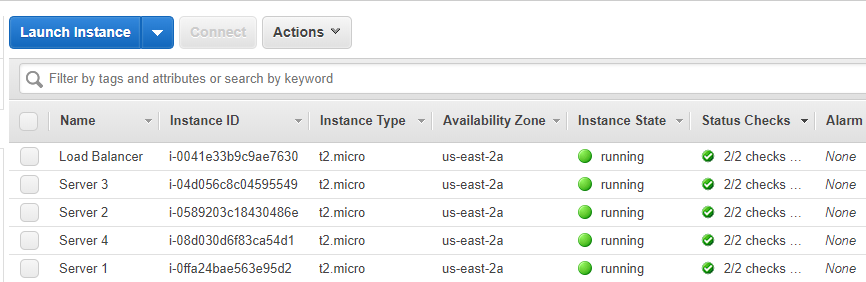
**JichenDai\_CS 524\_Lab\_#2**

## Create the Amazon EC2 instances

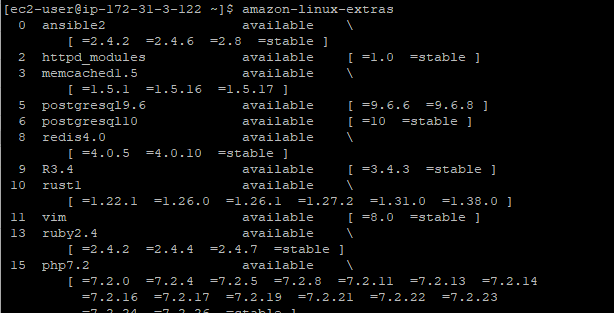
By follow the steps specified in Lab\_#1, five instances is first created.



## Install *Nginx* on each instance

Use ***aamzon-linux-extras*** to get information about extras.





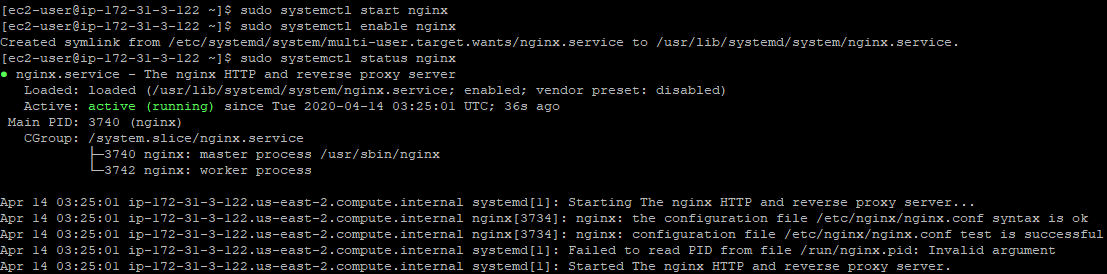
Enable nginx



Remove matedate and install nginx.

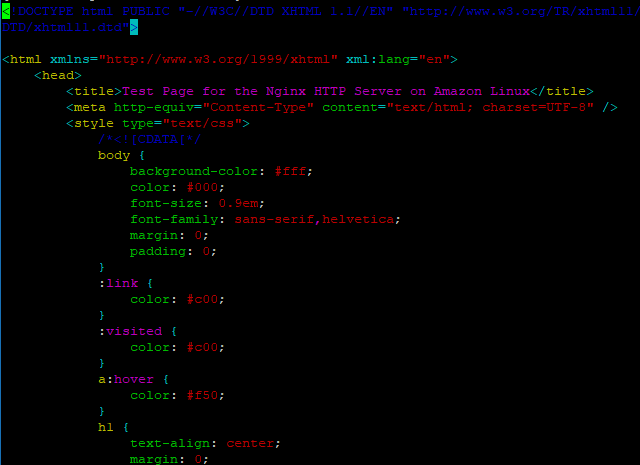


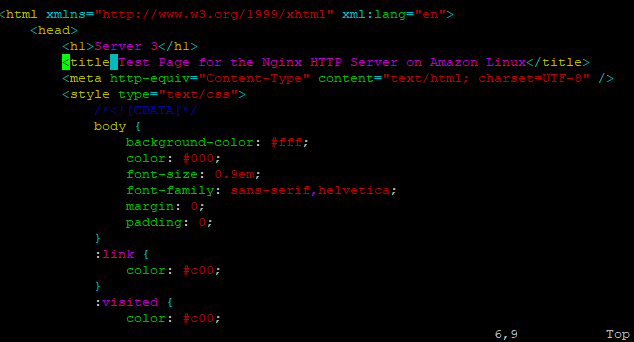
Start and enable nginx using: ***sudo systemctl start nginx*** and ***sudo systemctl enable nginx.***



To verify the *Nginx* is working, visit the DNS in google chrome.

