Project Report

HPC Cluster development using xCAT Provisioning

Submitted in partial fulfillment for the award of
Post Graduate Diploma in High-Performance Computing
System Administration from C-DAC ACTS (Pune)

Guided by :- Raushan Gami

Presented By :- Suryadev Chaudhary

Centre of Development of Advanced Computing (C-DAC), Pune

ACKNOWLEDGEMENT

This project "HPC Cluster Development using xCAT Provisioning"
was a great learning experience for us and we are submitting
this work to Advanced Computing Training School (CDAC ACTS).

As a cohesive team, we would like to extend our heartfelt appreciation
to all those who have contributed to the successful completion of this project.
Your collective support, guidance, and encouragement were instrumental in making
this endeavor a reality, and we are deeply grateful for your contributions.

We would like to express our sincere gratitude to Mr Nkhilesh,
our project guide, for their exceptional leadership and guidance throughout the project.
Their expertise, vision, and dedication were pivotal in steering
our team towards success.

A special thanks to Ms. Swati Salunkhe for their consistent encouragement
and the motivation they provided during challenging times
. Their positive attitude and belief in our capabilities bolstered our confidence.

Furthermore, we want to express our appreciation to our colleagues
who supported us in various capacities. From brainstorming sessions to late-night
discussions, your contributions have been invaluable.

To everyone who played a role in the success of this project,
your impact will forever be etched in our team's achievement.
Thank you for being an integral part of our journey.

From:

Er. Suryadev Chaudhary

TABLE OF CONTENTS

Introduction:-

High-Performance Computing (HPC) clusters play a pivotal role in accelerating complex computations, simulations, and data analysis. Managing these clusters efficiently is a challenge that the Extreme Cloud Administration Toolkit (xCAT) addresses. In this project, we will provide an overview of xCAT and explore how it can be integrated with LDAP, Nagios, and Slurm to optimize HPC cluster management.

1. Extreme Cloud Administration Toolkit (xCAT)

xCAT is an open-source infrastructure tool designed to streamline the management of clusters, data centers, and cloud environments. It offers a comprehensive set of features for deploying, provisioning, and managing nodes within an HPC environment. xCAT's modular architecture and flexible design make it an ideal choice for handling complex cluster configurations.

2. LDAP Integration

Lightweight Directory Access Protocol (LDAP) integration enhances the security and centralizes user management within an HPC cluster. By integrating xCAT with LDAP, you can achieve centralized authentication and authorization, simplifying user access control and ensuring a consistent user experience across the cluster.

3. Nagios Integration

Nagios is a powerful monitoring and alerting system that keeps a vigilant eye on the health and performance of your HPC cluster. Integrating xCAT with Nagios enables real-time monitoring of nodes, resources, and services. This integration enhances cluster reliability by promptly detecting and addressing issues, ensuring smooth operation and minimizing downtime.

4. Slurm Integration

Slurm, an open-source job scheduler and resource manager, optimizes workload distribution and resource allocation in HPC clusters. By integrating xCAT with Slurm, you can achieve seamless job scheduling, resource management, and workload balancing. This integration enhances cluster utilization and efficiency while maintaining optimal performance.

5. Benefits of Integration

-Efficient Management: The integration of xCAT, LDAP, Nagios, and Slurm creates a unified ecosystem that simplifies cluster administration, user management, monitoring, and job scheduling.

- Enhanced Security: LDAP integration centralizes user authentication and authorization, reducing security risks and ensuring consistent access control policies.

- Real-time Monitoring: Nagios integration provides continuous monitoring of cluster health, enabling administrators to proactively address issues before they impact performance.

- Optimized Workloads: Slurm integration allows for efficient resource allocation and job scheduling, maximizing cluster utilization and throughput.

Conclusion

In this introduction, we've outlined the capabilities of xCAT and its integration with LDAP, Nagios, and Slurm in the context of HPC cluster management. These integrations collectively enhance cluster efficiency, security, and reliability, enabling administrators to create and maintain high-performance computing environments that meet the demands of modern scientific, research, and computational workloads. In the following sections of this guide, we will delve deeper into the installation, configuration, and usage of these integrations to empower you to build and manage a robust HPC cluster.

Project Overview:-

Deploying a Cluster using xCAT, LDAP, Slurm, and Nagios**

Goals and Objectives:

This project aims to create a functional high-performance computing (HPC) cluster consisting of one master node and two worker nodes using the Extreme Cloud Administration Toolkit (xCAT). The cluster will be based on CentOS 7 and will be provisioned on bare-metal machines. The deployment will involve integrating LDAP for centralized user management, Slurm for resource allocation and job scheduling, and Nagios for real-time monitoring.

Project Objectives:

1. Cluster Deployment: Set up a 3-machine cluster, comprising one master node and two worker nodes, using xCAT's provisioning capabilities.

