

Lab #2

1) Create Amazon EC2 Instances:

Step 1: Sign in Amazon AWS account:

The screenshot shows the AWS Educate website at awseducate.com/student/s/awssite. The top navigation bar includes links for Apps, 210-451 - Cisco Pra..., Your Customized Te..., Portfolio, Career Pathways, Badges, Jobs, AWS Account, and Logout. A banner at the top says "Your Customized Textbook List | The Stevens Institute of Technology Campus Store".



AWS Educate Starter Account

Your cloud journey has only just begun. Use your AWS Educate Starter Account to access the AWS Console and resources, and start building in the cloud!

[AWS Educate Starter Account](#)

Your account has an estimated **98** credits remaining and access will end on **Feb 13, 2021**.

Note: Clicking this button will take you to a third party site managed by Vocareum, Inc. ("Third Party Servicer"). In addition to the AWS Educate terms of service, your use of the AWS Educate Starter Account is governed by the Third Party Servicer's terms, including its Privacy Policy. AWS assumes no responsibility or liability and makes no representations or warranties regarding services provided by a Third Party Servicer.

The screenshot shows the Vocareum website at labs.vocareum.com/main/main.php?m=editor&nav=1&asnid=14334&stepid=14335. The top navigation bar includes links for Apps, 210-451 - Cisco Pra..., Your Customized Te..., Home, My Classes, Help, and ttungeka@stevens.edu. The main content area is titled "Welcome to your AWS Educate Account". It provides a brief introduction and links to the AWS Console. On the right, there is a "Your AWS Account Status" section with three cards: "Active" (full access), "\$98.62" (remaining credits), and "2:59" (session time). At the bottom, a note cautions users to use the account responsibly and shut down instances when not in use.

Step 2:

The screenshot shows the AWS Management Console homepage. At the top, there's a navigation bar with links for 'Apps', 'Services' (selected), 'Resource Groups', and user information ('vocstartsoft/user623773=ttun...', 'N. Virginia', 'Support'). Below the navigation is the title 'AWS Management Console'. On the left, there's a sidebar titled 'AWS services' with sections for 'Find Services' (a search bar), 'Recently visited services' (listing EC2 and Billing), and 'All services'. To the right, there are two main content areas: 'Access resources on the go' (with a link to the AWS Console Mobile App) and 'Explore AWS' (sections for 'S3 Intelligent-Tiering' and 'AWS IQ').

Step 3:

Click on Services dropdown menu, Select EC2 under Compute:

The screenshot shows the 'Services' dropdown menu in the AWS Management Console. The 'Compute' section is expanded, showing various services like EC2, Lightsail, Lambda, Batch, Elastic Beanstalk, Serverless Application Repository, AWS Outposts, and EC2 Image Builder. The 'EC2' service is selected. Other sections visible include 'Storage' (S3, EFS, FSx, S3 Glacier, Storage Gateway, AWS Backup), 'Blockchain' (Amazon Managed Blockchain), 'Analytics' (Athena, EMR, CloudSearch, Elasticsearch Service, Kinesis, QuickSight), 'End User Computing' (WorkSpaces, AppStream 2.0, WorkDocs, WorkLink), 'Quantum Technologies' (Amazon Braket), 'Management & Governance' (AWS Organizations, CloudWatch, AWS Auto Scaling, CloudFormation, CloudTrail, Config), 'Security, Identity, & Compliance' (IAM, Resource Access Manager, Cognito), and 'Internet Of Things' (IoT Core, FreeRTOS, IoT 1-Click, IoT Analytics, IoT Device Defender, IoT Device Management, IoT Events, IoT Greengrass, IoT SiteWise, IoT Things Graph). At the bottom of the dropdown, there are links for 'Build a web app' and 'Build using virtual servers'.

Step 4: On EC2 Dashboard, Click on Launch Instance:

Launch instance

To get started, launch an Amazon EC2 instance, which is a virtual server in the cloud.

Launch Instance

Note: Your instances will launch in the US East (N. Virginia) Region

Scheduled events

US East (N. Virginia)
No scheduled events

Migrate a machine

Use CloudEndure Migration to simplify, expedite, and automate large-scale migrations from physical, virtual, and cloud-based infrastructure to AWS.

Select the Amazon Linux 2 AMI as shown below:

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

Step 1: Choose an Amazon Machine Image (AMI)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace; or you can select one of your own AMIs.

Cancel and Exit

Search for an AMI by entering a search term e.g. "Windows"

Quick Start

My AMIs	Amazon Linux 2 AMI (HVM, SSD Volume Type) - ami-0fc61db8544a617ed (64-bit x86) / ami-0f90a34c9df977efb (64-bit Arm)	Select
AWS Marketplace	Amazon Linux 2 comes with five years support. It provides Linux kernel 4.14 tuned for optimal performance on Amazon EC2, systemd 219, GCC 7.3, Glibc 2.26, Binutils 2.29.1, and the latest software packages through extras.	<input checked="" type="radio"/> 64-bit (x86) <input type="radio"/> 64-bit (Arm)
Community AMIs	Amazon Linux AMI 2018.03.0 (HVM), SSD Volume Type - ami-09a5b0b7edf08843d	Select
<input type="checkbox"/> Free tier only <small>(i)</small>	The Amazon Linux AMI is an EBS-backed, AWS-supported image. The default image includes AWS command line tools, Python, Ruby, Perl, and Java. The repositories include Docker, PHP, MySQL, PostgreSQL, and other packages.	64-bit (x86)

Select t2.micro as instance type and Click on Preview and Launch button:

Screenshot of the AWS EC2 Instance Launch Wizard Step 2: Choose an Instance Type.

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
<input type="checkbox"/>	General purpose	t2.nano	1	0.5	EBS only	-	Low to Moderate	Yes
<input checked="" type="checkbox"/>	General purpose	t2.micro <small>Free tier eligible</small>	1	1	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	General purpose	t2.small	1	2	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	General purpose	t2.medium	2	4	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	General purpose	t2.large	2	8	EBS only	-	Low to Moderate	Yes

Cancel Previous Review and Launch Next: Configure Instance Details

Click on Edit Security Group on the next page:

Step 7: Review Instance Launch

Root Device Type: ebs Virtualization type: hvm

Instance Type [Edit instance type](#)

Instance Type	ECUs	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance
t2.micro	Variable	1	1	EBS only	-	Low to Moderate

Security Groups [Edit security groups](#)

Security group name: launch-wizard-5
Description: launch-wizard-5 created 2020-04-03T12:16:13.627-04:00

Type	Protocol	Port Range	Source	Description
This security group has no rules				

Cancel Previous Launch

Review the Security Group:

Open port 80 and 22:

Security group name: launch-wizard-8

Security group ID: sg-0c1557e5d4a0f89b2

Description: launch-wizard-8 created 2020-04-03T13:46:45.767-04:00

VPC ID: vpc-cc0435b6

Owner: 112619478712

Inbound rules count: 3 Permission entries

Outbound rules count: 1 Permission entry

Type	Protocol	Port range	Source	Description - optional
HTTP	TCP	80	0.0.0.0/0	-
HTTP	TCP	80	::/0	-
SSH	TCP	22	76.116.178.250/32	-

Click on Launch:

Since this instance is for LoadBalancer, give the key that name and download the 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: LoadBalance

Download Key Pair

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.

Cancel **Launch Instances**

After downloading the key-pair, click on Launch Instances:

The screenshot shows the AWS CloudWatch Launch Status page. It displays a green checkmark icon and the message "Your instances are now launching". Below this, it says "The following instance launches have been initiated: i-0e41636c31bac77c9" and provides a link to "Hide launch log". A table shows the status of various steps: "Creating security groups" is successful (sg-0c68a016e9b284f8c); "Authorizing inbound rules" and "Initiating launches" are also successful; and "Launch initiation complete" is listed. The status bar at the bottom indicates "Launch Status" and "N. Virginia".

Launch Status

The screenshot continues the CloudWatch Launch Status page. It shows a section titled "Get notified of estimated charges" with a note about creating billing alerts for estimated charges. Below this, a section titled "How to connect to your instances" provides instructions: instances are launching and will reach the "running" state; users can monitor them via the "View Instances" link. The status bar at the bottom indicates "Launch Status" and "N. Virginia".

In the View Instances tab on EC2 Console, this instance can be viewed. I have named this one as Load Balancer:

The screenshot shows the AWS EC2 Instances page. The left sidebar lists navigation options like EC2 Dashboard, Instances, and Images. The main area shows a table of instances. One instance is selected and highlighted with a blue border, labeled "LoadBalancer" with Instance ID "i-0e41636c31bac77c9". The table includes columns for Name, Instance ID, Instance Type, Availability Zone, Instance State, Status Checks, Alarm Status, and Public DNS (IPv4). The Public DNS is listed as "ec2-54-80-242-77.compute-1.amazonaws.com". At the bottom, there are tabs for Description, Status Checks, Monitoring, and Tags, along with links for Feedback and English (US).

Note: For Load Balancer, security group should be:

Inbound rules

Type	Protocol	Port range	Source	Description - optional
SSH	TCP	22	Custom	0.0.0.0/0
HTTP	TCP	80	Custom	0.0.0.0/0

Add rule

⚠️ NOTE: Any edits made on existing rules will result in the edited rule being deleted and a new rule created with the new details. This will cause traffic that depends on that rule to stop working until the new rule is established.

Similarly, repeating the above steps, I have created 4 more instances, namely, Server1, Server2, Server 3 and Server 4 which can be viewed in the View Instances tab on EC2 Console:

Step 4: For Server 1, Server 2, Server 3, Server 4; IP address of my machine should be in the Security Group:

```
C:\ Administrator: Command Prompt

Ethernet adapter Ethernet:
Media State . . . . . : Media disconnected
Connection-specific DNS Suffix . : hsd1.nj.comcast.net

Wireless LAN adapter Local Area Connection* 2:
Media State . . . . . : Media disconnected
Connection-specific DNS Suffix . :

Wireless LAN adapter Local Area Connection* 3:
Media State . . . . . : Media disconnected
Connection-specific DNS Suffix . :

Wireless LAN adapter Wi-Fi:
Connection-specific DNS Suffix . : hsd1.nj.comcast.net
IPv6 Address. . . . . : 2601:81:4101:5280::80bb
IPv6 Address. . . . . : 2601:81:4101:5280:3c7e:8367:992e:f9ae
Temporary IPv6 Address. . . . . : 2601:81:4101:5280:cd96:ab9a:3e60:295f
Link-local IPv6 Address . . . . . : fe80::3c7e:8367:992e:f9ae%9
IPv4 Address. . . . . : 10.0.0.13
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : fe80::504c:9eff:febb:3ebb%9
                                         10.0.0.1

Ethernet adapter Bluetooth Network Connection 2:
```

So I used 10.0.0.13/32(which was not my IP address)in Security group so only my machine would be able to communicate with the instances, but I was not able to connect. This step has to be done for all servers 1 to 4. Also, their corresponding Key Pairs have to be downloaded.

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: launch-wizard-6

Description: launch-wizard-6 created 2020-04-03T13:39:38.194-04:00

Type	Protocol	Port Range	Source	Description
SSH	TCP	22	Custom 10.0.0.13/32	e.g. SSH for Admin Desktop

Add Rule

Review and Launch

Step 5: The highlighted ones show four servers and one load balancer that was configured.

EC2 Dashboard

Instances

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)
Server2	i-00b136b7a117c7574	t2.micro	us-east-1b	running	2/2 checks ...	None	ec2-54-80-150-192.co...
Server3	i-02c8a9d1904dfcfea	t2.micro	us-east-1d	running	2/2 checks ...	None	ec2-3-83-45-38.comput...
	i-0677ae41850c3ff96	t2.micro	us-east-1b	terminated		None	-
Server4	i-0d787f6d060da86aa	t2.micro	us-east-1b	running	Initializing	None	ec2-3-80-179-81.comp...
Server1	i-0ddec3278642ef16b	t2.micro	us-east-1b	running	2/2 checks ...	None	ec2-3-90-236-228.com...
LoadBalancer	i-0e41636c31bac77c9	t2.micro	us-east-1b	running	2/2 checks ...	None	ec2-54-210-153-235.co...

