

DevSecOps Training

DAY – 3

Today's IP Address details:

S.No	Ip Address- Day3	User Name	Password	Participants Name
Trainer	54.145.123.246	root	redhat	Trainer
1	3.88.0.246	root	redhat	alpana
2	3.83.174.192	root	redhat	baljeet
3	52.91.28.139	root	redhat	Devaraj
4	3.84.20.57	root	redhat	ganesh
5	3.91.155.119	root	redhat	kaustubh
6	3.95.13.210	root	redhat	manish
7	44.204.159.220	root	redhat	manoj
8	54.234.219.215	root	redhat	naveen
9	44.201.228.47	root	redhat	neha
10	54.172.176.72	root	redhat	pravallika
11	18.212.73.246	root	redhat	ramanand sai
12	52.70.92.37	root	redhat	revanth
13	54.157.202.235	root	redhat	rohit
14	3.91.200.184	root	redhat	rudra
15	54.161.183.202	root	redhat	sahitya
16	3.88.0.178	root	redhat	sampat
17	3.87.72.137	root	redhat	sangamesh
18	54.146.194.73	root	redhat	sashi
19	34.227.151.48	root	redhat	shashidhar
20	52.90.62.11	root	redhat	shweta
21	54.175.233.148	root	redhat	sudheer
22	34.226.246.88	root	redhat	uday
23	54.172.190.13	root	redhat	vaishnavi
24	3.88.196.199	root	redhat	vishali
25	44.204.172.209	root	redhat	sankalp
26	18.212.164.28	root	redhat	balakrishna

Trainer IP: 54.145.123.246

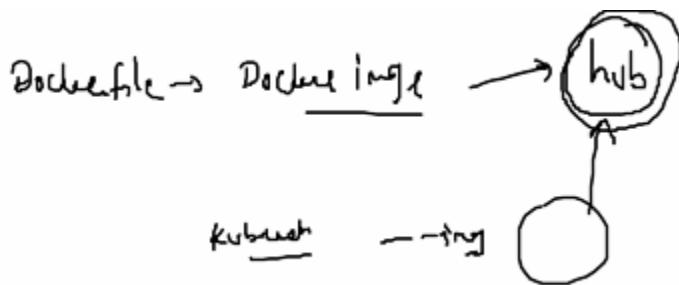
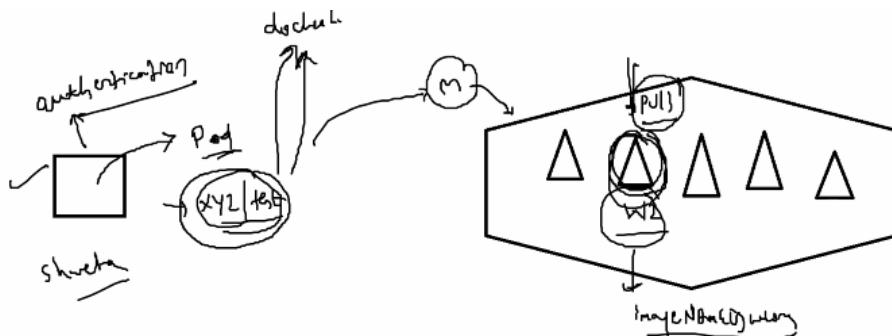
Details of my machine:

IP Address: 3.88.196.199

Username: root

Password: redhat

Flow – How to create a custom image from the docker file and deploy it into Kubernetes cluster:



Trainer is trying to upload authentication file in all the machine to make the kubernetes cluster up and ready:

```
https://aws.amazon.com/amazon-linux-2/
[ec2-user@node3 ~]$ sudo -i
[root@node3 ~]# systemctl enable --now docker kubelet
Created symlink from /etc/systemd/system/multi-user.target.wants/docker.service to /usr,
```

```
[root@node2 ~]# systemctl enable --now docker kubelet
Created symlink from /etc/systemd/system/multi-user.target.wants/docker.service.
```

Nodes are now seen:

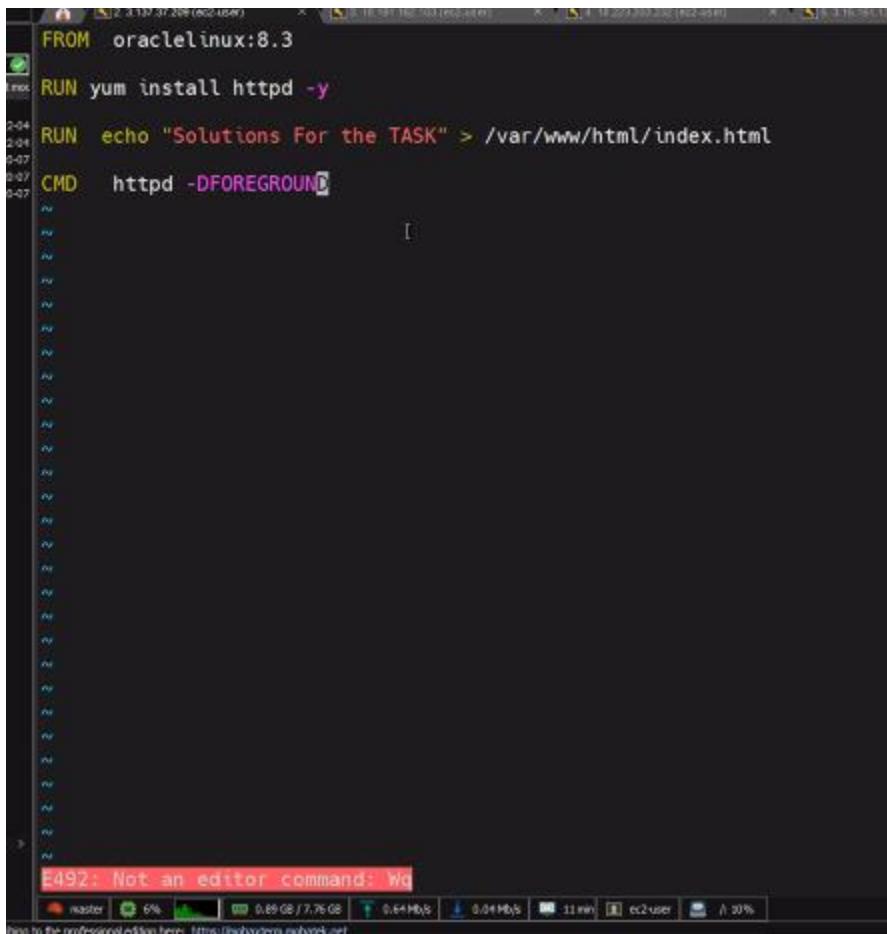
```
[root@master ~]# kubectl get nodes
NAME     STATUS   ROLES      AGE     VERSION
master   Ready    control-plane,master   21h    v1.23.6
node1    Ready    <none>    20h    v1.23.6
node2    Ready    <none>    20h    v1.23.6
node3    Ready    <none>    20h    v1.23.6
node4    Ready    <none>    20h    v1.23.6
node5    Ready    <none>    20h    v1.23.6
[root@master ~]#
```

Yesterday's post task is being explained by the trainer:

Create a folder to have a dockerfile:

```
node5  Ready    <none>    20h    v1.23.6
[root@master ~]# mkdir docker
[root@master ~]# cd docker
[root@master docker]# vim Dockerfile
```

Docker file:



A screenshot of a terminal window showing a Dockerfile. The file content is as follows:

```
FROM oraclelinux:8.3
RUN yum install httpd -y
RUN echo "Solutions For the TASK" > /var/www/html/index.html
CMD httpd -DFOREGROUND
```

The terminal shows some scrollback at the top and ends with an error message: "E492: Not an editor command: wq". The status bar at the bottom indicates the session is on "master" and has 6% usage.

Build the image using the docker file

```
[root@master docker]# docker build -t mayanktest .
Sending build context to Docker daemon 2.048kB
Step 1/4 : FROM oraclelinux:8.3
8.3: Pulling from library/oraclelinux
```

Check for the docker images: mayank test

```
[root@master docker]# docker images
REPOSITORY          TAG      IMAGE ID      CREATED     SIZE
mayanktest          latest   373987aa2ad7  8 seconds ago  393MB
calico/cni          v3.22.2  be7dfc21ba2e  12 days ago  236MB
calico/pod2daemon-flexvol  v3.22.2  d6660bf471e1  12 days ago  19.7MB
calico/node         v3.22.2  fd1608dbbc19  12 days ago  198MB
k8s.gcr.io/kube-apiserver  v1.23.6  8fa62c12256d  12 days ago  135MB
k8s.gcr.io/kube-proxy        v1.23.6  4c0375452406  12 days ago  112MB
k8s.gcr.io/kube-controller-manager  v1.23.6  df77b72818ad2  12 days ago  125MB
k8s.gcr.io/kube-scheduler       v1.23.6  595f327f224a  12 days ago  53.5MB
k8s.gcr.io/etcdd            3.5.1-0  25f8c7f3da61  5 months ago  293MB
k8s.gcr.io/coredns/coredns    v1.8.6   a4ca41631cc7  6 months ago  46.8MB
k8s.gcr.io/pause             3.6     6270bb605e12  8 months ago  683kB
oraclelinux           8.3     816d99f0bbe8  12 months ago  224MB
```

