Lab Assignment-2

Steps to achieve the Load Balancing:

- Install AWS CLI as we can create EC2 through command line and run scripts based on CLI. Prerequisite of Python 2.7+ is required.

```
$ pip install --upgrade --user awscli
```

- Check if aws is installed properly

```
$ aws --version
aws-cli/1.11.76 Python/2.7.10 Darwin/16.5.0 botocore/1.5.39
```

- Configure AWS by providing access key , security key and region

```
$ aws configure
AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/
bPxRfiCYEXAMPLEKEY
Default region name [None]: us-west-2
Default output format [None]: json
```

- Made Script to launch multiple instance of EC2 and shown in video and have kept script file in folder.

Details of Script explained:-

We created an array to hold names of the EC2 instance named instanceNames which can be assigned when instance is launched.

- As we want to create 5 instances and set some properties on those instance I am creating one instance at a time and setting those properties.

```
while [ $count -lt 5 ]
do
```

```
echo ${instanceNames[count]}
```

```
instanceId=$(aws ec2 run-instances --image-id
ami-7172b611 --security-group-ids sg-c7e225be --count 1
--instance-type t2.micro --key-name administrator-key-
pair-useast2 --block-device-mappings "[{\"DeviceName\":
```

```
\"/dev/xvda\",\"Ebs\":{\"VolumeSize\":
8,\"DeleteOnTermination\":true}}]" --user-data install-
nginx.sh --query 'Instances[0].InstanceId'
| sed 's/.//;s/.$//')
echo $instanceId
```

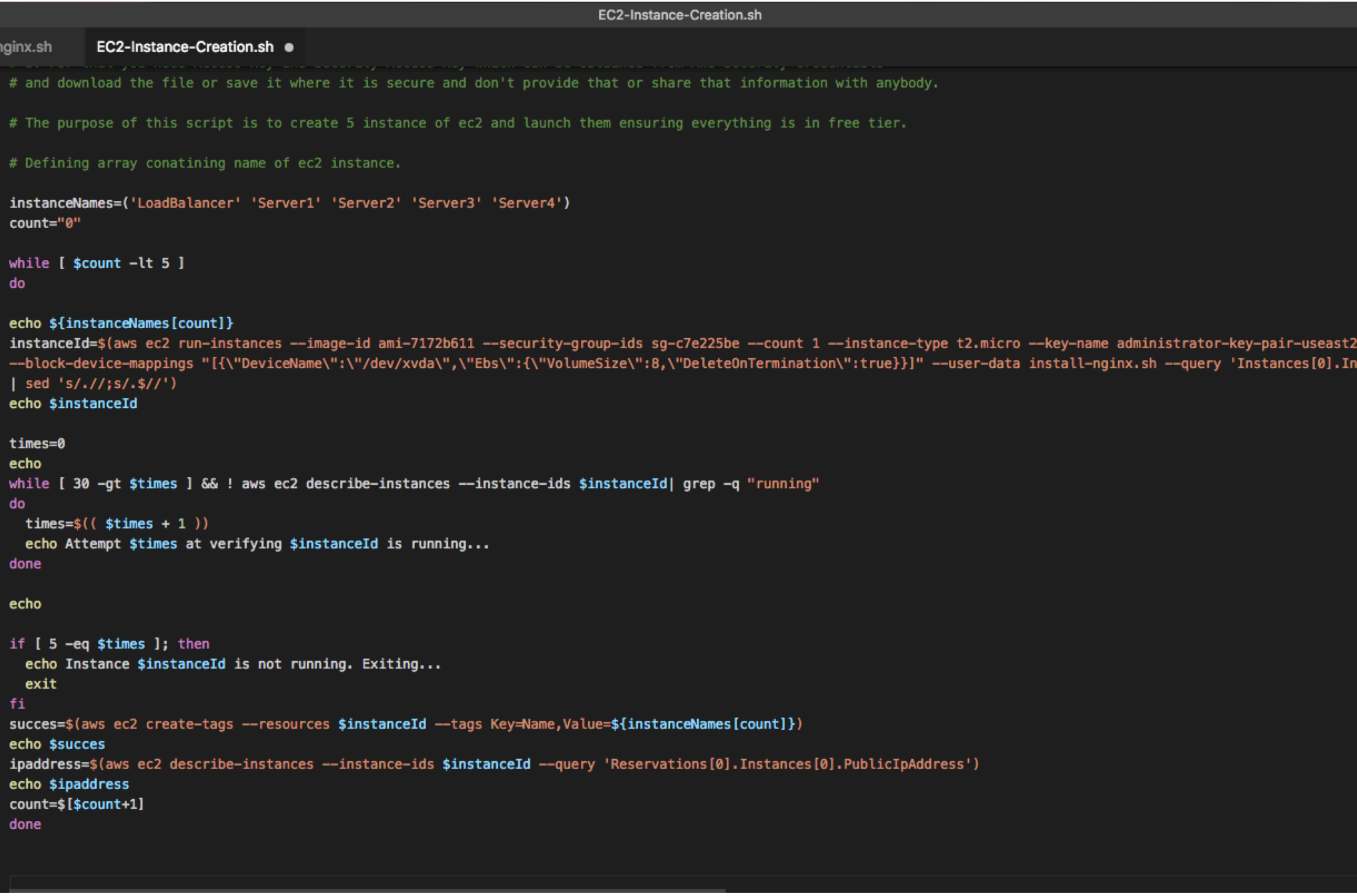
- Here
  - `aws ec2` in the command says we are going to execute aws command on ec2.
  - `run-instances` allows to launch the instance
  - `—image-id` describes the image of operating system used for launching and we have selected Linux free tier image.
  - `—Security-group -` allows us to bind the instance with the security rules to acess that instance and sg-c7e22be is the id of the security group we created earlier.
  - `—Count -` number of instances to create with the same property. Here we could have directly written 5 to create 5 such instances but I have to set some properties mentioned later using each instance id.Therefore its 1.
  - `—instance-type-` instance type is used for providing cpu performance we have selected t2.micro which falls under free tier.
  - `—Key-name` - allows us to associate private key to recognize the instance is accused by authorized user allocating key name that we have created earlier.
  - `—block-device-mapping-` allows us to allocate additional space with the instance while running. it provides various options we have used to set name and allocate 8 gb as minimum of 8 GB we have to allocate to terminate when the instance is terminated property.
  - `—user-data` - this property allows us to send additional file or script to be executed when the instance is launched. So on boot it will execute in this case install-nginx.sh which has commands to install nginx after updating yum which can be seen later in figure.
  - `—query` - returns the instance id which got started in the following format - “i-6238736989723”
  - we are using sed to remove 1st and last character of string returned to remove “ “ from the instance id.

```
times=0
echo
while [ 30 -gt $times ] && ! aws ec2 describe-
instances --instance-ids $instanceId| grep -q
"running"
do
  times=$(( $times + 1 ))
  echo Attempt $times at verifying $instanceId is
running...
done

echo

if [ 5 -eq $times ]; then
  echo Instance $instanceId is not running.
Exiting...
  exit
fi
succes=$(aws ec2 create-tags --resources $instanceId
--tags Key=Name,Value=${instanceNames[count]})
echo $succes
```

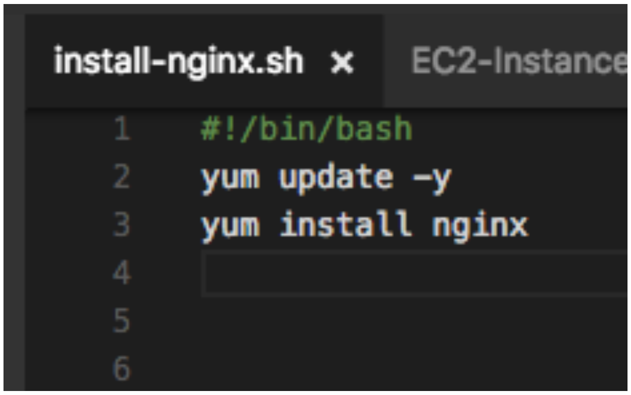
- Here we are checking by using aws ec2 describe-instance command that instance has started running as it takes few minutes o allocate resources and start up just like linux and if its running we are naming the instance using tags.

```
ipaddress=$(aws ec2 describe-instances --instance-ids
$instanceId --query
'Reservations[0].Instances[0].PublicIpAddress')
echo $ipaddress
count=$[$count+1]
done
```

- Above code just returns the ipaddress of the instance which can be useful for interacting with instance using ssh.



```
# and download the file or save it where it is secure and don't provide that or share that information with anybody.

# The purpose of this script is to create 5 instance of ec2 and launch them ensuring everything is in free tier.

# Defining array conatining name of ec2 instance.

instanceNames=('LoadBalancer' 'Server1' 'Server2' 'Server3' 'Server4')
count="0"

while [ $count -lt 5 ]
do

echo ${instanceNames[count]}
instanceId=$(aws ec2 run-instances --image-id ami-7172b611 --security-group-ids sg-c7e225be --count 1 --instance-type t2.micro --key-name administrator-key-pair-useast2
--block-device-mappings "[{\"DeviceName\":\"/dev/xvda\",\"Ebs\":{\"VolumeSize\":8,\"DeleteOnTermination\":true}}]" --user-data install-nginx.sh --query 'Instances[0].In
| sed 's/.//;s/.$//')
echo $instanceId

times=0
echo
while [ 30 -gt $times ] && ! aws ec2 describe-instances --instance-ids $instanceId| grep -q "running"
do
  times=$(( $times + 1 ))
  echo Attempt $times at verifying $instanceId is running...
done

echo

if [ 5 -eq $times ]; then
  echo Instance $instanceId is not running. Exiting...
  exit
fi
succes=$(aws ec2 create-tags --resources $instanceId --tags Key=Name,Value=${instanceNames[count]})
echo $succes
ipaddress=$(aws ec2 describe-instances --instance-ids $instanceId --query 'Reservations[0].Instances[0].PublicIpAddress')
echo $ipaddress
count=$[$count+1]
done
```



```
#!/bin/bash
yum update -y
yum install nginx
```

