**HOW TO SETUP SLURM - TWO NODES CLUSTER**

**Set the environment**

1. Install python3

sudo apt update

sudo apt install python3

1. Install pip

sudo apt install python3-pip

1. Install openstack client

sudo pip3 install python-openstackclient

1. Enable chameleon feature

sudo pip3 install git+https://github.com/chameleoncloud/python-blazarclient@chameleoncloud/xena

1. Do CLI Authentication by following the instructions from this [page](https://chameleoncloud.readthedocs.io/en/latest/technical/cli.html#cli-authentication).

*Do not set up Two-FA.*

1. Get and use OpenstackRC script by following the instructions from this [page](https://chameleoncloud.readthedocs.io/en/latest/technical/cli.html#the-openstack-rc-script)**.**

*In general, we use TACC instead of CHI.*

*If you see the following error in openstack:*

*“Unrecognized schema …” → you didn’t supply the correct password*

*“ Missing value …” → you haven’t sourced the OpenstackRC*

1. Set the environment variables

export MASTER\_LEASE\_NAME=<set a preferable string for your master lease name>

export INSTANCE\_LEASE\_NAME=<set a preferable string for your slave lease name>

export END\_DATE=<set a preferable date by using this format "YYYY-MM-DD HH:MM">

export KEY\_PAIR\_NAME=<set a preferable string for your key pair name>

export PRIVATE\_KEY\_NAME=<set a preferable string for your private key name>

export MASTER\_INSTANCE\_NAME=<set a preferable string for your master instance name>

export SLAVE\_INSTANCE\_NAME=<set a preferable string for your slave instance name>

For example:

export MASTER\_LEASE\_NAME=sbf-master-lease

export SLAVE\_LEASE\_NAME=sbf-slave-lease

export END\_DATE="2023-05-28 23:59"

export KEY\_PAIR\_NAME=test-key

export PRIVATE\_KEY\_NAME=test-private-key

export MASTER\_INSTANCE\_NAME=sbf-master

export SLAVE\_INSTANCE\_NAME=sbf-slave

1. Make them permanent

echo -e "export MASTER\_LEASE\_NAME=$MASTER\_LEASE\_NAME\nexport SLAVE\_LEASE\_NAME=$SLAVE\_LEASE\_NAME\nexport END\_DATE='$END\_DATE'\nexport KEY\_PAIR\_NAME=$KEY\_PAIR\_NAME\nexport PRIVATE\_KEY\_NAME=$PRIVATE\_KEY\_NAME\nexport MASTER\_INSTANCE\_NAME=$MASTER\_INSTANCE\_NAME\nexport SLAVE\_INSTANCE\_NAME=$SLAVE\_INSTANCE\_NAME" >> ~/.bashrc && source ~/.bashrc

**Create Instances in Chameleon**

1. Create a reservation for master instance.

openstack reservation lease create \

--reservation min=1,max=1,resource\_type=physical:host,resource\_properties='["=", "$node\_type", "compute\_haswell\_ib"]' \

--end-date "$END\_DATE" \

$MASTER\_LEASE\_NAME

1. Create a reservation for slave instance

openstack reservation lease create \

--reservation min=1,max=1,resource\_type=physical:host,resource\_properties='["=", "$node\_type", "compute\_haswell\_ib"]' \

--end-date "$END\_DATE" \

$SLAVE\_LEASE\_NAME

1. Store the master’s reservation id to an environment variable.

export MASTER\_RESERVATION\_ID=$(openstack reservation lease show $MASTER\_LEASE\_NAME | grep \"id\": | tail -1 | (read h; echo ${h:28:36}))

1. Store the slave’s reservation id to an environment variable

export SLAVE\_RESERVATION\_ID=$(openstack reservation lease show $SLAVE\_LEASE\_NAME | grep \"id\": | tail -1 | (read h; echo ${h:28:36}))