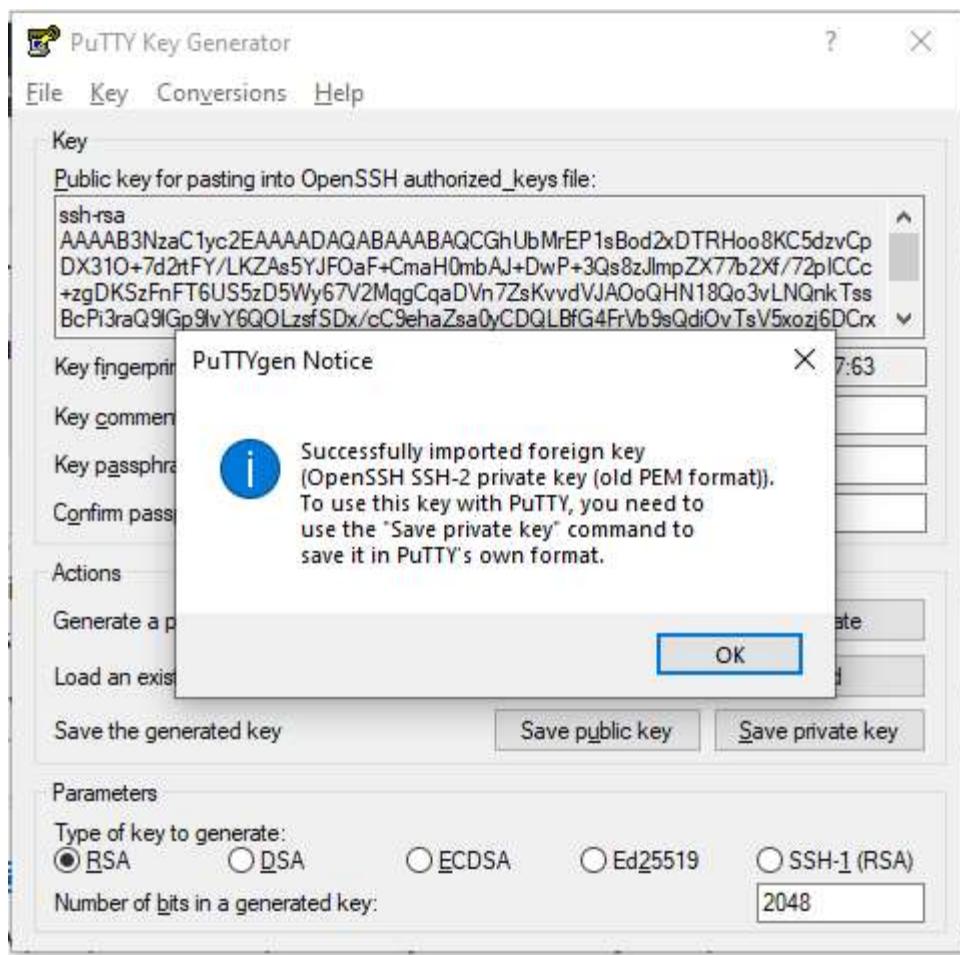
Instances: i-0e41636c31bac77c9 (LoadBalancer), i-0ddec3278642ef16b (Server1), i-0d787f6d060da86aa (Server4), i-02c8a9d1904dfcfea (Server3), i-00b136b7a117c7574 (Server2)

Description **Status Checks** **Monitoring** **Tags**

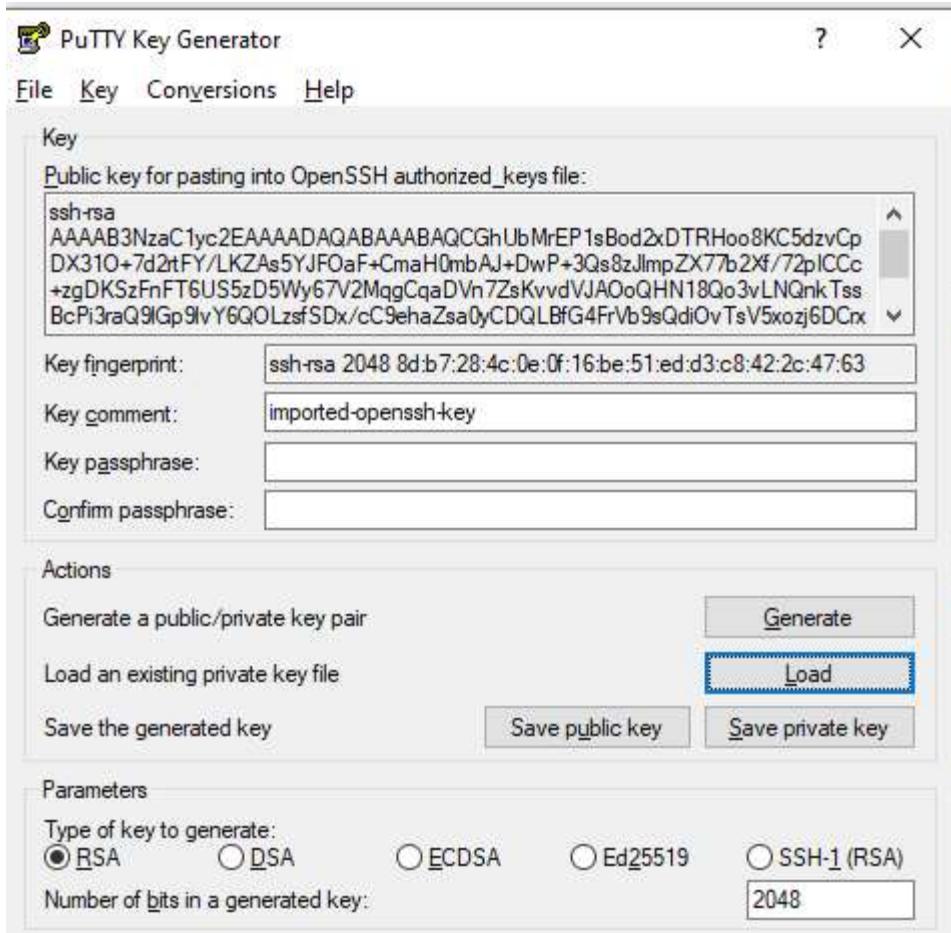
Feedback **English (US)**

Step 6:

In order to connect to LoadBalancer Console, firstly the .pem key has to be converted to .ppk using pUTTYGen:



Click on Save Private Key:



In this way, LoadBalancer.ppk file is created.

Step 7:

Start PuTTY (from the Start menu, choose All Programs, PuTTY, PuTTY)

In Session Category, For Host Name or IP Address, input the Public DNS of the LoadBalancer, highlighted in the figure below: Ensure that the Port value is 22 and Under Connection type, select SSH.

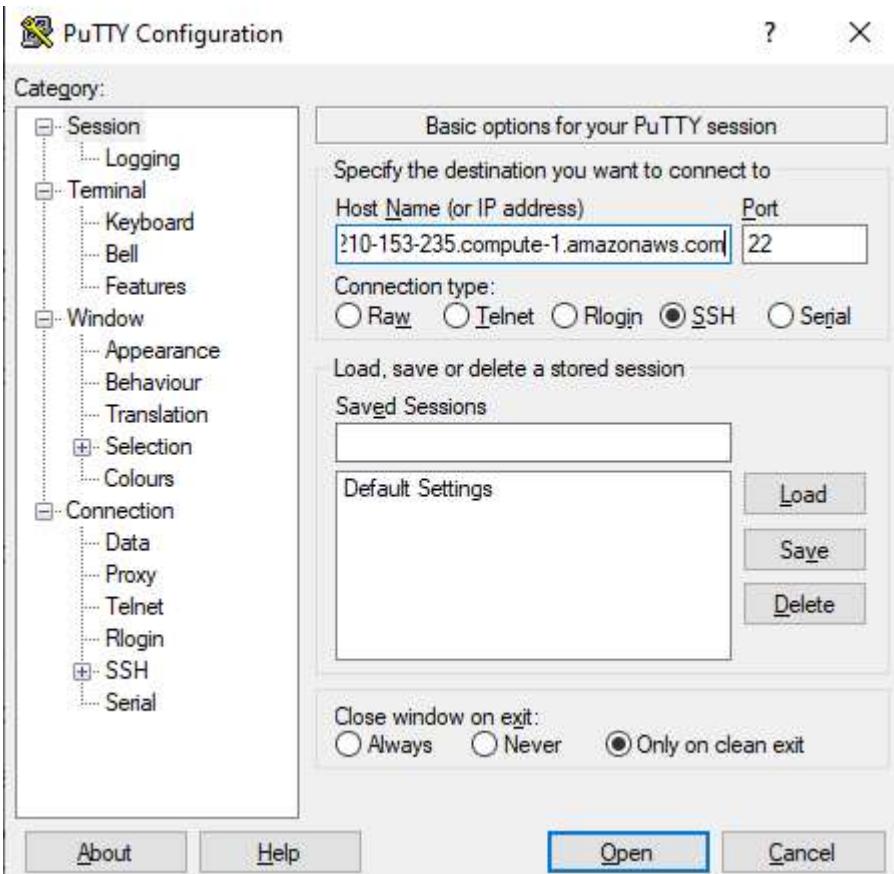
Screenshot of the AWS CloudWatch Metrics console showing a list of EC2 instances and a detailed view of one instance.

Instances Overview:

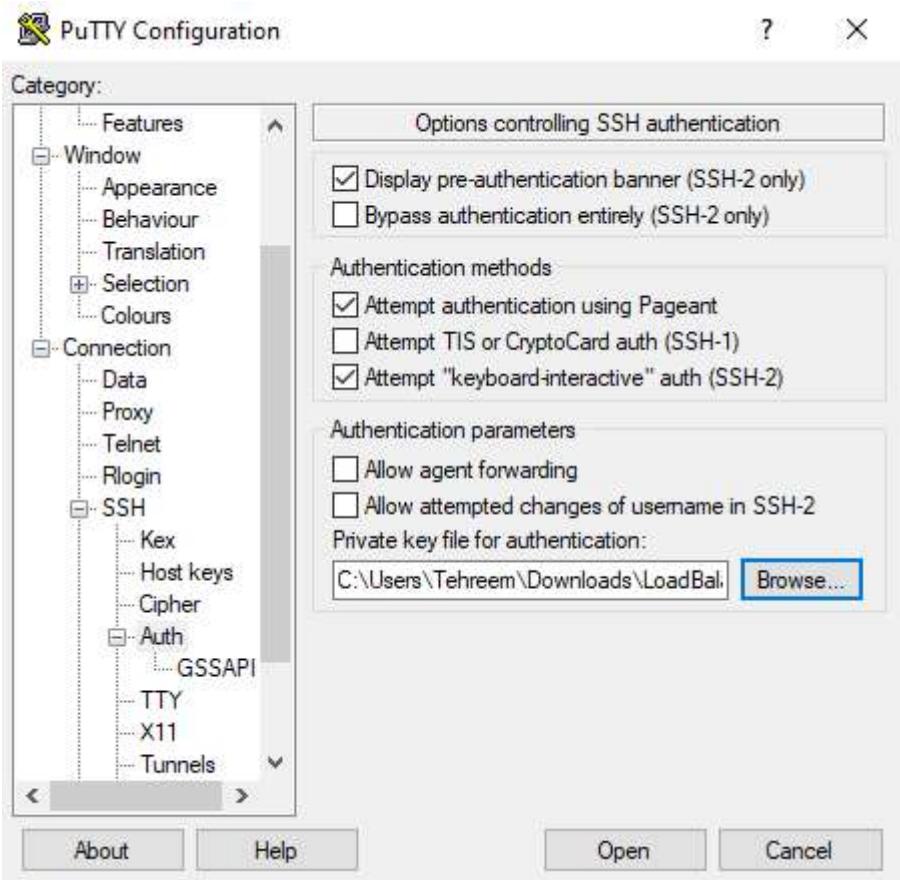
Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)
LoadBalancer	i-0e41636c31bac77c9	t2.micro	us-east-1b	running	2/2 checks ...	None	ec2-54-210-153-235.co...
Server1	i-0ddec3278642ef16b	t2.micro	us-east-1b	running	2/2 checks ...	None	ec2-3-90-236-228.com...
Server2	i-00b136b7a117c7574	t2.micro	us-east-1b	running	2/2 checks ...	None	ec2-54-80-150-192.co...
Server3	i-02c8a9d1904dfcfea	t2.micro	us-east-1d	running	2/2 checks ...	None	ec2-3-83-45-38.comput...
Server4	i-0d787f6d060da86aa	t2.micro	us-east-1b	running	2/2 checks ...	None	ec2-3-80-179-81.com...
	i-0677ae41850c3ff96	t2.micro	us-east-1b	terminated		None	

Detailed Instance View:

Instance ID: i-0e41636c31bac77c9
 Instance state: running
 Instance type: t2.micro
 Finding: You may not have permission to access
 Public DNS (IPv4): ec2-54-210-153-235.compute-1.amazonaws.com
 IPv4 Public IP: 54.210.153.235
 IPv6 IPs: -
 Elastic IPs: -



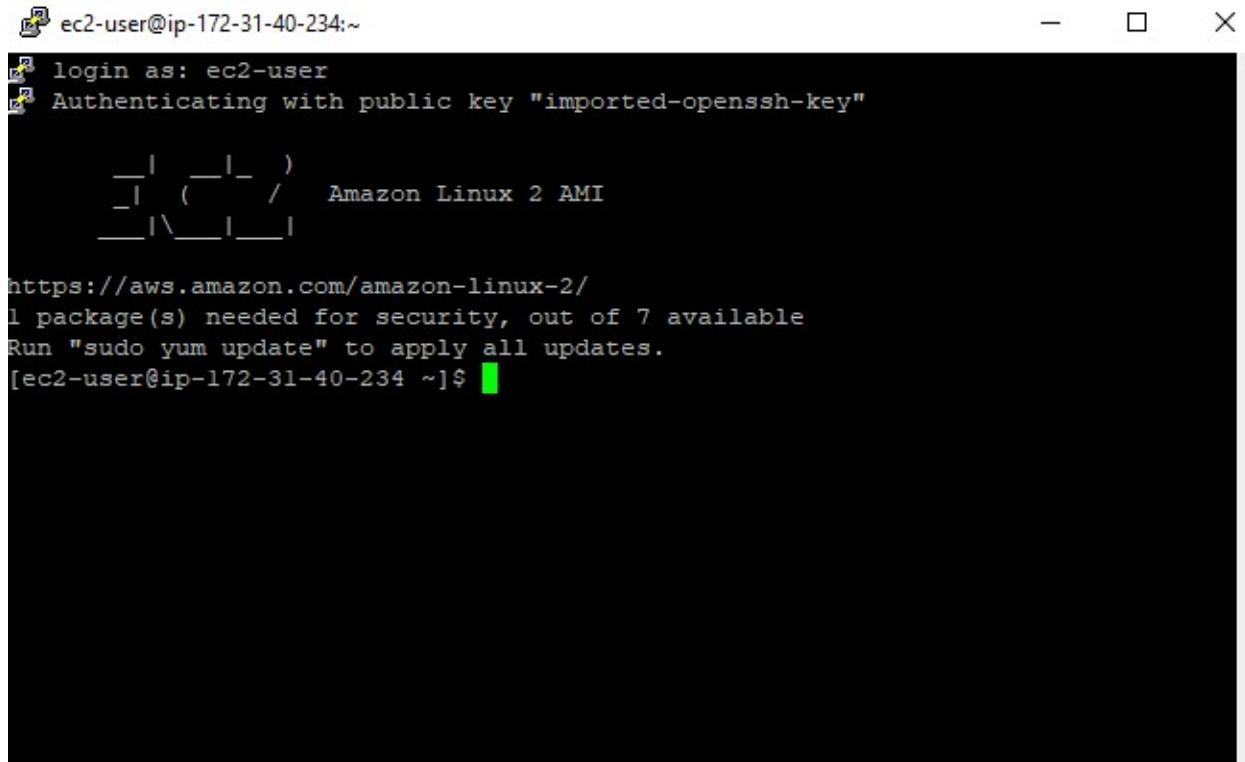
Under Category, Choose Connections and expand it. Now Choose Auth and browse the LoadBalancer.ppk file created in the previous step:



Click Open.