Tag the image:

```
[root@master docker]# docker tag mayanktest docker.io/mdhack/collinsimage
[root@master docker]# docker images
REPOSITORY          TAG      IMAGE ID      CREATED        SIZE
mayanktest          latest   373987aa2ad7  42 seconds ago  393MB
mdhack/collinsimage latest   373987aa2ad7  42 seconds ago  393MB
calico/cni          v3.22.2  b67dfc21ba2e  12 days ago   236MB
calico/pod2daemon-flexvol v3.22.2  d6660bf471e1  12 days ago   19.7MB
calico/node          v3.22.2  fd1608dbbc19  12 days ago   198MB
k8s.gcr.io/kube-apiserver v1.23.6  8fa62c12256d  12 days ago   135MB
k8s.gcr.io/kube-proxy    v1.23.6  4c0375452406  12 days ago   112MB
k8s.gcr.io/kube-scheduler v1.23.6  595f327f224a  12 days ago   53.5MB
k8s.gcr.io/kube-controller-manager v1.23.6  df7b72818ad2  12 days ago   125MB
k8s.gcr.io/etcd       3.5.1-0   25f8c7f3da61  5 months ago  293MB
k8s.gcr.io/coredns/coredns v1.8.6   a4ca41631cc7  6 months ago  46.8MB
k8s.gcr.io/pause       3.6      6270bb605e12  8 months ago  683kB
oraclelinux          8.3     816d99f0bbe8  12 months ago  224MB
[root@master docker]#
```

Docker login:

```
[root@master docker]# docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker ID, head over
to https://hub.docker.com to create one.
Username: mdhack
Password:
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
[root@master docker]#
```

Push the image into the docker hub:

```
[root@master docker]# docker push mdhack/collinsimage
Using default tag: latest
The push refers to repository [docker.io/mdhack/collinsimage]
491700d43e7c: Pushed
2b05feb5d9a: Pushing [=====] 39.35MB/168.8MB
02a3a073ed48: Mounted from library/oraclelinux
[
```

You can see the image in the docker hub now:

This is optional: Recommended not to do this

```
[root@master docker]# kubectl delete all --all --force
warning: Immediate deletion does not wait for confirmation that the running resource has been terminated. The resource may continue to run on the cluster indefinitely.
pod "mayank-9c4fb955-lnsxs" force deleted
[root@master docker]# kubecr[
```

Create the deployment:

```
[root@master docker]# pod mayank-9c4fb955-lnsxs force deleted
[root@master docker]# kubectl create deployment modi --image mdhack/collinsimage
deployment.apps/modi created
[root@master docker]#
```

Get pods:

```
[root@master docker]# kubectl get pods
NAME           READY   STATUS    RESTARTS   AGE
mayank-9c4f4b955-w8rs5  1/1     Running   0          28s
modi-7d498cd976-49862  0/1     ContainerCreating   0          4s
[root@master docker]# kubectl get pods
NAME           READY   STATUS    RESTARTS   AGE
mayank-9c4f4b955-w8rs5  1/1     Running   0          32s
modi-7d498cd976-49862  1/1     Running   0          8s
[root@master docker]# kubectl get del
```

Get all:

```
[root@master docker]# kubectl get all
NAME             READY   STATUS    RESTARTS   AGE
pod/modi-7d498cd976-49862  1/1     Running   0          32s
NAME           READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/modi  1/1        1           1          32s
NAME           DESIRED  CURRENT   READY   AGE
replicaset.apps/modi-7d498cd976  1         1           1          32s
```

To find the IP address of that pod and perform curl:

```
[root@caset.apps/modi-7d498cd976 ~]# kubectl get all -o wide
NAME           READY   STATUS    RESTARTS   AGE   IP           NODE   NOMINATED NODE   READINESS GATES
pod/modi-7d498cd976-49862  1/1     Running   0      36s  192.168.0.36  node2  <none>        <none>
NAME           READY   UP-TO-DATE   AVAILABLE   AGE   CONTAINERS   IMAGES          SELECTOR
deployment.apps/modi  1/1        1           1          36s  collinsimage  mdhack/collinsimage  app=modi
NAME           DESIRED  CURRENT   READY   AGE   CONTAINERS   IMAGES          SELECTOR
replicaset.apps/modi-7d498cd976  1         1           1          36s  collinsimage  mdhack/collinsimage  app=modi,pod-template-
hash=7d498cd976
[root@master docker]# curl 192.168.0.26
<body bgcolor='blue'>
<h1>Welcome</h1>
<h2>Hello From MDHack/Mayank Server</h2>
<h3>Apache Server</h3>

[root@master docker]# curl 192.168.0.36
Solutions For the TASK
[root@master docker]#
```