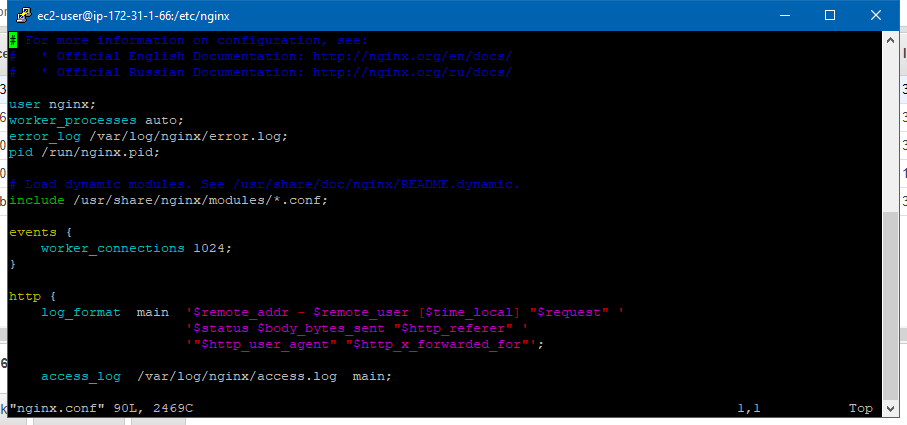
Using **vim** to open the index.html to edit *index.html*. Add <h1>Server N</h1> to it.





