Creating A Cybera Cluster With Hadoop

Updated: February 2020

In this tutorial, we will start 4 machines on Cybera Rapid Access Cloud which will function later as the worker and master nodes for Hadoop. After creating your account on RAC, log into your account here.
Following the steps outlined in the Basic Guide for Cybera, create 4 "m1.small" instances. While creating the instances, make sure that the names you assign to your machines are in compliance with the restrictions for valid hostnames. To be safe, do not use characters other than a-z, 1-9 and '-' in your hostnames. Choose 'Ubuntu 14.04' as your image.

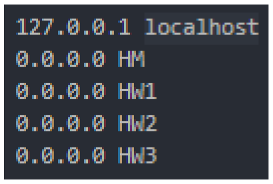
Cybera Instance Setup

1. Create account
2. Create a key pair
    a. Log-in to the Rapid Access Cloud dashboard at https://cloud.cybera.ca.
    b. In the left-hand panel under "Compute", click "Access & Security".
    c. Click the "Key Pair" tab, then click "+Create Key Pair".
    d. Enter a [key_pair_name], then click "Create Key Pair". The browser will automatically download a file named [key_pair_name].pem.
    e. Move or save this file on your computer somewhere you will remember. It will be used when accessing instances created with this key pair.
3. Modify the default security group
    a. In the left-hand panel under "Network", click "SecurityGroups".
    b. Click the "Manage Rules" button on the right-hand side associated with the "default" security group.
    c. Click "Manage Rules" for the default security group
    d. Click "Add Rule"
    e. In the "Port" box, input "22" and click "Add"
4. Launce 4 instances
    a. Under "Computer" section, navigate to "Instances" tab
    b. Click "Launch Instance"
    c. In the "Details" tab, specify the following parameters. **Do not yet click "Launch"**:
        i. Availability Zone: (nova)
        ii. Instance Name: [your_instance_name]
        iii. Flavor: (m1.small)
        iv. Instance Count: 4
        v. Instance Boot Source: (Boot from image)
        vi. Image Name: Ubuntu 14.04
    d. Click "Access & Security" tab within the Launch Instance field and select the [key_pair_name] created earlier in the tutorial. The default security group should be checked as well.
    e. Click the "Launch" button. It should take the instance less than 2 minutes to launch; progress can be monitored in the "Status" column and should say "Active" when ready.
5. Allocate and associate a floating IP
    a. In the left-hand panel under "Compute", click "Instances".
    b. Click the Action drop-down button on the right-hand side and select "Associate Floating IP".
    c. Click on the "+" sign next to "Select an IP address".
    d. There is only one pool of addresses available (nova) and the quota shows only one IP address from that pool, so simply click "Allocate IP".
    e. After the IP address has been allocated, click the "Associate" button the right-hand side. Under the Instances summary, your Instance should now have three IP addresses, including a publicly accessible IPv4 address.

Instance Configuration

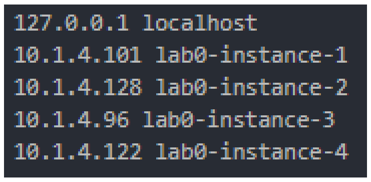
After the VMs are up and running, note down the IP addresses of each of the four machines. These addresses will be used to modify the config and hosts files found in D2L. After some preparation, you will be transferring these files from your operating system to the Linux instance on the cloud. To prep the files for transfer:

1. Modifying the hosts File
   a. On D2L, in the "Setting Up your Cybera Cluster" module, there is a template of the hosts file attached. Each of your VMs you should add make up line in the hosts file with the following format:
      i. [VM IP address] [VM name]
   b. Note that the IP address and hostname are separated with a single space. The file contains a line defining the localhost, which can be left untouched.
   c. Example:



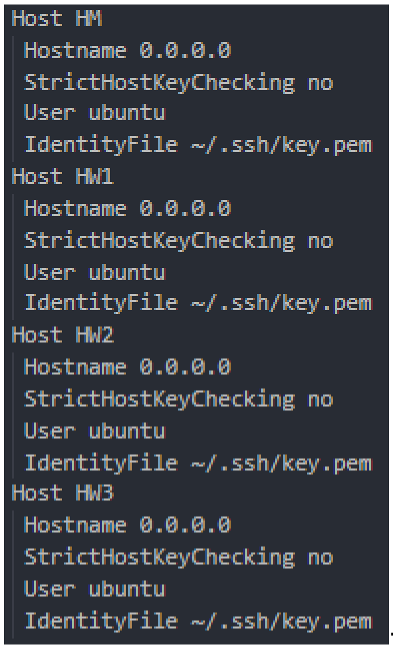
```
127.0.0.1 localhost
0.0.0.0 HM
0.0.0.0 HW1
0.0.0.0 HW2
0.0.0.0 HW3
```

→



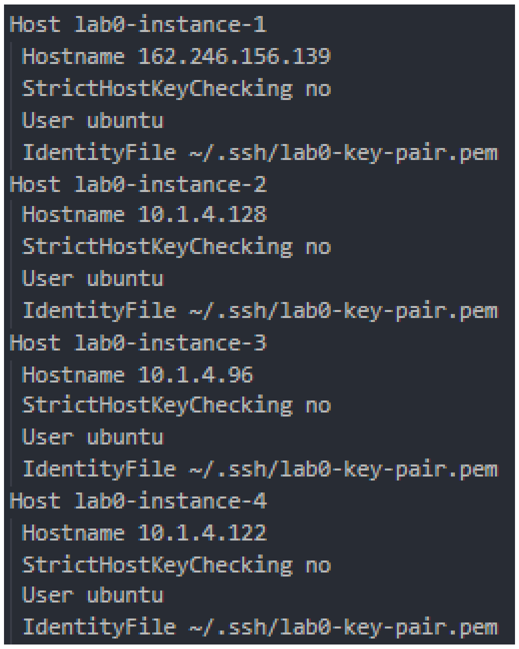
```
127.0.0.1 localhost
10.1.4.101 lab0-instance-1
10.1.4.128 lab0-instance-2
10.1.4.96 lab0-instance-3
10.1.4.122 lab0-instance-4
```