1. Create a key pair for accessing the instances using ssh.

openstack keypair create $KEY\_PAIR\_NAME --private-key $PRIVATE\_KEY\_NAME

1. Store current directory path to an environment variable as a private key’s path.

echo "export KEY\_LOCATION=$(pwd)" >> ~/.bashrc && source ~/.bashrc

1. Change permission of key pair

chmod 600 $PRIVATE\_KEY\_NAME

1. Create master instance

openstack server create \

--image ecb2ffc4-90a8-4c8b-b9ed-a141392d8359 \

--flavor baremetal \

--key-name $KEY\_PAIR\_NAME \

--nic net-id=1a03cf65-8fd6-4fce-94fd-bbaabe68a8e1 \

--hint reservation=$MASTER\_RESERVATION\_ID \

$MASTER\_INSTANCE\_NAME

1. Create slave instance

openstack server create \

--image ecb2ffc4-90a8-4c8b-b9ed-a141392d8359 \

--flavor baremetal \

--key-name $KEY\_PAIR\_NAME \

--nic net-id=1a03cf65-8fd6-4fce-94fd-bbaabe68a8e1 \

--hint reservation=$SLAVE\_RESERVATION\_ID \

$SLAVE\_INSTANCE\_NAME

1. Create floating IP for master instance and store it to an environment variable

export MASTER\_FLOATING\_IP=$(openstack floating ip create public | grep floating\_ip\_address | (read h; echo ${h:24:15}))

1. Create floating IP for slave instance and store it to an environment variable

export SLAVE\_FLOATING\_IP=$(openstack floating ip create public | grep floating\_ip\_address | (read h; echo ${h:24:15}))

1. Set the $MASTER\_FLOATING\_IP and $SLAVE\_FLOATING\_IP as permanent environment variables

echo -e "export MASTER\_FLOATING\_IP=$MASTER\_FLOATING\_IP\nexport SLAVE\_FLOATING\_IP=$SLAVE\_FLOATING\_IP" >> ~/.bashrc && source ~/.bashrc

1. Associate floating IP to master instance

openstack server add floating ip $MASTER\_INSTANCE\_NAME $MASTER\_FLOATING\_IP

1. Associate floating IP to slave instance

openstack server add floating ip $SLAVE\_INSTANCE\_NAME $SLAVE\_FLOATING\_IP

**SSH to the remote instances from local computer**

**In the following sections you need to access master instance and slave instance. For now, you can skip this section but if you encounter instruction to do ssh from local computer, you can follow these instructions.**

1. In your local computer, go to the directory where the private key is located.

cd $KEY\_LOCATION

1. Do SSH to master instance or remote instance

**Master**

ssh -i $PRIVATE\_KEY\_NAME cc@$MASTER\_FLOATING\_IP

**Slave**

ssh -i $PRIVATE\_KEY\_NAME cc@$SLAVE\_FLOATING\_IP

**Setup Slurm in master instance**

1. Do ssh from your local computer to master instance
2. Install slurm

wget https://www.ni-sp.com/wp-content/uploads/2019/10/SLURM\_Ubuntu\_installation.sh

export VER=20.11.8 # latest 20.11

export VER=21.08.6

export VER=20.11.9

export VER=22.05.9

bash SLURM\_Ubuntu\_installation.sh

1. Edit compute nodes specifications in /etc/slurm/slurm.conf according to slave instance specification.

NodeName=ubuntu NodeAddr=SLAVE\_IP CPUs=SLAVE\_CPUS

SLAVE\_IP will be replaced by slave’s fixed ip address and SLAVE\_CPUS will be replaced by number of slave’s CPUs. You can get these information by following the instructions below:

1. Do ssh from your local computer to slave instance.
2. On the slave instance, install net-tools

sudo apt install net-tools

1. Check slave’s fixed ip address by running

ifconfig

1. Check number of slave’s CPUs using

lscpu

1. Restart slurmctld

sudo systemctl daemon-reload

If you want to run slurmctld on the background use this command

sudo systemctl restart slurmctld

But if you want to run slurmctld in debugging mode use this command

sudo /usr/sbin/slurmctld -D -vvv

**Setup Slurm in slave instance**

1. Do ssh from your local computer to slave instance
2. Install slurm

wget https://www.ni-sp.com/wp-content/uploads/2019/10/SLURM\_Ubuntu\_installation.sh

export VER=20.11.8 # latest 20.11

export VER=21.08.6

export VER=20.11.9

export VER=22.05.9

bash SLURM\_Ubuntu\_installation.sh

1. Copy /etc/munge/munge.key from the master instance to /etc/munge/ on the slave instance
2. Enable ssh connection between master and slave

To enable ssh connection we should create ssh public key in master instance and add it to the slave’s authorized\_keys

**Master**

1. Create MASTER\_PUBLIC\_KEY

ssh-keygen

1. Get MASTER\_PUBLIC\_KEY by copying the output of this following command

cat ~/.ssh/id\_rsa.pub

**Slave**

1. Add MASTER\_PUBLIC\_KEY to the authorized\_keys (replace the MASTER\_PUBLIC\_KEY string in the following command using the copied public key from master instance.)

echo MASTER\_PUBLIC\_KEY >> ~/.ssh/authorized\_keys

1. Copy

**Master**

1. Copy munge to home directory

sudo cp /etc/munge/munge.key ~/

1. Add read permission to other user

sudo chmod o+r ~/munge.key

We need to add read permissions to other users so that other users, besides the root user who owns this file, can copy this file.

