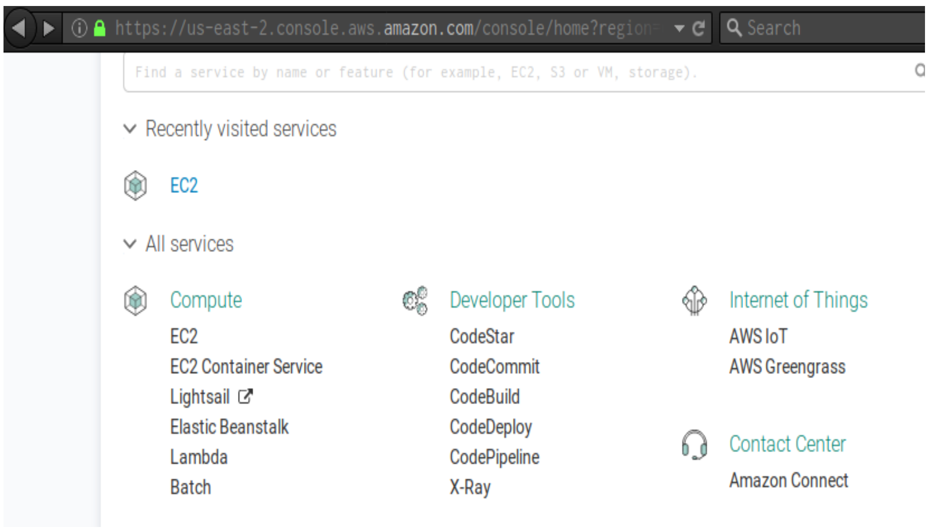
ECE 579: Cloud Computing, Fall 2017

Recitation 1: Getting started with AWS instance

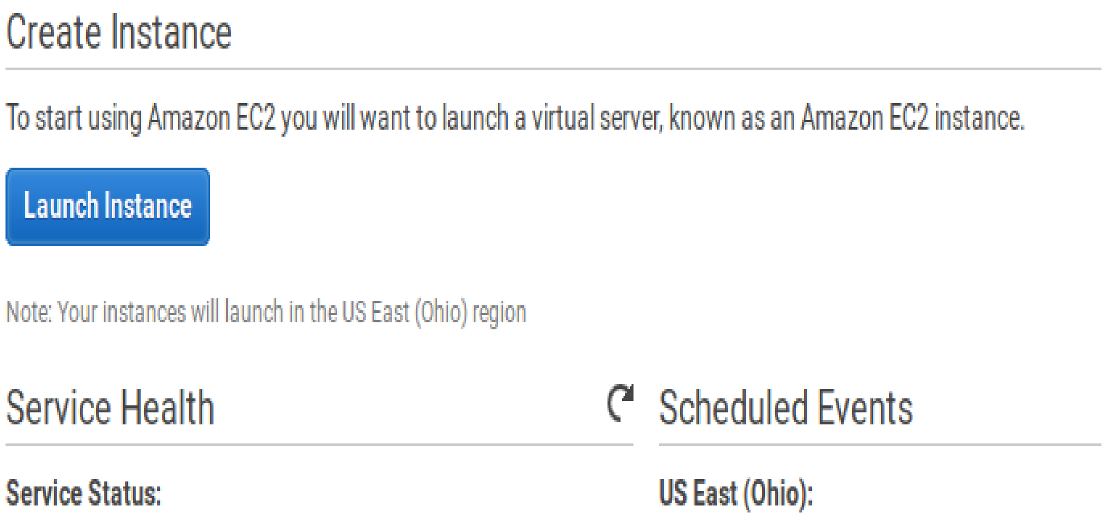
This is tutorial will introduce you to setting up an AWS ec2 instance. You can follow similar procedure to setup other instances for your projects. Familiarity with any Linux distribution is recommended.

Setting up AWS ec2 instance

1. Sign in to the console.
2. Services > Compute > **EC2**



3. Select launch instance.



Create Instance

To start using Amazon EC2 you will want to launch a virtual server, known as an Amazon EC2 instance.