## Configure the load balancer

using vim to open *nginx.conf.*



*Replace and add the following content:*

*events {*

*worker\_connections 768;*

*}*

*http {*

*upstream myapp {*

*#ip\_hash;*

*server [SERVER\_PUBLIC\_DNS\_NAME] weight=1;*

*server [SERVER\_PUBLIC\_DNS\_NAME] weight=1;*

*server [SERVER\_PUBLIC\_DNS\_NAME] weight=1;*

*server [SERVER\_PUBLIC\_DNS\_NAME] weight=1;*

*}*

*server {*

*listen 80;*

*server\_name myapp.com;*

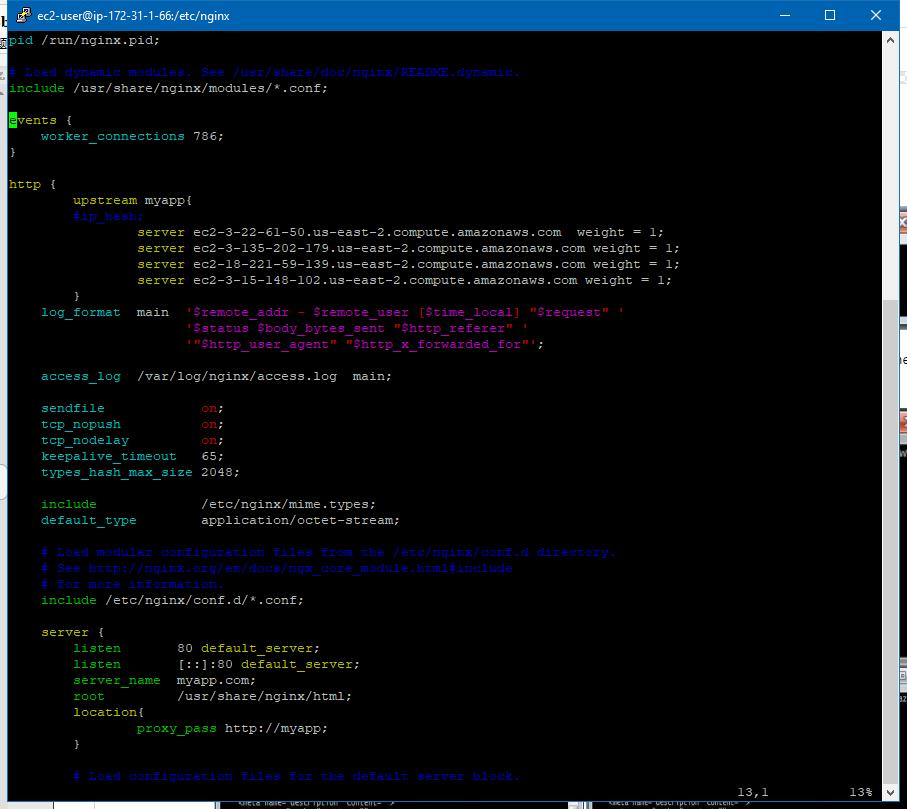
*location / {*

*proxy\_pass http://myapp;*

*}*

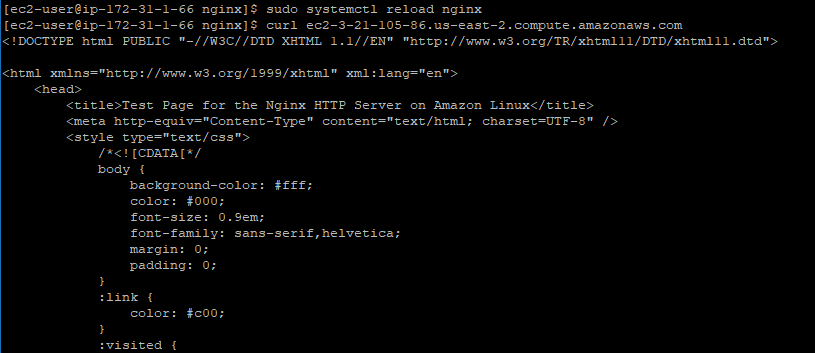
*}*

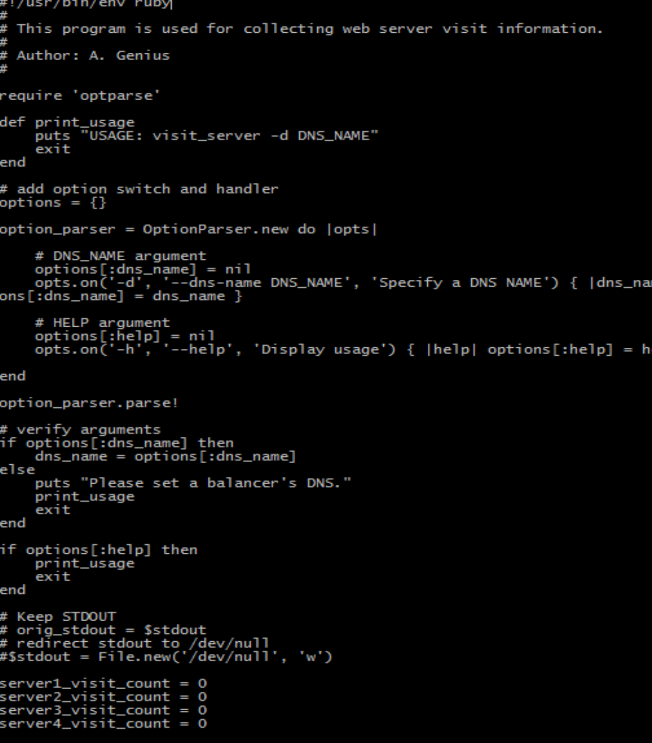
*}*



Reload nginx using: ***sudo systemctl reload nginx***.