Click Yes on the Security Warning as I trust this instance.

Enter login as ec2-user:



```
[ec2-user@ip-172-31-40-234:~]
[ec2-user@ip-172-31-40-234:~] login as: ec2-user
[ec2-user@ip-172-31-40-234:~] Authenticating with public key "imported-openssh-key"

 _ _|_ ( ___|_ )   Amazon Linux 2 AMI
   \_\_|__|_|

https://aws.amazon.com/amazon-linux-2/
1 package(s) needed for security, out of 7 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-40-234 ~]$
```

Now run this command:

To download Nginx key:

```
sudo wget http://nginx.org/keys/nginx_signing.key
```

```
ec2-user@ip-172-31-40-234:~  
login as: ec2-user  
Authenticating with public key "imported-openssh-key"  
  
Amazon Linux 2 AMI  
  
https://aws.amazon.com/amazon-linux-2/  
1 package(s) needed for security, out of 7 available  
Run "sudo yum update" to apply all updates.  
[ec2-user@ip-172-31-40-234 ~]$ sudo wget http://nginx.org/keys/nginx_signing.key  
--2020-04-03 18:44:20-- http://nginx.org/keys/nginx_signing.key  
Resolving nginx.org (nginx.org)... 62.210.92.35, 95.211.80.227, 2001:laf8:4060:a  
004:21::e3  
Connecting to nginx.org (nginx.org)|62.210.92.35|:80... connected.  
HTTP request sent, awaiting response... 200 OK  
Length: 1561 (1.5K) [text/plain]  
Saving to: 'nginx_signing.key'  
  
100%[=====] 1,561 --.-K/s in 0s  
  
2020-04-03 18:44:20 (178 MB/s) - 'nginx_signing.key' saved [1561/1561]  
[ec2-user@ip-172-31-40-234 ~]$
```

Step 7:

Since sudo apt-key command was not working on my machine, I installed it using amazon-linux-extra:

```
[root@ip-172-31-40-234:~
```

```
 42 php7.4           available [ =stable ]
[root@ip-172-31-40-234 ~]# sudo amazon-linux-extras install nginx1.12
Topic nginx1.12 has end-of-support date of 2019-09-20
Installing nginx
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Cleaning repos: amzn2-core amzn2extra-docker amzn2extra-nginx1.12
12 metadata files removed
4 sqlite files removed
0 metadata files removed
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core                                | 2.4 kB  00:00
amzn2extra-docker                          | 1.8 kB  00:00
amzn2extra-nginx1.12                      | 1.3 kB  00:00
(1/6): amzn2-core/2/x86_64/group_gz      | 2.5 kB  00:00
(2/6): amzn2-core/2/x86_64/updateinfo    | 199 kB  00:00
(3/6): amzn2extra-docker/2/x86_64/updateinfo | 69 B   00:00
(4/6): amzn2extra-docker/2/x86_64/primary_db | 64 kB   00:00
(5/6): amzn2extra-nginx1.12/2/x86_64/primary_db | 23 kB   00:00
(6/6): amzn2-core/2/x86_64/primary_db     | 39 MB   00:00
Resolving Dependencies
--> Running transaction check
--> Package nginx.x86_64 1:1.12.2-2.amzn2.0.2 will be installed
--> Processing Dependency: nginx-filesystem = 1:1.12.2-2.amzn2.0.2 for package:
1:nginx-1.12.2-2.amzn2.0.2.x86_64
--> Processing Dependency: nginx-all-modules = 1:1.12.2-2.amzn2.0.2 for package:
1:nginx-1.12.2-2.amzn2.0.2.x86_64
--> Processing Dependency: nginx-filesystem for package: 1:nginx-1.12.2-2.amzn2.
0.2.x86_64
--> Processing Dependency: libprofiler.so.0()(64bit) for package: 1:nginx-1.12.2
-2.amzn2.0.2.x86_64
--> Running transaction check
--> Package gperftools-libs.x86_64 0:2.6.1-1.amzn2 will be installed
--> Package nginx-all-modules.noarch 1:1.12.2-2.amzn2.0.2 will be installed
--> Processing Dependency: nginx-mod-stream = 1:1.12.2-2.amzn2.0.2 for package:
1:nginx-all-modules-1.12.2-2.amzn2.0.2.noarch
--> Processing Dependency: nginx-mod-mail = 1:1.12.2-2.amzn2.0.2 for package: 1:
nginx-all-modules-1.12.2-2.amzn2.0.2.noarch
--> Processing Dependency: nginx-mod-http-xslt-filter = 1:1.12.2-2.amzn2.0.2 for
 package: 1:nginx-all-modules-1.12.2-2.amzn2.0.2.noarch
--> Processing Dependency: nginx-mod-http-perl = 1:1.12.2-2.amzn2.0.2 for packag
e: 1:nginx-all-modules-1.12.2-2.amzn2.0.2.noarch
--> Processing Dependency: nginx-mod-http-image-filter = 1:1.12.2-2.amzn2.0.2 fo
r package: 1:nginx-all-modules-1.12.2-2.amzn2.0.2.noarch
--> Processing Dependency: nginx-mod-http-geoip = 1:1.12.2-2.amzn2.0.2 for packa
```

```

Dependencies Resolved

=====
Package          Arch    Version      Repository      Size
=====
Installing:
nginx           x86_64  1:1.12.2-2.amzn2.0.2 amzn2extra-nginx1.12 533 k
Installing for dependencies:
fontconfig       x86_64  2.10.95-11.amzn2.0.2 amzn2-core        231 k
fontpackages-filesystem noarch 1.44-8.amzn2      amzn2-core        10 k
gd              x86_64  2.0.35-26.amzn2.0.2 amzn2-core        147 k
gperftools-libs x86_64  2.6.1-1.amzn2      amzn2-core        274 k
libXll           x86_64  1.6.5-2.amzn2.0.2 amzn2-core        614 k
libXll-common   noarch  1.6.5-2.amzn2.0.2 amzn2-core        164 k
libXau           x86_64  1.0.8-2.1.amzn2.0.2 amzn2-core        29 k
libXpm           x86_64  3.5.12-1.amzn2.0.2 amzn2-core        57 k
libpng           x86_64  2:1.5.13-7.amzn2.0.2 amzn2-core        214 k
libxcb           x86_64  1.12-1.amzn2.0.2 amzn2-core        216 k
libxslt          x86_64  1.1.28-5.amzn2.0.2 amzn2-core        243 k
nginx-all-modules noarch 1:1.12.2-2.amzn2.0.2 amzn2extra-nginx1.12 17 k
nginx-filesystem noarch 1:1.12.2-2.amzn2.0.2 amzn2extra-nginx1.12 17 k
nginx-mod-http-geoip  x86_64 1:1.12.2-2.amzn2.0.2 amzn2extra-nginx1.12 24 k
nginx-mod-http-image-filter
                      x86_64 1:1.12.2-2.amzn2.0.2 amzn2extra-nginx1.12 27 k
nginx-mod-http-perl   x86_64 1:1.12.2-2.amzn2.0.2 amzn2extra-nginx1.12 37 k
nginx-mod-http-xslt-filter
                      x86_64 1:1.12.2-2.amzn2.0.2 amzn2extra-nginx1.12 26 k
nginx-mod-mail      x86_64 1:1.12.2-2.amzn2.0.2 amzn2extra-nginx1.12 55 k
nginx-mod-stream    x86_64 1:1.12.2-2.amzn2.0.2 amzn2extra-nginx1.12 76 k
stix-fonts          noarch 1.1.0-5.amzn2      amzn2-core        1.3 M

Transaction Summary
=====
Install 1 Package (+20 Dependent packages)

Total download size: 4.2 M
Installed size: 11 M
Is this ok [y/d/N]: 

```

*sudo yum install command was not working for my system.

Step 8:

The following screen shows that nginx is running on Load Balancer machine

```
ec2-user@ip-172-31-40-234:~
```

```
https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-172-31-40-234 ~]$ sudo /etc/init.d/nginx start
sudo: /etc/init.d/nginx: command not found
[ec2-user@ip-172-31-40-234 ~]$ sudo yum install nginx
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core
                                         | 2.4 kB     00:00
Package 1:nginx-1.12.2-2.amzn2.0.2.x86_64 already installed and latest version
Nothing to do
[ec2-user@ip-172-31-40-234 ~]$ sudo /etc/init.d/nginx-1.12.2-2.amzn2.0.2.x86_64
sudo: /etc/init.d/nginx-1.12.2-2.amzn2.0.2.x86_64: command not found
[ec2-user@ip-172-31-40-234 ~]$ sudo /etc/init.d/nginx-1.12.2
sudo: /etc/init.d/nginx-1.12.2: command not found
[ec2-user@ip-172-31-40-234 ~]$ sudo amazon-linux-extras
  0 ansible2           available   \
    [ =2.4.2  =2.4.6  =2.8  =stable ]
  2 httpd_modules      available   [ =1.0  =stable ]
  3 memcached1.5       available   \
    [ =1.5.1  =1.5.16  =1.5.17 ]
  4 *nginx1.12=latest  enabled     [ =1.12.2 ]
  5 postgresql9.6       available   [ =9.6.6  =9.6.8 ]
  6 postgresql10        available   [ =10 ]
  8 redis4.0            available   [ =4.0.5  =4.0.10.1 ]
```

On putting public IP Address of LoadBalancer as URL:

So nginx is running successfully on LoadBalancer.



3) Change loadbalancer config file:

Step 1:

Editing the configuration file on LoadBalancer and viewing it:



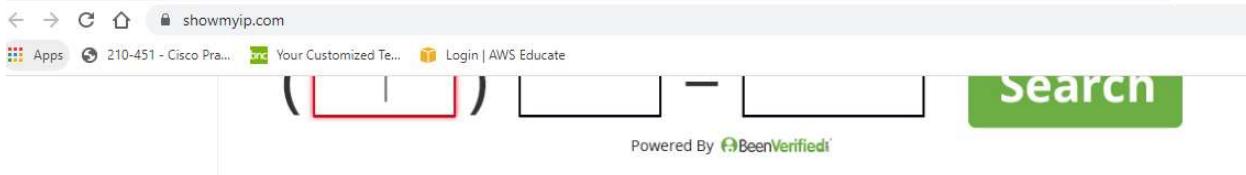
```
root@ip-172-31-40-234:/home/ec2-user
[...]
events { worker_connections 768; }http { upstream myapp {
#ip_hash; server ec2-52-91-113-8.compute-1.amazonaws.com weight=1; server ec2-52-90-49-169.compute-1.amazonaws.com weight=1; server ec2-52-87-243-94.compute-1.amazonaws.com weight=1; server ec2-18-234-178-34.compute-1.amazonaws.com weight=1; } server { listen 80; server_name myapp.com; location / { proxy_pass http://myapp; } }}
```

Load modular configuration files from the /etc/nginx/conf.d directory.
See http://nginx.org/en/docs/ngx_core_module.html#include
for more information.

Step 9:

Since I put my machine address as 10.0.0.13/32; which was not my public IP Address; I was not able to connect to any of the servers.

So now my public IPAddress is:



I changed this by going in the security group of my server 1 , server 2, server 3 and server 4 and now I can connect successfully to my ec2 instance! Now only my machine will be able to access these servers.

Details:	
Your IPv4	76.116.178.250
Your IPv6	2601:81:4101:5280:3c7e:8367:992ef9ae
Country	United States
Region	New Jersey
City	Fords
ZIP	08863

I changed this by going in the security group of my server 1 , server 2, server 3 and server 4 and now I can connect successfully to my ec2 instance! Now only my machine will be able to access these servers.

Inbound rules [Info](#)

Type Info	Protocol Info	Port range Info	Source Info	Description - optional Info
SSH	TCP	22	Custom ▾	<input type="text" value="76.116.178.250"/> <input type="button" value="X"/> 32

[Add rule](#)

⚠ NOTE: Any edits made on existing rules will result in the edited rule being deleted and a new rule created with the new details. This will cause traffic that depends on that rule to be dropped for a very brief period of time until the new rule can be created.

2) Install Nginx on each instance:

Step 1:

Install nginx on servers 1 to 4 using command `sudo amazon-linux-extras install nginx1.12`

Like the one used in step 8. Now all servers have nginx installed and enabled.