Note: Your instances will launch in the US East (Ohio) region

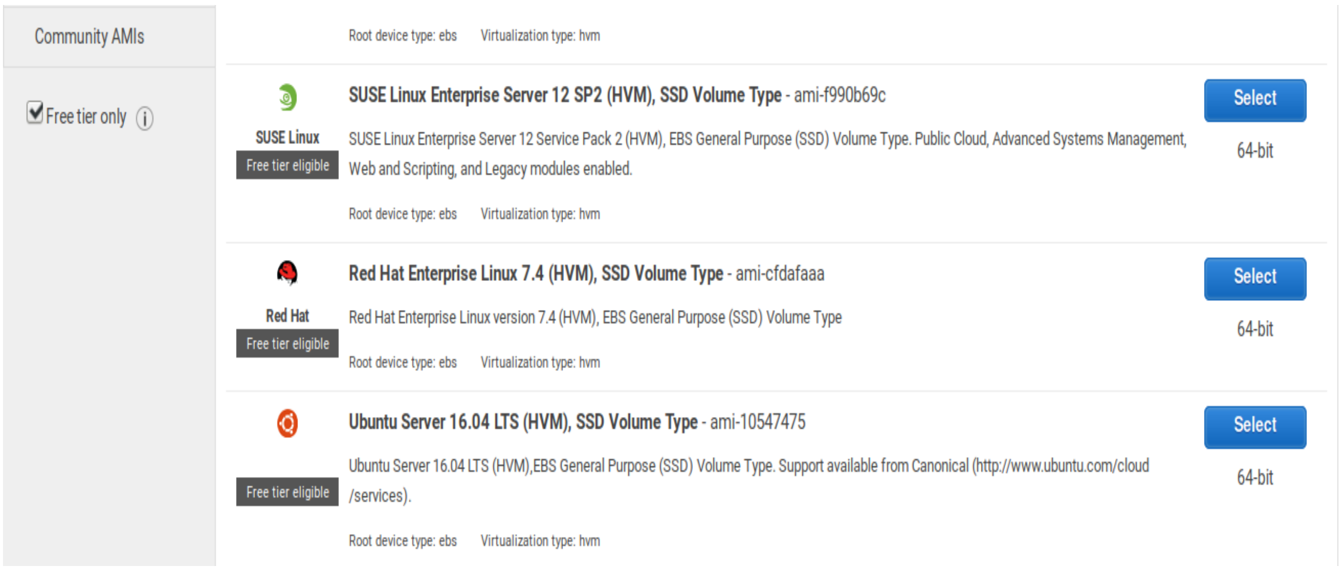
Service Health

Scheduled Events

Service Status:

US East (Ohio):