Check in node 2:

```
[root@node2 ~]# systemctl enable --now docker kubelet
Created symlink from /etc/systemd/system/multi-user.target.wants/docker.service to /usr/lib/systemd/system/kubelet.service.
[root@node2 ~]# docker images
REPOSITORY          TAG      IMAGE ID      CREATED       SIZE
mdhack/collinsimage latest   373987aa2ad7  4 minutes ago  393MB
vishali007/postimage 1.0     60e49d69ed90  17 hours ago  393MB
shwetabansal05/firstimage latest   729721f6d3d8  22 hours ago  393MB
alpana87/firstimage latest   1da998ca7116  23 hours ago  393MB
pravallikagunda/pravalimage latest   f8a6c8f108b6  23 hours ago  393MB
nginx               latest   fa5269854a5e  6 days ago   142MB
calico/kube-controllers v3.22.2  a1a88662416b  12 days ago  132MB
calico/cni           v3.22.2  be7dfc21ba2e  12 days ago  236MB
calico/pod2daemon-flexvol v3.22.2  d6660bf471e1  12 days ago  19.7MB
calico/node          v3.22.2  fd1608dbbc19  12 days ago  198MB
k8s.gcr.io/kube-proxy v1.23.6  4c0375452406  12 days ago  112MB
k8s.gcr.io/coredns/coredns v1.8.6   a4ca41631cc7  6 months ago  46.8MB
k8s.gcr.io/pause      3.6     6270bb605e12  8 months ago  683KB
[root@node2 ~]#
```

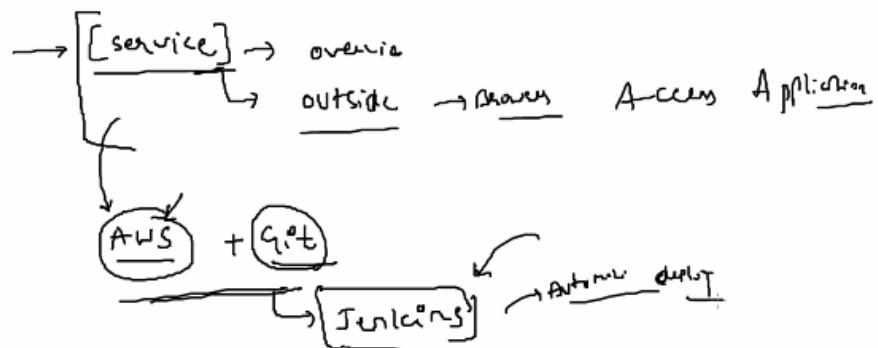
Agenda:

- Service

-> Overview

-> Outside the browser how to access the application.

- AWS
- Git
- Jenkins – Whatever you do now, it can be automated!

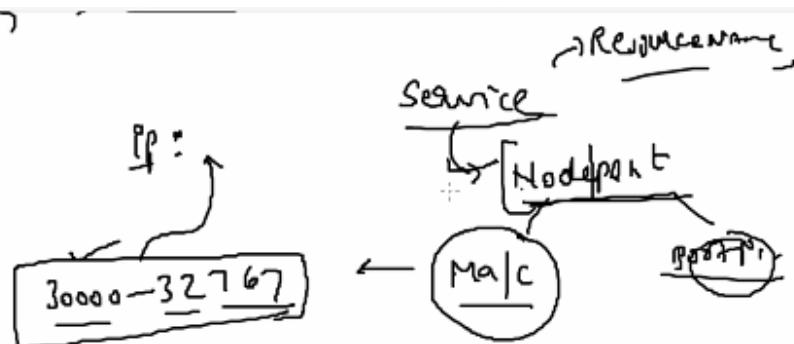


Embedded System Security:

Access:

Expose deployment: In docker we use -p (port expose)

Service -> Node port -> Machine port number -> Range is fixed – 3000 – 32767 (ip address)



If container is working on 80 port number, you have to tell that container is working on 80 port. If 8080 – port, you have to tell 8080 port number.

In production environment, we can't use this command:

```
[root@master docker]# kubectl delete all --all --force
[2] 13:37:37.209 (ec2-user) * [3] 13:38:19.192.103 (ec2-user) * [4] 14:18:23.209.232 (ec2-user) * [5] 15:16:16.134 (ec2-user)
warning: Immediate deletion does not wait for confirmation that the runn
ce may continue to run on the cluster indefinitely.
pod "modi-7d498cd976-49862" force deleted
deployment.apps "modi" force deleted
replicaset.apps "modi-7d498cd976" force deleted
[root@master docker]#
```

We will create a deployment with simple image:

```
[root@master docker]# kubectl create deployment mayank --image=mdhack/mayanknginximage
deployment.apps/mayank created
[root@master docker]# kubectl get pods
```

Get the pod details:

```
[root@master docker]# kubectl get pods
NAME           READY   STATUS    RESTARTS   AGE
mayank-6876cf4c88-q2j2l  1/1     Running   0          7s
[root@master docker]# kubectl get all
NAME           READY   STATUS    RESTARTS   AGE
pod/mayank-6876cf4c88-q2j2l  1/1     Running   0          9s
NAME           READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/mayank      1/1     1           1           9s
NAME           DESIRED  CURRENT   READY   AGE
replicaset.apps/mayank-6876cf4c88  1       1         1        9s
[root@master docker]#
```

Service name of the cloud will be different, but service will be same. It is pretty much same to adapt other cloud vendor.