2. LDAP Integration: Implement LDAP (latest version) for centralized user authentication and authorization across the cluster.

3. Slurm Integration: Configure Slurm as the cluster's resource manager, enabling efficient resource allocation and job scheduling.

4. Nagios Monitoring: Integrate Nagios to monitor the health and performance of the cluster's components in real-time.

Project Steps:-

CODE:-

```
#Disable Firewall

systemctl stop firewalld.service; systemctl stop firewalld

systemctl disable firewalld.service; systemctl disable firewalld


#Disable SELINUX

setenforce 0

sed -i 's/^SELINUX=.*/SELINUX=disabled/g' /etc/sysconfig/selinux


#Disable DHCP on the interface

private_interface=$(zenity --entry --title="Private network's interface required" --text="Please mention the interface of private network : ")

mkdir -p /root/inputs

touch /root/inputs/private_interface

echo "$private_interface" > /root/inputs/private_interface



#Taking inputs Hostname, IP, Domain name


hostname_master=$(zenity --entry --title="Hostname required" --text="Provide a hostname to your master.")

touch /root/inputs/hostname_master

echo "$hostname_master" > /root/inputs/hostname_master

#read -p "Provide a hostname to your master : " hostname_master



privateip_network=$(ip a show dev $private_interface | grep -oP 'inet \K[\d.]+')

echo "nameserver $privateip_network" >> /etc/resolv.conf    #this command not running in bash

touch /root/inputs/private_ip

echo "$privateip_network" > /root/inputs/private_ip


zenity \

--info \
```

*--text="current ip is $privateip_network.\n DO NOT CHANGE THE IP" *

*--title="xCAT setup" *

--ok-label="OK"

domain_name=$(zenity --entry --title="Domain name required" --text="Please provide a domain name.")

touch /root/inputs/domain_name

echo "$domain_name" > /root/inputs/domain_name

#read -p "Please provide a domain name --Example 'avicii.in' -- : " domain_name

hostnamectl set-hostname $hostname_master.$domain_name

2. xCAT Installation and Configuration:

- Install xCAT on the master node.

#Updating yum and Utilities

yum install zenity -y

yum install xterm -y

yum install ntp -y

yum update -y

yum install yum-utils -y

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

#wget -P /etc/yum.repos.d https://xcat.org/files/xcat/repos/yum/xcat-dep/rh7/x86_64/xcat-dep.repo --no-check-certificate

yum install epel-release -y

yum clean all

yum makecache

#Making xcat directory which will contain xcat-core and xcat-dep

mkdir -p ~/xcat

cd ~/xcat/

#downloading xCAT

wget https://raw.githubusercontent.com/xcat2/xcat-core/master/xCAT-server/share/xcat/tools/go-xcat -O - >/tmp/go-xcat

chmod +x /tmp/go-xcat

#This will install xcat

/tmp/go-xcat install -y

wget https://xcat.org/files/xcat/xcat-core/2.16.x_Linux/xcat-core/xcat-core-2.16.5-linux.tar.bz2

wget https://xcat.org/files/xcat/xcat-dep/2.x_Ubuntu/?C=M;O=D

wget https://xcat.org/files/xcat/xcat-dep/2.x_Linux/xcat-dep-202212061505.tar.bz2

#Extracting xcat-core

tar xvf xcat-core-2.16.5-linux.tar.bz2

tar jxvf xcat-dep-202212061505.tar.bz2

cd ~/xcat/xcat-core

./mklocalrepo.sh

 # Inside xcat-dep

cd ~/xcat/xcat-dep/

cd rh7

cd ppc64le/

./mklocalrepo.sh

#yum install xCAT -y

#chmod 777 /etc/profile.d/xcat.sh

#SOURCE

source /etc/profile.d/xcat.sh

- Configure xCAT's database, DHCP, and TFTP services for provisioning.

```
domain_name=`cat /root/inputs/domain_name`

hostname_master=`cat /root/inputs/hostname_master`

private_interface=`cat /root/inputs/private_interface`

private_ip=`cat /root/inputs/private_ip`


chdef -t site domain=$domain_name

dhcpinterfaces="chdef -t site dhcpinterfaces='$hostname_master|$private_interface'"

eval "$dhcpinterfaces"

worker_password=$(zenity --entry --title="Password" --text="Provide a common password for all worker nodes : ")

#read -p "Provide Password for worker node : " worker_password

passwd="chtab key=system passwd.username=root passwd.password=$worker_password"

eval "$passwd"

#chtab key=system passwd.username=root passwd.password=$worker_password

makedhcp -n

makedns -n

makentp

systemctl restart dhcpd
```