4. Choose an AMI: Select free tier only. We will use Ubuntu Server 16.04 for our experiment.



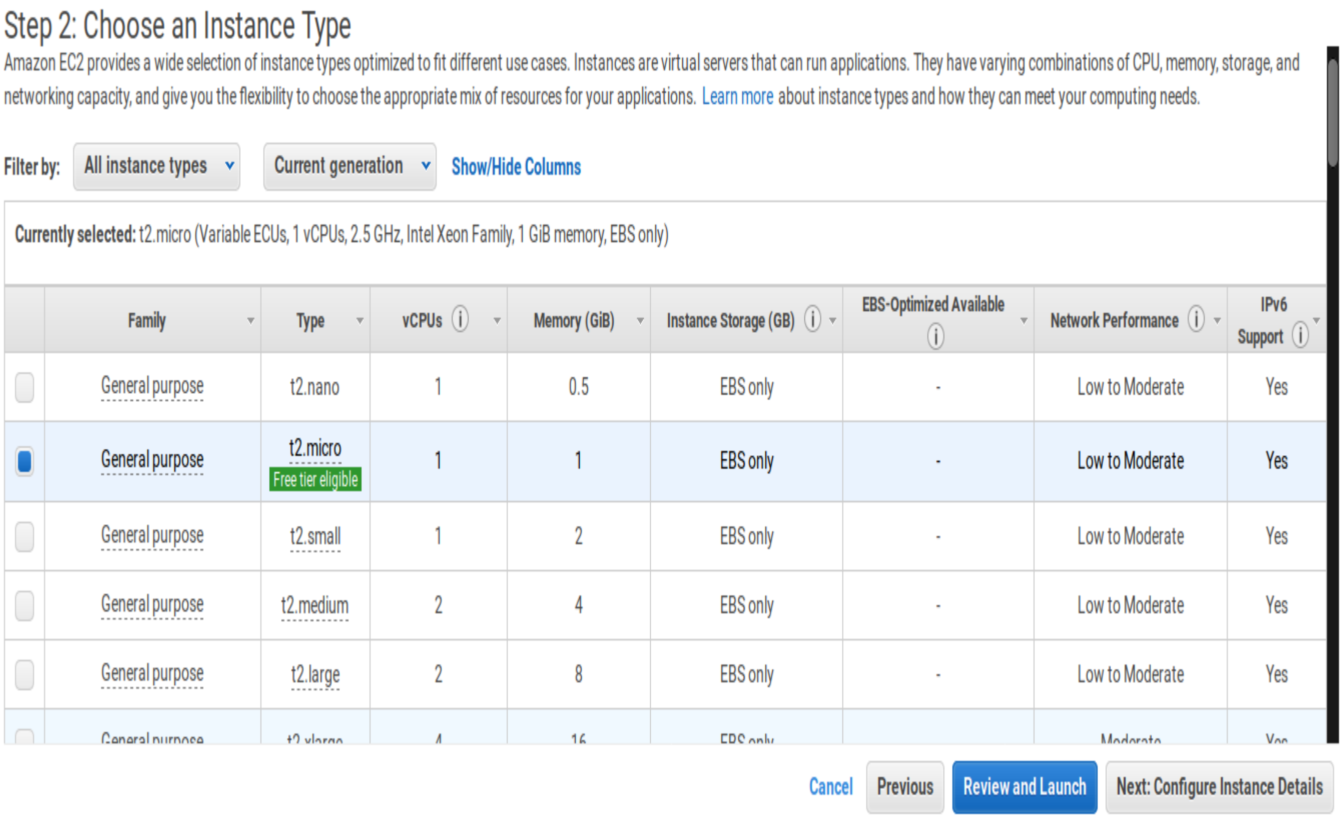
Community AMIs

☑ Free tier only ⓘ

Root device type: ebs   Virtualization type: hvm

SUSE Linux Free tier eligible	**SUSE Linux Enterprise Server 12 SP2 (HVM), SSD Volume Type** - ami-f990b69c SUSE Linux Enterprise Server 12 Service Pack 2 (HVM), EBS General Purpose (SSD) Volume Type. Public Cloud, Advanced Systems Management, Web and Scripting, and Legacy modules enabled. Root device type: ebs Virtualization type: hvm	Select 64-bit
Red Hat Free tier eligible	**Red Hat Enterprise Linux 7.4 (HVM), SSD Volume Type** - ami-cfdafaaa Red Hat Enterprise Linux version 7.4 (HVM), EBS General Purpose (SSD) Volume Type Root device type: ebs Virtualization type: hvm	Select 64-bit
Free tier eligible	**Ubuntu Server 16.04 LTS (HVM), SSD Volume Type** - ami-10547475 Ubuntu Server 16.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud /services). Root device type: ebs Virtualization type: hvm	Select 64-bit

5. Choose an Instance Type: Select General purpose "t2.micro". Click on the next button.



Step 2: Choose an Instance Type

Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. Learn more about instance types and how they can meet your computing needs.

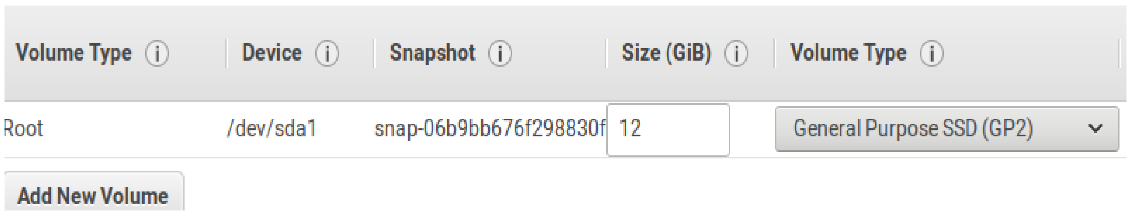
Filter by: All instance types ▾ Current generation ▾ Show/Hide Columns

Currently selected: t2.micro (Variable ECUs, 1 vCPUs, 2.5 GHz, Intel Xeon Family, 1 GiB memory, EBS only)