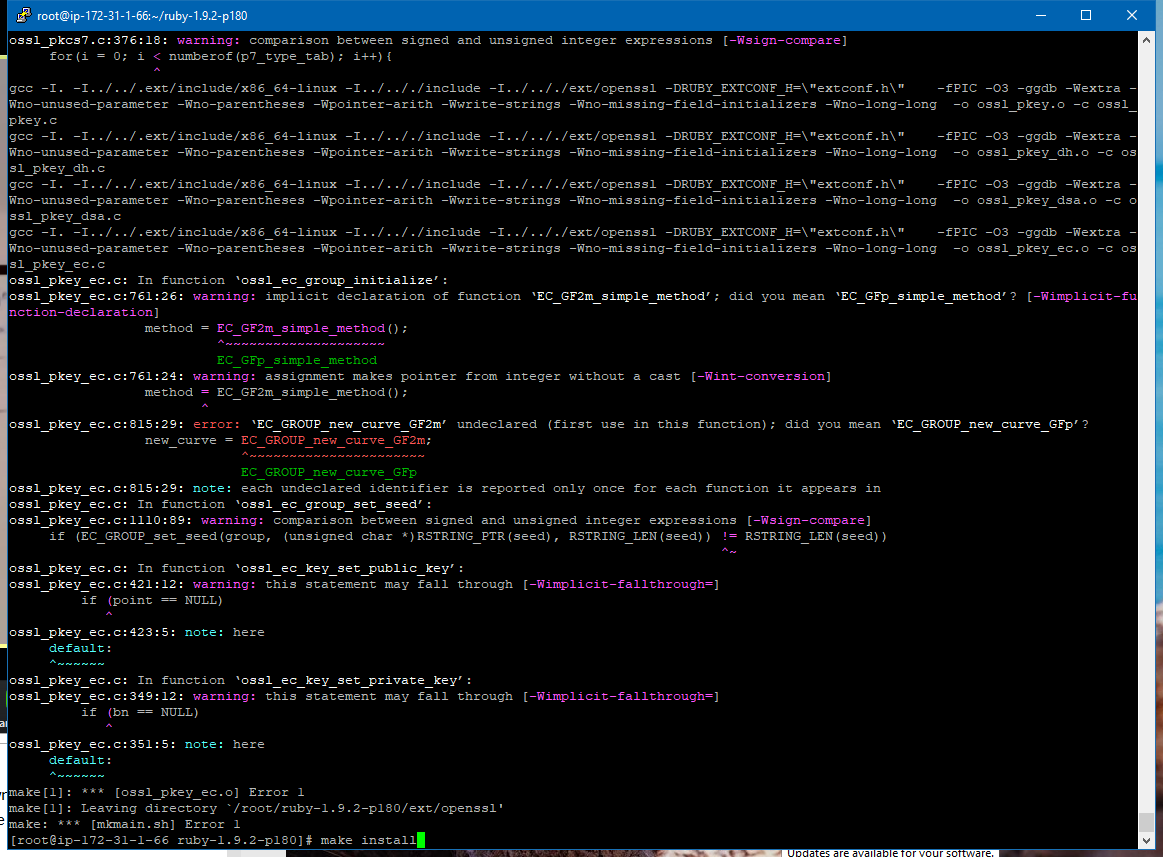
Use the ***curl ec2-3-21-105-86.us-east-2.compute.amazonaws.com*** command visit the balancer.



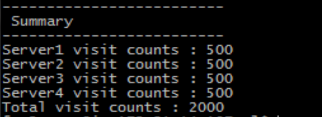
Edit visit\_server suing: ***vim visit\_server***

## **Collect the information on visits to your site**

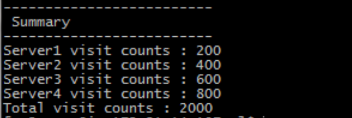
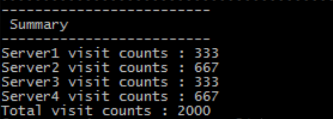
## Install Ruby



Use **ruby visit\_server -d ec2-172-31-1-66.us-east-2.compute.amazonaws.com** to see result of scenario 1.



Then, change the weight of four server into **1: 2: 3: 4** and **1: 2: 1: 2** in *nginx.conf*. Get different result:



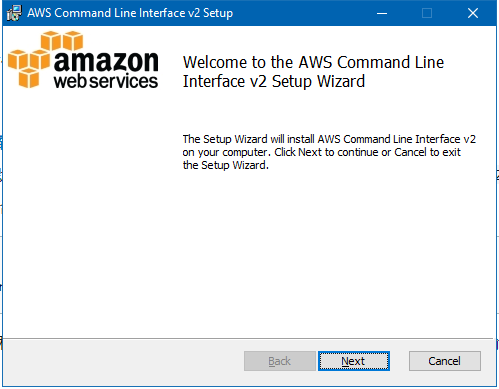
## **My observations**

In my opinion, on the one hand, the technology of load balancing is very useful, since it allows us to handle a large bunch of job by dividing them into multi-servers. On the other hand, this technology also help us to implement failover: when one of the server shut down, other servers can take the responsibility.