3. Node configuration

CODE:-

```
get http://mirrors.nxtgen.com/centos-mirror/7.9.2009/isos/x86_64/CentOS-7-x86_64-DVD-2009.iso

copycds CentOS-7-x86_64-DVD-2009.iso

genimage centos7.9-x86_64-netboot-compute

touch /install/netboot/compute.synclists

echo "/etc/passwd -> /etc/passwd" >> /install/netboot/compute.synclist

echo "/etc/group -> /etc/group" >> /install/netboot/compute.synclist

echo "/etc/shadow -> /etc/shadow" >> /install/netboot/compute.synclist
```

path_of_computelist="\"/install/netboot/compute.synclists\""

compute_list="chdef -t osimage centos7.9-x86_64-netboot-compute synclists=$path_of_computelist"

eval "$compute_list"

chdef -t osimage centos7.9-x86_64-netboot-compute synclists="/install/netboot/compute.synclists" #This does not run in bash, run this command manually on terminal

#packimage centos7.9-x86_64-netboot-compute

name_of_node=$(zenity --entry --title="Node name required" --text="Please provide a name to your node : ")

#read -p "Please provide a name to your node : " name_of_node

ip_of_node=$(zenity --entry --title="ip required" --text="Please provide the ip which you want it to be assigned to $name_of_node1 : ")

#read -p "Please provide the ip which you want it to be assigned to $name_of_node1 : " ip_of_node

mac_of_node=$(zenity --entry --title="ip required" --text="Please mention the mac address of $name_of_node in colon-hexadecimal notation : ")

#read -p "Please mention the mac address in of $name_of_node1 in colon-hexadecimal notation : " mac_of_node

mkdef -t node -o $name_of_node groups=all netboot=pxe ip=$ip_of_node mac=$mac_of_node provmethod=centos7.9-x86_64-netboot-compute

#echo "$ip_of_node $name_of_node $name_of_node.$domain_name" >> /etc/hosts

#echo "nameserver $master_privateip" >> /etc/resolv.conf

makehosts

makenetworks

makedhcp -n

makedns -n

makentp

systemctl restart dhcpd

nodeset $name_of_node osimage=centos7.9-x86_64-netboot-compute

whiptail --msgbox " $name_of_node is ready, please start the machine . Press Enter to continue." 10 50

LDAP Integration:

- Install and configure the latest version of LDAP on the master node.
- Integrate xCAT with LDAP for centralized user management, enabling users to authenticate across the cluster.

CODE:-

```
yum -y install openldap-servers openldap-clients
cp /usr/share/openldap-servers/DB_CONFIG.example /var/lib/ldap/DB_CONFIG
 chown ldap. /var/lib/ldap/DB_CONFIG
 systemctl start slapd
 systemctl enable slapd
 slappasswd
 vim  chrootpw.ldif
                dn: olcDatabase={0}config,cn=config
                changetype: modify
                add: olcRootPW
                olcRootPW: {SSHA}ZG80X2fsWssEhBMBaAa6zegqKCXtOwVq
~
 ldapadd -Y EXTERNAL -H ldapi:/// -f chrootpw.ldif
 ldapadd -Y EXTERNAL -H ldapi:/// -f /etc/openldap/schema/cosine.ldif
 ldapadd -Y EXTERNAL -H ldapi:/// -f /etc/openldap/schema/nis.ldif
 ldapadd -Y EXTERNAL -H ldapi:/// -f /etc/openldap/schema/inetorgperson.ldif
 vim 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=hpcsa,dc=in" read by * none

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

                dn: olcDatabase={2}hdb,cn=config
                changetype: modify
                replace: olcRootDN
                olcRootDN: cn=Manager,dc=hpcsa,dc=in

                dn: olcDatabase={2}hdb,cn=config
                changetype: modify
                add: olcRootPW
                olcRootPW: {SSHA}HlLk9Qviw3rfsF9UNGOGaNf++0XL/Xun

                dn: olcDatabase={2}hdb,cn=config
                changetype: modify
                add: olcAccess
                olcAccess: {0}to attrs=userPassword,shadowLastChange by
                  dn="cn=Manager,dc=hpcsa,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=hpcsa,dc=in" write by * read

                  142  ldapmodify -Y EXTERNAL -H ldapi:/// -f chdomain.ldif
```

```
  143  vim basedomain.ldif
dn: dc=hpcsa,dc=in
objectClass: top
objectClass: dcObject
objectclass: organization
o: hpcsa in
dc: hpcsa

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

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

dn: ou=Group,dc=hpcsa,dc=in
objectClass: organizationalUnit
ou: Group
ldapadd -x -D cn=Manager,dc=hpcsa,dc=in -W -f basedomain.ldif
vim ldapuser.ldif
dn: uid=hpcsa1,ou=People,dc=hpcsa,dc=in
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
cn: hpcsa1
sn: Linux
userPassword: {SSHA}ZG80X2fsWssEhBMBaAa6zegqKCXtOwVq
loginShell: /bin/bash
uidNumber: 1003
gidNumber: 1003
homeDirectory: /home/hpcsa1

dn: cn=hpcsa1,ou=Group,dc=hpcsa,dc=in
objectClass: posixGroup
cn: hpcsa1
gidNumber: 1003
memberUid: hpcsa1

ldapadd -x -D cn=Manager,dc=hpcsa,dc=in -W -f ldapuser.ldif
systemctl start slapd
systemctl status slapd
yum --installroot=/install/netboot/centos7.9/x86_64/compute/rootimg/ install openldap-clients
exports CHROOT=/install/netboot/centos7.9/x86_64/compute/rootimg
export CHROOT=/install/netboot/centos7.9/x86_64/compute/rootimg
echo $CHROOT
chroot $CHROOT
```