Take IP address of cluster: using kubectl get pods -o wide

Expose the deployment on port 80:

```
[root@master docker]# kubectl expose deployment mayank --port 80
service/mayank exposed
[root@master docker]# k
```

Get service:

```
service/mayank exposed
[root@master docker]# kubectl get service
NAME    TYPE    CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
mayank  ClusterIP  10.98.5.49  <none>        80/TCP   4s
[root@master docker]#
```

We want Type: node port. So delete the service

```
mayank  ClusterIP  10.98.5.49  <none>        80/TCP   4s
[root@master docker]# kubectl delete service mayank
service "mayank" deleted
[root@master docker]# kubectl expose deployment mayank --port 80
```

Expose they deployment in port 80:

```
service/mayank deleted
[root@master docker]# kubectl expose deployment mayank --port 80 --type NodePort
service/mayank exposed
[root@master docker]# kubectl get service
NAME    TYPE    CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
mayank  NodePort  10.104.59.216  <none>        80:30568/TCP  2s
[root@master docker]#
```

Check the website:

The screenshot shows a LinkedIn user profile for 'Mayank Modi'. The profile picture is a selfie of a man with glasses and a dark shirt. Below the picture, the name 'Mayank Modi' is displayed in bold. Underneath the name, it says 'Devops Trainer' and 'Rajasthan'. There is a 'Contact' button at the bottom of the card.

Taking an ip address and exposing your application in a port number.

```
[root@master docker]# kubectl create deployment collinsapp --image=quay.io/mayan123modi/simple-webapp
deployment.apps/collinsapp created
[root@master docker]#
```

This work on 80 port number by default:

The terminal window shows the command 'kubectl get deployments.apps' being run. The output lists two deployment entries: 'collinsapp' and 'mayank'. Both are in the 'READY' state with 1 pod each, and they are both 'UP-TO-DATE' and 'AVAILABLE'. The 'collinsapp' deployment is 27 seconds old, and the 'mayank' deployment is 4m28s old.

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
collinsapp	0/1	1	0	27s
mayank	1/1	1	1	4m28s

Edit the deployment:

```

progressDeadlineSeconds: 600
replicas: 1
revisionHistoryLimit: 10
selector:
  matchLabels:
    app: collinsapp
strategy:
  rollingUpdate:
    maxSurge: 25%
    maxUnavailable: 25%
  type: RollingUpdate
template:
  metadata:
    creationTimestamp: null
    labels:
      app: collinsapp
  spec:
    containers:
      - image: quay.io/mayan123modi/simple-webapp
        imagePullPolicy: Always
        name: simple-webapp
        resources: {}
        terminationMessagePath: /dev/termination-log
        terminationMessagePolicy: File
    dnsPolicy: ClusterFirst
    restartPolicy: Always
    schedulerName: default-scheduler
    securityContext: {}
    terminationGracePeriodSeconds: 30
status:
  conditions:
    - lastTransitionTime: "2022-04-27T04:30:59Z"
      lastUpdateTime: "2022-04-27T04:30:59Z"
      message: Deployment does not have minimum availability.
      reason: MinimumReplicasUnavailable

```

Get pod command:

```

[root@master docker]# kubectl get pods
NAME           READY   STATUS      RESTARTS   AGE
collinsapp-7847bb86f6-hc2tn  0/1     ImagePullBackOff  0          85s
collinsapp-7c455fbb89-tngxn  0/1     ContainerCreating  0          3s
mayank-6876cf4c88-q2j2l     1/1     Running     0          5m26s
[root@master docker]# kubectl get pods
NAME           READY   STATUS      RESTARTS   AGE
collinsapp-7c455fbb89-tngxn  1/1     Running     0          10s
mayank-6876cf4c88-q2j2l     1/1     Running     0          5m33s
[root@master docker]#

```

Check the IP address:

```

[root@master docker]# kubectl get pods -o wide
NAME           READY   STATUS      RESTARTS   AGE   IP          NODE   NOMINATED-NODE   SCHEDULER   ATTEMPTED-REASON
collinsapp-7c455fbb89-tngxn  1/1     Running     0          10s   192.168.0.14   node5   <none>
mayank-6876cf4c88-q2j2l     1/1     Running     0          5m33s  192.168.0.252  node3   <none>
[root@master docker]# curl 192.168.0.14

```

Curl:

```
mayank-08/6c14c88-q2) ~ 1/1 Running 0 5m36s 192.168.0.14:8080
[root@master docker]# curl 192.168.0.14:8080
<!doctype html>
<title>Hello from Flask</title>
<body style="background: #e74c3c;"></body>
<div style="color: #e4e4e4;
    text-align: center;
    height: 90px;
    vertical-align: middle;">
<h1>Hello from collinsapp-7c455fbb89-tngxn!</h1>
</div>[root@master docker]#
```