2. Creating the config File
   a. On D2L, in the "Setting Up your Cybera Cluster" module, there is a template of the config file. For each of your VMs, you should add 5 lines with the following format:
      i. Host [VM name]
      ii. Hostname [VM IP]
      iii. StrictHostKeyChecking no
      iv. User ubuntu
      v. IdentityFile ~/.ssh/[your_key].pem
   b. This format is already present in the file. You can just substitute the IP addresses and hostnames of your machines. When adding details for the master node in this file, use the Public IP (floating IP) that is assigned to your master node as the [VM IP] (it's unverified whether this is necessary).
   c. Example (lab0-instance-1 is master node):



```
Host HM
 Hostname 0.0.0.0
 StrictHostKeyChecking no
 User ubuntu
 IdentityFile ~/.ssh/key.pem
Host HW1
 Hostname 0.0.0.0
 StrictHostKeyChecking no
 User ubuntu
 IdentityFile ~/.ssh/key.pem
Host HW2
 Hostname 0.0.0.0
 StrictHostKeyChecking no
 User ubuntu
 IdentityFile ~/.ssh/key.pem
Host HW3
 Hostname 0.0.0.0
 StrictHostKeyChecking no
 User ubuntu
 IdentityFile ~/.ssh/key.pem
```

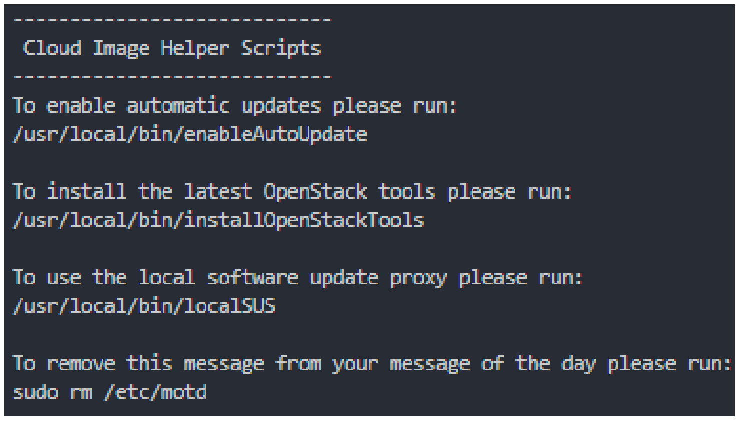
→



```
Host lab0-instance-1
 Hostname 162.246.156.139
 StrictHostKeyChecking no
 User ubuntu
 IdentityFile ~/.ssh/lab0-key-pair.pem
Host lab0-instance-2
 Hostname 10.1.4.128
 StrictHostKeyChecking no
 User ubuntu
 IdentityFile ~/.ssh/lab0-key-pair.pem
Host lab0-instance-3
 Hostname 10.1.4.96
 StrictHostKeyChecking no
 User ubuntu
 IdentityFile ~/.ssh/lab0-key-pair.pem
Host lab0-instance-4
 Hostname 10.1.4.122
 StrictHostKeyChecking no
 User ubuntu
 IdentityFile ~/.ssh/lab0-key-pair.pem
```

Now that your host and config files are prepped, you must move them from your computer into the Linux virtual machines. This can be done in a few ways; however, we will just create new files in Linux and paste the contents of our prepared files into them. Entering and navigating these instances will involve using basic Linux commands.

3. Connect to the master node (i.e. the instance with the allocated floating IP address)
    a. Open git bash (windows) or equivalent terminal (mac) and navigate to the folder containing your key pair file. If using git bash, can simply navigate to folder, right click, and select "git bash here".
    b. Enter `ssh -i [key_pair_name].pem ubuntu@[floating_ip_address]`
        i. Note that the square brackets indicate that you need to insert your own information.
    c. Answer "yes" when prompted: Are you sure you want to continue connecting (yes/no)?
    d. You should now be connected to the instance and see something like:

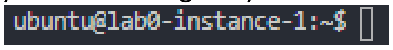


```
----------------------------
 Cloud Image Helper Scripts
----------------------------
To enable automatic updates please run:
/usr/local/bin/enableAutoUpdate

To install the latest OpenStack tools please run:
/usr/local/bin/installOpenStackTools

To use the local software update proxy please run:
/usr/local/bin/localSUS

To remove this message from your message of the day please run:
sudo rm /etc/motd
```

Now that you're working on the master node, you can copy our files over. You can do this using the SCP command. However, if you're uncomfortable with Linux, the following steps will help you get more comfortable with basic navigation and file manipulation. Then, you can learn to use the SCP command to move large files after some introduction. Since you are navigating through the command line, you will need to know some basic Linux commands to do so:

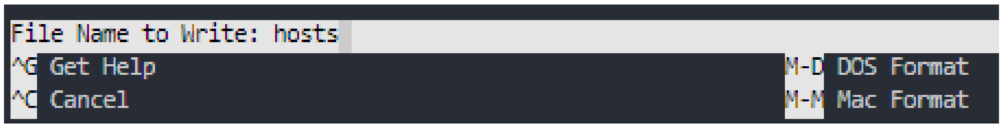
- `cd [pathName]` → Changes the current working directory to another path/location
    - `cd ..` → Moves up one folder (outside of current folder)
- `ls` → Lists the contents of the current working directory
- `sudo nano [fileName]` → Opens named file (creates named file if it doesn't exist)
- `rm [filename]` → Deletes named file