Slurm Resource Management using Ansible:

Ansible inventory

```
[all]

localhost ansible_host=192.168.10.199

n4 ansible_host=192.168.10.203

[nodes]

n4 ansible_host=192.168.10.203
```

Master Playbook

```
############### Only Master ################################
- name: Configure SLURM Control Node
  hosts: localhost
  become: true
  tasks:

    - name: Download SLURM tarball
      get_url:
        url: https://download.schedmd.com/slurm/slurm-20.11.9.tar.bz2
        dest: /root/slurm-20.11.9.tar.bz2  # Specify the destination path

    - name: extract SLURM tar file
      command: rpmbuild -ta /root/slurm-20.11.9.tar.bz2

    - name: Create /home/rpms directory
      file:
        path: /home/rpms
        state: directory

    - name: Copy RPMs to /home/rpms
      synchronize:
        src: /root/rpmbuild/RPMS/x86_64/
        dest: /home/rpms/

    - name: List directory content
      find:
        paths: /home/rpms/
        patterns: "*.rpm"
      register: rpm_file_result


    - name: Install RPM file
      yum:
        name: "{{ rpm_file_result.files[0].path }}"
```

```
      state: present
    when: rpm_file_result.matched > 0


  - name: Generate MUNGE key
    command: /usr/sbin/create-munge-key > /etc/munge/munge.key
    changed_when: false


  - name: Copy slurm.conf.example to /etc/slurm/slurm.conf
    copy:
      src: /etc/slurm/slurm.conf.example
      dest: /etc/slurm/slurm.conf


  - name: Create slurmctld.log
    file:
      path: /var/log/slurm/slurmctld.log
      state: touch

  - name: Change ownership of slurmctld.log
    file:
      path: /var/log/slurm/slurmctld.log
      owner: slurm
      group: slurm

  - name: Create slurm_jobacct.log and slurm_jobcomp.log
    file:
      path: /var/log/slurm_jobacct.log
      state: touch

  - name: Change ownership of slurm_jobacct.log and slurm_jobcomp.log
    file:
      path: /var/log/slurm_jobacct.log
      owner: slurm
      group: slurm

  - name: Change ownership of slurm_jobcomp.log
    file:
      path: /var/log/slurm_jobcomp.log
      owner: slurm


  - name: Edit slurm.conf
    lineinfile:
      path: /etc/slurm/slurm.conf
      regexp: '^ControlMachine='
      line: 'ControlMachine=master'

- name: Run slurmd -C command
  hosts: nodes
  become: true
  tasks:

    - name: Execute Another Playbook on Different Host
      run_once: true
      command: ansible-playbook ./hosts node_info.yml
```

```
      register: node_info_output

- name: Run slurmd -C command
  hosts: localhost
  become: true
    tasks:
      - name: Save Playbook Output to File
        deate_to: localhost
        local_action:
          module: copy
          content: "{{ node_info_output.stdout }}"
          dest: ./node_info.txt

      - name: Remove last 2 lines from file
        ansible.builtin.lineinfile:
          path: /path/to/your/file.txt
          regexp: '^.*$'
          backrefs: yes
          state: absent
          before: -2

      - name: Append Playbook Output to File
        shell: "echo '{{ node_info_output.stdout }}' >> /etc/slurm/slurm.conf"

      - name: Get last line of the file
        ansible.builtin.shell: tail -n 1 /path/to/your/file.txt
        register: last_line

      - name: Edit last line in the file
        ansible.builtin.shell: echo "{{ last_line.stdout | regex_replace('old_pattern', 'new_content') }}" > /tmp/temp_file &&
mv /tmp/temp_file /etc/slurm.slurm.conf
```