Example for server 1:

ec2-user@ip-172-31-32-147:~

```
[ec2-user@ip-172-31-32-147 ~]$ sudo amazon-linux-extras install nginx1.12
Topic nginx1.12 has end-of-support date of 2019-09-20
Installing nginx
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Cleaning repos: amzn2-core amzn2extra-docker amzn2extra-nginx1.12
12 metadata files removed
4 sqlite files removed
0 metadata files removed
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core | 2.4 kB 00:00
amzn2extra-docker | 1.8 kB 00:00
amzn2extra-nginx1.12 | 1.3 kB 00:00
(1/6): amzn2-core/2/x86_64/group_gz | 2.5 kB 00:00
(2/6): amzn2-core/2/x86_64/updateinfo | 199 kB 00:00
(3/6): amzn2extra-nginx1.12/2/x86_64/primary_db | 23 kB 00:00
(4/6): amzn2extra-docker/2/x86_64/primary_db | 64 kB 00:00
(5/6): amzn2extra-docker/2/x86_64/updateinfo | 69 B 00:00
(6/6): amzn2-core/2/x86_64/primary_db | 39 MB 00:00
Resolving Dependencies
--> Running transaction check
--> Package nginx.x86_64 1:1.12.2-2.amzn2.0.2 will be installed
--> Processing Dependency: nginx-filesystem = 1:1.12.2-2.amzn2.0.2 for package:
1:nginx-1.12.2-2.amzn2.0.2.x86_64
--> Processing Dependency: nginx-all-modules = 1:1.12.2-2.amzn2.0.2 for package:
1:nginx-1.12.2-2.amzn2.0.2.x86_64
--> Processing Dependency: nginx-filesystem for package: 1:nginx-1.12.2-2.amzn2.
0.2.x86_64
--> Processing Dependency: libprofiler.so.0() (64bit) for package: 1:nginx-1.12.2
-2.amzn2.0.2.x86_64
--> Running transaction check
--> Package gperftools-libs.x86_64 0:2.6.1-1.amzn2 will be installed
--> Package nginx-all-modules.noarch 1:1.12.2-2.amzn2.0.2 will be installed
--> Processing Dependency: nginx-mod-stream = 1:1.12.2-2.amzn2.0.2 for package:
1:nginx-all-modules-1.12.2-2.amzn2.0.2.noarch
--> Processing Dependency: nginx-mod-mail = 1:1.12.2-2.amzn2.0.2 for package: 1:
nginx-all-modules-1.12.2-2.amzn2.0.2.noarch
--> Processing Dependency: nginx-mod-http-xslt-filter = 1:1.12.2-2.amzn2.0.2 for
package: 1:nginx-all-modules-1.12.2-2.amzn2.0.2.noarch
--> Processing Dependency: nginx-mod-http-perl = 1:1.12.2-2.amzn2.0.2 for packag
e: 1:nginx-all-modules-1.12.2-2.amzn2.0.2.noarch
--> Processing Dependency: nginx-mod-http-image-filter = 1:1.12.2-2.amzn2.0.2 fo
r package: 1:nginx-all-modules-1.12.2-2.amzn2.0.2.noarch
--> Processing Dependency: nginx-mod-http-geoip = 1:1.12.2-2.amzn2.0.2 for packa
ge: 1:nginx-all-modules-1.12.2-2.amzn2.0.2.noarch
```

This below screen shows it is enabled/running:

```

ec2-user@ip-172-31-32-147:~ Verifying : 1:nginx-mod-http-geoip-1.12.2-2.amzn2.0.2.x86_64
Verifying : 1:nginx-mod-http-perl-1.12.2-2.amzn2.0.2.x86_64
Verifying : 1:nginx-mod-stream-1.12.2-2.amzn2.0.2.x86_64
Verifying : 1:nginx-mod-mail-1.12.2-2.amzn2.0.2.x86_64
Verifying : 1:nginx-filesystem-1.12.2-2.amzn2.0.2.noarch
Verifying : 1:nginx-1.12.2-2.amzn2.0.2.x86_64
Verifying : libXau-1.0.8-2.1.amzn2.0.2.x86_64
Verifying : libXll-common-1.6.5-2.amzn2.0.2.noarch
Verifying : fontconfig-2.10.95-11.amzn2.0.2.x86_64
Verifying : libXpm-3.5.12-1.amzn2.0.2.x86_64
Verifying : 2:libpng-1.5.13-7.amzn2.0.2.x86_64

Installed:
  nginx.x86_64 1:1.12.2-2.amzn2.0.2

Dependency Installed:
  fontconfig.x86_64 0:2.10.95-11.amzn2.0.2
  gperftools-libs.x86_64 0:2.6.1-1.amzn2
  libXau.x86_64 0:1.0.8-2.1.amzn2.0.2
  libxcb.x86_64 0:1.12-1.amzn2.0.2
  nginx-filesystem.noarch 1:1.12.2-2.amzn2.0.2
  nginx-mod-http-perl.x86_64 1:1.12.2-2.amzn2.0.2
  nginx-mod-stream.x86_64 1:1.12.2-2.amzn2.0.2
  fontpackages-filesystem.noarch 0:1.44-8.amzn2
  libXll.x86_64 0:1.6.5-2.amzn2.0.2
  libXpm.x86_64 0:3.5.12-1.amzn2.0.2
  libxslt.x86_64 0:1.1.28-5.amzn2.0.2
  nginx-mod-http-geoip.x86_64 1:1.12.2-2.amzn2.0.2
  nginx-mod-http-xslt-filter.x86_64 1:1.12.2-2.amzn2.0.2
  stix-fonts.noarch 0:1.1.0-5.amzn2

Complete!
  0 ansible2           available   \
    [ =2.4.2  =2.4.6  =2.8  =stable ]
  2 httpd_modules       available   [ =1.0  =stable ]
  3 memcached1.5        available   \
    [ =1.5.1  =1.5.16  =1.5.17 ]
  4 *nginx1.12=latest   enabled     [ =1.12.2 ]
  5 postgresql9.6        available   [ =9.6.6  =9.6.8 ]
  6 postgresql10         available   [ =10 ]
  8 redis4.0            available   [ =4.0.5  =4.0.10 ]
  9 R3.4                 available   [ =3.4.3  =stable ]
 10 rustl                available   \
    [ =1.22.1  =1.26.0  =1.26.1  =1.27.2  =1.31.0  =1.38.0 ]
 11 vim                  available   [ =8.0 ]
 13 ruby2.4              available   \
    [ =2.4.2  =2.4.4  =2.4.7  =stable ]
 15 php7.2              available   \
    [ =7.2.0  =7.2.4  =7.2.5  =7.2.8  =7.2.11  =7.2.13  =7.2.14
      =7.2.16  =7.2.17  =7.2.19  =7.2.21  =7.2.22  =7.2.23

```

Testing if nginx has started on server 1:

In browser, type <http://ec2-52-91-113-8.compute-1.amazonaws.com>

In case of server 1, ec2-52-91-113-8.compute-1.amazonaws.com is the public DNS.

```

root@ip-172-31-43-176:/home/ec2-user
[ =1.8.0_192  =1.8.0_202  =1.8.0_212  =1.8.0_222  =1.8.0_232
  =1.8.0_242 ]
28 firecracker      available   [ =0.11  =stable ]
29 golang1.11       available   \
  [ =1.11.3  =1.11.11  =1.11.13  =stable ]
30 squid4           available   [ =4 ]
31 php7.3           available   \
  [ =7.3.2  =7.3.3  =7.3.4  =7.3.6  =7.3.8  =7.3.9  =7.3.10
    =7.3.11  =7.3.13  =stable ]
32 lustre2.10       available   \
  [ =2.10.5  =2.10.8  =stable ]
33 java-openjdk11   available   [ =11  =stable ]
34 lynis             available   [ =stable ]
35 kernel-ng         available   [ =stable ]
36 BCC               available   [ =0.x ]
37 mono              available   [ =5.x ]
38 nginx1           available   [ =stable ]
39 ruby2.6           available   [ =2.6  =stable ]
40 mock              available   [ =stable ]
41 postgresql11     available   [ =11  =stable ]
42 php7.4            available   [ =stable ]
* Extra topic has reached end of support.
[root@ip-172-31-43-176 ec2-user]# nginx
[root@ip-172-31-43-176 ec2-user]#

```

Security Groups (1/1) [Info](#)

[C](#) Actions ▾ [Create security group](#)

Filter security groups

search: sg-0c1557e5d4a0f89b2 X Clear filters

<input checked="" type="checkbox"/>	Security group ID	▲ Security group name	VPC ID	Description	Owner
<input checked="" type="checkbox"/>	sg-0c1557e5d4a0f89b2	launch-wizard-8	vpc-cc0435b6	launch-wizard-8 create...	11261947

Type	Protocol	Port range	Source	Description - optional
HTTP	TCP	80	0.0.0.0/0	-
HTTP	TCP	80	::/0	-
SSH	TCP	22	76.116.178.250/32	-

For some reason in my machine, in order to start nginx I had to go to root using sudo su command and then type command nginx and then only the web-page loaded! Also, I made sure to add port 80 in inbound security rules for my instances.

A screenshot of a web browser window. The address bar shows 'Not secure | ec2-52-90-49-169.compute-1.amazonaws.com'. Below the address bar, there are links for 'Apps', '210-451 - Cisco Pra...', 'Your Customized Te...', 'Login | AWS Educate', and a star icon. The main content area has a dark blue header with the text 'Welcome to nginx on Amazon Linux!'. Below the header, a message states: 'This page is used to test the proper operation of the nginx HTTP server after it has been installed. If you can read this page, it means that the web server installed at this site is working properly.' A blue box contains the text 'Website Administrator'. At the bottom, there is a note about the default index.html page and instructions to edit the configuration file.

Repeat this step for all servers 1,2,3, and 4. And make sure nginx is working by inputting public DNS in the browser for each one of them!

Step 2:

Now index.html has to be edited for Server1:

```
[?] login as: ec2-user
[?] Authenticating with public key "imported-openssh-key"
Last login: Sat Apr  4 19:24:05 2020 from 76.116.178.250

      _\   _\   )
      _\  ( _\  /   Amazon Linux 2 AMI
      _\ \_\_|\__|_


https://aws.amazon.com/amazon-linux-2/
1 package(s) needed for security, out of 7 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-32-147 ~]$ sudo su
[root@ip-172-31-32-147 ec2-user]# vi /usr/share/nginx/html/index.html
[root@ip-172-31-32-147 ec2-user]# ^C
[root@ip-172-31-32-147 ec2-user]# vi /usr/share/nginx/html/index.html
[root@ip-172-31-32-147 ec2-user]# vi /usr/share/nginx/html/index.html
[root@ip-172-31-32-147 ec2-user]# ~
bash: /root: Is a directory
[root@ip-172-31-32-147 ec2-user]# vi /usr/share/nginx/html/index.html
[root@ip-172-31-32-147 ec2-user]# vi /usr/share/nginx/html/index.html
[root@ip-172-31-32-147 ec2-user]# vi /usr/share/nginx/html/index.html
[root@ip-172-31-32-147 ec2-user]# vi /usr/share/nginx/html/index.html
[root@ip-172-31-32-147 ec2-user]#
```

Write permissions were not given in ec2user mode so I switched to root mode:

Edited the index.html file to include Server 1 header in h1 tag:

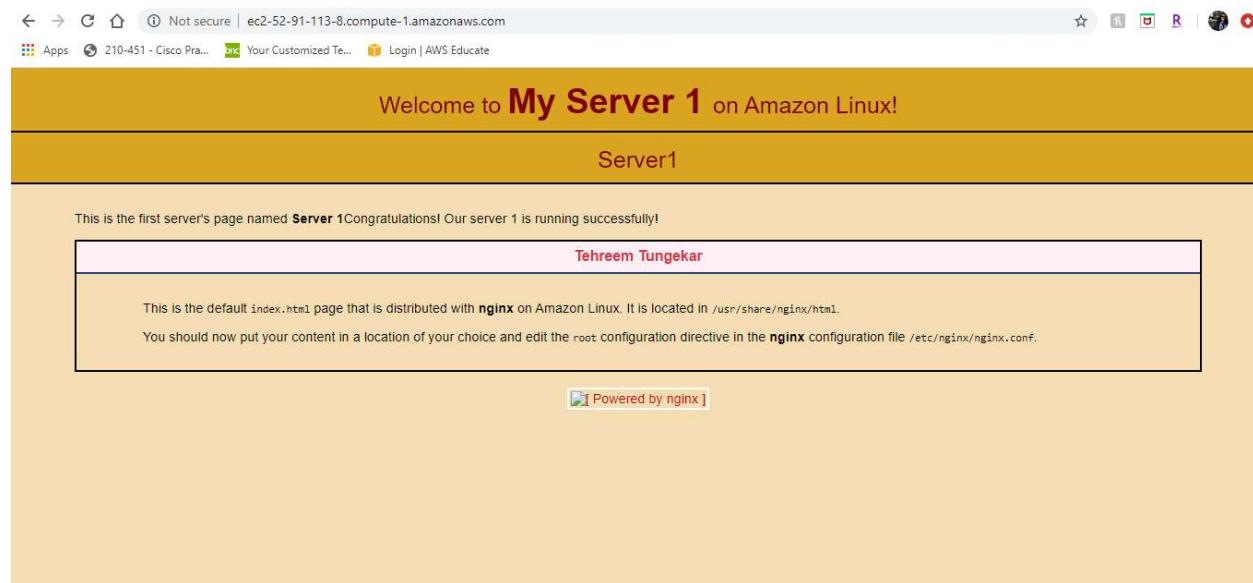
```
root@ip-172-31-32-147:/home/ec2-user
hr {
    display: none;
}
.content {
    padding: 1em 5em;
}
.alert {
    border: 2px solid #000;
}

img {
    border: 2px solid #fff;
    padding: 2px;
    margin: 2px;
}
a:hover img {
    border: 2px solid #294172;
}
.logos {
    margin: 1em;
    text-align: center;
}
/*]]>/*
</style>
</head>

<body>
    <h1>Welcome to <strong>My Server 1 </strong> on Amazon Linux!</h1>

<header><h1>Server1</h1></header>
<div class="content">
    <p>This is the first server's page named
    <strong>Server 1</strong>Congratulations! Our server 1 is running successfully!</p>

    <div class="alert">
        <h2>Tehreem Tungekar</h2>
        <div class="content">
            <p>This is the default <tt>index.html</tt> page that
            is distributed with <strong>nginx</strong> on
            Amazon Linux. It is located in
            <tt>/usr/share/nginx/html</tt>.</p>
        <p>You should now put your content in a location of</p>
    </div>
</div>
```



Not secure | ec2-52-90-49-169.compute-1.amazonaws.com

Apps 210-451 - Cisco Pr... Your Customized Te... Login | AWS Educate

Welcome to My Server 2 on Amazon Linux!