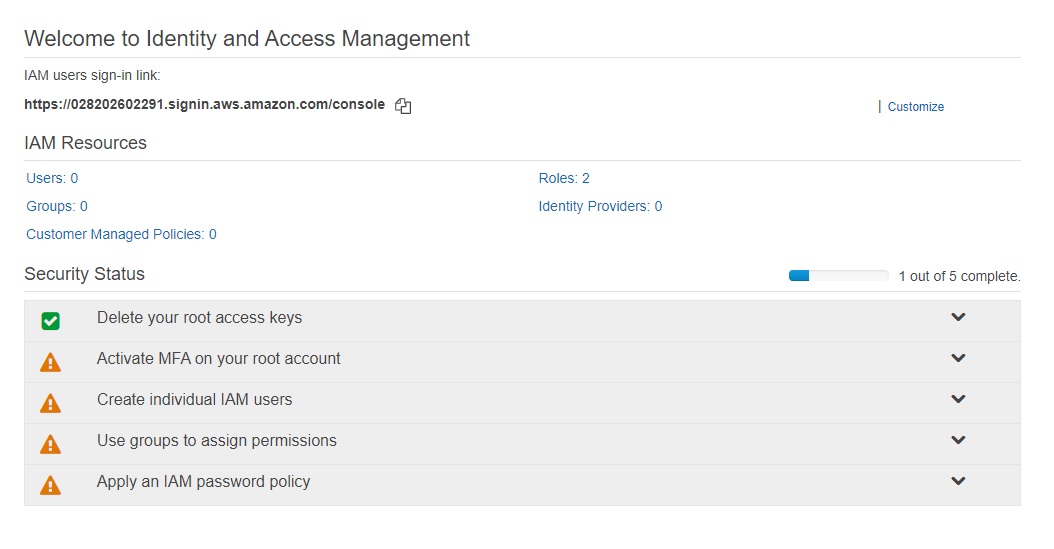
**Additional Steps:**

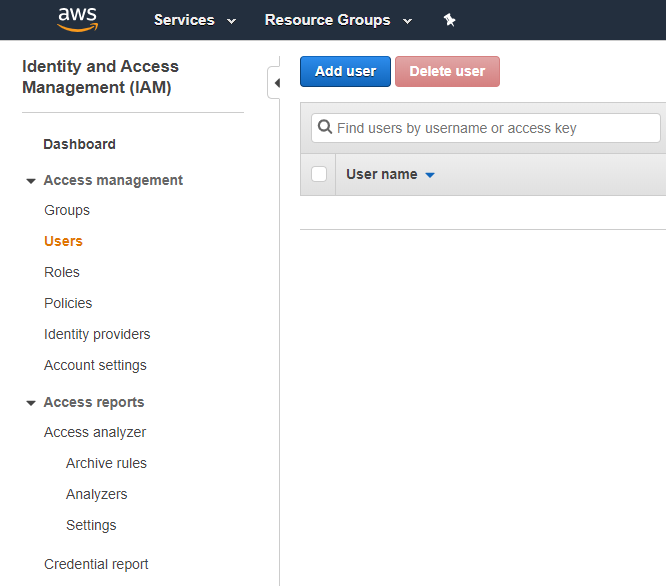
Creating instance using Command Line

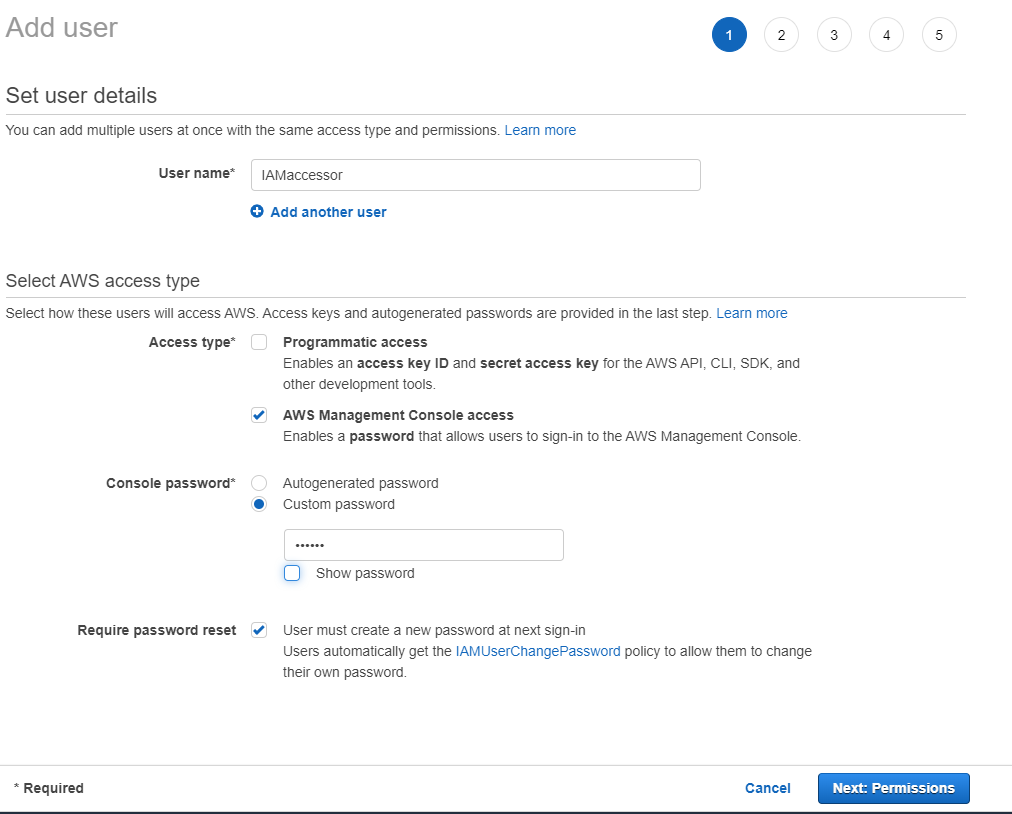
Download and Install the command line:



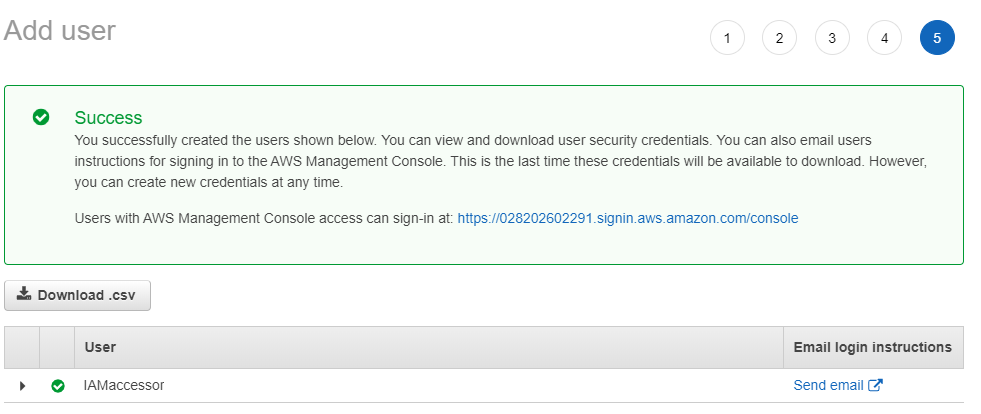
Create a user following steps below:



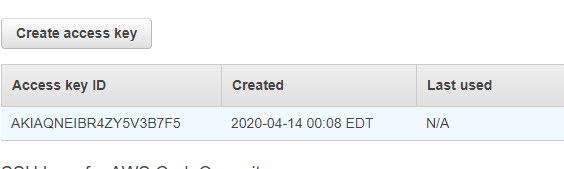




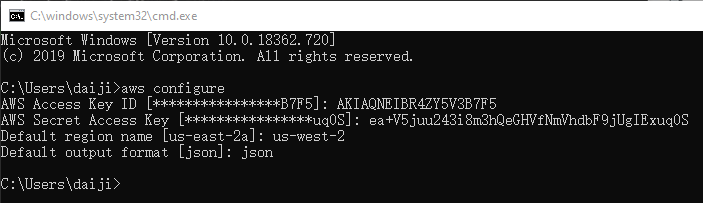
A success message will appear after user is created successfully.



Now, select the *user* just created and click on *Security credentials*, then click *Create access key*, a new access key is created.