All machines Playbook(to be configured on all devices)

```
############ All Machine ###############
---
- name: Configure SLURM Control Node
  hosts: all
  become: true
  tasks:
        # All Machine
    - name: Install dependency packages
      yum:
       name: "{{ item }}"
       state: present
      loop:
        - epel-release
        - munge
        - munge-libs
        - munge-devel
        - rpm-build
        - python3
        - readline-devel
```

```
      - perl-ExtUtils-MakeMaker
      - pam-devel
      - gcc
      - mysql-devel

  - name: Set SLURMUSER Environment Variable
    set_fact:
      SLURMUSER: 900

  - name: Create SLURM Group
    command: groupadd -g {{ SLURMUSER }} slurm

  - name: Create SLURM User
    command: useradd -m -c "SLURM workload manager" -d /var/lib/slurm -u {{ SLURMUSER }} -g slurm -s /bin/bash
slurm
    environment:
      SLURMUSER: 900

  - name: Create /var/spool/slurm directory and set ownership/permissions
    block:
      - file:
          path: /var/spool/slurm
          state: directory
      - file:
          path: /var/spool/slurm
          owner: slurm
          group: slurm
      - file:
          path: /var/spool/slurm
          mode: '0755'

  - name: Create /var/log/slurm directory and set ownership/permissions
    block:
      - file:
          path: /var/log/slurm
          state: directory
      - file:
          path: /var/log/slurm
          owner: slurm
          group: slurm
      - file:
          path: /var/log/slurm
          mode: '0755'
      - command: chown -R slurm:slurm /var/log/slurm

  - name: Copy MUNGE key to remote nodes
    copy:
      src: "/etc/munge/munge.key"
      dest: "/etc/munge/munge.key"
      mode: 0400

  - name: Start and enable MUNGE service
    systemd:
      name: munge
      state: started
      enabled: yes
```

Nagios Installation and Integration:

Ansible inventory

```
[all]
localhost ansible_host=192.168.10.199
n4 ansible_host=192.168.10.203


[nodes]
n4 ansible_host=192.168.10.203
```

Master Playbook

```
---
- name: Install OpenHPC repository and httpd
  hosts: your_host_group
  become: yes  # This allows Ansible to execute commands with root privileges (sudo)
  varsi_files:
    - nagios_username: root
    - nagios_password: my_app

  tasks:
    - name: Install OpenHPC repository
      yum:
        name: http://build.openhpc.community/OpenHPC:/1.3/CentOS_7/x86_64/ohpc-release-1.3-1.el7.x86_64.rpm
        state: present

    - name: Install dependency
      yum:
         name: "{{ item }}"
       state: present
      loop:
        - httpd
        - yum-utils

   - name: Download xCAT repository file
     get_url:
       url: https://xcat.org/files/xcat/repos/yum/latest/xcat-core/xcat-core.repo
       dest: /etc/yum.repos.d/xcat-core.repo

   - name: Install ohpc-nagios package
     yum:
       name: ohpc-nagios
```

```
  state: present

- name: Start Nagios service
  systemd:
    name: nagios
    state: started

- name: Enable Nagios service at boot
  systemd:
    name: nagios
    enabled: yes

    - name: Start Nagios service
  systemd:
    name: httpd
    state: started

- name: Enable Nagios service at boot
  systemd:
    name: httpd
    enabled: yes


- name: Create Nagios user
  htpasswd:
    path: /etc/nagios/passwd
    name: admin
    password: admin
    mode: add

- name: Remove lines
  lineinfile:
    path: /etc/nagios/conf.d/hosts.cfg
    regexp: '.*'  # Match all lines
    state: absent
    backrefs: yes
    validate: 'cat %s'

 - name: Replace line
   replace:
    path: /etc/nagios/conf.d/hosts.cfg
    regexp: 'members\ HOSTNAME1\,HOSTNAME2\,HOSTNAME3\,HOSTNAME4'
    replace: 'members\ client'


 - name: Replace line
   replace:
    path: /etc/nagios/conf.d/hosts.cfg
    regexp: 'host_name\ HOSTNAME1'
    replace: 'host_name\ client'

 - name: Get IP address
   shell: cat /etc/hosts | grep client | awk '{ print $1 }'
   register: ipp_output

     - name: Replace line with IP address
  lineinfile:
```

```
    path: /etc/nagios/conf.d/hosts.cfg
    regexp: 'address\ HOST1_IP'
    line: 'address\ {{ ipp_output.stdout }}'

- name: Change ownership
  file:
    path: /etc/nagios/conf.d/hosts.cfg
    owner: nagios
    group: nagios

- name: Copy file
  copy:
    src: /path/to/services.cfg.example
    dest: /etc/nagios/conf.d/services.cfg

- name: Change ownership
  file:
    path: /etc/nagios/conf.d/services.cfg
    owner: nagios
    group: nagios

- name: Restart Nagios service
  systemd:
    name: nagios
    state: restarted
```