Server 2

This page is used to test the proper operation of the **nginx** HTTP server after it has been installed. If you can read this page, it means that the web server installed at this site is working properly.

Tehreem Tungekar

This is the default index.html page that is distributed with **nginx** on Amazon Linux. It is located in /usr/share/nginx/html.
You should now put your content in a location of your choice and edit the root configuration directive in the **nginx** configuration file /etc/nginx/nginx.conf.

[Powered by nginx]

Not secure | ec2-52-87-243-94.compute-1.amazonaws.com

Apps 210-451 - Cisco Pr... Your Customized Te... Login | AWS Educate

Welcome to My Server 3 on Amazon Linux!

Server 3

This page is used to test the proper operation of the **nginx** HTTP server after it has been installed. If you can read this page, it means that the web server installed at this site is working properly.

Tehreem Tungekar

This is the default index.html page that is distributed with **nginx** on Amazon Linux. It is located in /usr/share/nginx/html.
You should now put your content in a location of your choice and edit the root configuration directive in the **nginx** configuration file /etc/nginx/nginx.conf.

[Powered by nginx]



Step 3:

Using \$curl dnsnameofloadbalancer:

```
[ec2-user@ip-172-31-40-234:~]
[ec2-user@ip-172-31-40-234 ~]$ curl ec2-54-210-153-235.compute-1.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 Amazon Linux</title>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <style type="text/css">
        /*<![CDATA[*]>
        body {
background-color: wheat;

        font-size: 0.9em;
        font-family: sans-serif,helvetica;
        margin: 0;
        padding: 0;
    }
    :link {
        color: #c00;
    }
    :visited {
        color: #c00;
    }
    a:hover {
        color: #f50;
    }
    h1 {
        text-align: center;
        margin: 0;
        padding: 0.6em 2em 0.4em;
        background-color:goldenrod;
color:maroon;

        font-weight: normal;
        font-size: 1.75em;
        border-bottom: 2px solid #000;
    }
    h1 strong {
```

As it can be seen, first , load is sent to Server 2 .

```
ec2-user@ip-172-31-40-234:~
```

```
        margin: 0;
        padding: 0.5em;
        border-bottom: 2px solid #294172;
    }
    hr {
        display: none;
    }
    .content {
        padding: 1em 5em;
    }
    .alert {
        border: 2px solid #000;
    }

    img {
        border: 2px solid #ffff;
        padding: 2px;
        margin: 2px;
    }
    a:hover img {
        border: 2px solid #294172;
    }
    .logos {
        margin: 1em;
        text-align: center;
    }
    /*]]>/*
</style>
</head>

<body>

<h1>Welcome to <strong>My Server 2 </strong> on Amazon Linux!</h1>
<header><h1>Server 2</h1></header>
<div class="content">
    <p>This page is used to test the proper operation of the
    <strong>nginx</strong> HTTP server after it has been
    installed. If you can read this page, it means that the
    web server installed at this site is working
    properly.</p>
    <div class="alert">
        <h2>Tehreem Tungekar</h2>
        <div class="content">
```

Next time, load is sent to Server 3:

```

ec2-user@ip-172-31-40-234:~ 
border: 2px solid #000;
}

img {
    border: 2px solid #fff;
    padding: 2px;
    margin: 2px;
}
a:hover img {
    border: 2px solid #294172;
}
.logos {
    margin: 1em;
    text-align: center;
}
/*]]> */
</style>
</head>

<body>
<h1>Welcome to <strong>My Server 3 </strong> on Amazon Linux!</h1>
<header><h1>Server 3</h1></header>

<div class="content">
<p>This page is used to test the proper operation of the
<strong>nginx</strong> HTTP server after it has been
installed. If you can read this page, it means that the
web server installed at this site is working
properly.</p>

<div class="alert">
<h2>Tehreem Tungekar</h2>
<div class="content">
<p>This is the default <tt>index.html</tt> page that
is distributed with <strong>nginx</strong> on
Amazon Linux. It is located in
<tt>/usr/share/nginx/html</tt>.</p>

<p>You should now put your content in a location of
your choice and edit the <tt>root</tt> configuration
directive in the <strong>nginx</strong>
configuration file

```

4) Collect information about visits to your site:

Step 1:

```

#!/usr/bin/env ruby
#
# This program is used for collecting web server visit information.
#
# Author: A. Genius
#
require 'optparse'
def print_usage
  puts "USAGE: visit_server -d DNS_NAME"
  exit
end
# add option switch and handler
options = {}
option_parser = OptionParser.new do |opts|
  # DNS_NAME argument
  options[:dns_name] = nil
  opts.on('-d', '--dns-name DNS_NAME', 'Specify a DNS NAME') { |dns_name|

```

```

options[:dns_name] = dns_name }
# HELP argument
options[:help] = nil
opts.on('-h', '--help', 'Display usage') { |help| options[:help] = help }
end
option_parser.parse!
# verify arguments
if options[:dns_name] then
  dns_name = options[:dns_name]
else
  puts "Please set a balancer's DNS."
  print_usage
  exit
end
if options[:help] then
  print_usage
  exit
end
# Keep STDOUT
#orig_stdout = $stdout
# redirect stdout to /dev/null
#$stdout = File.new('/dev/null', 'w')
server1_visit_count = 0
server2_visit_count = 0
server3_visit_count = 0
server4_visit_count = 0
# starting to visit load balancing server
puts "Starting to visit load balancing server"
2000.times do
  # visit load balancer
  #o = `curl #{dns_name}`
  o = `curl -s #{dns_name}`
  if o =~ /server\s*1/i
    server1_visit_count += 1
  elsif o =~ /server\s*2/i
    server2_visit_count += 1
  elsif o =~ /server\s*3/i
    server3_visit_count += 1
  elsif o =~ /server\s*4/i
    server4_visit_count += 1
  end
  print "."
end
# redirect output to stdout
#$stdout = orig_stdout
# print visit information

```

```
puts
puts '-----'
puts ' Summary'
puts '-----'
puts "Server1 visit counts : " + server1_visit_count.to_s
puts "Server2 visit counts : " + server2_visit_count.to_s
puts "Server3 visit counts : " + server3_visit_count.to_s
puts "Server4 visit counts : " + server4_visit_count.to_s
puts "Total visit counts : " + (server1_visit_count + server2_visit_count + server3_visit_count +
server4_visit_count).to_s
```

Installed ruby 2.6, as it can be seen in the below screen and wrote a program using command \$ vim visit_server and pasted the above highlighted code in it.

Scenario 1:

 root@ip-172-31-40-234:/home/ec2-user

```
events {
    worker_connections 768;
}

http {
    upstream myapp {
        #ip_hash;
        server ec2-52-91-113-8.compute-1.amazonaws.com weight=1;
        server ec2-52-90-49-169.compute-1.amazonaws.com weight=1;
        server ec2-52-87-243-94.compute-1.amazonaws.com weight=1;
        server ec2-18-234-178-34.compute-1.amazonaws.com weight=2;
    }
    server {
        listen 80;
        server_name myapp.com;
        location / {
            proxy_pass http://myapp;
        }
    }
}
```

As it can be seen from the above screen, since in this scenario, equal weights are assigned, 1,1,1 and 1 to all the servers, load is equally distributed amongst all the servers as 500 each.

Extra Scenario:

Now we need to change the configuration file of load balancer server in order to assign unequal weights to all servers and then we can see the output:

Firstly, modifying the conf file:

```
[root@ip-172-31-40-234:~]# su - ec2-user
[ec2-user@ip-172-31-40-234 ~]$ vi /etc/nginx/nginx.conf
```

```
root@ip-172-31-40-234:/home/ec2-user
events {
    worker_connections 768;
}
http {
    upstream myapp {
        #ip_hash;
        server ec2-52-91-113-8.compute-1.amazonaws.com weight=1;
        server ec2-52-90-49-169.compute-1.amazonaws.com weight=3;
        server ec2-52-87-243-94.compute-1.amazonaws.com weight=2;
        server ec2-18-234-178-34.compute-1.amazonaws.com weight=1;
    }
    server {
        listen 80;
        server_name myapp.com;
        location / {
            proxy_pass http://myapp;
        }
    }
}
```

In this case, Server 1 and Server 4 are assigned weight 1, server 2 is assigned weight 3 and server 3 is assigned weight 2.

Now checking using visit server script: ruby visit_server -d ec2-54-210-153-235.compute-1.amazonaws.com

First, since the config file has been modified, restart the nginx service using command:

```
sudo systemctl restart nginx
```

Then check using ruby script:

```
[root@ip-172-31-40-234 ~]# /home/ec2-user/nginx -v
nginx: [emerg] bind() to 0.0.0.0:80 failed (98: Address already in use)
nginx: [emerg] bind() to 0.0.0.0:80 failed (98: Address already in use)
nginx: [emerg] bind() to 0.0.0.0:80 failed (98: Address already in use)
nginx: [emerg] still could not bind()
[root@ip-172-31-40-234 ~]# nginx -v
nginx version: nginx/1.12.2
[root@ip-172-31-40-234 ~]# sudo systemctl restart nginx
[root@ip-172-31-40-234 ~]# ruby visit_server -d ec2-54-210-153-235.compute-1.amazonaws.com
Starting to visit load balancing server
.
.
.
-----  
Summary-----  
Server1 visit counts : 286  
Server2 visit counts : 857  
Server3 visit counts : 571  
Server4 visit counts : 286  
Total visit counts : 2000
[root@ip-172-31-40-234 ~]#
```

Since the weight is unevenly distributed, the counts have also been unevenly distributed among the four servers.

Scenario 2:

Using command: **vi /etc/nginx/nginx.conf**

Assign weight 1 to server 1, weight 2 to server 2, weight 3 to server 3 and weight 4 to server 4;

 root@ip-172-31-40-234:/home/ec2-user

```
events {
    worker_connections 768;
}

http {
    upstream myapp {
        #ip_hash;
        server ec2-52-91-113-8.compute-1.amazonaws.com weight=1;
        server ec2-52-90-49-169.compute-1.amazonaws.com weight=2;
        server ec2-52-87-243-94.compute-1.amazonaws.com weight=3;
        server ec2-18-234-178-34.compute-1.amazonaws.com weight=4;
    }
    server {
        listen 80;
        server_name myapp.com;
        location / {
            proxy_pass http://myapp;
        }
    }
}
```

Restarting the service and checking the visits:

 root@ip-172-31-40-234:/home/ec2-user
[root@ip-172-31-40-234 ec2-user]# sudo systemctl restart nginx
[root@ip-172-31-40-234 ec2-user]# ruby visit_server -d ec2-54-210-153-235.compute-1.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
[root@ip-172-31-40-234 ec2-user]#

Since more weight is assigned to Server 4, it gets the highest counts, i.e. 800 counts, followed by server 3 getting 600 counts as it has weight 3; server 2 gets 600 counts as it has weight 2 and the least counts are given to server 1 as it has the minimum weight 1 assigned to it.

Scenario 3:

In this scenario, server 1 and server 3 get weight 1, server 2 and server 4 get weight 2.

 root@ip-172-31-40-234:/home/ec2-user

```
events {
    worker_connections 768;
}
http {
    upstream myapp {
        #ip_hash;
        server ec2-52-91-113-8.compute-1.amazonaws.com weight=1;
        server ec2-52-90-49-169.compute-1.amazonaws.com weight=2;
        server ec2-52-87-243-94.compute-1.amazonaws.com weight=1;
        server ec2-18-234-178-34.compute-1.amazonaws.com weight=2;
    }
    server {
        listen 80;
        server_name myapp.com;
        location / {
            proxy_pass http://myapp;
        }
    }
}
```

Saving this file and restarting the nginx service.

Visit server script shows:

As it can be seen, Server 1 and 3 which have a weight 1 assigned get 333 counts each. While servers 2 and 4 get double the count as compared to servers 1 and 3 because their weights are double.

Additional Steps:

Lastly, terminate all instances:

The screenshot shows the AWS Management Console with the EC2 service selected. The left sidebar lists navigation options like EC2 Dashboard, Events, Tags, Reports, Limits, Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Scheduled Instances, Capacity Reservations, and Images. The main content area displays a table of EC2 instances. The table has columns for Name, Instance ID, Instance Type, Availability Zone, Instance State, Status Checks, Alarm Status, and Public DNS (IPv4). Five instances are listed: LoadBalancer (stopped), Server1 (stopped), Server2 (stopped), Server3 (stopped), and Server4 (stopped). Each instance row includes a checkbox, a color-coded status indicator, and a yellow gear icon for actions. A search bar at the top of the table allows filtering by tags and attributes or keyword. The bottom of the page shows a summary of selected instances and a navigation bar with links for Description, Status Checks, Monitoring, and Tags.

