=hpcsa
```

```
ControlMachine=master
#ControlAddr=
#BackupController=
#BackupAddr=
#
SlurmUser=slurm
#SlurmdUser=root
SlurmctldPort=6817
SlurmdPort=6818
AuthType=auth/munge
#JobCredentialPrivateKey=
#JobCredentialPublicCertificate=
StateSaveLocation=/var/spool/slurm/ctld
SlurmdSpoolDir=/var/spool/slurm/d
SwitchType=switch/none
MpiDefault=none
SlurmctldPidFile=/var/run/slurmctld.pid
SlurmdPidFile=/var/run/slurmd.pid
ProctrackType=proctrack/pgid
#PluginDir=
#FirstJobId=
ReturnToService=0
#MaxJobCount=
#PlugStackConfig=
#PropagatePrioProcess=
#PropagateResourceLimits=
#PropagateResourceLimitsExcept=
#Prolog=
#Epilog=
#SrunProlog=
#SrunEpilog=
#TaskProlog=
#TaskEpilog=
#TaskPlugin=
#TrackWCKey=no
#TreeWidth=50
#TmpFS=
#UsePAM=
#
# TIMERS
```

```
SlurmctldTimeout=300
SlurmdTimeout=300
InactiveLimit=0
MinJobAge=300
KillWait=30
Waittime=0
#
# SCHEDULING
SchedulerType=sched/backfill
#SchedulerAuth=
SelectType=select/cons_tres
SelectTypeParameters=CR_Core
#PriorityType=priority/multifactor
#PriorityDecayHalfLife=14-0
#PriorityUsageResetPeriod=14-0
#PriorityWeightFairshare=100000
#PriorityWeightAge=1000
#PriorityWeightPartition=10000
#PriorityWeightJobSize=1000
#PriorityMaxAge=1-0
#
# LOGGING
SlurmctldDebug=info
SlurmctldLogFile=/var/log/slurmctld.log
SlurmdDebug=info
SlurmdLogFile=/var/log/slurmd.log
JobCompType=jobcomp/none
#JobCompLoc=
#
# ACCOUNTING
#JobAcctGatherType=jobacct_gather/linux
#JobAcctGatherFrequency=30
#
#AccountingStorageType=accounting_storage/slurmdbd
#AccountingStorageHost=
#AccountingStorageLoc=
#AccountingStoragePass=
#AccountingStorageUser=
#
# COMPUTE NODES
```

#NodeName=linux[1-32] Procs=1 State=UNKNOWN
NodeName=node1 CPUs=2 Boards=1 SocketsPerBoard=2 CoresPerSocket=1 ThreadsPerCore=1 RealMemory=5666 State=UNKNOWN
NodeName=node2 CPUs=2 Boards=1 SocketsPerBoard=2 CoresPerSocket=1 ThreadsPerCore=1 RealMemory=5666 State=UNKNOWN
PartitionName=standard Nodes=ALL Default=YES MaxTime=INFINITE State=UP

- **To check if clients are online :**

[root@master ~]# sinfo

```
File  Edit  View  Search  Terminal  Help
[root@master ~]# sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
debug        up   infinite      2   idle node[1-2]
standard*    up   infinite      2   idle node[1-2]
[root@master ~]#
```