Configure Nagios worker nodes, and critical services.

```
---
- name: Start and Enable NRPE service
  hosts: nodes
  become: yes  # This allows Ansible to execute commands with root privileges (sudo)
  tasks:
    - name: Start NRPE service
      systemd:
        name: nrpe
        state: started

    - name: Enable NRPE service at boot
      systemd:
        name: nrpe
        enabled: yes
```

Expected Outcomes:

- *A fully functional HPC cluster with one master node and two worker nodes.*
- *Centralized user authentication and authorization through LDAP.*
- *Resource allocation and job scheduling managed by Slurm.*
- *Real-time monitoring of cluster components and services using Nagios.*

User Guide

Basic Commands

xCAT (Extreme Cloud Administration Toolkit) provides a set of commands to help with the provisioning and management of computing clusters. These commands are designed to simplify tasks like deploying operating systems, managing nodes, configuring hardware, and more. Here are some key xCAT provisioning commands along with explanations for each:

1. nodeset:
   - This command is used to set properties on nodes or groups of nodes.
   - Example: `nodeset compute[01-10] netboot=xnba`

2. rpower:
   - Used to remotely power on, power off, reboot, or reset nodes.
   - Example: `rpower compute01 on`
3. nodesetstate:
   - This command sets the provisioning state of nodes.
   - Example: `nodesetstate compute[01-10] osimage`
4. chtab:
   - Allows you to modify the properties in the xCAT database.
   - Example: `chtab -t node -o compute01 installnic=eth0`
5. makeconservercf:
   - Generates a configuration file for the `conserver` tool, used for console access.
   - Example: `makeconservercf compute[01-10] > /etc/conserver.cf`
6. lsdef:
   - Displays the definitions of nodes, groups, or other objects in the xCAT database.
   - Example: `lsdef compute01`
7. mkdef:
   - Creates or modifies the definitions of nodes, groups, or other objects in the xCAT database.
   - Example: `mkdef -t node compute01 mac=00:11:22:33:44:55`
8. tabrestore:
   - Restores the xCAT database from a backup file.
   - Example: `tabrestore /tmp/xcatbackup`
9. nodesta:
   - Provides information about the status of nodes.
   - Example: `nodestat compute[01-10]`
10. genimage:
    - Generates an OS image for network booting.

   - Example: `genimage rhel7.9-x86_64-netboot`
11. makedhcp:
   - Generates the DHCP configuration files for network booting.
   - Example: `makedhcp -n`
12. makedns:
   - Generates the DNS configuration files for xCAT.
   - Example: `makedns -n`
13. lsimage:
   - Lists the available OS images managed by xCAT.
   - Example: `lsimage`
14. rinstall:
   - Initiates the installation of the operating system on nodes.
   - Example: `rinstall compute[01-10]`

These are just a few examples of xCAT provisioning commands. xCAT provides a comprehensive set of tools for managing various aspects of cluster deployment, configuration, and maintenance. When using these commands, it's essential to refer to the xCAT documentation for a more detailed understanding of their usage, options, and potential variations based on your cluster's specific configuration.

Troubleshooting

Common Issues and Solutions:-

- Login issue on worker node
    - resolved by using chtab command for user and password generation

- Unable to boot node on bare metal
    - we were using centos 8 on master and was trying to boot centos 7.9
    - then to resolve this we generated image by centos 8 and resolved the issue

conclusion of troubleshooting:-

*1. **User and Password Generation:**
The `chtab` command can be useful for managing user and password information in xCAT. This command allows you to set or modify various attributes for nodes, including authentication credentials. Generating and managing user accounts and passwords appropriately is crucial for securing your cluster.*

*2. **Operating System Compatibility:**
The issue with booting a CentOS 7.9 node on a CentOS 8 master highlights the importance of operating system compatibility. xCAT's image generation functionality can be very helpful in creating custom images that are compatible with your specific environment.*

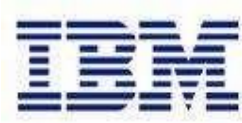
*3. **Image Generation:**
Generating a custom image based on CentOS 8 to resolve the booting issue was a practical solution. Creating custom images tailored to your specific requirements can ensure that your cluster runs smoothly and efficiently.*

*4. **Operating System Version Matching:**
Ensuring that the operating system versions of your master and worker nodes match can help prevent compatibility issues. While it's possible to manage nodes with different OS versions, it may require additional configuration and considerations.*

*5. **Documentation and Support:**
Keeping track of the steps you took to resolve these issues can be valuable for future reference. It's also a good idea to consult official documentation, user forums, or support channels when encountering complex or unique problems.*

Community and Contributions

Who's using xCAT:-



- *xCAT is an open-source tool for automating deployment, scaling, and management of bare metal servers and virtual machines developed by IBM*