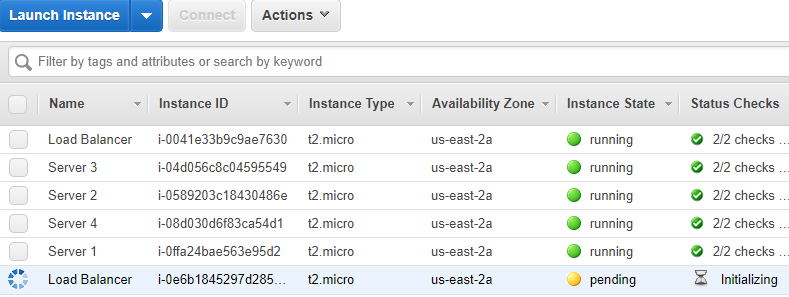
Then, open command line in computer and input following commands.



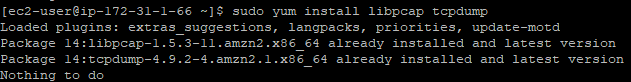
Finally, launch an instance using the command below:



Now, We have an additional instance,



**Collect and analyze packages**  
Check whether tcpdump packages is installed



Running tcpdump command and create report in dumpfile.txt.

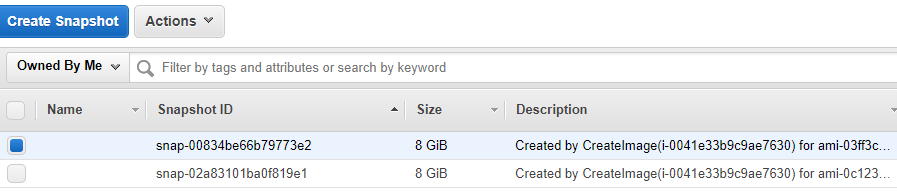


Running tcpdump command again and create dumpfile2.txt.

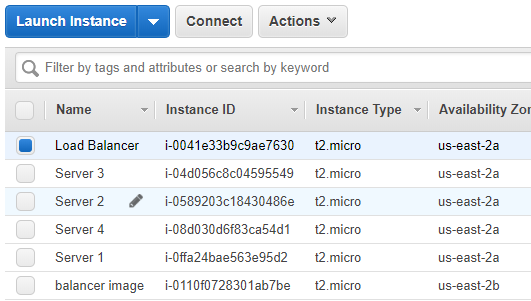


**EC2 backup and restore:**

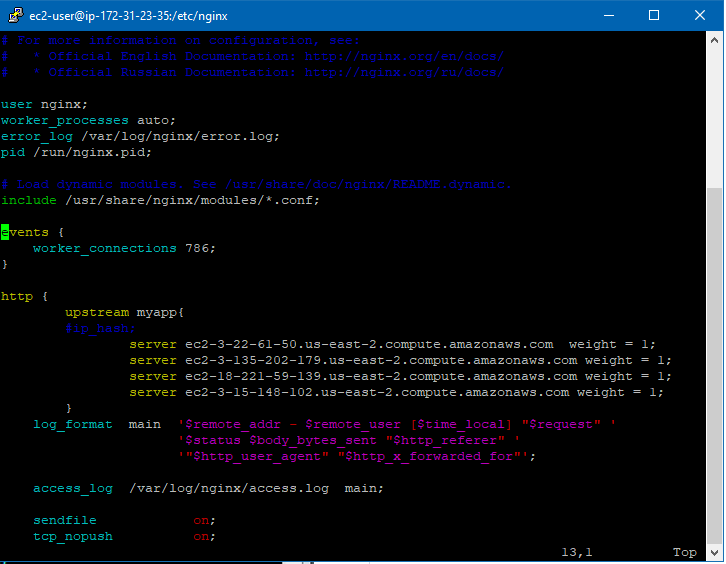
Create an image of load balancer, you can just right click your load balancer and click ‘image’:( I created two by accident)



Create an instance with this image:



Then open this image instance in PuTTY and go into nginx.conf.



**Since the nginx.conf in this instance is the same as the nginx.conf in load balancer, we can come to the conclusion that they have same files.**

**This is not the original load balancer, because we can see this ip 172.31.23.35 is different from load balancer(172.31.1.66).**