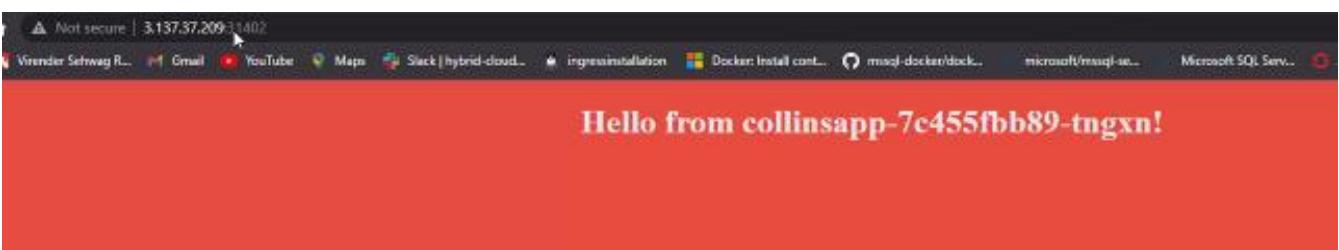
To expose a node port type:

```
</div>[root@master docker]# kubectl expose deployment collinsapp --port 8080 --type NodePort
service/collinsapp exposed
[root@master docker]# ku
```

Get service:

```
[root@master docker]# kubectl get service
NAME      TYPE      CLUSTER-IP   EXTERNAL-IP   PORT(S)      AGE
collinsapp  NodePort  10.96.169.244 <none>        8080:31402/TCP  3s
mayank     NodePort  10.104.59.216  <none>        80:30563/TCP   3m40s
[root@master docker]#
```

Check the website:



Get the details of deployments:

```
mayank     NodePort  10.104.59.216  <none>        80:30563/TCP   3m40s
[root@master docker]# kubectl get deployments.apps collinsapp
NAME      READY   UP-TO-DATE  AVAILABLE  AGE
collinsapp 1/1     1          1          2m44s
[root@master docker]#
```

To scale the application: increase in load

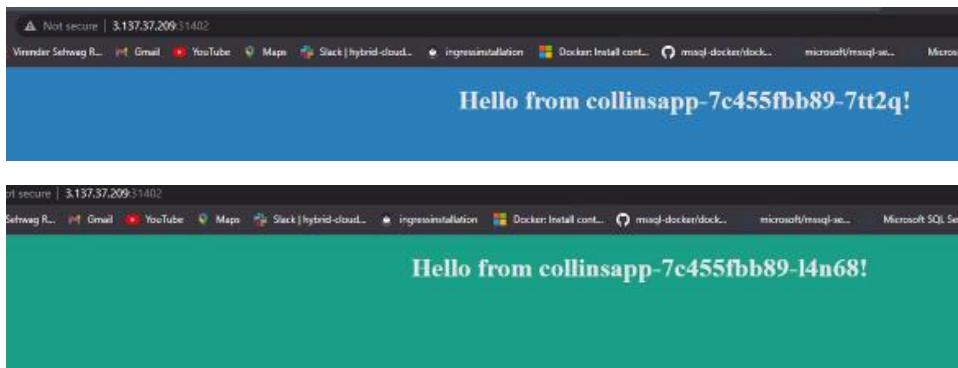
```
[root@master docker]# kubectl get pods
NAME           READY   STATUS    RESTARTS   AGE
collinsapp-7c455fbb89-tngxn  1/1     Running   0          87s
mayank-6876cf4c88-q2j2l   1/1     Running   0          6m50s
[root@master docker]# kubectl
alpha      autoscale  create      exec      logs      rollout    version
annotate   certificate debug      explain    options    run       wait
api-resources cluster-info delete    expose     patch     scale
api-versions completion  describe  get       plugin    set
apply      config      diff       help      port-forward taint
attach     cordon      drain     kustomize proxy     top
auth      cp          edit      label     replace    uncordon
[root@master docker]# kubectl
```

Get pods:

```
[root@master docker]# kubectl get pods
NAME           READY   STATUS    RESTARTS   AGE
collinsapp-7c455fbb89-2pdkb  1/1     Running   0          5s
collinsapp-7c455fbb89-4w6qx  0/1     ContainerCreating   0          5s
collinsapp-7c455fbb89-7tt2q  0/1     ContainerCreating   0          5s
collinsapp-7c455fbb89-jdf2s  0/1     ContainerCreating   0          5s
collinsapp-7c455fbb89-l4n68  0/1     ContainerCreating   0          5s
collinsapp-7c455fbb89-pqz94  0/1     ContainerCreating   0          5s
collinsapp-7c455fbb89-pzl4g  1/1     Running   0          5s
collinsapp-7c455fbb89-r6kst  0/1     ContainerCreating   0          5s
collinsapp-7c455fbb89-tngxn  1/1     Running   0          2m3s
collinsapp-7c455fbb89-z5kvq  1/1     Running   0          5s
mayank-6876cf4c88-q2j2l    1/1     Running   0          7m26s
[root@master docker]#
```