	Family	Type	vCPUs ⓘ	Memory (GiB)	Instance Storage (GB) ⓘ	EBS-Optimized Available ⓘ	Network Performance ⓘ	IPv6 Support ⓘ
☐	General purpose	t2.nano	1	0.5	EBS only	-	Low to Moderate	Yes
■	General purpose	t2.micro Free tier eligible	1	1	EBS only	-	Low to Moderate	Yes
☐	General purpose	t2.small	1	2	EBS only	-	Low to Moderate	Yes
☐	General purpose	t2.medium	2	4	EBS only	-	Low to Moderate	Yes
☐	General purpose	t2.large	2	8	EBS only	-	Low to Moderate	Yes
☐	General purpose	t2.xlarge	4	16	EBS only		Moderate	Yes

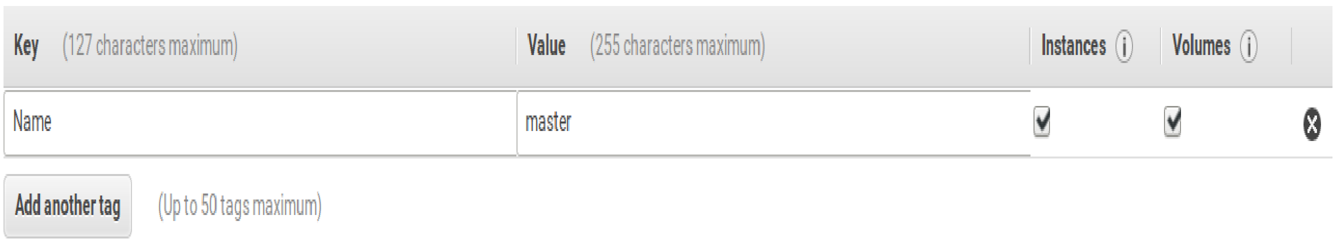
6. Configure Instance Details: Set number of instances to 3 and proceed to modify storage.

7. Add Storage: Change storage capacity from 8 Gib to 12 Gib. Click on Next button.



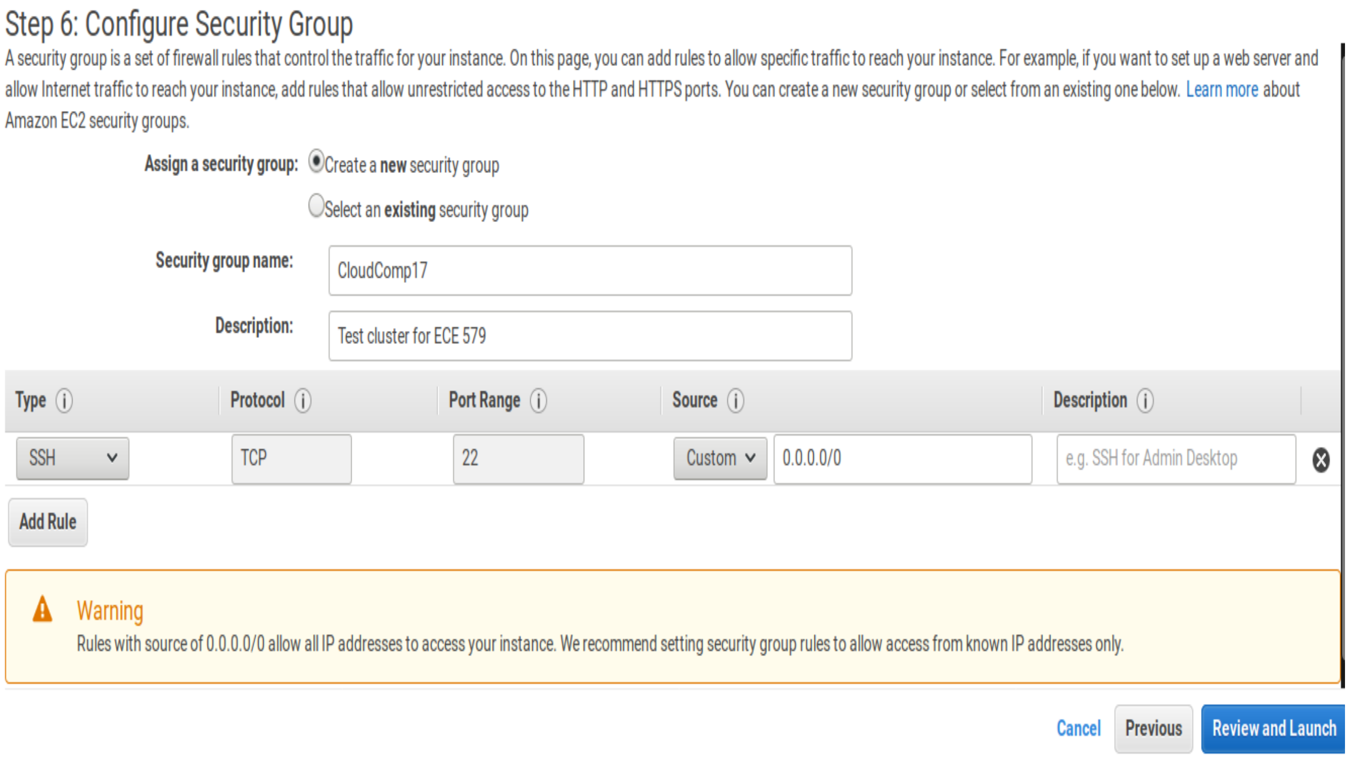
Volume Type ⓘ	Device ⓘ	Snapshot ⓘ	Size (GiB) ⓘ	Volume Type ⓘ
Root	/dev/sda1	snap-06b9bb676f298830f	12	General Purpose SSD (GP2)