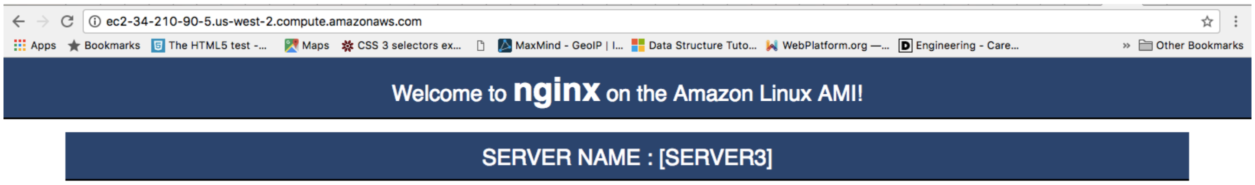
Figure 1.1 EC2-instance-creation

Figure 1.2 install.nginx.sh

- Verified installation of nginx by opening public dos of every instance like follows:

- Updated index.html as follows to display server name

```
[root@ip-172-31-32-234 ~]# cd /usr/share/nginx/html
[root@ip-172-31-32-234 html]# vi index.html
```



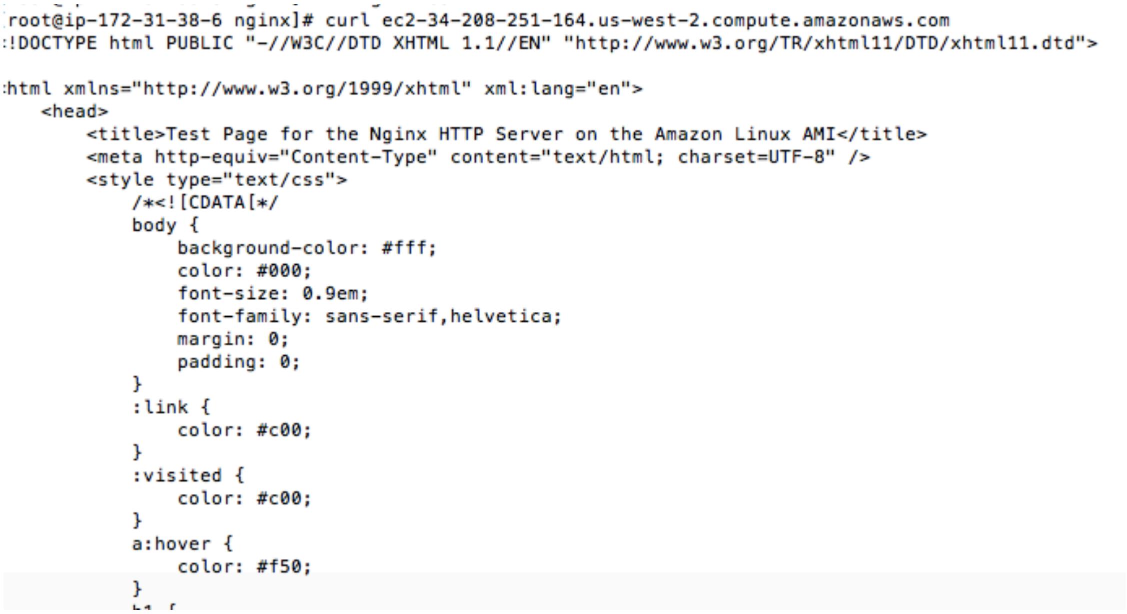
Welcome to **nginx** on the Amazon Linux AMI!

SERVER NAME : [SERVER3]

- Performed same procedure for Server1, Server2, Server,3 and Server4
- In Loadbalancer updated the nginx.conf file as follows and reloaded nginx server

```
[root@ip-172-31-38-6 ~]# /etc/init.d/nginx status
nginx is stopped
[root@ip-172-31-38-6 ~]# /etc/init.d/nginx start
Starting nginx:                                            [  OK  ]
[root@ip-172-31-38-6 ~]# /etc/init.d/nginx status
nginx (pid  2586) is running...
[root@ip-172-31-38-6 ~]# vi /etc/nginx/nginx.conf
[root@ip-172-31-38-6 ~]# /etc/init.d/nginx reload
Reloading nginx:                                           [  OK  ]
```

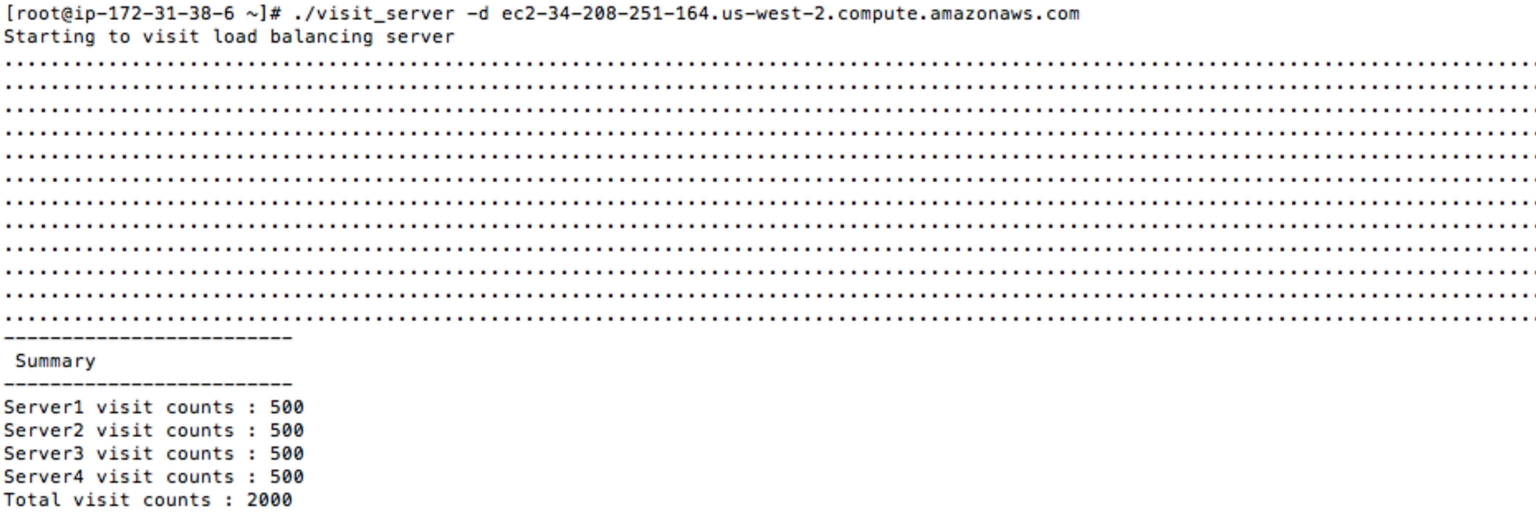
- Tested Curl ec2-34-208-251-164.us-west-2.compute.amazonaws.com
- It displays on console the index page of server 1, on 1st request and distribute request between 4 servers in round robin fashion.



```
root@ip-172-31-38-6 nginx]# curl ec2-34-208-251-164.us-west-2.compute.amazonaws.com
:!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">

:html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
    <head>
        <title>Test Page for the Nginx HTTP Server on the Amazon Linux AMI</title>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
        <style type="text/css">
            /*<![CDATA[*/
            body {
                background-color: #fff;
                color: #000;
                font-size: 0.9em;
                font-family: sans-serif,helvetica;
                margin: 0;
                padding: 0;
            }
            :link {
                color: #c00;
            }
            :visited {
                color: #c00;
            }
            a:hover {
                color: #f50;
            }
```