You will be able to tell what node/directory you are working in by the location shown in the terminal:



```
ubuntu@lab0-instance-1:~$
```

4. Moving everything to the Master Node
    a. Moving hosts to the master node
        i. In the master node, navigate to the /etc/ folder using `cd /etc/`
        ii. Enter `sudo nano hosts` to create and open a hosts file
        iii. Copy and paste the contents of your prepared "hosts" file into the newly created file
        iv. Press ctrl+O to save changes. This will prompt a message asking you to name the new file. Ensure that the name does not have a ".txt" extension and press enter.



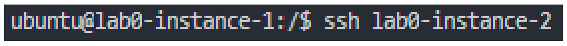
```
File Name to Write: hosts
^G Get Help                                          M-D DOS Format
^C Cancel                                            M-M Mac Format
```

        v. Press ctrl+X to exit the file
    b. Moving config to the master node
        i. In the master node, navigate to the ~/.ssh/ folder using `cd ~/.ssh/`
        ii. Enter `sudo nano config` to create and open a config file
        iii. Copy and paste the contents of your prepared "config" file into the newly created file
        iv. Press ctrl+O to save changes. This will prompt a message asking you to name the new file. Ensure that the name does not have a ".txt" extension and press enter.
        v. Press ctrl+X to exit the file

c. Moving key pair to the master node
   i. Repeat the config steps with your key pair file (it belongs in the same directory). Make sure to name it exactly the same as your prepped key-pair file with the ".pem" extension.

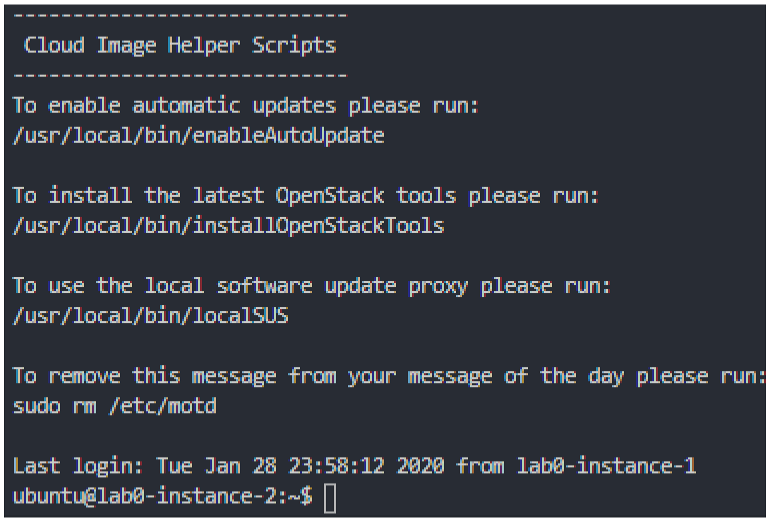
5. Testing
   a. If you have correctly created your files, you should be able to ssh (traverse) into all of your VMs from your master node without a password. If you're asked for a password or you get a "PERMISSION DENIED" message, your config file is not created correctly.
   b. Example:

Entering ssh command: `ssh [workerNodeName]`



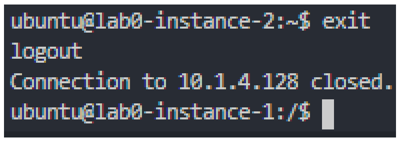
```
ubuntu@lab0-instance-1:/$ ssh lab0-instance-2
```

Response:



```
----------------------------
 Cloud Image Helper Scripts
----------------------------
To enable automatic updates please run:
/usr/local/bin/enableAutoUpdate

To install the latest OpenStack tools please run:
/usr/local/bin/installOpenStackTools

To use the local software update proxy please run:
/usr/local/bin/localSUS

To remove this message from your message of the day please run:
sudo rm /etc/motd

Last login: Tue Jan 28 23:58:12 2020 from lab0-instance-1
ubuntu@lab0-instance-2:~$
```

Exiting and returning to master node: `exit`



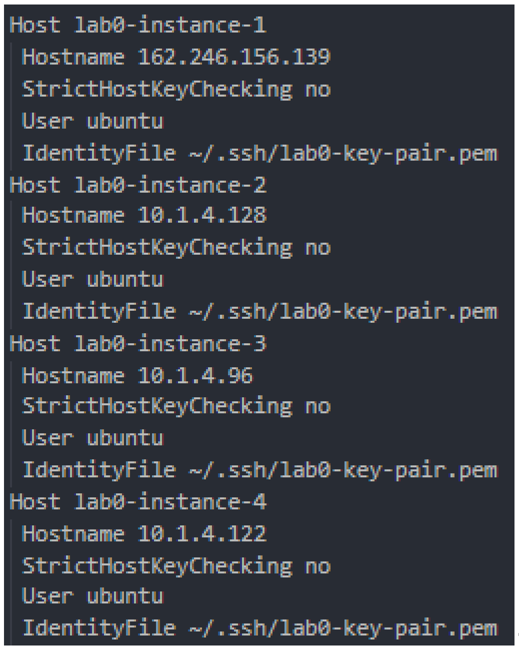
```
ubuntu@lab0-instance-2:~$ exit
logout
Connection to 10.1.4.128 closed.
ubuntu@lab0-instance-1:/$
```

Note that you will only be able to move between worker nodes through your master node. For example, assuming node1 is your master node, you will not be able to navigate to node 3 from node 2. Instead, you will have to edit node 2 and then use the ssh command to enter node 3.