8. Tag Instance: Type "Name" in *Key* and "master" in corresponding *Value*. Now we will configure a security group for this instance.



Key (127 characters maximum)	Value (255 characters maximum)	Instances	Volumes	
Name	master	✓	✓	⊗

9. Configure Security Group: Set name for your security group and provide a description. Set "SSH" as Type. Also, allow "HTTP" type as a new rule.



Step 6: Configure Security Group

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. Learn more about Amazon EC2 security groups.

Assign a security group: [✓ Create a **new** security group] Select an **existing** security group

Security group name: CloudComp17

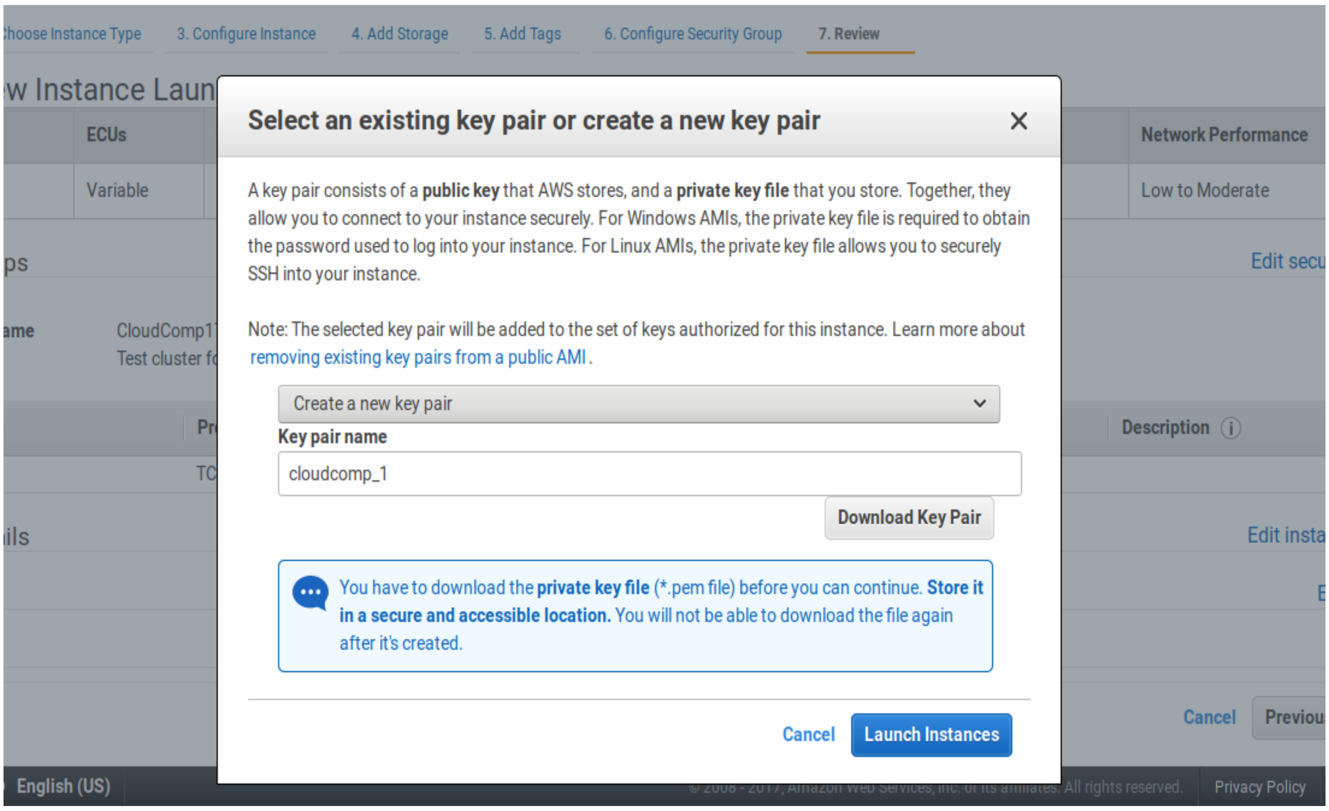
Description: Test cluster for ECE 579

Type	Protocol	Port Range	Source		Description
SSH	TCP	22	Custom	0.0.0.0/0	e.g. SSH for Admin Desktop

Warning
Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

10. Review the configuration details of your instance. Click on Launch button.

11. Create a new key pair for your instance and download it. Next, launch the instance.


Select an existing key pair or create a new key pair

A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.

Note: The selected key pair will be added to the set of keys authorized for this instance. Learn more about removing existing key pairs from a public AMI .

Create a new key pair

Key pair name

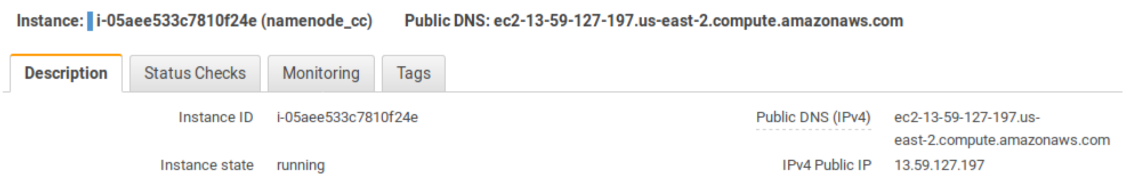
cloudcomp_1