1. Copy munge.key from master to slave using ssh protocol

scp ~/munge.key cc@SLAVE\_IP:~/

We first move the file to the home directory because on the slave instance the authorized user is 'cc', not root, so we can't use the sudo command on scp.

1. Remove munge.key in the home directory

sudo rm ~/munge.key

**Slave**

1. Remove munge.key in /etc/munge

sudo rm /etc/munge/munge.key

1. Copy munge.key (the one we just copied from the master instance) from home directory to /etc/munge directory

sudo cp ~/munge.key /etc/munge

1. Change the owner of slave’s munge.key to munge user so that it is the same as the master’s munge.key

sudo chown munge /etc/munge/munge.key

1. Change the group of slave’s munge.key to munge group so that it is the same as the master’s munge.key

sudo chgrp munge /etc/munge/munge.key

1. Change permission

sudo chmod 400 /etc/munge/munge.key

1. Remove munge.key in the home directory

sudo rm ~/munge.key

1. Update munge service in both instances (**WARNING! munge.key will not be updated if we update in slave instance only**).

**Master**

sudo systemctl daemon-reload

sudo systemctl restart munge

**Slave**

sudo systemctl daemon-reload

sudo systemctl restart munge

1. Set master’s ip address as the SlurmctldHost value in the /etc/slurm/slurm.conf file

SlurmctldHost=MASTER\_IP

MASTER\_IP will be replaced by master’s fixed ip address. You can get this information by following the instructions below:

1. Do ssh from your local computer to master instance.
2. On the master instance, install net-tools

sudo apt install net-tools

1. Check master’s fixed ip address by running

ifconfig

1. Edit compute nodes specification in /etc/slurm/slurm.conf according to slave instance specification.

NodeName=ubuntu NodeAddr=SLAVE\_IP CPUs=SLAVE\_CPUS

1. Restart slurmd

sudo systemctl daemon-reload

If you want to run slurmd on the background use this command

sudo systemctl restart slurmd

But if you want to run slurmd in debugging mode use this command

sudo /usr/sbin/slurmd -D -vvv

**Setup SLURM accounting on the master instance**

1. Create database for slurmdbd

sudo mysql

mysql> create user 'USER'@'localhost';

mysql> grant all on slurm\_acct\_db.\* TO 'USER'@'localhost';

mysql> SHOW ENGINES; #Verify we have InnoDB support

mysql> create database slurm\_acct\_db;

mysql> exit

The USER string is preferable and it will be used as the value of StorageUser key on the next step.

1. Create the /etc/slurm/slurmdbd.conf file with the following content

AuthInfo=/var/run/munge/munge.socket.2

AuthType=auth/munge

DbdHost=localhost

LogFile=/var/log/slurmdbd.log

PidFile=/var/run/slurmdbd.pid

SlurmUser=slurm

StorageType=accounting\_storage/mysql

StorageUser=USER

1. Change ownership of the /etc/slurm/slurmdbd.conf to slurm user

sudo chown slurm /etc/slurm/slurmdbd.conf

1. Change permission of the /etc/slurm/slurmdbd.conf

sudo chmod 600 /etc/slurm/slurmdbd.conf

1. Set slurmdbd plugin location in the /etc/slurm/slurm.conf file.

AccountingStorageType=accounting\_storage/slurmdbd

1. Set job accounting gather type in the /etc/slurm/slurm.conf file

JobAcctGatherType=jobacct\_gather/linux

1. Start slurmdbd service

sudo systemctl daemon-reload

sudo systemctl restart slurmdbd

1. To verify, run

sacct

If successful, the following table will appear



1. Try to submit a job
2. Create a bash script job

sudo nano myjobscript

1. Write the content

#!/bin/bash

#SBATCH --partition=test

#SBATCH -o %j.out

#SBATCH -e %j.err

#SBATCH --time=01:00

# executable

sleep 1m

1. Submit job

sbatch myjobscript

1. Check the submitted job information using

sacct

And you will see the submitted job inside the table.