Cluster Setup on Cybera

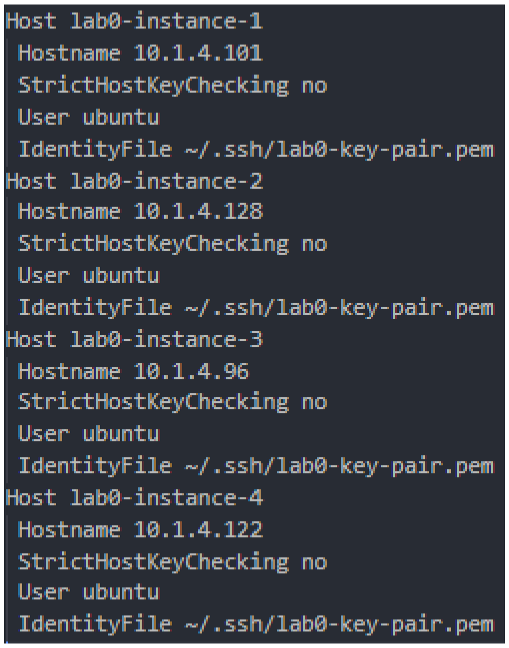
Before starting this tutorial, please ensure that all steps of the previous tutorial are complete. If they are, you should have your **hosts** file updated according to your cluster IPs in the /etc/ folder and your **config** and **[your_key].pem** files in the ~/.ssh/ folder. All the IPs in hosts and config files should be the internal IPs of the machine (e.g. 10.X.X.X).

- The previous tutorial asked you to use the "floating IP address" rather than the "internal IP address" in your hosts file. I'm not sure why this tutorial wants it the other way around. When I was doing my setup, I went back into my config file and switched the IP back to my internal IP as follows:



```
Host lab0-instance-1
 Hostname 162.246.156.139
 StrictHostKeyChecking no
 User ubuntu
 IdentityFile ~/.ssh/lab0-key-pair.pem
Host lab0-instance-2
 Hostname 10.1.4.128
 StrictHostKeyChecking no
 User ubuntu
 IdentityFile ~/.ssh/lab0-key-pair.pem
Host lab0-instance-3
 Hostname 10.1.4.96
 StrictHostKeyChecking no
 User ubuntu
 IdentityFile ~/.ssh/lab0-key-pair.pem
Host lab0-instance-4
 Hostname 10.1.4.122
 StrictHostKeyChecking no
 User ubuntu
 IdentityFile ~/.ssh/lab0-key-pair.pem
```

→



```
Host lab0-instance-1
 Hostname 10.1.4.101
 StrictHostKeyChecking no
 User ubuntu
 IdentityFile ~/.ssh/lab0-key-pair.pem
Host lab0-instance-2
 Hostname 10.1.4.128
 StrictHostKeyChecking no
 User ubuntu
 IdentityFile ~/.ssh/lab0-key-pair.pem
Host lab0-instance-3
 Hostname 10.1.4.96
 StrictHostKeyChecking no
 User ubuntu
 IdentityFile ~/.ssh/lab0-key-pair.pem
Host lab0-instance-4
 Hostname 10.1.4.122
 StrictHostKeyChecking no
 User ubuntu
 IdentityFile ~/.ssh/lab0-key-pair.pem
```

Also, double check that the "IdentityFile" variable in the **config** file has the same name as the key pair file on your master node (see above example). If your **config** or **hosts** files associate the floating IP with the master node's entry, please change it to the internal IP. Set the permission of your **config** file and **.pem** file to 600, if not so already (I didn't do this). By now, you should be able to ssh into all 3 worker nodes from your master node.

In order to complete the installation and configuration of Hadoop on the cluster, download the following files from D2L and copy them to your master node as you did with the host and config files. Just place them in the main directory. Remember to include any ".py" extensions and exclude any ".txt" extensions when using the `sudo nano [fileName]` command.

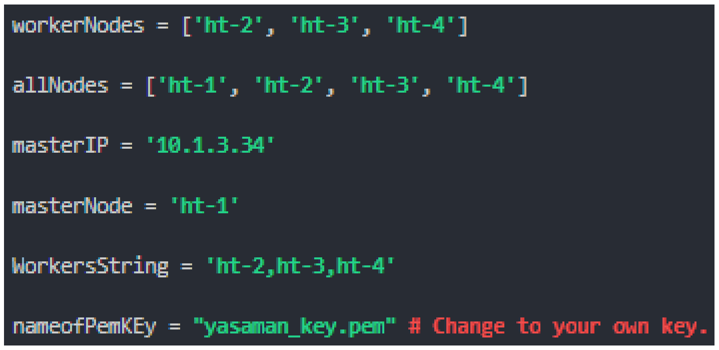
a. **setupSSH.py**
b. **installHadoop.py**
c. **configureHadoop.py**
d. **MasterScript.py**
e. **WorkerScript.py**
f. **profile.txt**

If you accidentally include the .txt extension when creating **profile.txt**, remove it using: `mv profile.txt profile`

Before using these files, you'll have to modify the following variables in **setupSSH.py**, **installHadoop.py** and **configureHadoop.py**:

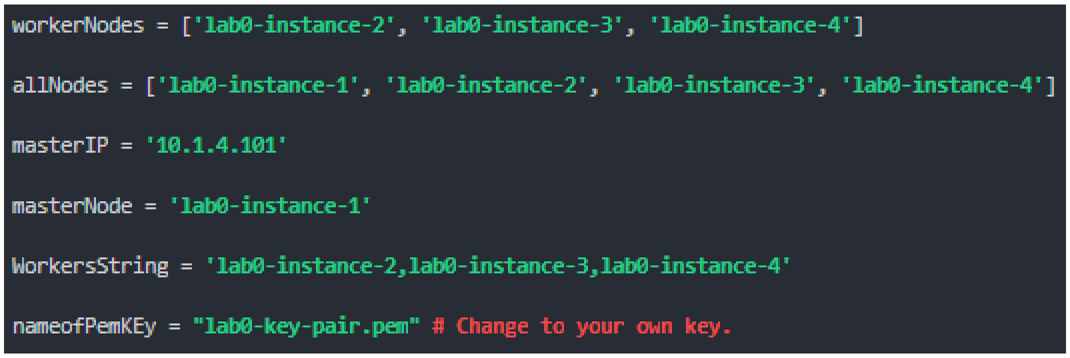
1. Add the hostnames of your worker nodes to the "workerNodes" array at the beginning of each script.
2. Add the hostnames of your master and worker nodes to the "allNodes" array.
3. Set the "masterIP" variable to the **internal IP address** of your master node VM (the one starting like 10.X.X.X)
4. Set the "masterNode" variable to the hostname of your master node.
5. Insert the hostnames of your worker nodes into the "WorkersString" variable separating the names by comma(,). Make sure to not insert any empty spaces in the string.
6. Set the value of "nameofPemKEy" variable to your ".pem" file name.

Before modifications:



```
workerNodes = ['ht-2', 'ht-3', 'ht-4']

allNodes = ['ht-1', 'ht-2', 'ht-3', 'ht-4']

masterIP = '10.1.3.34'

masterNode = 'ht-1'

WorkersString = 'ht-2,ht-3,ht-4'

nameofPemKEy = "yasaman_key.pem" # Change to your own key.
```

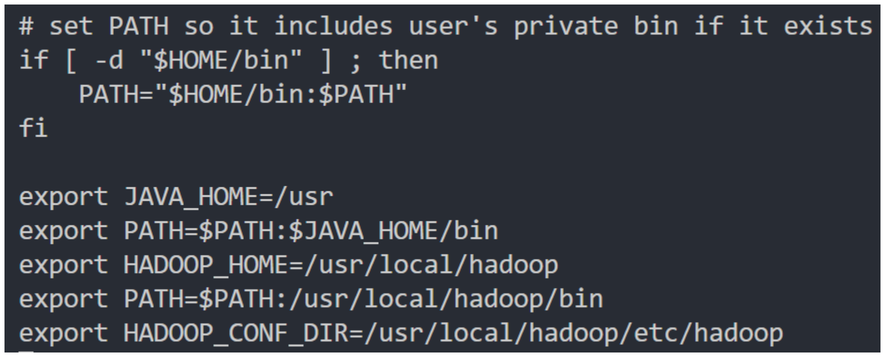
After modifications:



```
workerNodes = ['lab0-instance-2', 'lab0-instance-3', 'lab0-instance-4']

allNodes = ['lab0-instance-1', 'lab0-instance-2', 'lab0-instance-3', 'lab0-instance-4']

masterIP = '10.1.4.101'

masterNode = 'lab0-instance-1'

WorkersString = 'lab0-instance-2,lab0-instance-3,lab0-instance-4'

nameofPemKEy = "lab0-key-pair.pem" # Change to your own key.
```

You'll be completing this process in three steps:

1. Set up SSH
   a. Run the file setupSSH.py on your master node using `python setupSSH.py`
   b. Login to each of your worker nodes and see if the **hosts** and **config** files have been successfully copied to the /etc/ and ~/.ssh/ folders respectively. If yes, the setup is successful.
2. Install Hadoop on all nodes
   a. Run the file **installHadoop.py** on the master node using python `python installHadoop.py`. It should take a little while for Hadoop to be installed on all nodes.
   b. After the script is done running, in your main directory, copy the contents of **profile** into the very bottom of **.profile** (note the period) on your <u>master and worker nodes</u>. ".profile" is a hidden file and can't be seen in the directory (unless you use the command `ls -la`)



```
# set PATH so it includes user's private bin if it exists
if [ -d "$HOME/bin" ] ; then
    PATH="$HOME/bin:$PATH"
fi

export JAVA_HOME=/usr
export PATH=$PATH:$JAVA_HOME/bin
export HADOOP_HOME=/usr/local/hadoop
export PATH=$PATH:/usr/local/hadoop/bin
export HADOOP_CONF_DIR=/usr/local/hadoop/etc/hadoop
```

   c. Run the following command: "`. profile`" (note the use of a period). Nothing should visually happen after the command is used.
   d. To test if the command worked (i.e. if the **.profile** file has run successfully), echo any of the environment variables listed inside the file and see if they return the same value as specified in the file. For example, if you run the command `echo $HADOOP_CONF_DIR`, it should return `/usr/local/hadoop/etc/hadoop/`.

3. Configure Hadoop
   a. Run **configureHadoop.py** on the master node to set the configuration files of Hadoop. This file will automatically run the **MasterScript.py** and **WorkerScript.py** files.