You have to download the **private key file** (*.pem file) before you can continue. **Store it in a secure and accessible location.** You will not be able to download the file again after it's created.

Now you'll be able to see your launched instances in *Intances >> Running Instances*.

Your dashboard will have 3 running instances named **master**. You rename by clicking on it.

Copy public dns for your instance in a file. For example- an instance named "namenode_cc" has DNS- *ec2-13-59-127-197.us-......com*


Instance: i-05aee533c7810f24e (namenode_cc)   Public DNS: ec2-13-59-127-197.us-east-2.compute.amazonaws.com

Instance ID	i-05aee533c7810f24e	Public DNS (IPv4)	ec2-13-59-127-197.us-east-2.compute.amazonaws.com
Instance state	running	IPv4 Public IP	13.59.127.197

Logging into your ec2 Instances

We will use "ssh" for remote logging into your created instances. For illustration purposes, "**local**" will refer to your local machine console i.e your personal computer and for commands to be executed on ec2 instance/node console, "**node**" will be used.

1. Copy download key file to your ~/.ssh directory.

```
local$ cp ~/path/to/download/dir/your_key_name.pem ~/.ssh
```

2. Go to your ssh directory and find your key.

```
local$ cd ~/.ssh && ls -la # -a lists hidden files as well.
```

3. Change permissions for *your_key_name.pem* file to read and write for the user.

```
local$ sudo chmod 600 ~/.ssh/your_key_name.pem
```

