1. Make an Amazon AWS account or sign in using pre-registered username / password
2. Go to EC2 and launch instance using (ami-628c8a0a) – instance already has base R / Rstudio installed - storage set at 10 Gb – the default
   1. Create a security group
      1. SSH – port 22 –IP anywhere – or us designated IP address depending on use
      2. HTTP – port 80
      3. Custom TCP – port 10187
3. Create or use an existing key pair and save to separate folder on desktop (ex. amazon-key.pem)
4. Go to AWS dashboard – go to volumes under elastic block store
   1. Create volume with specific storage space (e.g. 100gb)
      1. Attach the volume to master instance based on instance ID and location
5. Login on to server via command – call key directory and modify
   1. cd desktop
   2. cd key
   3. chmod 400 amazon-key.pem
   4. To actually login use -> ssh –i amazon-key.pem root@(public DNS)
      1. Ex. ssh –i amazon-key.pem root@ec2-54-174-158-1.compute-1.amazonaws.com
6. \*\*Notes – in order to install software to the server\*\* - use if R is not preloaded on instance
   1. in linux command line (bash), search for software
      1. sudo apt-cache search rstudio
      2. sudo apt-get install rstudio-server
   2. install.packages for R – that would have all packages needed to be installed for R
7. Go to command line (bash) – login to the server & mount volume
   1. ssh –i amazon-key.pem root@ec2-54-174-75-252.compute-1.amazonaws.com
   2. Will be prompted to use Ubuntu rather than root
      1. ssh –i amazon-key.pem ubuntu@ec2-54-174-75-252.compute-1.amazonaws.com
   3. Mount
      1. Type and hit return on sudo su
      2. mkfs.ext4 /dev/xvdh (this is the path of the volume created – the “s” is replaced with “xv”)
      3. mkdir –m 000 /vol
      4. echo “/dev/xvdh /vol auto noatime 0 0” | sudo tee –a /etc/fstab
      5. sudo mount /vol
8. Now deal with the SSH keys – necessary to be on ubuntu user of master server
   1. su ubuntu
   2. cd
   3. ssh-keygen –t dsa (just hit return when asked any question about passphrase)
   4. cat ~/.ssh/ id\_dsa.pub >> ~/.ssh/authorized\_keys
   5. chmod 644 ~/.ssh/authorized\_keys
9. Start up base R and install all packages required for cluster and other code
   1. In terminal
      1. R
      2. Install.packages(“doSNOW”)
      3. Install. Packages(“itertools”)
      4. Install all other packages needed for desired computation
   2. This step is necessary for all the nodes to have the packages installed to run the cluster commands and also in order for the desired to code to work on each node
      1. If not followed packages have to manually installed to each node
      2. An alternative method of install packages to node will be explored
10. Now to create the nodes
    1. Go to AWS dashboard – click on instance and under actions – select “create image”
    2. Name image name as something specific – then create – images will be added to Images tab under AMIs
       1. It will take a while to process (rebooting master instance)
    3. Select the image – click Launch
       1. Select instance type desired and the quantity – (if you wanted 8 slaves then you would choose 8 here)
       2. Make sure that the instances share the same security group and placement groups (location) as master
11. Add the Public DNS names of the slave machines into a new file /vol/nodelist
    1. Go to command – log into the master
    2. Go to AWS dashboard – go to instances – copy the Public DNS of each slave
    3. In command, type: cat > nodelist.txt (hit return)
       1. Paste Public DNS names
       2. Press return
       3. Press Control + d (done)
    4. Type: cat nodelist.txt
       1. Should present the DNS names
    5. \*\*When copying the Public DNS names of many nodes\*\* - use this method
       1. Go to instances – select the slaves – copy the selected Public DNS and the instance ID attached
       2. Go to command – log into the master
       3. In command, type: cat > nodelist.txt (hit return)
          1. Paste Public DNS names
          2. Press return
          3. Press Control + d
          4. Type: cat nodelist.txt | cut -d' ' -f2 > nodelist2.txt
12. Connection between Master and nodes established
    1. Test connection, type: ssh ec2-54-174-75-252.compute-1.amazonaws.com ls (Public DNS of a slave)
    2. ssh ec2-54-174-75-252.compute-1.amazonaws.com ls –ka
    3. ssh ec2-54-174-75-252.compute-1.amazonaws.com ls –la
13. Install AWS CLI (for Mac)
    1. Necessary to have Pip installed
       1. sudo easy\_install pip
    2. Next Install AWS CLI using Pip (on Mac)
       1. sudo pip install awscli
       2. sudo pip install –upgrade awscli
14. Configure the AWS CLI
    1. Go to command – stay as user
       1. aws configure
          1. Follow steps
          2. Enter Access Key ID and Secret Access Key
          3. Enter Region and output format (json)
15. Call the cluster and perform parallel computation in R
    1. First, it is very useful to disable unidentified host confirmation alerts that appear when connecting to the instances
       1. Go to command – stay as user
          1. StrictHostKeyChecking=no
    2. Go to command – ssh login onto the master node
       1. Start up R
          1. R

library(doSNOW)

library(itertool)

setDefaultClusterOptions(port=10187)

machines <- readLines("nodelist2.txt")

## 'each' refers to the number of cores for each node being used

machines <- rep(c("localhost", machines), each = 1)

## In order to make and start the cluster

clust <- makeCluster(machines, type = "SOCK")

registerDoSNOW(clust)

## Now perform desired parallel computation here

## In order to stop the cluster

StopCluster(cl)

## Remember to stop instances in EC2 console to avoid unnessary cost