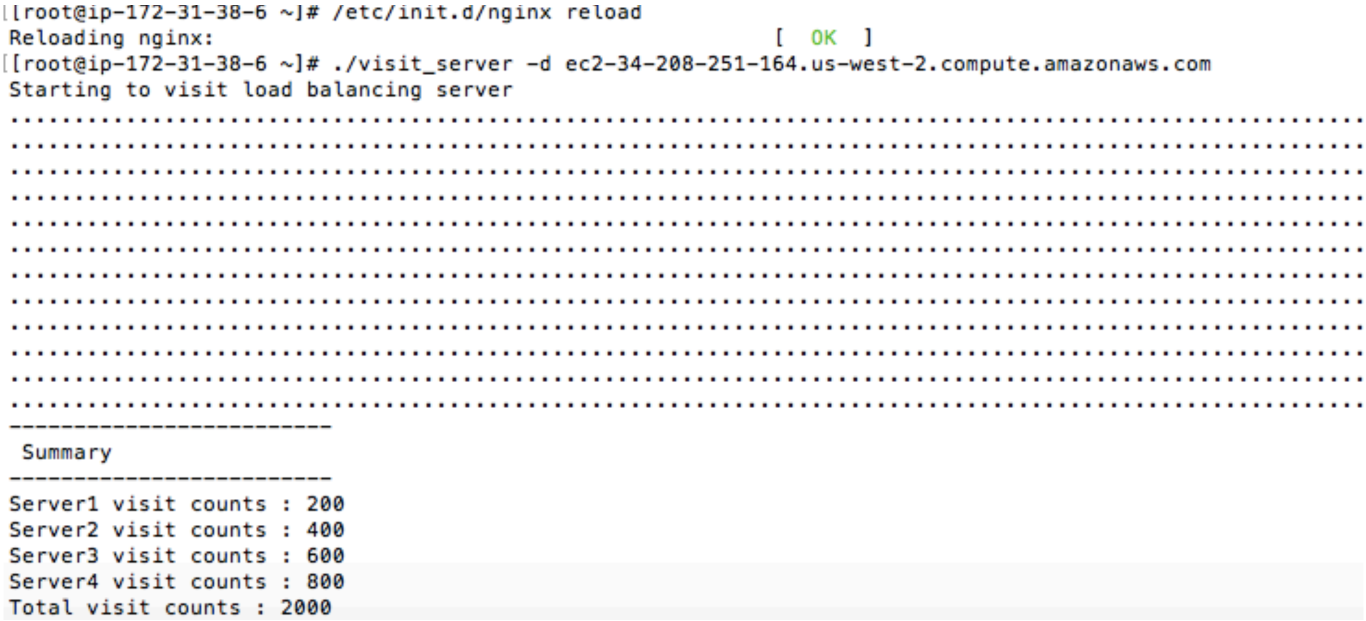
- Created and saved visit_server file on LoadBalancer instance and changes the permissions to executable to execute that file.

```
[root@ip-172-31-38-6 ~]# vi visit_server
[root@ip-172-31-38-6 ~]# chmod +x visit_server
```



```
[root@ip-172-31-38-6 ~]# ./visit_server -d ec2-34-208-251-164.us-west-2.compute.amazonaws.com
Starting to visit load balancing server
.....................................................................................................................................
.....................................................................................................................................
.....................................................................................................................................
.....................................................................................................................................
.....................................................................................................................................
.....................................................................................................................................
.....................................................................................................................................
.....................................................................................................................................
.....................................................................................................................................
.....................................................................................................................................
.....................................................................................................................................
.....................................................................................................................................
-------------------------
 Summary
-------------------------
Server1 visit counts : 500
Server2 visit counts : 500
Server3 visit counts : 500
Server4 visit counts : 500
Total visit counts : 2000
```

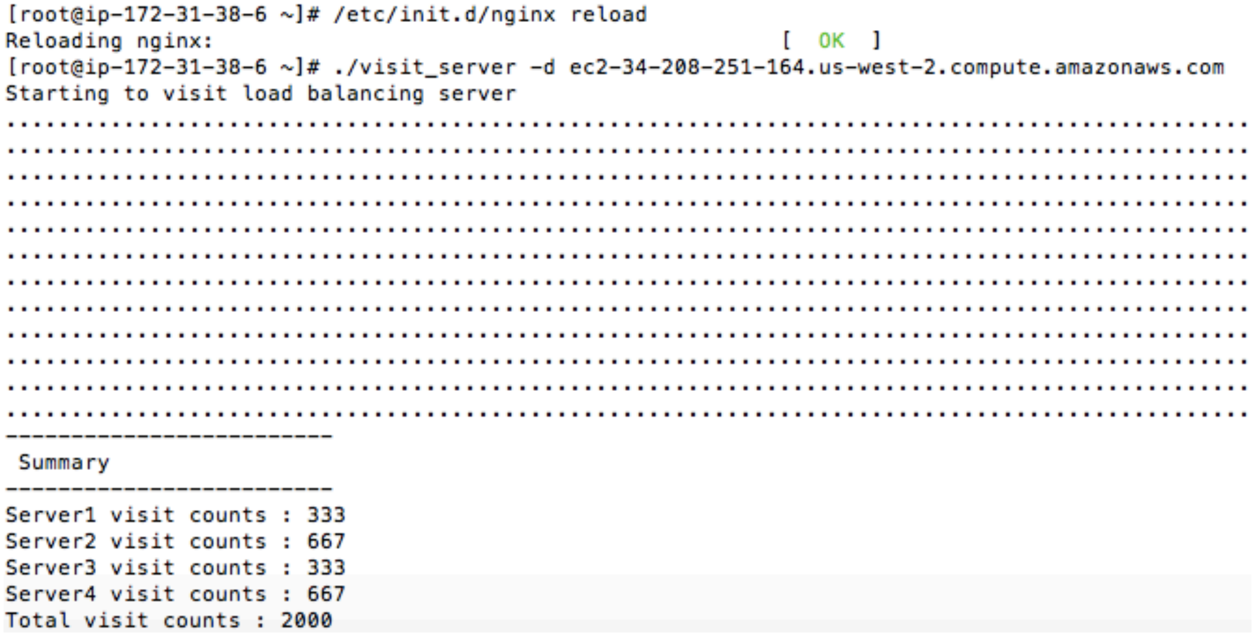
Above is scenario 1 with weight equally distributed 1 to all servers



```
[root@ip-172-31-38-6 ~]# /etc/init.d/nginx reload
Reloading nginx:                                       [  OK  ]
[root@ip-172-31-38-6 ~]# ./visit_server -d ec2-34-208-251-164.us-west-2.compute.amazonaws.com
Starting to visit load balancing server
.........................................................................................................
.........................................................................................................
.........................................................................................................
.........................................................................................................
.........................................................................................................
.........................................................................................................
.........................................................................................................
.........................................................................................................
.........................................................................................................
.........................................................................................................
.........................................................................................................
.........................................................................................................
-------------------------
 Summary
-------------------------
Server1 visit counts : 200
Server2 visit counts : 400
Server3 visit counts : 600
Server4 visit counts : 800
Total visit counts : 2000
```

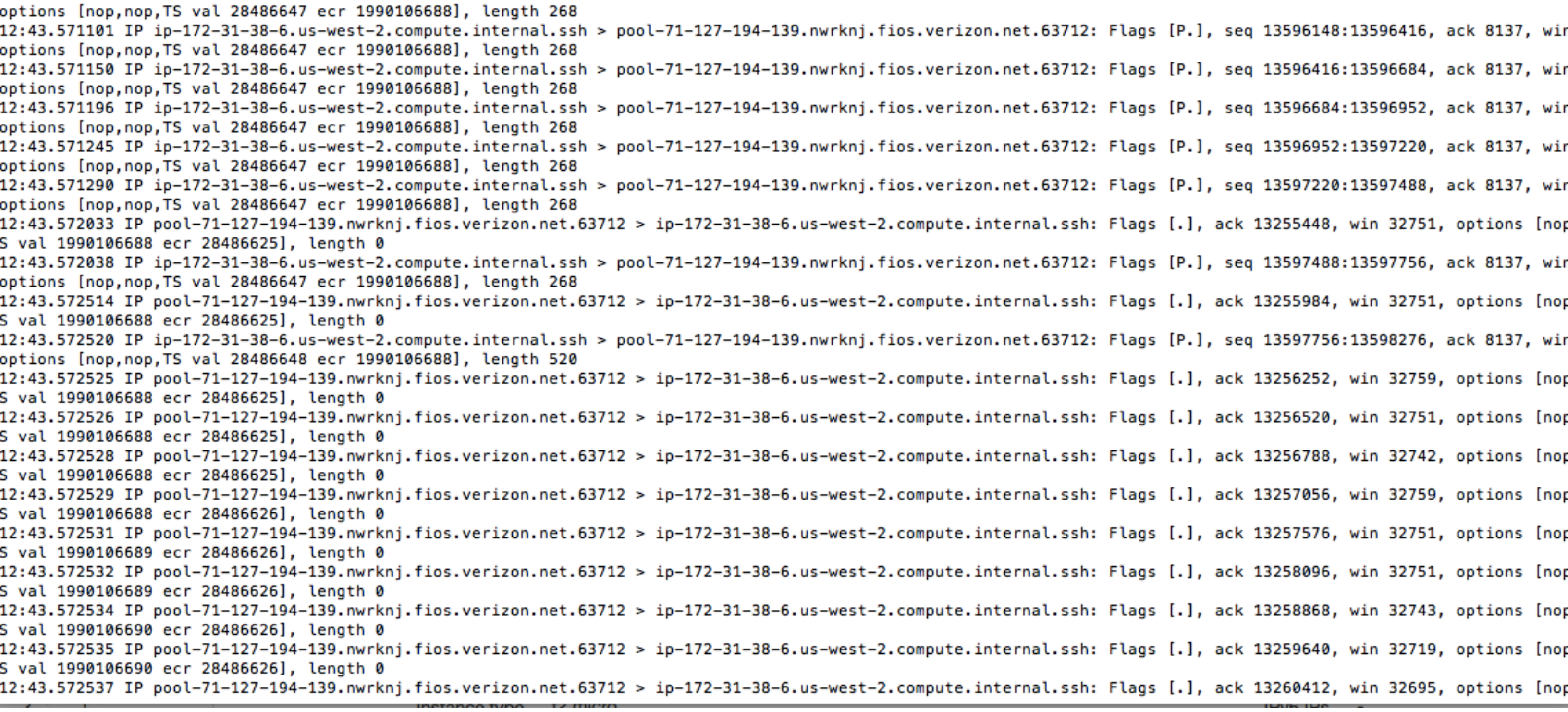
Above is Scenario 2 with weight distributed as 1,2 3,4 in nginx.conf