Now, total 9 pods are running

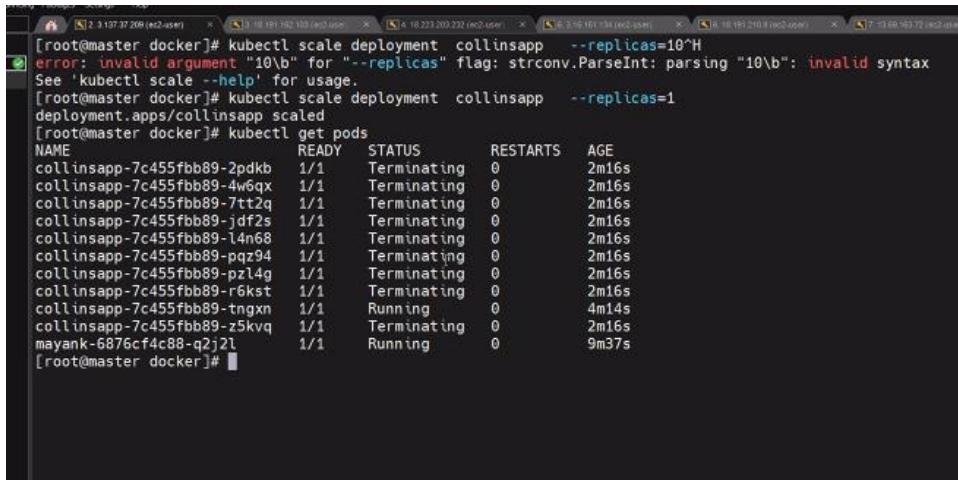
In these 10 pods, where will my request go? It can go anywhere; it works by Round Robin Fashion.



Whenever I refresh, it takes different number? Since its Round Robin

```
> Round Robbin -- one by one - my req is only on every pod --- INternal Load BALANCER
```

This is how you can scale up the application:



```
[root@master docker]# kubectl scale deployment collinsapp --replicas=10^H
error: invalid argument "10\b" for "--replicas" flag: strconv.ParseInt: parsing "10\b": invalid syntax
See 'kubectl scale --help' for usage.
[root@master docker]# kubectl scale deployment collinsapp --replicas=1
deployment.apps/collinsapp scaled
[root@master docker]# kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
collinsapp-7c455fbb89-2pdkb  1/1     Terminating   0          2m16s
collinsapp-7c455fbb89-4w6qx  1/1     Terminating   0          2m16s
collinsapp-7c455fbb89-7tt2q  1/1     Terminating   0          2m16s
collinsapp-7c455fbb89-jdf2s  1/1     Terminating   0          2m16s
collinsapp-7c455fbb89-l4n68  1/1     Terminating   0          2m16s
collinsapp-7c455fbb89-pqz94  1/1     Terminating   0          2m16s
collinsapp-7c455fbb89-pz14g  1/1     Terminating   0          2m16s
collinsapp-7c455fbb89-r6kst  1/1     Terminating   0          2m16s
collinsapp-7c455fbb89-tngnx  1/1     Running     0          4m14s
collinsapp-7c455fbb89-z5kvq  1/1     Terminating   0          2m16s
mayank-6876fc4c88-q2j2l    1/1     Running     0          9m37s
[root@master docker]#
```

Commands shared by the Trainer:

173 kubectl get all

174 kubectl get pods -o wide

175 curl 192.168.0.33

176 kubectl describe deployments.apps mayank

177 history

178 cat deployment.yaml

179 curl 192.168.0.52

180 kubectl get pods -o wide -n sangamesh

181 kubectl describe pod -n sangamesh sangameshhub-df8c768dd-mklq4

182 curl 192.168.0.56

183 curl 192.168.0.58

184 kubectl get nodes

185 ip a

186 kubectl get nodes

187 ip a

188 vim .kube/config

189 kubectl get nodes

190 systemctl status kubelt

191 systemctl status kubelet

192 systemctl start kubelet

```
193 systemctl status kubelet
194 kubectl get nodes
195 systemctl status kubelet
196 kubectl get nodes
197 systemctl status docker
198 systemctl enable --now docker
199 systemctl enable --now kubelet
200 systemctl status docker
201 kubectl get nodes
202 mkdir docker
203 cd docker
204 vim Dockerfile
205 docker build -t mayanktest .
206 docker images
207 docker tag mayanktest docker.io/mdhack/collinsimage
208 docker images
209 docker login
210 docker push mdhack/collinsimage
211 kubectl get all
212 kubectl delete all --all --force
213 kubectl create deployment modi --image mdhack/collinsimage
214 kubectl get pods
215 kubectl get dep
216 kubectl get deployments.apps
217 kubectl delete deployments.apps mayank
218 kubectl get all
219 kubectl get all -o wide
220 curl 192.168.0.26
221 curl 192.168.0.36
222 kubectl get all
```