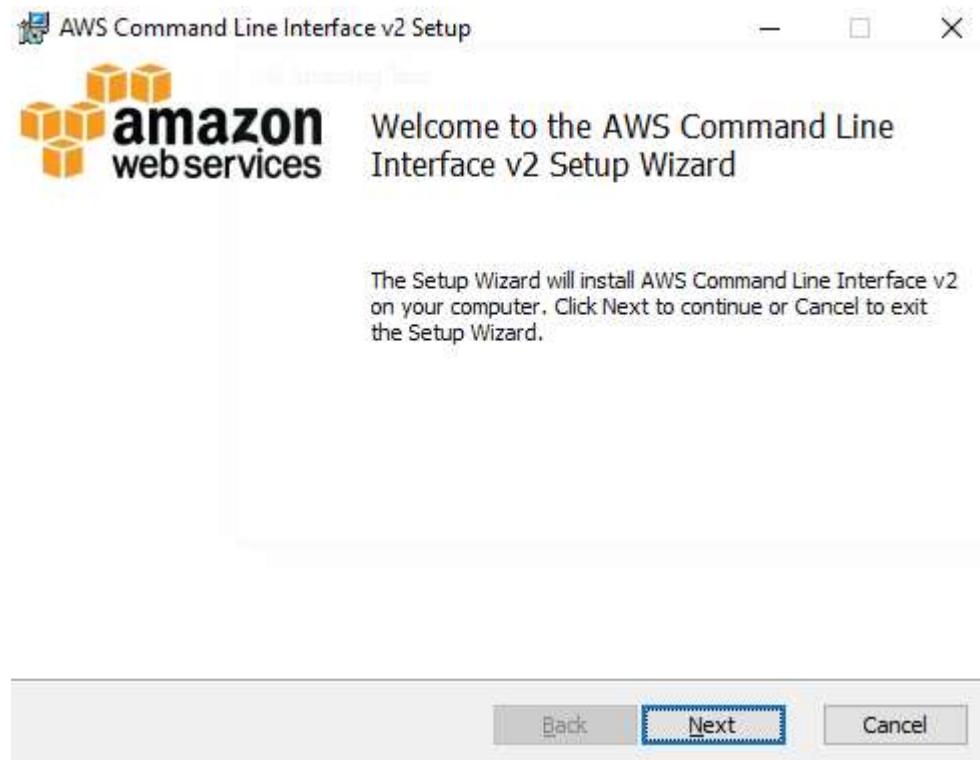
	Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)
<input type="checkbox"/>	LoadBalancer	i-0e41636c31bac77c9	t2.micro	us-east-1b	stopped	None	None	
<input checked="" type="checkbox"/>	Server1	i-0ddec3278642ef16b	t2.micro	us-east-1b	stopped	None	None	
<input checked="" type="checkbox"/>	Server2	i-00b136b7a117c7574	t2.micro	us-east-1b	stopped	None	None	
<input checked="" type="checkbox"/>	Server3	i-02c8a9d1904dfcfea	t2.micro	us-east-1d	stopped	None	None	
<input checked="" type="checkbox"/>	Server4	i-0d787f6d060da86aa	t2.micro	us-east-1b	stopped	None	None	

Steps for creating EC2 instance through CLI:

Step 1:

Download the AWS CLI MSI installer for Windows (64-bit) at
<https://awscli.amazonaws.com/AWSCLIV2.msi>

Run the downloaded file



Click Next, Accept the terms, Check the Path and click Install.

Step 2: Creating IAM access key:

Search IAM in AWS Console :

← → ⌂ ⌂ console.aws.amazon.com/iam/home?region=us-east-1#/home

Apps 210-451 - Cisco Pr... Your Customized Te... Login | AWS Educate

aws Services Resource Groups

Identity and Access Management (IAM)

Dashboard

Access management

Groups

Users

Roles

Policies

Identity providers

Account settings

Access reports

Access analyzer

Archive rules

Analyzers

Settings

Credential report

Welcome to Identity and Access Management

We encountered the following errors while processing your request:

User: am:aws sts::112619478712.assumed-role/vocstartsoft/user623773=tungeka@stevens.edu is not authorized to perform: iam>ListAccountAliases on resource: * with an explicit deny

IAM Resources

Users: 0 Roles: 12

Groups: 0 Identity Providers: 0

Customer Managed Policies: 0

Security Status 0 out of 4 complete.

Activate MFA on your root account

Create individual IAM users

Use groups to assign permissions

Apply an IAM password policy

Feature Spotlight

Introduction to AWS IAM

Additional Information

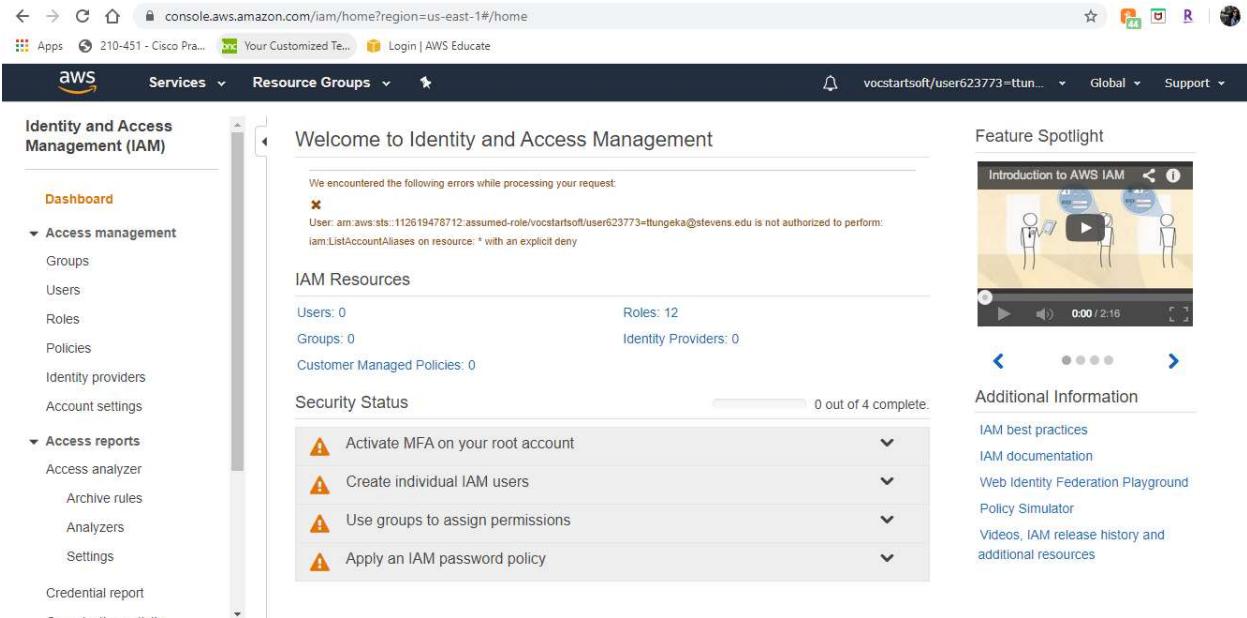
IAM best practices

IAM documentation

Web Identity Federation Playground

Policy Simulator

Videos, IAM release history and additional resources



Click Users in the left side:

← → ⌂ ⌂ console.aws.amazon.com/iam/home?region=us-east-1#/users

Apps 210-451 - Cisco Pr... Your Customized Te... Login | AWS Educate

aws Services Resource Groups

Identity and Access Management (IAM)

Dashboard

Access management

Groups

Users

Roles

Policies

Identity providers

Account settings

Access reports

Access analyzer

Archive rules

Analyzers

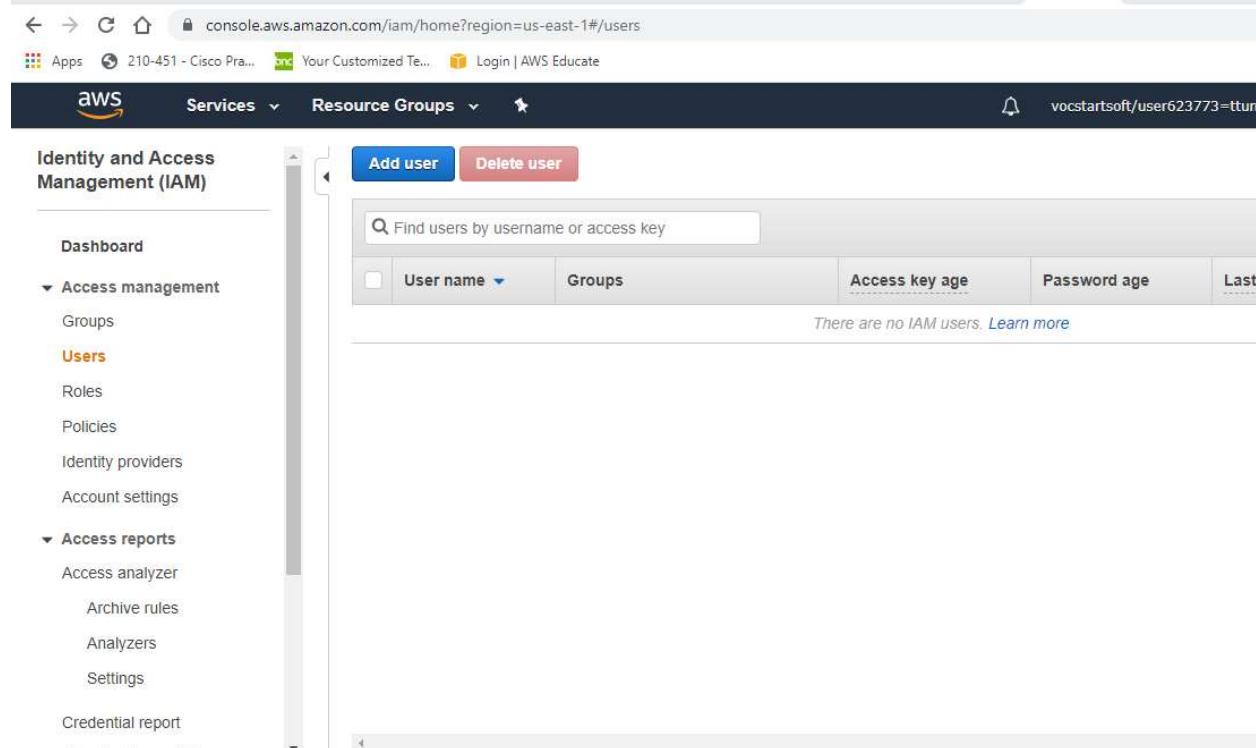
Settings

Credential report

Add user Delete user

Find users by username or access key

<input type="checkbox"/>	User name	Groups	Access key age	Password age	Last
There are no IAM users. Learn more					



Click on Add User:

← → ⌂ ⌂ console.aws.amazon.com/iam/home?region=us-east-1#/users\$new?step=details

Apps 210-451 - Cisco Pra... Your Customized Te... Login | AWS Educate

Services Resource Groups

Set user details

You can add multiple users at once with the same access type and permissions. [Learn more](#)

User name* tehreemtungekar

[+ Add another user](#)

Select AWS access type

Select how these users will access AWS. Access keys and autogenerated passwords are provided in the last step. [Learn more](#)

Access type* Programmatic access
Enables an **access key ID** and **secret access key** for the AWS API, CLI, SDK, and other development tools.

AWS Management Console access
Enables a **password** that allows users to sign-in to the AWS Management Console.

Console password* Autogenerated password Custom password

...
 Show password

* Required [Cancel](#) [Next: Permissions](#)

Click on Next after giving a username and creating a password.

Since I don't want to create any group, I clicked Next, also, I did not create any Tags

← → ⌂ ⌂ console.aws.amazon.com/iam/home?region=us-east-1#/users\$new?step=review&accessKey&login&userNames=tehreemtungekar&passwordReset&... ⌂ ⌂

Apps 210-451 - Cisco Pra... Your Customized Te... Login | AWS Educate

Services Resource Groups

Review

Review your choices. After you create the user, you can view and download the autogenerated password and access key.

User details

User name	tehreemtungekar
AWS access type	Programmatic access and AWS Management Console access
Console password type	Custom
Require password reset	Yes
Permissions boundary	Permissions boundary is not set

Permissions summary

The user shown above will be added to the following groups.