Below is the Scenario 3 with weight distributed as 1,2,1,2



```
[root@ip-172-31-38-6 ~]# /etc/init.d/nginx reload
Reloading nginx:                                           [  OK  ]
[root@ip-172-31-38-6 ~]# ./visit_server -d ec2-34-208-251-164.us-west-2.compute.amazonaws.com
Starting to visit load balancing server
...............................................................................................
...............................................................................................
...............................................................................................
...............................................................................................
...............................................................................................
...............................................................................................
...............................................................................................
...............................................................................................
...............................................................................................
...............................................................................................
...............................................................................................
...............................................................................................
-------------------------
 Summary
-------------------------
Server1 visit counts : 333
Server2 visit counts : 667
Server3 visit counts : 333
Server4 visit counts : 667
Total visit counts : 2000
```

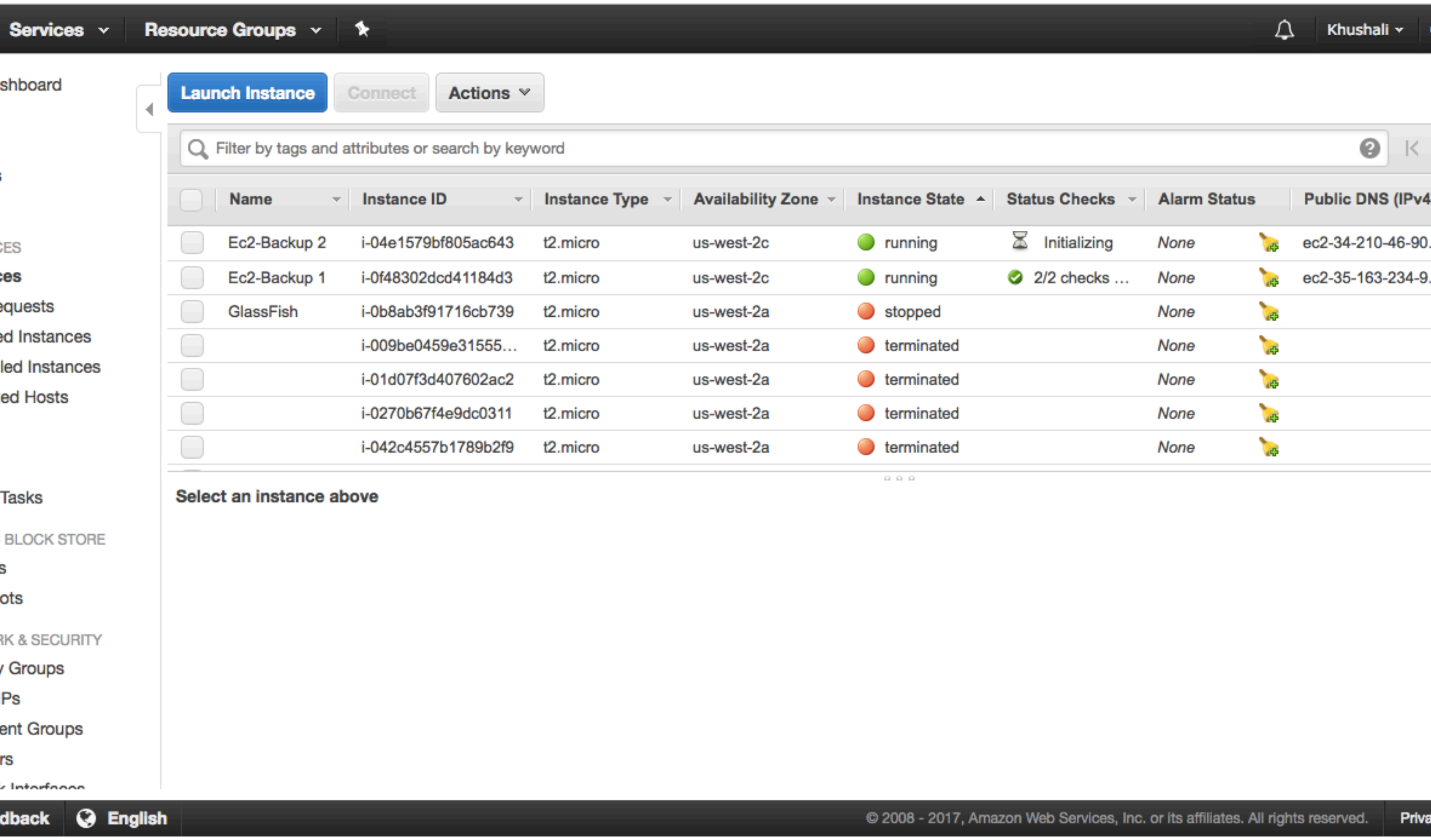
For tcp dump installed tcpdump using yum and after running tcp dump and infinite loop of ack– acknowledgements and seq– sequence of packets in response to ack were found which are going on in an endless loop



```
options [nop,nop,TS val 28486647 ecr 1990106688], length 268
12:43.571101 IP ip-172-31-38-6.us-west-2.compute.internal.ssh > pool-71-127-194-139.nwrknj.fios.verizon.net.63712: Flags [P.], seq 13596148:13596416, ack 8137, wir
options [nop,nop,TS val 28486647 ecr 1990106688], length 268
12:43.571150 IP ip-172-31-38-6.us-west-2.compute.internal.ssh > pool-71-127-194-139.nwrknj.fios.verizon.net.63712: Flags [P.], seq 13596416:13596684, ack 8137, wir
options [nop,nop,TS val 28486647 ecr 1990106688], length 268
12:43.571196 IP ip-172-31-38-6.us-west-2.compute.internal.ssh > pool-71-127-194-139.nwrknj.fios.verizon.net.63712: Flags [P.], seq 13596684:13596952, ack 8137, wir
options [nop,nop,TS val 28486647 ecr 1990106688], length 268
12:43.571245 IP ip-172-31-38-6.us-west-2.compute.internal.ssh > pool-71-127-194-139.nwrknj.fios.verizon.net.63712: Flags [P.], seq 13596952:13597220, ack 8137, wir
options [nop,nop,TS val 28486647 ecr 1990106688], length 268
12:43.571290 IP ip-172-31-38-6.us-west-2.compute.internal.ssh > pool-71-127-194-139.nwrknj.fios.verizon.net.63712: Flags [P.], seq 13597220:13597488, ack 8137, wir
options [nop,nop,TS val 28486647 ecr 1990106688], length 268
12:43.572033 IP pool-71-127-194-139.nwrknj.fios.verizon.net.63712 > ip-172-31-38-6.us-west-2.compute.internal.ssh: Flags [.], ack 13255448, win 32751, options [nop
S val 1990106688 ecr 28486625], length 0
12:43.572038 IP ip-172-31-38-6.us-west-2.compute.internal.ssh > pool-71-127-194-139.nwrknj.fios.verizon.net.63712: Flags [P.], seq 13597488:13597756, ack 8137, wir
options [nop,nop,TS val 28486647 ecr 1990106688], length 268
12:43.572514 IP pool-71-127-194-139.nwrknj.fios.verizon.net.63712 > ip-172-31-38-6.us-west-2.compute.internal.ssh: Flags [.], ack 13255984, win 32751, options [nop
S val 1990106688 ecr 28486625], length 0
12:43.572520 IP ip-172-31-38-6.us-west-2.compute.internal.ssh > pool-71-127-194-139.nwrknj.fios.verizon.net.63712: Flags [P.], seq 13597756:13598276, ack 8137, wir
options [nop,nop,TS val 28486648 ecr 1990106688], length 520
12:43.572525 IP pool-71-127-194-139.nwrknj.fios.verizon.net.63712 > ip-172-31-38-6.us-west-2.compute.internal.ssh: Flags [.], ack 13256252, win 32759, options [nop
S val 1990106688 ecr 28486625], length 0
12:43.572526 IP pool-71-127-194-139.nwrknj.fios.verizon.net.63712 > ip-172-31-38-6.us-west-2.compute.internal.ssh: Flags [.], ack 13256520, win 32751, options [nop
S val 1990106688 ecr 28486625], length 0
12:43.572528 IP pool-71-127-194-139.nwrknj.fios.verizon.net.63712 > ip-172-31-38-6.us-west-2.compute.internal.ssh: Flags [.], ack 13256788, win 32742, options [nop
S val 1990106688 ecr 28486625], length 0
12:43.572529 IP pool-71-127-194-139.nwrknj.fios.verizon.net.63712 > ip-172-31-38-6.us-west-2.compute.internal.ssh: Flags [.], ack 13257056, win 32759, options [nop
S val 1990106688 ecr 28486626], length 0
12:43.572531 IP pool-71-127-194-139.nwrknj.fios.verizon.net.63712 > ip-172-31-38-6.us-west-2.compute.internal.ssh: Flags [.], ack 13257576, win 32751, options [nop
S val 1990106689 ecr 28486626], length 0
12:43.572532 IP pool-71-127-194-139.nwrknj.fios.verizon.net.63712 > ip-172-31-38-6.us-west-2.compute.internal.ssh: Flags [.], ack 13258096, win 32751, options [nop
S val 1990106689 ecr 28486626], length 0
12:43.572534 IP pool-71-127-194-139.nwrknj.fios.verizon.net.63712 > ip-172-31-38-6.us-west-2.compute.internal.ssh: Flags [.], ack 13258868, win 32743, options [nop
S val 1990106690 ecr 28486626], length 0
12:43.572535 IP pool-71-127-194-139.nwrknj.fios.verizon.net.63712 > ip-172-31-38-6.us-west-2.compute.internal.ssh: Flags [.], ack 13259640, win 32719, options [nop
S val 1990106690 ecr 28486626], length 0
12:43.572537 IP pool-71-127-194-139.nwrknj.fios.verizon.net.63712 > ip-172-31-38-6.us-west-2.compute.internal.ssh: Flags [.], ack 13260412, win 32695, options [nop
```

In image it can be seen every set responds to same ack number like pool–71–127–194–139 packets to a particular ack can be gathered by arranging sequence number in ascending order.