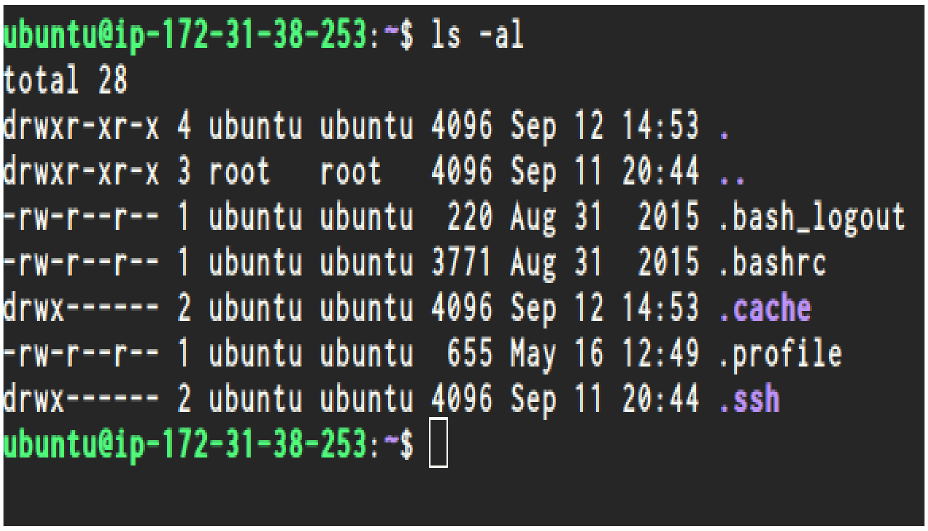
4. Connect to your ec2 instance using your *.pem file and instance Public DNS. Type "yes", if prompted.

```
local$ ssh -i ~/.ssh/your_key_name.pem ubuntu@your_node_public_dns
# This translates to-
local$ ssh -i ~/.ssh/your_key_name.pem ubuntu@ec2-13-59-127-197.us-......com
```

5. You will now be dropped into your instance/node console.

```
node$ sudo apt-get upgrade  # upgrade installed packages
node$ ls -al
```

6. Now list files in your home directory.



```
ubuntu@ip-172-31-38-253:~$ ls -al
total 28
drwxr-xr-x 4 ubuntu ubuntu 4096 Sep 12 14:53 .
drwxr-xr-x 3 root   root   4096 Sep 11 20:44 ..
-rw-r--r-- 1 ubuntu ubuntu  220 Aug 31  2015 .bash_logout
-rw-r--r-- 1 ubuntu ubuntu 3771 Aug 31  2015 .bashrc
drwx------ 2 ubuntu ubuntu 4096 Sep 12 14:53 .cache
-rw-r--r-- 1 ubuntu ubuntu  655 May 16 12:49 .profile
drwx------ 2 ubuntu ubuntu 4096 Sep 11 20:44 .ssh
ubuntu@ip-172-31-38-253:~$
```

Mount and Unmount

We will now mount and unmount drives of our ec2 instance.

Caution: *Do not write to a drive mounted on two distinct locations. This may cause disk failure.*

1. Check the drive name mounted on **MOUNTPOINT / (root)**.

```
node$ lsblk -lb # It should be "xvda1"
```

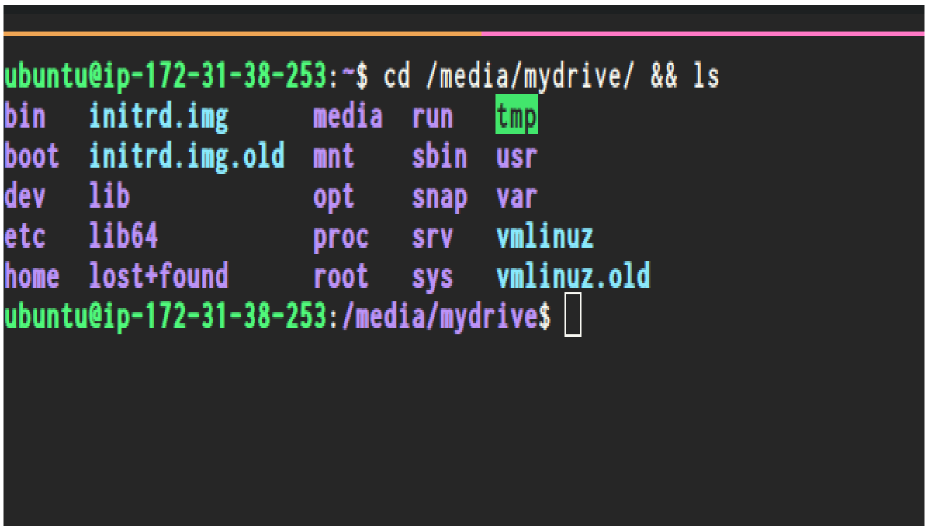
2. Make a folder named "mydrive" in /media to mount your current root directory on it.

```
node$ sudo mkdir -p /media/mydrive
node$ cd /media && ls -la # You should have a folder called mydrive
```