Type	Name
Managed policy	IAMUserChangePassword

Tags

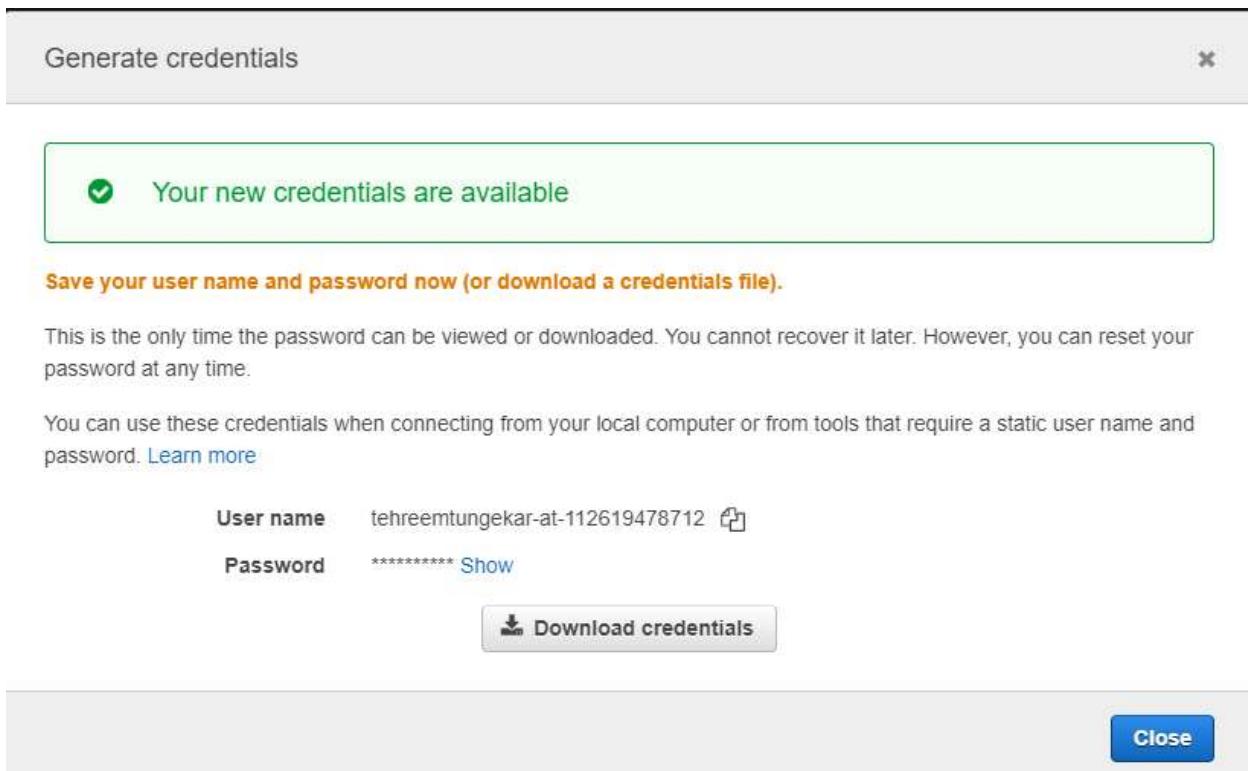
[Cancel](#) [Previous](#) [Create user](#)

Click on Create user

The screenshot shows the AWS IAM User Management interface. At the top, there are buttons for 'Add user' and 'Delete user'. A search bar at the top left contains the placeholder 'Find users by username or access key'. On the right, it says 'Showing 1 result'. Below the search bar is a table with columns: 'User name' (sorted), 'Groups', 'Access key age', 'Password age', 'Last activity', and 'MFA'. One row is visible, showing 'tehreemtungekar' under 'User name', 'None' under 'Groups', 'None' under 'Access key age', 'None' under 'Password age', 'None' under 'Last activity', and 'None' under 'MFA'.

The screenshot shows the AWS IAM Security Credentials page for the user 'tehreemtungekar'. The left sidebar has sections for 'Identity and Access Management (IAM)' like Dashboard, Access management, Users, Policies, Identity providers, Account settings, and more. The main content area has tabs for 'Permissions', 'Groups', 'Tags', 'Security credentials' (which is selected), and 'Access Advisor'. Under 'Sign-in credentials', there's a 'Summary' section with a warning: 'User does not have console management access' (N/A). It also lists 'Console password', 'Assigned MFA device', and 'Signing certificates' (None). Under 'Access keys', there's a 'Create access key' button and a table with columns 'Access key ID', 'Created', 'Last used', and 'Status'. The table shows 'No results'. At the bottom, there's a link 'GOALS FOR AWS IAM Development'.

Due to limited access, the warnings are shown. Click on Create Access and Generate Credentials:



Click on Download Credentials.

Step 3: Open command prompt and enter command aws configure. Then fill data as per downloaded credential file

```
Administrator: Command Prompt - aws configure
Microsoft Windows [Version 10.0.18362.720]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>aws configure
AWS Access Key ID [*****60CA]: tehreemtungekar-at-112619478712
AWS Secret Access Key [*****Vv/C]: 6VCQ27/0H9ZBbYwL7i9km/VT1tRqCqMxZuqubg9DKsI=
Default region name [us-east-1]: us-east-1
Default output format [json]: json
```

Step 4: Creating security group:

The screenshot shows the AWS IAM service's "Attach policy" interface. At the top, there is a search bar and a filter dropdown set to "Filter". Below this is a table titled "Showing 7 results" containing a list of IAM entities:

Name	Type
tehreemtungekar	User
EMR_AutoScaling_DefaultRole	Role
EMR_DefaultRole	Role
EMR_EC2_DefaultRole	Role
robomaker_students	Role

At the bottom right of the table are "Cancel" and "Attach policy" buttons.

Click on attach policy and check if the policy is attached successfully.

The screenshot shows the AWS IAM service's "Policies" summary page for the "AdministratorAccess" policy. On the left, there is a navigation sidebar with options like "Dashboard", "Groups", "Users", "Roles", "Policies", "Access reports", "Access analyzer", "Analyzers", "Settings", "Credential report", and "Organization activity".

The main area displays the "Summary" of the "AdministratorAccess" policy. It shows the "Policy ARN" as "arn:aws:iam::aws:policy/AdministratorAccess" and the "Description" as "Provides full access to AWS services and resources". Below this, there are tabs for "Permissions", "Policy usage", "Policy versions", and "Access Advisor".

The "Permissions" tab is selected, showing a table with one result:

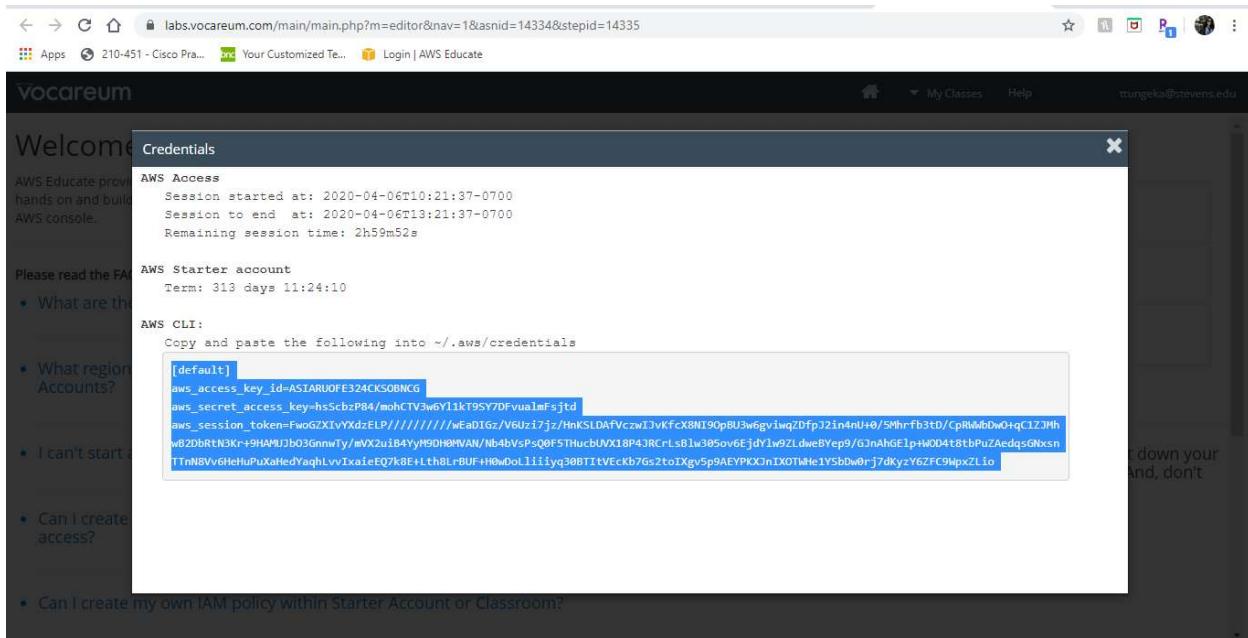
Name	Type
tehreemtungekar	User

At the bottom of the "Permissions" section are "Attach" and "Detach" buttons.

So it seems there were some issues with IAM users, which can be resolved as:

First login to AWS account and click on Account details.

Then click on Account Details and get the details below:



Paste this in C:/Users/Tehreem/.aws/credentials file

```
credentials - Notepad
File Edit Format View Help
[default]
aws_access_key_id=ASIARUOFE324CKSOBNCG
aws_secret_access_key=hsScbzP84/mohCTV3w6Y11kT9SY7DFvualmFsjtd
aws_session_token=FwoGZXIvYXdzELP//////////wEaDIGz/V6Uzi7jz/HnKSLDAfVczwIjvKfcX8NI90pBU3w6gvivqZDfpJ2in4nU+0/5Mhrlfb3tD/CpRlwBdwO+qC1ZJMh
wIB0bRtb3Kc+r9HAMJbO3GnnwTy/mV2uiB4YyM9DH0MAN/Nb4bVsPsQ0F5ThucbUVX18P4JRCrlsB1w305ov6EjdY1w9ZLdweBYep9/GJnAh6Elp+H0D4t8tbPuZAedqsGNxsnTTnN8Vv6HeHuPuXaHedYaqhLvvIxai
EQ7k8E+Lth8LrBUF+H0wDoLiiiyq30BTItVEcKb7Gs2toIXgv5p9AEYPKKJnIXOTWHe1YsbDw0rj7dkyzY6ZFC9WpxZLio
```

Save this file. Now run aws configure through cmd in run as administrator mode:

```
Administrator: Command Prompt
Microsoft Windows [Version 10.0.18362.720]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>aws configure
AWS Access Key ID [*****BNCG]: ASTARUOFE324CKSOBNCG
AWS Secret Access Key [*****sjtd]: hsScbzP84/mohCTV3w6Yl1kT9SY7DFvualmFsjtd
Default region name [us-east-1]: us-east-1
Default output format [json]: json

C:\WINDOWS\system32>aws ec2 create-security-group --group-name my-sg1 --description "My new sec"
{
    "GroupId": "sg-0b9d6a7d80863478e"
}

C:\WINDOWS\system32>script
'script' is not recognized as an internal or external command,
operable program or batch file.

C:\WINDOWS\system32>
```

After creating the security group, we need to

```
C:\WINDOWS\system32>aws ec2 authorize-security-group-ingress --group-id sg-0b9d6a7d80863478e --protocol tcp --port 80 --cidr 76.116.178.250/32
C:\WINDOWS\system32>aws ec2 authorize-security-group-ingress --group-id sg-0b9d6a7d80863478e --protocol tcp --port 22 --cidr 76.116.178.250/32
C:\WINDOWS\system32>aws ec2 describe-security-groups --group-ids sg-0b9d6a7d80863478e
{
    "SecurityGroups": [
        {
            "IpPermissionsEgress": [
                {
                    "IpProtocol": "-1",
                    "PrefixListIds": [],
                    "IpRanges": [
                        {
                            "CidrIp": "0.0.0.0/0"
                        }
                    ],
                    "UserIdGroupPairs": [],
                    "Ipv6Ranges": []
                }
            ],
            "Description": "My new sec",
            "IpPermissions": [
                {
                    "PrefixListIds": [],
                    "FromPort": 80,
                    "IpRanges": [
                        {
                            "CidrIp": "76.116.178.250/32"
                        }
                    ],
                    "ToPort": 80,
                    "IpProtocol": "tcp",
                    "UserIdGroupPairs": [],
                    "Ipv6Ranges": []
                },
                {
                    "PrefixListIds": [],
                    "FromPort": 22,
```

The aws ec2 describe-security-groups command displays the ports just added.

I have added port 3389 just for testing it.

```
Administrator: Command Prompt
{
    "PrefixListIds": [],
    "FromPort": 22,
    "IpRanges": [
        {
            "CidrIp": "76.116.178.250/32"
        }
    ],
    "ToPort": 22,
    "IpProtocol": "tcp",
    "UserIdGroupPairs": [],
    "Ipv6Ranges": []
},
{
    "PrefixListIds": [],
    "FromPort": 3389,
    "IpRanges": [
        {
            "CidrIp": "76.116.178.250/32"
        }
    ],
    "ToPort": 3389,
    "IpProtocol": "tcp",
    "UserIdGroupPairs": [],
    "Ipv6Ranges": []
},
],
"GroupName": "my-sg1",
"VpcId": "vpc-cc0435b6",
"OwnerId": "112619478712",
"GroupId": "sg-0b9d6a7d80863478e"
}
]
}

C:\WINDOWS\system32>
```

Now we need to create a key-pair for this instance.

```
C:\WINDOWS\system32>aws ec2 create-key-pair --key-name Tehreemkey --query 'KeyMaterial' --output text > Tehreemkey.pem
C:\WINDOWS\system32>aws ec2 describe-key-pairs --key-name Tehreemkey
{
    "KeyPairs": [
        {
            "KeyName": "Tehreemkey",
            "KeyFingerprint": "f9:b7:10:f4:a8:44:1b:a4:b6:87:58:9e:42:6d:1c:5c:32:f4:a3:47"
        }
    ]
}
```

Take ami id from the EC2 console: ami-0fc61db8544a617ed

```
C:\WINDOWS\system32>aws ec2 run-instances --image-id ami-0fc61db8544a617ed --count 1 --instance-type t2.micro --key-name Tehreemkey --security-group-ids sg-0b9d6a7d80863478e
{
    "Instances": [
        {
            "Monitoring": {
                "State": "disabled"
            },
            "PublicDnsName": "",
            "StateReason": {
                "Message": "pending",
                "Code": "pending"
            },
            "State": {
                "Code": 0,
                "Name": "pending"
            },
            "EbsOptimized": false,
            "LaunchTime": "2020-04-06T18:06:35.000Z",
            "PrivateIpAddress": "172.31.80.162",
            "ProductCodes": [],
            "VpcId": "vpc-cc0435b6",
            "StateTransitionReason": "",
            "InstanceId": "i-0ea3038aeb63e4a6b",
            "ImageId": "ami-0fc61db8544a617ed",
            "PrivateDnsname": "ip-172-31-80-162.ec2.internal",
            "KeyName": "Tehreemkey",
            "SecurityGroups": [
                {
                    "GroupName": "my-sg1",
                    "GroupId": "sg-0b9d6a7d80863478e"
                }
            ],
            "ClientToken": "",
            "SubnetId": "subnet-ddb457fc",
            "InstanceType": "t2.micro",
            "NetworkInterfaces": [
                {
                    "Status": "in-use",
                    "MacAddress": "12:ed:42:83:a6:dd",
                    "Description": "Amazon Elastic Network Adapter"
                }
            ]
        }
    ]
}
```

Instance has been created in AWS EC2 dashboard:

The screenshot shows the AWS EC2 Instances dashboard. On the left, there's a navigation sidebar with links like EC2 Dashboard, Events, Tags, Reports, Limits, INSTANCES (with sub-links for Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Scheduled Instances, Capacity Reservations), IMAGES (with sub-links for AMIs, Bundle Tasks), and a New EC2 Experience link.