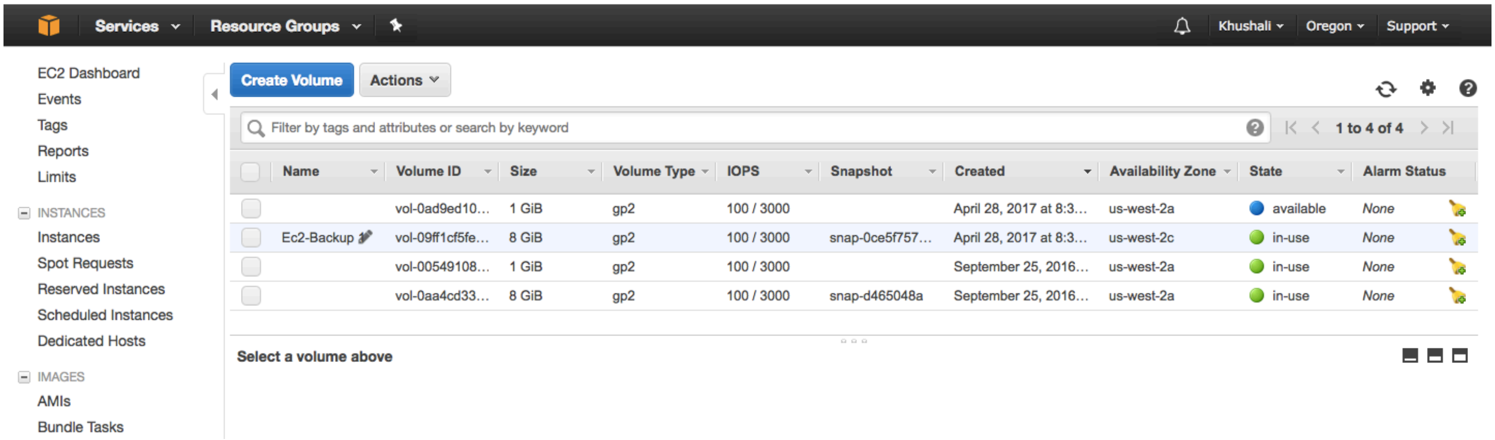
```
Additional step:-
1) Created a new instance named EC2-backup 1 where we are going to
   attach a new EBS and mount on this instance and unmount and mount
   on EC2-Backup 2
```



Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4
Ec2-Backup 2	i-04e1579bf805ac643	t2.micro	us-west-2c	● running	⌛ Initializing	None	ec2-34-210-46-90.
Ec2-Backup 1	i-0f48302dcd41184d3	t2.micro	us-west-2c	● running	✓ 2/2 checks ...	None	ec2-35-163-234-9.
GlassFish	i-0b8ab3f91716cb739	t2.micro	us-west-2a	● stopped		None	
	i-009be0459e31555...	t2.micro	us-west-2a	● terminated		None	
	i-01d07f3d407602ac2	t2.micro	us-west-2a	● terminated		None	
	i-0270b67f4e9dc0311	t2.micro	us-west-2a	● terminated		None	
	i-042c4557b1789b2f9	t2.micro	us-west-2a	● terminated		None	

Select an instance above

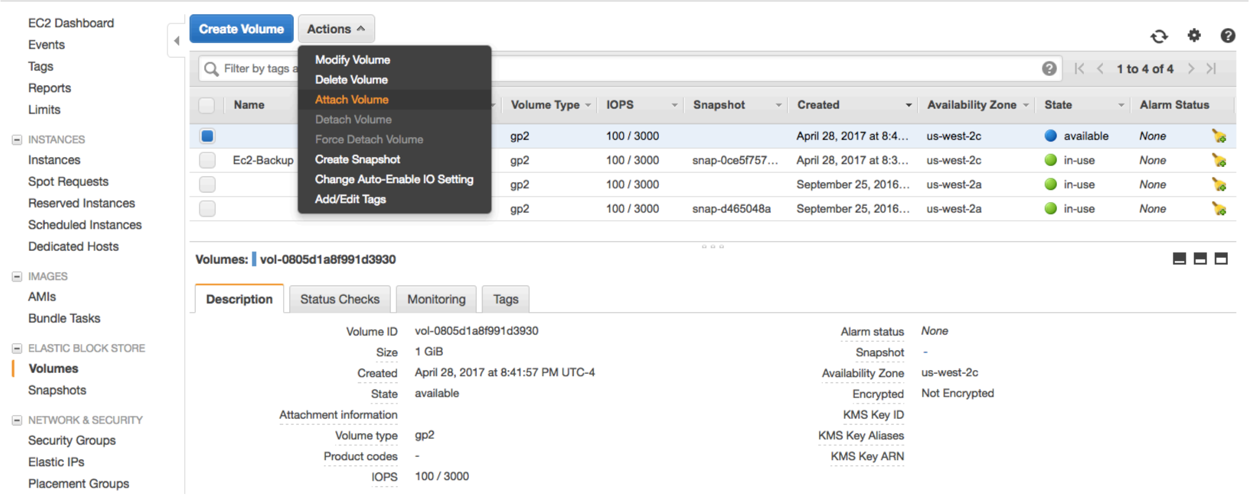
```
2) Created a new volume which is not attached to any instance which is
shown in blue available in stats.
```



Name	Volume ID	Size	Volume Type	IOPS	Snapshot	Created	Availability Zone	State	Alarm Status
	vol-0ad9ed10...	1 GiB	gp2	100 / 3000		April 28, 2017 at 8:3...	us-west-2a	● available	None
Ec2-Backup	vol-09ff1cf5fe...	8 GiB	gp2	100 / 3000	snap-0ce5f757...	April 28, 2017 at 8:3...	us-west-2c	● in-use	None
	vol-00549108...	1 GiB	gp2	100 / 3000		September 25, 2016...	us-west-2a	● in-use	None
	vol-0aa4cd33...	8 GiB	gp2	100 / 3000	snap-d465048a	September 25, 2016...	us-west-2a	● in-use	None

Select a volume above

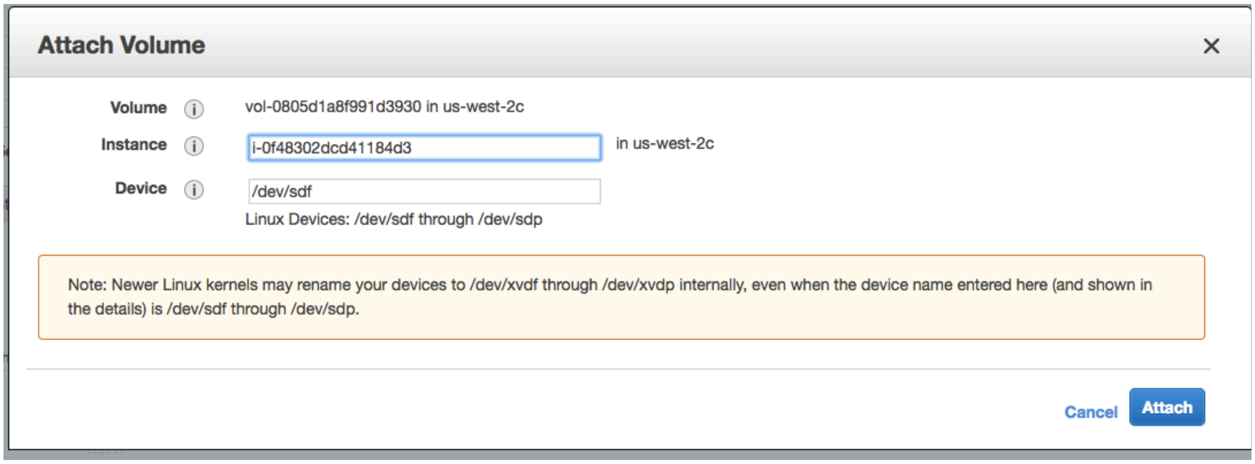
3) Attaching the volume to EC2-Backup 1



	Name	Volume Type	IOPS	Snapshot	Created	Availability Zone	State	Alarm Status
☑		gp2	100 / 3000		April 28, 2017 at 8:4…	us-west-2c	● available	None
□	Ec2-Backup	gp2	100 / 3000	snap-0ce5f757…	April 28, 2017 at 8:3…	us-west-2c	● in-use	None
□		gp2	100 / 3000		September 25, 2016…	us-west-2a	● in-use	None
□		gp2	100 / 3000	snap-d465048a	September 25, 2016…	us-west-2a	● in-use	None