Testing Hadoop

By now all your nodes should have Hadoop installed and configured on them. In order to test the correct configuration and installation of the cluster, do the following:

1. Format HDFS using the following command on master node: `hdfs namenode -format`
2. Start all Hadoop Services using the following command on master node: `$HADOOP_HOME/sbin/start-all.sh`
3. Using the `jps` command, you can see which services are running:
   a. Running `jps` on your master node should yield the following services:
      i. Jps
      ii. SecondaryNameNode
      iii. ResourceManager
      iv. NameNode
   b. Running `jps` on your worker nodes should yield the following services:
      i. NodeManager
      ii. DataNode
      iii. Jps
4. In order to enable Hadoop Streaming, copy the hadoop-streaming-2.7.6.jar to $HADOOP_HOME using the following command on the master node:
   a. `cp /usr/local/hadoop/share/hadoop/tools/lib/hadoop-streaming-2.7.6.jar /usr/local/hadoop/`

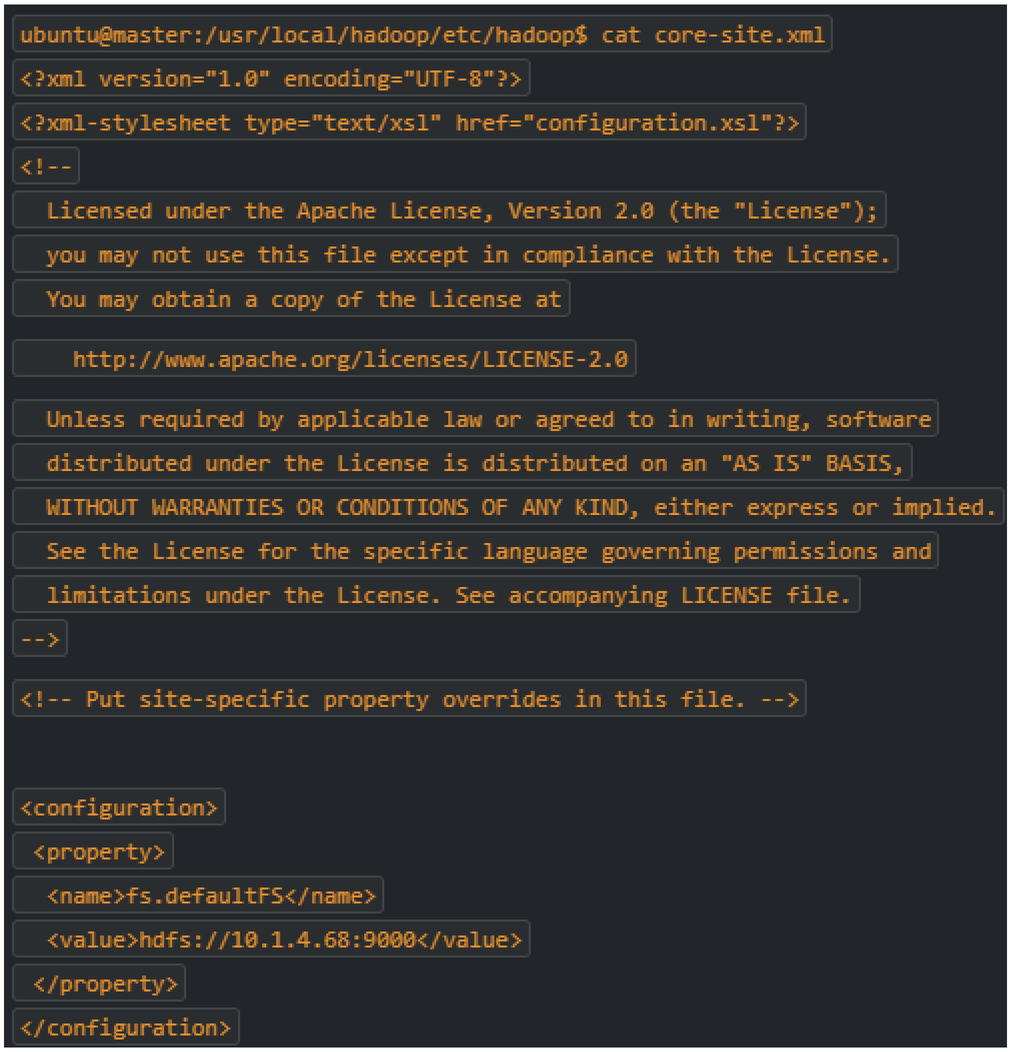
You now have a fully functioning Hadoop cluster. Unless it isn't functioning. In that case, check out the following troubleshooting section.

Troubleshooting

[Fatal Error] core-site.xml:27:2: The markup in the document following the root element must be well-formed.

If you are getting this error after running start-all.sh and your jps is showing up as a single line, follow these steps:

1. There is an additional <property> clause in $HADOOP_CONF_DIR/core-site.xml which needs to be removed. Sudo nano edit this file and remove the additional property file so that your core-site.xml looks like this (replace master IP with your own master IP address):



```
ubuntu@master:/usr/local/hadoop/etc/hadoop$ cat core-site.xml
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->


<configuration>
 <property>
  <name>fs.defaultFS</name>
  <value>hdfs://10.1.4.68:9000</value>
 </property>
</configuration>
```

2. Modify this file on all worker nodes as well.
3. Run `hdfs namenode -format` on the master node
4. Run `$HADOOP_HOME/sbin/start-all.sh` on the master node
5. Check that all processes are running with `jps` on master and worker nodes (see Testing Hadoop #3)

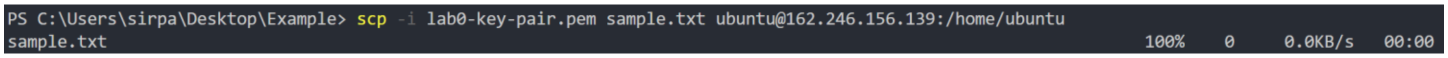
Transferring files to Linux

If you're using nodes for processing big data, you're likely going to want a better method of loading files onto your nodes aside from copying and pasting.

