SLURM cluster

1. Launch two EC2 instances, AMI ami-7d342906, instance type t2.medium
This is the Bioconductor 3.6 AMI but it has not been updated in a while.

2. Install NTP for time synchronization
Possibly not necessary; Ubuntu 16.04 has synchronization built in but NTP is recommended for time-sensitive applications.
Refererence: https://www.eecis.udel.edu/~mills/ntp/html/index.html

```
sudo timedatectl set-ntp no
sudo apt-get update
sudo apt-get install ntp
sudo ntpq -p   # check
```

3. Setup passwordless ssh (not necessary but useful)

```
head-node$ ssh-keygen   # accept default location, no password
# append to worker-node ~/.ssh/authorized_keys by sftp from local machine
# Note: append, don't copy, or the node will no longer be accessible
# Test:
head-node$ echo "Hello from 172.31.15.34!" > test.txt
head-node$ cat test.txt | ssh 172.31.11.140 "cat >test.txt"
# Check on worker-node whether file showed up.
```

4. Install, configure and launch munge
Reference: https://github.com/dun/munge/wiki/Installation-Guide

```
sudo apt-get install munge
sudo chmod 700 /etc/munge      # recommended
sudo chmod 711 /var/lib/munge
sudo chmod 700 /var/log/munge
# /dev/random is better but takes about 1 minute per 25 bytes
dd if=/dev/urandom bs=1 count=1024 >munge.key
# copy the key to /etc/munge/munge.key on all machines
head-node$ sudo cat /etc/munge/munge.key | ssh 172.31.11.140 "cat >munge.key"
worker-node$ sudo mv munge.key /etc/munge/munge.key
sudo /etc/init.d/munge start
# Test: note quotation marks in ssh command
$ munge -n
$ munge -n | unmunge
172.31.11.140$ munge -n | ssh 172.31.15.34 "unmunge"
172.31.15.34$ munge -n | ssh 172.31.1.140 "unmunge"
```

5. Install, configure and launch slurm

a. install Ubuntu slurm package

```
sudo apt-get install slurm-llnl   # everything necessary is in here
```

b. Make a group and a user who will run the slurm daemons. By convention these are called “slurm"

```
sudo groupadd slurm
sudo useradd -g slurm slurm
sudo mkdir -p /var/spool/slurm
sudo chown slurm:slurm /var/spool/slurm
sudo chmod 700 /var/spool/slurm
```

c. More configuration

```
# The node with local IP 172.31.7.132 will be called node000
# The node with local IP 172.31.9.26 will be called node001
# node000 will be a control node and a compute node. node001 will be a compute node only
# edit /etc/hostname on both nodes and reboot
# edit /etc/hosts on both nodes and add both IP addresses and names
# 172.31.7.132 node000
# 172.31.9.26 node001
```

d. Make a simple configuration file using the online easy configuration tool
Reference: https://slurm.schedmd.com/configurator.easy.html

```
sudo mv slurm.conf /etc/slurm-llnl/slurm.conf
sudo chown slurm:slurm /etc/slurm-llnl/slurm.conf
# put the IP addresses and node names into /etc/hosts
# Note: change ProcType=proctrack/cgroup to
ProctrackType=proctrack/linuxproc
# this is simpler and faster.

# Open ports 6818 and 6817 in AWS console

# launch daemons (-D for output to stdout, each -v adds verbosity)
sudo slurmctld -D -vvvvvvv  # on node000
sudo slurmd -D -vvvvvvv     # on both nodes
```

Test batch job: (script from FAS)

```
#!/bin/bash
#SBATCH -p debug  # partition (queue)
#SBATCH -N 1      # number of nodes
#SBATCH -n 1      # number of cores
#SBATCH -t 0-1:00   # time limit
#SBATCH -o slurm.%N.%j.out
#SBATCH -e slurm.%N.%j.err
hnm=`/bin/hostname`
echo "Hello from Slurm $hnm!"
for i in {1..1000}; do
echo $RANDOM >> SomeRandomNumbers.$1.txt
done
```

```
sort SomeRandomNumbers.$1.txt
`rm -f SomeRandomNumbers.$1.txt`

# then chmod +x and launch several jobs with
perl -e 'for $i (1..10) { $n = 500+$i; system("sbatch hello.sh $n"); }

# Output will be in /home/ubuntu on the node the job runs on.
```

Note: node001 needs to have same UID/GID as node000. This happens naturally if it's the first new user and group added. Both get UID/GID 1001.

Note: to cd to a directory that is protected, need to execute “sudo -i” to become root, then cd into the directory.

6. Relaunch from AMI’s. (necessary if different key pair needs to be used)
Need to reset /etc/hosts because private IP’s change.
Need to set up passwordless ssh again because private keys are different.
Need to reset /etc/hostname
Open ports 6818 and 6817 in security group.
Need to reconfigure NFS every time the IP addresses change.

7. Install NFS and export /home and /scratch:

```
node000$ sudo apt-get update
node000$ sudo apt-get install nfs-kernel-server
node001$ sudo apt-get update
node001$ sudo apt-get install nfs-common

node000$ sudo vim /etc/exports
# Add line: /home 172.31.9.26(rw,sync,no_root_squash,no_subtree_check)
node000$ sudo systemctl restart nfs-kernel-server
# open port 2049 on the host (node000)

node001$ sudo mkdir -p /nfs/home
node001$ sudo mount 172.31.7.132:/home /nfs/home
node001$ sudo vim /etc/fstab
# (to mount at launch) append line: (not tested)
# 172.31.7.132:/home /nfs/home nfs auto,nofail,noatime,nolock,intr,tcp,actimeo=1800 0 0

# This will be a problem for the batchtools.
# The registry will be /home/ruser/registry on node000 but /nfs/home/ruser/registry on
# all other nodes.
# Instead: set up /scratch accessible for RWX to all users.
# Probably a security issue, but OK for a test cluster.

node000$ sudo mkdir -p /scratch
node000$ sudo chown nobody:nogroup /scratch
node000$ sudo vim /etc/exports
# Add line:
# /scratch 172.31.9.26(rw,sync,no_subtree_check)
node000$ sudo systemctl restart nfs-kernel-server
node000$ sudo chmod 777 /scratch
```

```
node001$ sudo mkdir -p /scratch
node001$ sudo mount 172.31.7.132:/scratch /scratch
node001$ sudo chmod 777 /scratch
# (to mount at launch) append line to /etc/fstab: (not tested)
# 172.31.7.132:/scratch /scratch nfs auto,nofail,noatime,nolock,intr,tcp,actimeo=1800 0 0
```

8. Update Bioconductor. Install batchtools, geuvPack, gQTLstats.
(Save to an AMI, because the Bioconductor update takes about an hour.)
Launch R in /scratch and run as described in EQTL notes.