Volumes: vol-0805d1a8f991d3930

Volume ID	vol-0805d1a8f991d3930	Alarm status	None
Size	1 GiB	Snapshot	-
Created	April 28, 2017 at 8:41:57 PM UTC-4	Availability Zone	us-west-2c
State	available	Encrypted	Not Encrypted
Attachment information		KMS Key ID	
Volume type	gp2	KMS Key Aliases	
Product codes	-	KMS Key ARN	
IOPS	100 / 3000		



Volume: vol-0805d1a8f991d3930 in us-west-2c

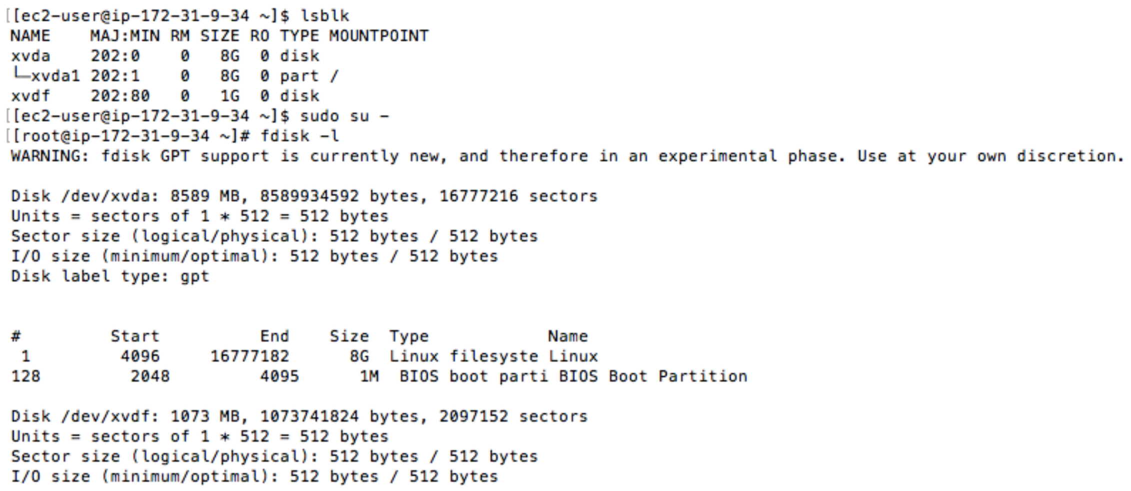
Instance: i-0f48302dcd41184d3 in us-west-2c

Device: /dev/sdf

Linux Devices: /dev/sdf through /dev/sdp

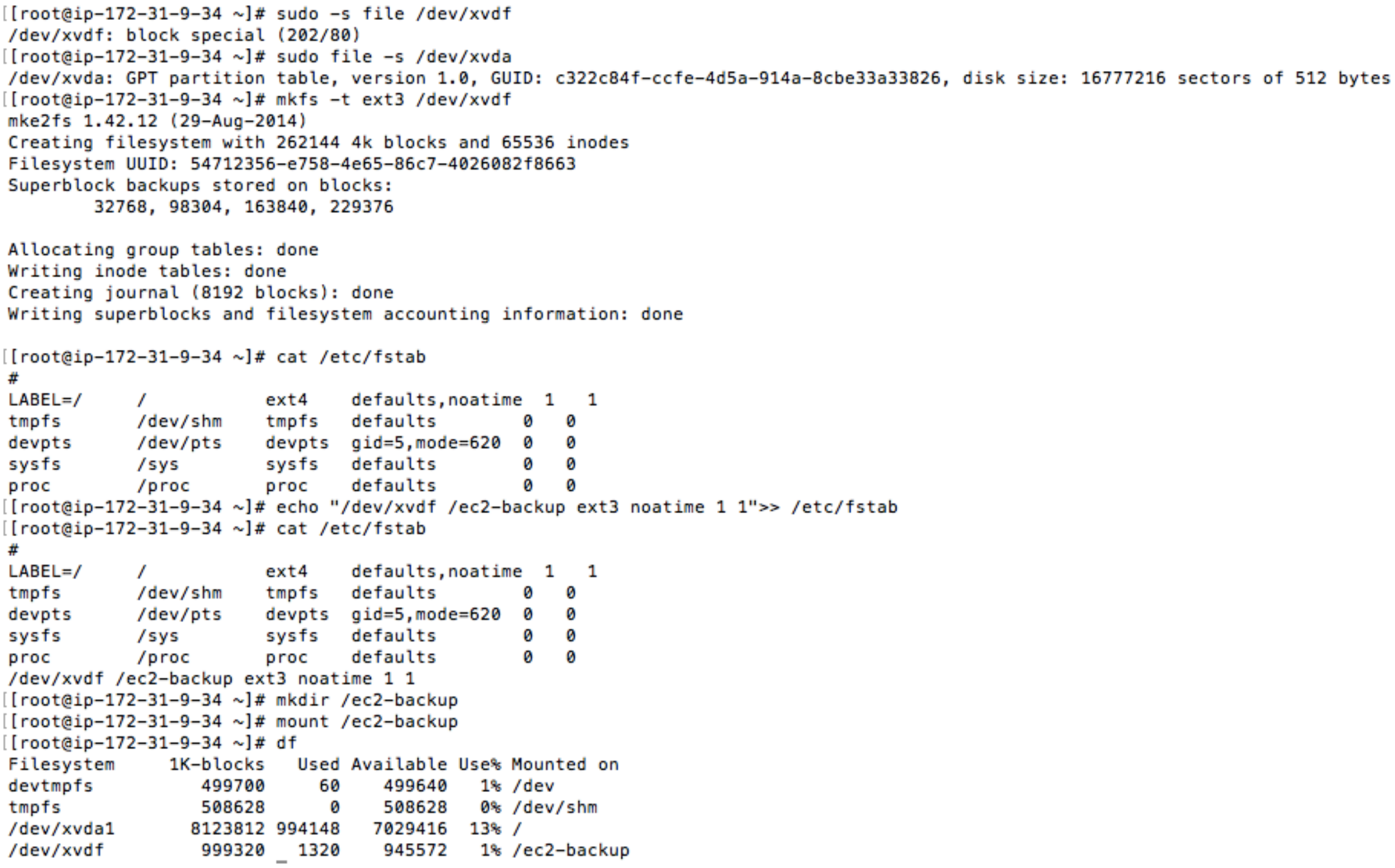
Note: Newer Linux kernels may rename your devices to /dev/xvdf through /dev/xvdp internally, even when the device name entered here (and shown in the details) is /dev/sdf through /dev/sdp.

4) After connecting to the instance by doing ssh on terminal we fire lsblk command which shows us disks attached with the instance.

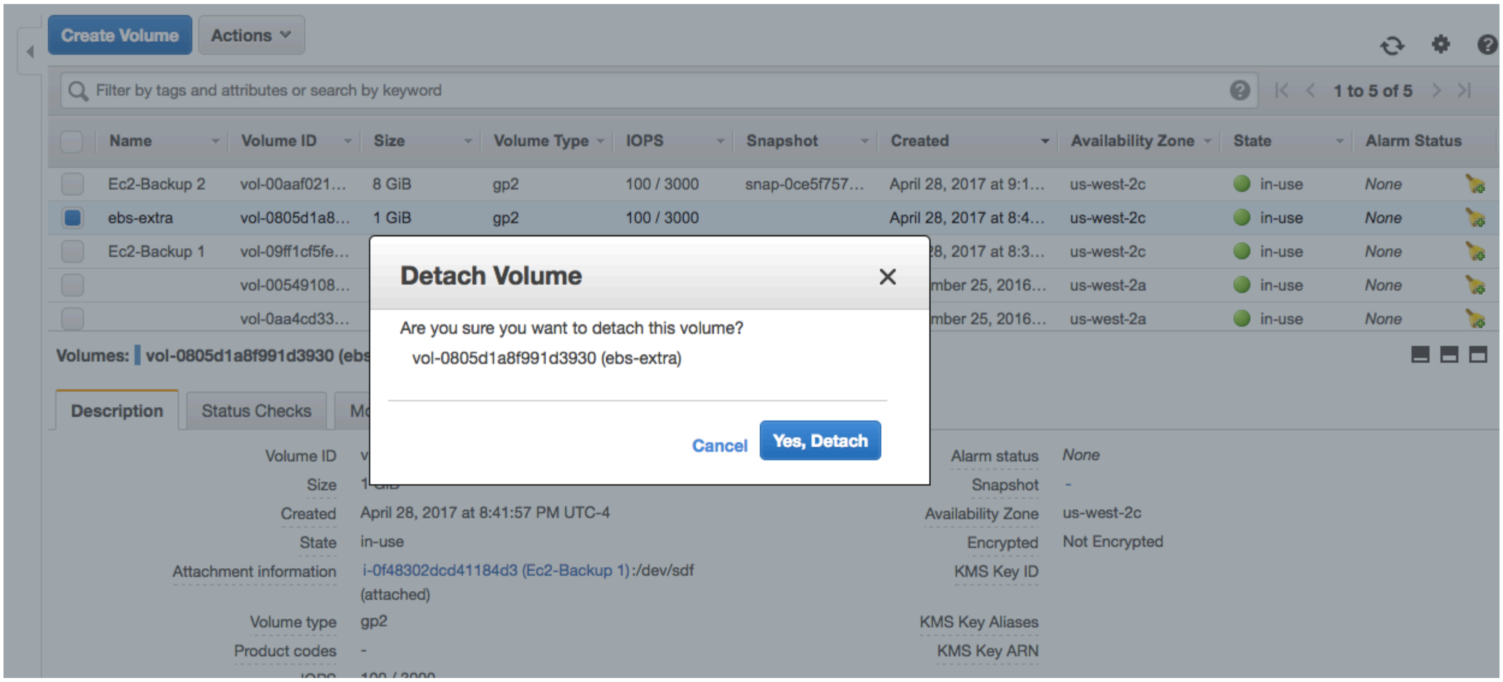


```
[ec2-user@ip-172-31-9-34 ~]$ lsblk
NAME    MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
xvda    202:0    0   8G  0 disk
└─xvda1 202:1    0   8G  0 part /
xvdf    202:80   0   1G  0 disk
[ec2-user@ip-172-31-9-34 ~]$ sudo su -
[root@ip-172-31-9-34 ~]# fdisk -l
WARNING: fdisk GPT support is currently new, and therefore in an experimental phase. Use at your own discretion.

Disk /dev/xvda: 8589 MB, 8589934592 bytes, 16777216 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: gpt


#         Start          End    Size  Type            Name
 1         4096     16777182     8G  Linux filesyste Linux
128        2048         4095     1M  BIOS boot parti BIOS Boot Partition

Disk /dev/xvdf: 1073 MB, 1073741824 bytes, 2097152 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

5) We make a file system with proper extended memory here we are using ext 3 after making it file system we make an entry in /etc/fstab file which maintains all abs blocks and finally we make the directory /ec2-backup and mount it after mounting we check by df command which shows mounted filesystem, now it shows /dev/xvdf as /ec2-backup