The main area displays a table of instances. The columns are: Instance ID, Instance Type, Availability Zone, Instance State, Status Checks, Alarm Status, Public DNS (IPv4), IPv4 Public IP, and IP. There are six instances listed:

Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)	IPv4 Public IP	IP
i-0e41636c31bac77c9	t2.micro	us-east-1b	stopped	None	None	-	-	-
i-0ddc3278642ef16b	t2.micro	us-east-1b	stopped	None	None	-	-	-
i-00b136b7a117c7574	t2.micro	us-east-1b	stopped	None	None	-	-	-
i-02c8a9d1904dfcfea	t2.micro	us-east-1d	stopped	None	None	-	-	-
i-0d787f6d060da86aa	t2.micro	us-east-1b	stopped	None	None	-	-	-
i-0ea3038aeb63e4a6b	t2.micro	us-east-1d	running	2/2 checks ...	None	ec2-3-82-200-54.com...	3.82.200.54	-

Below the table, a message says "Select an instance above" with three small icons for selection.

console.aws.amazon.com/ec2/v2/home?region=us-east-1#instances:sort=tag:Name

Instance ID (IPv4)	IPv4 Public IP	IPv6 IPs	Key Name	Monitoring	Launch Time	Security Groups	Owner
-	-	-	LoadBalancer	disabled	April 3, 2020 at 1:22:50 PM ...	launch-wizard-5	112619478712
-	-	-	Server1	disabled	April 5, 2020 at 7:56:47 PM ...	launch-wizard-6	112619478712
-	-	-	Server2	disabled	April 4, 2020 at 1:07:53 PM ...	launch-wizard-7	112619478712
-	-	-	Server3	disabled	April 4, 2020 at 1:14:59 PM ...	launch-wizard-8	112619478712
-	-	-	Server4	disabled	April 5, 2020 at 7:26:55 PM ...	launch-wizard-9	112619478712
3-82-200-54.comp...	3.82.200.54	-	Tehreemkey	disabled	April 6, 2020 at 2:06:35 PM ...	my-sg1	112619478712

Tehreemkey is the key, as specified in Cmd and my sg1 security group has been assigned successfully.

console.aws.amazon.com/ec2/v2/home?region=us-east-1#SecurityGroups:search=sg-0b9d6a7d80863478e:sort=group-id

Type	Protocol	Port range	Source	Description - optional
HTTP	TCP	80	76.116.178.250/32	-
SSH	TCP	22	76.116.178.250/32	-
RDP	TCP	3389	76.116.178.250/32	-

The three security parameters specified have also been added. Port 3389 is just for testing purpose.

Shutting down instances:

```
C:\WINDOWS\system32>aws ec2 terminate-instances --instance-ids i-0ea3038aeb63e4a6b
{
    "TerminatingInstances": [
        {
            "InstanceId": "i-0ea3038aeb63e4a6b",
            "CurrentState": {
                "Code": 32,
                "Name": "shutting-down"
            },
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
        }
    ]
}

C:\WINDOWS\system32>
```

The screenshot shows the AWS Management Console interface for the EC2 service. The left sidebar has a tree view with 'New EC2 Experience' selected, followed by 'Events', 'Tags', 'Reports', 'Limits', 'INSTANCES' (selected), 'Images', and 'Bundle Tasks'. Under 'INSTANCES', there are links for 'Instance Types', 'Launch Templates', 'Spot Requests', 'Savings Plans', 'Reserved Instances', 'Dedicated Hosts', 'Scheduled Instances', and 'Capacity Reservations'. The main content area is titled 'Instances' with tabs for 'Launch Instance', 'Connect', and 'Actions'. A search bar at the top says 'Filter by tags and attributes or search by keyword'. Below it is a table with columns: Name, Instance ID, Instance Type, Availability Zone, Instance State, Status Checks, Alarm Status, and Public DNS (IPv4). The table shows six rows: 'LoadBalancer' (terminated), 'Server1' (terminated), 'Server2' (terminated), 'Server3' (terminated), 'Server4' (terminated), and 'i-0ea3038aeb63e4a...' (terminated). At the bottom of the table, there's a message 'Select an instance above'.

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)
LoadBalancer	i-0e41636c31bac77c9	t2.micro	us-east-1b	stopped	None	None	
Server1	i-0ddec3278642ef16b	t2.micro	us-east-1b	stopped	None	None	
Server2	i-00b136b7a117c7574	t2.micro	us-east-1b	stopped	None	None	
Server3	i-02c8a9d1904dfcfea	t2.micro	us-east-1d	stopped	None	None	
Server4	i-0d787f6d060da86aa	t2.micro	us-east-1b	stopped	None	None	
	i-0ea3038aeb63e4a...	t2.micro	us-east-1d	terminated	None	None	

Successfully terminated!

To record all the commands that were executed:

```
C:\WINDOWS\system32>doskey/history>history.txt
C:\WINDOWS\system32>
```

Now go to C:\WINDOWS\system32. There should be a file named history:

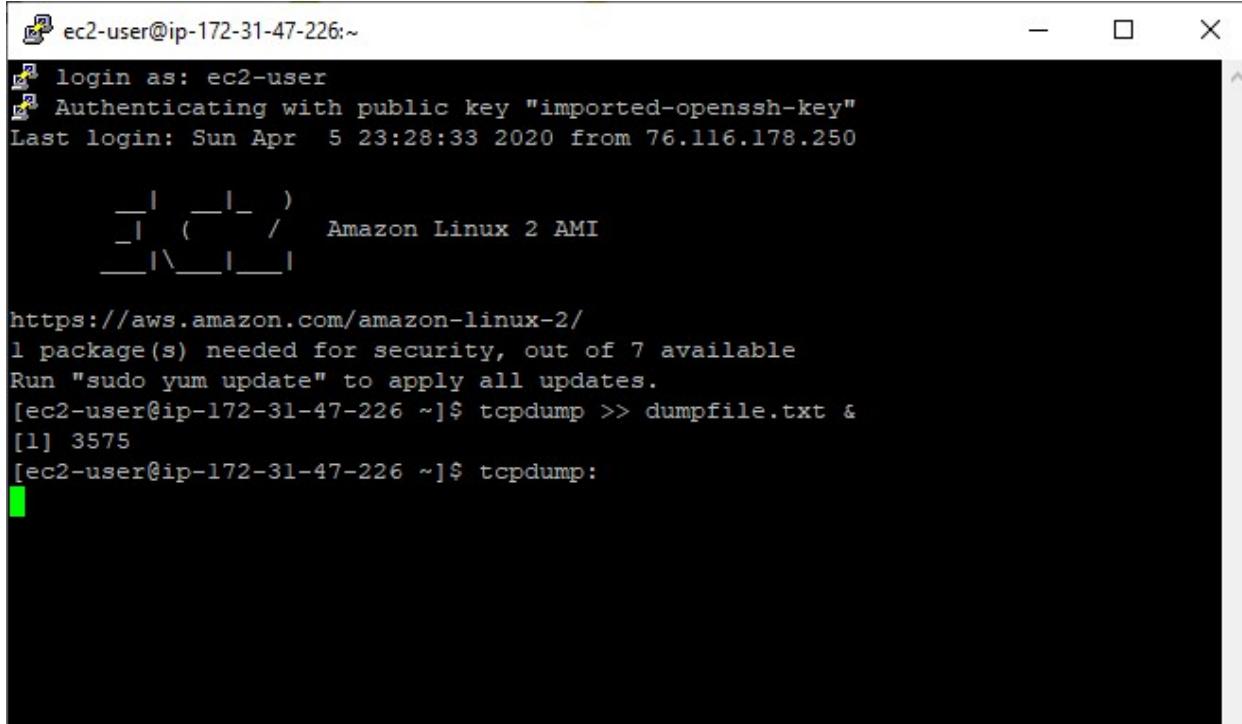
```

history - Notepad
File Edit Format View Help
|aws configure
aws ec2 create-security-group --group-name my-sg1 --description "My new sec"
script
start-transcript
Start-Transcript
    [[-Path] <String>]
    [-Append]
    [-Force]
    [-NoClobber]
    [-IncludeInvocationHeader]
    [-WhatIf]
    [-Confirm]
Start-Transcript -Path "C:\transcripts\transcript0.txt" -NoClobber
doskey/history>history.txt
aws ec2 authorize-security-group-ingress --group-id sg-0b9d6a7d80863478e --protocol tcp --port 3389 --cidr 76.116.178.250/32
aws ec2 authorize-security-group-ingress --group-id sg-0b9d6a7d80863478e --protocol tcp --port 80 --cidr 76.116.178.250/32
aws ec2 authorize-security-group-ingress --group-id sg-0b9d6a7d80863478e --protocol tcp --port 22 --cidr 76.116.178.250/32
aws ec2 describe-security-groups --group-ids sg-0b9d6a7d80863478e
aws ec2 create-key-pair --key-name Tehreemkey --query 'KeyMaterial' --output text > Tehreemkey.pem
aws ec2 describe-key-pairs --key-name Tehreemkey
aws ec2 run-instances --image-id ami-0fc61db8544a617ed --count 1 --instance-type t2.micro --key-name Tehreemkey --security-group-ids sg-0b9d6a7d80863478e
aws ec2 create-tags --resources i-0ea3038aeb63e4a6b --tags Key=Created through CLI,Value=MyInstance
aws ec2 create-tags --resources i-0ea3038aeb63e4a6b --tags Key=CreatedthroughCLI,Value=MyInstance
aws ec2 terminate-instances --instance-ids i-0ea3038aeb63e4a6b
doskey/history>history.txt

```

So all commands were recorded successfully!

Step 17: Running tcpdump command:



```

ec2-user@ip-172-31-47-226:~ 
login as: ec2-user
Authenticating with public key "imported-openssh-key"
Last login: Sun Apr  5 23:28:33 2020 from 76.116.178.250

           _\   _ ) 
          _ \  /  Amazon Linux 2 AMI
         __| \__|__| 

https://aws.amazon.com/amazon-linux-2/
1 package(s) needed for security, out of 7 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-47-226 ~]$ tcpdump >> dumpfile.txt &
[1] 3575
[ec2-user@ip-172-31-47-226 ~]$ tcpdump:

```

If this command is run on the server, it captures all the packets endlessly and the server has to be stopped forcefully.

```
root@ip-172-31-47-226:/home/ec2-user
comcast.net.49949: Flags [P.], seq 13704208:13704432, ack 8481, win 844, length
224
23:52:49.568225 IP ip-172-31-47-226.ec2.internal.ssh > c-76-116-178-250.hsd1.nj.
comcast.net.49949: Flags [P.], seq 13704432:13704656, ack 8481, win 844, length
224
23:52:49.568268 IP ip-172-31-47-226.ec2.internal.ssh > c-76-116-178-250.hsd1.nj.
comcast.net.49949: Flags [P.], seq 13704656:13704880, ack 8481, win 844, length
224
23:52:49.568310 IP ip-172-31-47-226.ec2.internal.ssh > c-76-116-178-250.hsd1.nj.
comcast.net.49949: Flags [P.], seq 13704880:13705104, ack 8481, win 844, length
224
23:52:49.568352 IP ip-172-31-47-226.ec2.internal.ssh > c-76-116-178-250.hsd1.nj.
comcast.net.49949: Flags [P.], seq 13705104:13705328, ack 8481, win 844, length
224
23:52:49.568396 IP ip-172-31-47-226.ec2.internal.ssh > c-76-116-178-250.hsd1.nj.
comcast.net.49949: Flags [P.], seq 13705328:13705552, ack 8481, win 844, length
224
23:52:49.568438 IP ip-172-31-47-226.ec2.internal.ssh > c-76-116-178-250.hsd1.nj.
comcast.net.49949: Flags [P.], seq 13705552:13705776, ack 8481, win 844, length
224
23:52:49.568480 IP ip-172-31-47-226.ec2.internal.ssh > c-76-116-178-250.hsd1.nj.
comcast.net.49949: Flags [P.], seq 13705776:13706000, ack 8481, win 844, length
224
```

So I made the packet capture go to a dumpfile named dumpfile.txt.