On terminal (Mac) or git bash (Windows), you can simply transfer files from your computer into your Linux nodes using the SCP command:

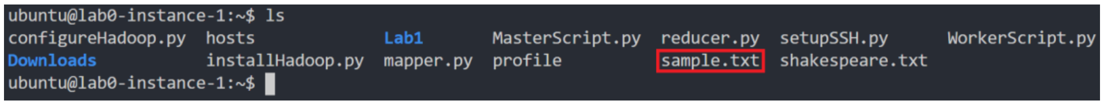
```
scp -i [key-pair].pem [relative path of local file] ubuntu@[floating IP]:[linux destination path]
```

For example, if you wanted to copy a file from your local computer to your main git directory, you'd use `/home/ubuntu/` as your destination path:



```
PS C:\Users\sirpa\Desktop\Example> scp -i lab0-key-pair.pem sample.txt ubuntu@162.246.156.139:/home/ubuntu
sample.txt                                                                                100%    0     0.0KB/s   00:00
```

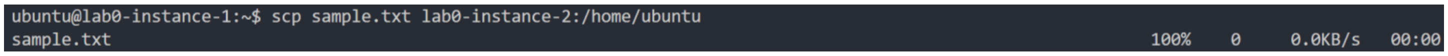
After logging in as usual, you'll be able to see the file in the named Linux directory:



```
ubuntu@lab0-instance-1:~$ ls
configureHadoop.py  hosts           Lab1       MasterScript.py  reducer.py  setupSSH.py      WorkerScript.py
Downloads           installHadoop.py mapper.py profile          sample.txt  shakespeare.txt
ubuntu@lab0-instance-1:~$ 
```

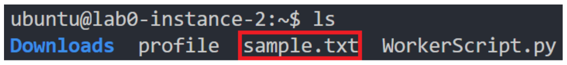
You can also copy files from your master node to your worker nodes with SCP:

```
scp [relative path of local file] [nodename]:[destination path]
```



```
ubuntu@lab0-instance-1:~$ scp sample.txt lab0-instance-2:/home/ubuntu
sample.txt                                                                                100%    0     0.0KB/s   00:00
```

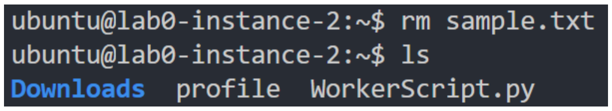
After switching to your other node, you'll be able to see the file in the named Linux directory:



```
ubuntu@lab0-instance-2:~$ ls
Downloads  profile  sample.txt  WorkerScript.py
```

Also, to remove unwanted files, just use the rm command:

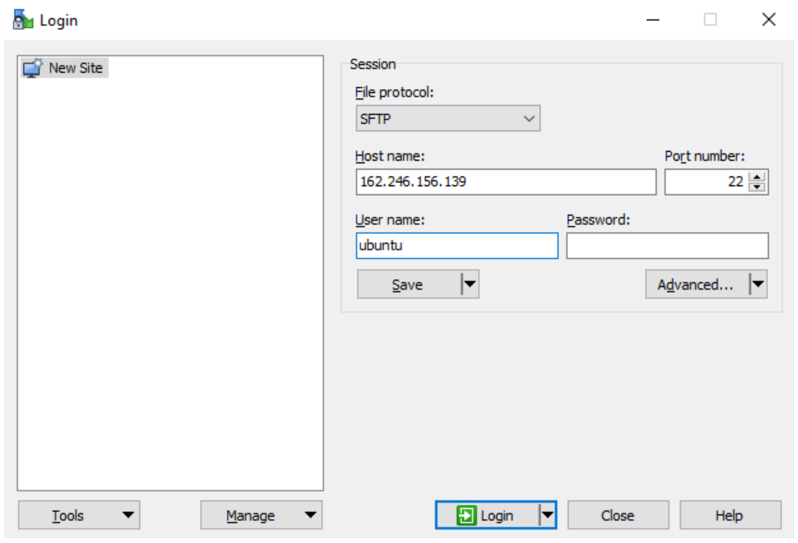
```
rm [filename]
```



```
ubuntu@lab0-instance-2:~$ rm sample.txt
ubuntu@lab0-instance-2:~$ ls
Downloads  profile  WorkerScript.py
```

If you're not a fan of using the console and consider yourself more of a "click and drag" kind of person. Windows users can install WinSCP. There is likely a similar service for Mac users:

1. Download WinSCP with all the default settings provided by the setup wizard.
2. Launch WinSCP
3. On the login screen, enter your floating IP as the host name and "ubuntu" as the username:



Login

New Site

Session

File protocol: SFTP

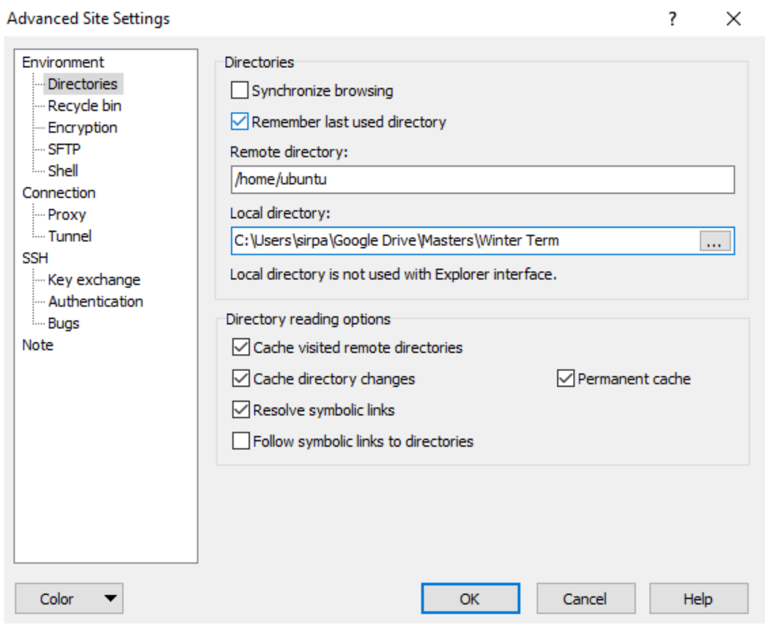
Host name: 162.246.156.139

Port number: 22

User name: ubuntu

Password:

4. Click the "Advanced" button
5. Under the "Directories" section, enter "/home/ubuntu/" as your remote directory and the location of the files you wish to copy as the local directory.



Advanced Site Settings

Directories

☐ Synchronize browsing

☑ Remember last used directory

Remote directory: /home/ubuntu

Local directory: C:\Users\sirpa\Google Drive\Masters\Winter Term

Local directory is not used with Explorer interface.

Directory reading options

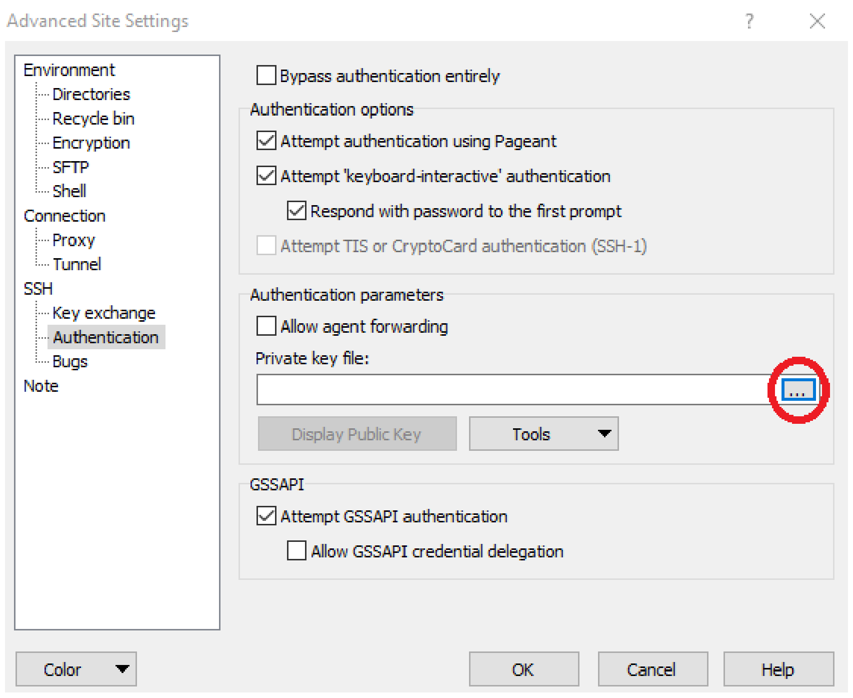
☑ Cache visited remote directories

☑ Cache directory changes ☑ Permanent cache

☑ Resolve symbolic links

☐ Follow symbolic links to directories

6. Under the "Authentication" section, click the […] button to begin linking your key-pair file.


Advanced Site Settings

Environment
- Directories
- Recycle bin
- Encryption
- SFTP
- Shell

Connection
- Proxy
- Tunnel

SSH
- Key exchange
- Authentication
- Bugs

Note

- [ ] Bypass authentication entirely

Authentication options
- [✓] Attempt authentication using Pageant
- [✓] Attempt 'keyboard-interactive' authentication
  - [✓] Respond with password to the first prompt
- [ ] Attempt TIS or CryptoCard authentication (SSH-1)

Authentication parameters
- [ ] Allow agent forwarding

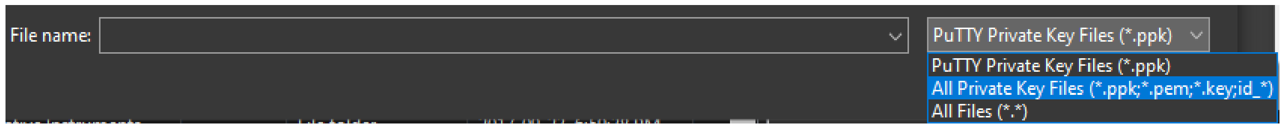
Private key file: ...

Display Public Key | Tools

GSSAPI
- [✓] Attempt GSSAPI authentication
  - [ ] Allow GSSAPI credential delegation

Color | OK | Cancel | Help

7. In the bottom right of your explorer window, indicate that you want to search for all private key files (including .pem) rather than just .ppk files.


File name:

PuTTY Private Key Files (*.ppk)
- PuTTY Private Key Files (*.ppk)
- All Private Key Files (*.ppk;*.pem;*.key;id_*)
- All Files (*.*)

8. Locate and select your [key-pair].pem file
9. Press "OK" when asked if you want to convert your OpenSSH private key (.pem file) into PuTTY format (.ppk file).
10. Rename the new .ppk file to match your [key-pair].pem file.
11. Press "OK" to exit advanced settings.
12. Click "Save" to ensure that you don't have to input any of this information again
13. Click "Login"
14. Click "Yes" to continue connecting to an unknown server and add its host key to a cache.

WinSCP is now setup. The left side shows your local files, the right side shows your node files. You can now copy files into your master node with a simple click and drag.