```
[[root@ip-172-31-9-34 ~]# sudo -s file /dev/xvdf
 /dev/xvdf: block special (202/80)
[[root@ip-172-31-9-34 ~]# sudo file -s /dev/xvda
 /dev/xvda: GPT partition table, version 1.0, GUID: c322c84f-ccfe-4d5a-914a-8cbe33a33826, disk size: 16777216 sectors of 512 bytes
[[root@ip-172-31-9-34 ~]# mkfs -t ext3 /dev/xvdf
 mke2fs 1.42.12 (29-Aug-2014)
 Creating filesystem with 262144 4k blocks and 65536 inodes
 Filesystem UUID: 54712356-e758-4e65-86c7-4026082f8663
 Superblock backups stored on blocks:
         32768, 98304, 163840, 229376

 Allocating group tables: done
 Writing inode tables: done
 Creating journal (8192 blocks): done
 Writing superblocks and filesystem accounting information: done

[[root@ip-172-31-9-34 ~]# cat /etc/fstab
 #
 LABEL=/     /           ext4    defaults,noatime  1   1
 tmpfs       /dev/shm    tmpfs   defaults        0   0
 devpts      /dev/pts    devpts  gid=5,mode=620  0   0
 sysfs       /sys        sysfs   defaults        0   0
 proc        /proc       proc    defaults        0   0
[[root@ip-172-31-9-34 ~]# echo "/dev/xvdf /ec2-backup ext3 noatime 1 1">> /etc/fstab
[[root@ip-172-31-9-34 ~]# cat /etc/fstab
 #
 LABEL=/     /           ext4    defaults,noatime  1   1
 tmpfs       /dev/shm    tmpfs   defaults        0   0
 devpts      /dev/pts    devpts  gid=5,mode=620  0   0
 sysfs       /sys        sysfs   defaults        0   0
 proc        /proc       proc    defaults        0   0
 /dev/xvdf /ec2-backup ext3 noatime 1 1
[[root@ip-172-31-9-34 ~]# mkdir /ec2-backup
[[root@ip-172-31-9-34 ~]# mount /ec2-backup
[[root@ip-172-31-9-34 ~]# df
 Filesystem     1K-blocks   Used Available Use% Mounted on
 devtmpfs          499700     60    499640   1% /dev
 tmpfs             508628      0    508628   0% /dev/shm
 /dev/xvda1       8123812 994148   7029416  13% /
 /dev/xvdf         999320 _ 1320    945572   1% /ec2-backup
```



☐	Name	Volume ID	Size	Volume Type	IOPS	Snapshot	Created	Availability Zone	State	Alarm Status
☐	Ec2-Backup 2	vol-00aaf021…	8 GiB	gp2	100 / 3000	snap-0ce5f757…	April 28, 2017 at 9:1…	us-west-2c	● in-use	None
☑	ebs-extra	vol-0805d1a8…	1 GiB	gp2	100 / 3000		April 28, 2017 at 8:4…	us-west-2c	● in-use	None
☐	Ec2-Backup 1	vol-09ff1cf5fe…					28, 2017 at 8:3…	us-west-2c	● in-use	None
☐		vol-00549108…					mber 25, 2016…	us-west-2a	● in-use	None
☐		vol-0aa4cd33…					mber 25, 2016…	us-west-2a	● in-use	None

Detach Volume ×

Are you sure you want to detach this volume?

vol-0805d1a8f991d3930 (ebs-extra)

Volumes: vol-0805d1a8f991d3930 (ebs

Volume ID v

Size 1

Created April 28, 2017 at 8:41:57 PM UTC-4

State in-use

Attachment information i-0f48302dcd41184d3 (Ec2-Backup 1):/dev/sdf (attached)

Volume type gp2

Product codes -

Alarm status None

Snapshot -

Availability Zone us-west-2c

Encrypted Not Encrypted

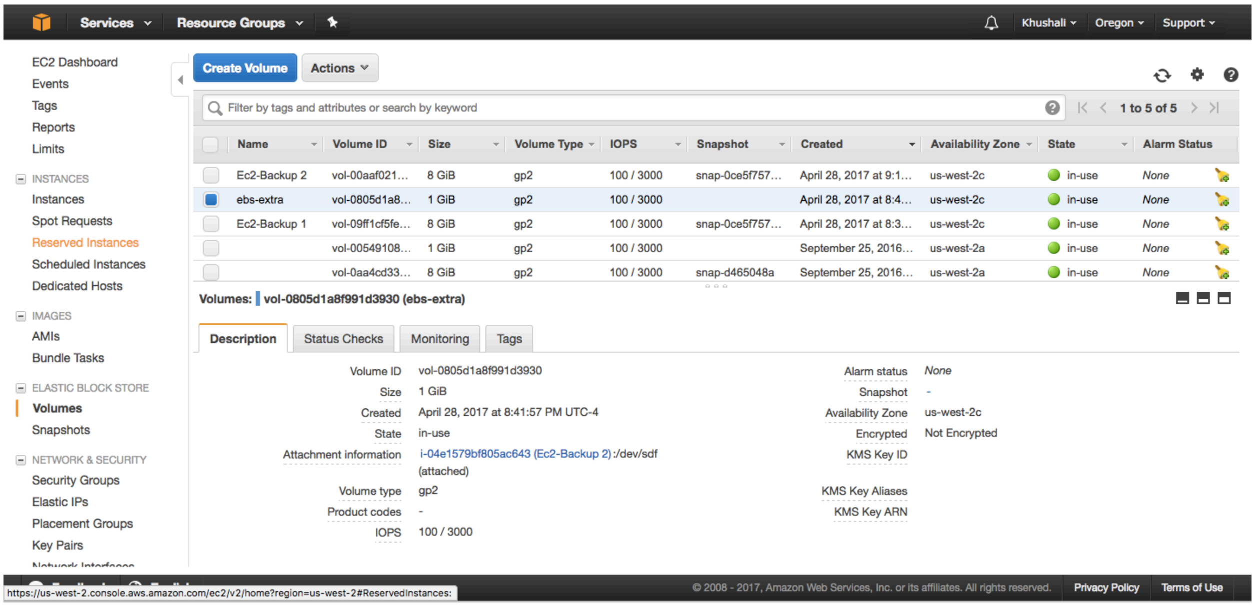
KMS Key ID

KMS Key Aliases

KMS Key ARN

6) As we are able to see in image above we are trying to detach the volume but also notice in second window below this it was attached to EC2-Bakcup 1

7) Here we attached eps-extra to another instance i.e EC2-backup 2



	Name	Volume ID	Size	Volume Type	IOPS	Snapshot	Created	Availability Zone	State	Alarm Status
□	Ec2-Backup 2	vol-00aaf021…	8 GiB	gp2	100 / 3000	snap-0ce5f757…	April 28, 2017 at 9:1…	us-west-2c	● in-use	None
■	ebs-extra	vol-0805d1a8…	1 GiB	gp2	100 / 3000		April 28, 2017 at 8:4…	us-west-2c	● in-use	None
□	Ec2-Backup 1	vol-09ff1cf5fe…	8 GiB	gp2	100 / 3000	snap-0ce5f757…	April 28, 2017 at 8:3…	us-west-2c	● in-use	None
□		vol-00549108…	1 GiB	gp2	100 / 3000		September 25, 2016…	us-west-2a	● in-use	None
□		vol-0aa4cd33…	8 GiB	gp2	100 / 3000	snap-d465048a	September 25, 2016…	us-west-2a	● in-use	None

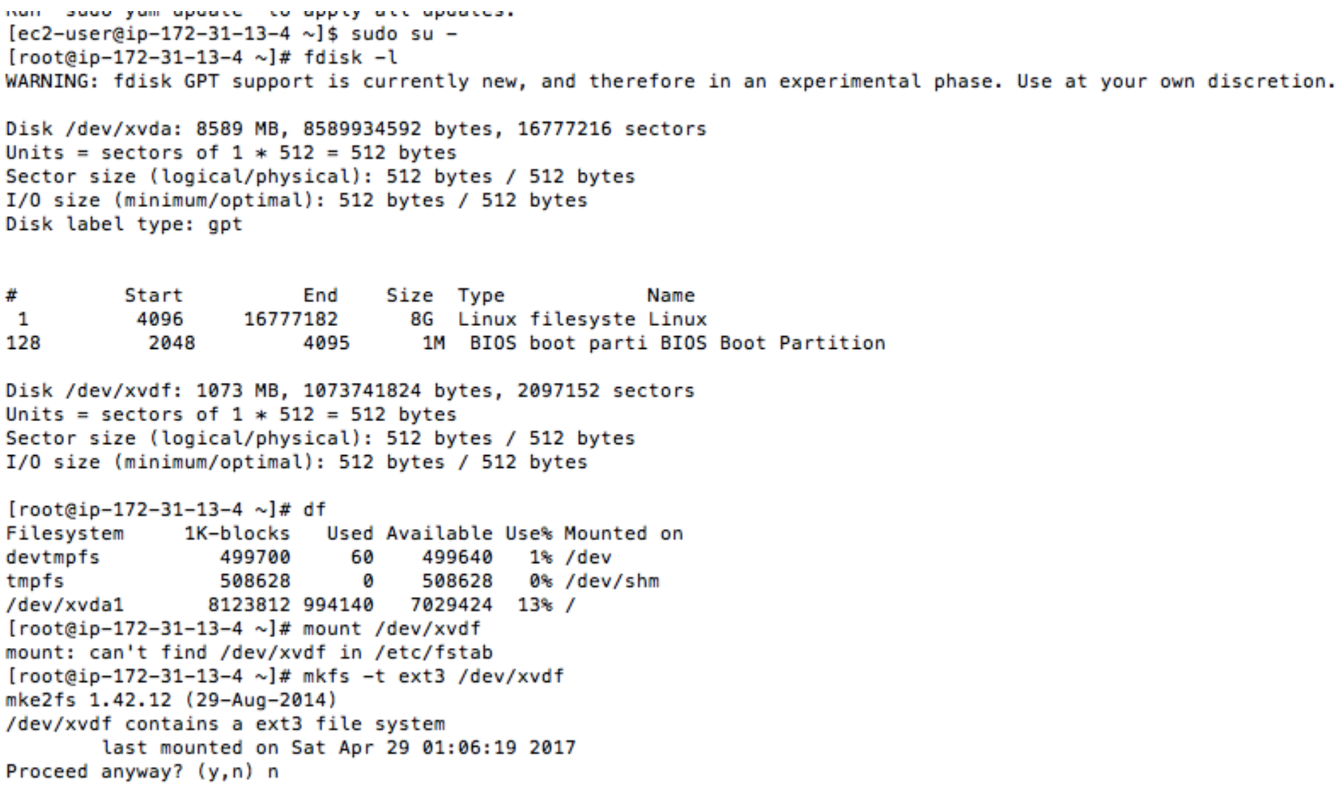
Volumes: vol-0805d1a8f991d3930 (ebs-extra)

Volume ID	vol-0805d1a8f991d3930	Alarm status	None
Size	1 GiB	Snapshot	-
Created	April 28, 2017 at 8:41:57 PM UTC-4	Availability Zone	us-west-2c
State	in-use	Encrypted	Not Encrypted
Attachment information	i-04e1579bf805ac643 (Ec2-Backup 2):/dev/sdf (attached)	KMS Key ID	
Volume type	gp2	KMS Key Aliases	
Product codes	-	KMS Key ARN	
IOPS	100 / 3000		

https://us-west-2.console.aws.amazon.com/ec2/v2/home?region=us-west-2#ReservedInstances:

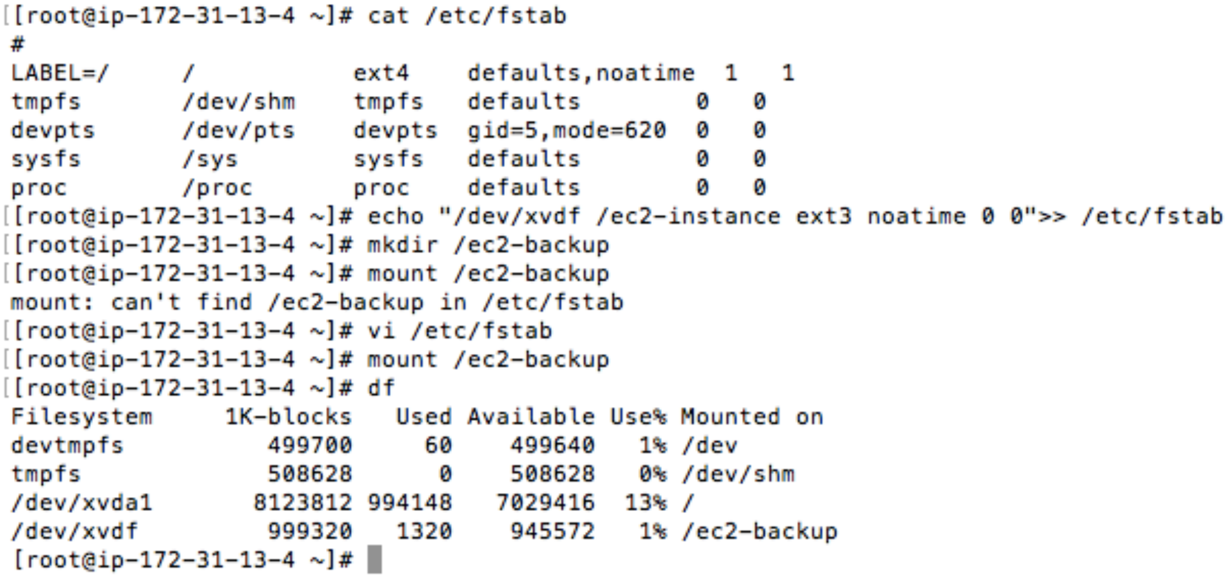
© 2008 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved.

8) For mounting this block we just need to make entry in /etc/fstab we have already made it available in file system so it should store that information and which can be seen when i try to make the same block in file system it asks to reformat as it is already a filesystem.