3. Mount your root drive to mydrive folder.

```
node$ sudo mount /dev/xvda1 /media/mydrive -t ext
node$ cd /media/mydrive && ls # Check mountpoint with $lsblk -lb
```

```
Console output should be-
```



```
ubuntu@ip-172-31-38-253:~$ cd /media/mydrive/ && ls
bin   initrd.img      media  run   tmp
boot  initrd.img.old  mnt    sbin  usr
dev   lib             opt    snap  var
etc   lib64           proc   srv   vmlinuz
home  lost+found      root   sys   vmlinuz.old
ubuntu@ip-172-31-38-253:/media/mydrive$
```

4. Get out of your /media/mydrive directory otherwise it'll throw error: "Volume is busy" while unmounting.

```
node$ cd ~
node$ sudo umount /media/mydrive
node$ cd /media/mydrive && ls # Empty directory
```

5. Type `exit` to leave ec2 instance/node console.

EC2 direct login

Remembering public DNS of your node can be pain. We will make a configuration file for our node specifying the key and public DNS for connection.

1. Run the `find_config.sh` .

```
local$ bash find_config.sh
```

2. Go into your ssh directory and edit `config` as per the following rules.

```
# Edit your config
# Replace the code with your key and ec2 public DNS
#
# It should look like this-
# Host namenode_cc
#     HostName  ec2-13-59-127-197.us-east-2.compute.amazonaws.com
#     User ubuntu
#     IdentityFile ~/.ssh/cloudcomp_1.pem

Host name_of_your_node
    HostName your_ec2_public_dns
    User ubuntu
    IdentityFile ~/.ssh/your_key_name.pem
```

3. Now you can directly login to your node.

```
local$ ssh name_of_your_node # Example- ssh namenode_cc
```

Make a webpage for your server

1. Login into your node. Install apache2 and my-sql-server

```
node$ sudo apt-get install apache2 mysql-server
```

2. Change the HTML content of the webpage.

```
node$ cd /var/www/html
node$ sudo vi index.html
```

```
# Replace index.html with the following code-

<HTML>
   <HEAD>
      <TITLE>
         A Small Hello
      </TITLE>
   </HEAD>
<BODY>
   <H1>Welcome to Cloud Computing!</H1>
   <P>You are in lecture 2!</P>
</BODY>
</HTML>
```

3. Now enter your public DNS in the url bar and you should see a greeting page.

Working with AWS Command Line Interface

You can manage your ec2 instances directly from your console. You can read about instance states and applicable charges here You can follow this guide to install aws-cli according to your OS environment.

- **Getting Access key ID**:

    i. Sign in to IAM console to get Access key ID.
    ii. Go to User tab on left and add user.
    iii. Enter username and select *Programmatic access* and *AWS Management Console access*.
    iv. Enter a custom password.
    v. De-select *Require password reset*.
    vi. Proceed and create a group.
    vii. Provide AdministratorAcess to the group.
    viii. Review to create user.
    ix. Copy Access Key ID and Secret Key.

- **Configure aws-cli**: On your local machine console type `aws configure`.

```
AWS Access Key ID [None]: AKIAI44QH8DHBEXAMPLE
AWS Secret Access Key [None]: je7MtGbClwBF/2Zp9Utk/h3yCo8nvbEXAMPLEKEY
Default region name [None]: us-east-2 # region as in your public DNS
Default output format [None]: text
```

- **Useful aws-cli commands**:

    - Start: `aws ec2 start-instances --instance-ids <your instance id>`
    - Stop: `aws ec2 stop-instances --instance-ids <your instance id>`
    - Reboot: `aws ec2 reboot-instances --instance-ids <your instance id>`
    - Terminate: `aws ec2 terminate-instances --instance-ids <your instance id>`

    Note: Instance ID will be specific for each instance and can be found in ec2 dashboard.

    Warning: A terminated instance cannot be de-terminated. Read AWS charging policy before using a paid instance.