- *xCAT is an official IBM high Performance Computing cluster management tool.*

Leibniz Supercomputing Centre

- *>9000 and >3000 compute nodes are managed by xCAT in SuperMUC super computer SuperMUC is the high-end supercomputer at the Leibniz Supercomputing Centre with more than 241,000 cores and a combined peak performance of the two installation phases of more than 6.8 PetaFlop/s.*
- *>6000 compute nodes are managed by xCAT in SuperMUC-NG super computer SuperMUC-NG has 311,040 compute cores in total with a main memory of 719 TB and a peak performance of 26.9 PetaFlop/*

Stanford Research Computing Center

- *The Stanford Research Computing Center (SRCC) is a joint effort of the Dean of Research and IT Services to build and support a comprehensive program to advance computational research at Stanford. That includes offering and supporting traditional high-performance computing (HPC) systems, as well as systems for high throughput and data-intensive computing. The SRCC also helps researchers transition their analyses and models from the desktop to more capable and plentiful resources, providing the opportunity to explore their data and answer research questions (on-premise or in the cloud) at a scale typically not possible on desktops or departmental servers.*
- *We use xCAT to deploy and maintain the Sherlock cluster, Stanford's shared HPC cluster that features over 25,000 CPU cores and supports over 4,000 users.*

Versatus HPC

- *100% focused on High Performance Computing, Versatus HPC is a Brazilian company focused on delivering the best solutions for HPC scenarios in the academia, industries and businesses. With 10 years of experience we allow the researchers to focus exclusively on R&D pursuing its scientific objectives.*
- *We are supporting / installing xCAT in lot of Universities and Companies in Brazil. We always defaults to xCAT as the main provisioner and we try at best to convince our customers to move from whatever provisioning solution, including proprietary ones, to xCAT*

RedLine

- *RedLine specializes in High Performance Computing (HPC) software and systems engineering, development, and support, from planning, designing, and implementation, to porting and performance tuning*
- *We have been using xCAT since early 2011. We have deployed numerous clusters on many different hardware platforms*

MEGWARE

Since its foundation in 1990, the Chemnitz (Germany) based company has been able to draw on extensive experience in the development and installation of HPC systems and Linux clusters. MEGWARE, as one of Europe's leading supercomputing specialists, has delivered and installed approximately 1,500 high-performance computers to research institutes, universities, and industry customers throughout Europe.

MEGWARE has been using xCAT since more than 10 years. xCAT is the favourite tool when it comes to provisioning HPC and AI clusters.

Glossary

Sure, here's a list of terms and acronyms related to High-Performance Computing (HPC) and xCAT (Extreme Cluster/Cloud Administration Toolkit):

HPC Terms and Acronyms:

1. HPC (High-Performance Computing): A field of computing that deals with the use of supercomputers and high-performance clusters to solve complex scientific, engineering, and data-intensive problems.

2. Cluster: A group of interconnected computers or servers that work together as a single system to perform computations.

3. Supercomputer: A highly powerful and advanced computer designed to perform extremely complex calculations at high speeds.

4. MPI (Message Passing Interface): A standardized and portable message-passing system that allows multiple processes to communicate with each other in parallel computing environments.

5. CUDA (Compute Unified Device Architecture): A parallel computing platform and programming model developed by NVIDIA for utilizing GPUs for general-purpose processing.

6. GPU (Graphics Processing Unit): A specialized processor designed for accelerating graphics rendering, but also used for general-purpose parallel computing tasks in HPC.

7. CPU (Central Processing Unit): The primary processing unit of a computer that executes instructions of a program.

8. InfiniBand: A high-speed interconnect technology used in HPC clusters for low-latency and high-bandwidth communication between nodes.

9. NUMA (Non-Uniform Memory Access): A memory architecture design where memory access times can vary depending on the location of data relative to the processor.

10. Parallel Computing: The use of multiple processors or cores to perform computations simultaneously, aimed at improving processing speed.

11. Petascale: Refers to computing systems capable of performing at least one quadrillion (10^15) calculations per second.

12. Exascale: Refers to computing systems capable of performing at least one quintillion (10^18) calculations per second.

13. HPC Middleware: Software layers that sit between the operating system and applications, providing services like job scheduling, resource management, and communication.

14. Job Scheduler: Software that manages the distribution of computational tasks across a cluster, ensuring efficient resource utilization.

xCAT Terms and Acronyms:

1. xCAT (Extreme Cluster/Cloud Administration Toolkit): An open-source software tool designed for the installation, deployment, and management of HPC clusters and cloud environments.

2. Node: A single computing entity within a cluster that can be a physical machine or a virtual instance.

3. BMC (Baseboard Management Controller): A hardware component that provides out-of-band management capabilities for remote monitoring and control of a server.

4. PXE (Preboot Execution Environment): A network protocol that allows a computer to boot from a network interface before loading the operating system from local storage.

5. OS Image: A complete snapshot of an operating system, including its software, configuration, and settings, used for deploying identical environments across cluster nodes.