```
Run  sudo yum update  to apply all updates.
[ec2-user@ip-172-31-13-4 ~]$ sudo su -
[root@ip-172-31-13-4 ~]# fdisk -l
WARNING: fdisk GPT support is currently new, and therefore in an experimental phase. Use at your own discretion.

Disk /dev/xvda: 8589 MB, 8589934592 bytes, 16777216 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: gpt


#         Start          End    Size  Type            Name
 1         4096     16777182      8G  Linux filesyste Linux
128        2048         4095      1M  BIOS boot parti BIOS Boot Partition

Disk /dev/xvdf: 1073 MB, 1073741824 bytes, 2097152 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes

[root@ip-172-31-13-4 ~]# df
Filesystem     1K-blocks   Used Available Use% Mounted on
devtmpfs          499700     60    499640   1% /dev
tmpfs             508628      0    508628   0% /dev/shm
/dev/xvda1       8123812 994140   7029424  13% /
[root@ip-172-31-13-4 ~]# mount /dev/xvdf
mount: can't find /dev/xvdf in /etc/fstab
[root@ip-172-31-13-4 ~]# mkfs -t ext3 /dev/xvdf
mke2fs 1.42.12 (29-Aug-2014)
/dev/xvdf contains a ext3 file system
        last mounted on Sat Apr 29 01:06:19 2017
Proceed anyway? (y,n) n
```

9) Finally mount the disk.



```
[[root@ip-172-31-13-4 ~]# cat /etc/fstab
#
LABEL=/     /           ext4    defaults,noatime  1   1
tmpfs       /dev/shm    tmpfs   defaults        0   0
devpts      /dev/pts    devpts  gid=5,mode=620  0   0
sysfs       /sys        sysfs   defaults        0   0
proc        /proc       proc    defaults        0   0
[[root@ip-172-31-13-4 ~]# echo "/dev/xvdf /ec2-instance ext3 noatime 0 0">> /etc/fstab
[[root@ip-172-31-13-4 ~]# mkdir /ec2-backup
[[root@ip-172-31-13-4 ~]# mount /ec2-backup
mount: can't find /ec2-backup in /etc/fstab
[[root@ip-172-31-13-4 ~]# vi /etc/fstab
[[root@ip-172-31-13-4 ~]# mount /ec2-backup
[[root@ip-172-31-13-4 ~]# df
Filesystem     1K-blocks   Used Available Use% Mounted on
devtmpfs          499700     60    499640   1% /dev
tmpfs             508628      0    508628   0% /dev/shm
/dev/xvda1       8123812 994148   7029416  13% /
/dev/xvdf         999320   1320    945572   1% /ec2-backup
[root@ip-172-31-13-4 ~]# █
```

Observations:

- While installing Nginx if Rule for 80 port is allowed on security group then only it will open using public dos name
- While mounting EBS additionally noticed we can store all database in this as its data doesn’t get deleted if the instance is terminated.
- Load balancer based on weights distributes the load to servers in round robin fashion.