6. Kickstart: A network installation system used by xCAT to automate the installation of operating systems on multiple nodes.

7. Nodeset: A group of nodes that share a similar configuration or purpose within the cluster.

8. xCAT Server: The central management node in an xCAT environment responsible for controlling and managing the cluster nodes.

9. SNMP (Simple Network Management Protocol): A protocol used for managing and monitoring network devices, often utilized for server hardware monitoring.

10. LDAP (Lightweight Directory Access Protocol): A protocol for accessing and maintaining distributed directory information services, often used for centralized user authentication and management.

11. HPCaaS (High-Performance Computing as a Service): A cloud computing model that provides HPC resources to users on-demand.

12. DB2: A relational database management system often used as the backend for xCAT's data management.

References

External Documentation

Official xCAT Project Website: The official xCAT project website typically hosts documentation, user guides, installation instructions, and other resources.

Website: https://xcat.org/
xCAT GitHub Repository: The xCAT source code and documentation are often available on GitHub. You can find the repository for xCAT and its documentation there.

GitHub Repository: https://github.com/xcat2/xcat-core
IBM Knowledge Center: IBM often provides documentation for xCAT through its Knowledge Center, offering detailed guides and information on various aspects of xCAT.

IBM xCAT Knowledge Center:
https://www.ibm.com/support/knowledgecenter/SSPHQG_2.1.0/
User Forums and Mailing Lists: xCAT users often share their experiences and solutions on user forums and mailing lists. This can be a valuable resource for troubleshooting and learning from others.

xCAT User Mailing List: https://lists.sourceforge.net/lists/listinfo/xcat-user
Documentation in the xCAT Source Code: The xCAT source code repository on GitHub might include a "docs" or "doc" directory with various documentation files and guides.

Online Communities and Forums: Besides official sources, you can find discussions, tutorials, and resources related to xCAT on various online communities and forums dedicated to HPC and cluster management.

Related Projects and Tools

xCAT (Extreme Cluster/Cloud Administration Toolkit) is a versatile tool used for managing and provisioning HPC clusters and cloud environments. If you're interested in projects related to xCAT scripting for HPC provisioning, you might want to explore the following areas and initiatives:

1. xCAT Community Projects: The xCAT community often contributes various scripts, plugins, and tools that extend xCAT's functionality. These projects can range from custom provisioning scripts to integration with other tools and technologies.

. Cluster Deployment and Management: Many HPC administrators and researchers create custom scripts to automate the deployment and management of their cluster environments using xCAT. These scripts might include automated node provisioning, OS installations, configuration management, and software deployment.

3. Monitoring and Health Checking: Developing scripts that utilize xCAT to monitor the health and performance of cluster nodes, including hardware components, can be valuable for maintaining a stable HPC environment.

4. Integration with Configuration Management Tools: Integrate xCAT with popular configuration management tools like Ansible, Puppet, or Chef to ensure consistent configurations across your HPC cluster.

5. GPU Management and Scheduling: If your cluster uses GPUs for parallel processing, scripting the management and scheduling of GPU resources with xCAT can be a useful project.

6. Power Management: Implement scripts that leverage xCAT to control and manage power consumption of cluster nodes, including power cycling and management of power states.

7. User Account and Authentication Management: Automating user account provisioning and authentication configuration across the cluster using xCAT can simplify user management tasks.

8. Data Management and Storage: Develop scripts to configure and manage storage resources within the cluster, ensuring efficient data management and access.

9. Cloud Integration: If your HPC environment spans both on-premises and cloud resources, scripting xCAT for hybrid cloud provisioning and management could be a project of interest.

10. Resource Allocation and Scheduling: Design scripts that interact with xCAT to optimize resource allocation and job scheduling within the cluster.

11. Documentation and Training Materials: Creating scripts to automate the generation of documentation and training materials for cluster users can help ensure that users have accurate information about cluster usage.

12. Backup and Recovery: Developing scripts that automate backup and recovery processes using xCAT can be crucial for maintaining data integrity and disaster recovery capabilities.

Conclusion

This project will result in the successful deployment of an HPC cluster using xCAT, LDAP, Slurm, and Nagios. The integrated technologies will collectively enhance the cluster's efficiency, security, and manageability, making it a powerful platform for computational workloads. This project demonstrates the comprehensive capabilities of xCAT and the seamless integration of various tools to create a robust and highly functional cluster environment.

References

https://www.liquidweb.com/kb/how-to-install-and-configure-ansible/ ----------for installation

https://docs.ansible.com/ansible/latest/playbook_guide/index.html ---------- for scripting

https://xcat-docs.readthedocs.io/en/stable/----------xCAT documentation

https://linuxhint.com/install-configure-linux-ldap/---------- ldap installation

https://installati.one/centos/8/slurm/----------slurm installation and config