223 kubectl get all -o wide

224 hsi

225 history

213 kubectl create deployment modi --image mdhack/collinsimage

214 kubectl get pods

215 kubectl get depl

216 kubectl get deployments.apps

217 kubectl delete deployments.apps mayank

218 kubectl get all

219 kubectl get all -o wide

220 curl 192.168.0.26

221 curl 192.168.0.36

222 kubectl get all

223 kubectl get all -o wide

224 hsi

225 history

226 kubectl delete all --all --force

227 kubectl create deployment mayank --image=mdhack/mayanknginximage

228 kubectl get pods

229 kubectl get all

230 kubectl get all -o wide

231 curl 192.168.0.252

232 kubectl expose deployment mayank --port 80

233 kubectl get service

234 kubectl delete service mayank

235 kubectl expose deployment mayank --port 80 --type NodePort

236 kubectl get service

237 kubectl create deployment collinsapp --image=quay.io/mayan123modi/simple-webapp

238 kubectl get all

```
239 kubectl get deployments.apps
240 kubectl get pods
241 kubectl describe pod collinsapp-7847bb86f6-hc2tn
242 kubectl edit deployments.apps collinsapp
243 kubectl get pods
244 kubectl get pods -o wide
245 curl 192.168.0.14:8080
246 kubectl expose deployment collinsapp --port 8080 --type NodePort
247 kubectl get service
248 kubectl get deployments.apps collinsapp
249 kubectl get pods
250 kubectl scale deployment collinsapp --replicas=10
251 kubectl get pods
252 <<X
253 kubectl scale deployment collinsapp --replicas=10
254 kubectl scale deployment collinsapp --replicas=1
255 kubectl get pods
256 history
```

TASK 1:

Q.1 CREATE A deployment using your own image and scale that upto 5 replicas and expose it on node port and access it from browser and the ip that u need to use is 3.137.37.209: port no.

Create deployment:

```
[root@vishali task3]# kubectl create deployment vishalideploy --image=vishali007
/vishaliimage:1.0
deployment.apps/vishalideploy created
```

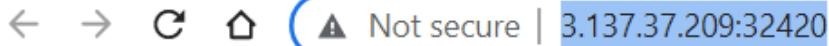
Scale:

```
[root@vishali task3]# kubectl scale deployment vishalideploy --replicas=5
deployment.apps/vishalideploy scaled
```

Expose it

```
[root@vishali task3]# kubectl expose deployment vishalideploy --port 80 --type NodePort
service/vishalideploy exposed
[root@vishali task3]# kubectl get service
NAME           TYPE      CLUSTER-IP   EXTERNAL-IP   PORT(S)        AGE
vishalideploy  NodePort  10.105.86.135 <none>       80:32420/TCP  3s
```

Check the website: <http://3.137.37.209:32420/>



Hello From Dockerfile

Q.2 create a deployment using quay.io/mayank123modi/simple-webapp image and expose it using nodeport and share the url with me

Create the deployment and expose it:

```
[root@vishali task3]# kubectl create deployment videploy --image=quay.io/mayank1/simple-webapp
deployment.apps/videploy created
[root@vishali task3]# kubectl expose deployment videploy --port 80 --type NodePort
service/videploy exposed
[root@vishali task3]# kubectl get service
NAME           TYPE      CLUSTER-IP   EXTERNAL-IP   PORT(S)        AGE
videploy       NodePort  10.99.61.33 <none>       80:31434/TCP  4s
vishalideploy  NodePort  10.105.86.135 <none>       80:32420/TCP  5m18s
vishdeploy     NodePort  10.111.217.91 <none>       80:30427/TCP  2m27s
[root@vishali task3]# kubectl get pods -o wide
NAME                  READY   STATUS    RESTARTS   AGE
pod1                 1/1     Running   1 (16h ago) 19h
 192.168.0.180   node1   <none>    <none>
pod2                 1/1     Running   1 (16h ago) 19h
 192.168.0.245   node3   <none>    <none>
postimagetask-6f4db4bcd5-vqj7d  0/1     ImagePullBackOff 0          17h
 192.168.0.244   node3   <none>    <none>
postimagetask3-db9b9dd8d-p8kf7  1/1     Running   1 (16h ago) 17h
 192.168.0.6     node2   <none>    <none>
posttask-59b9548d4c-qhxs7     0/1     ImagePullBackOff 0          17h
 192.168.0.53    node2   <none>    <none>
posttask3-7ff7f9b799-g22p2    0/1     ImagePullBackOff 0          17h
 192.168.0.15    node2   <none>    <none>
test1-6db788bdd4-mccwb       1/1     Running   1 (16h ago) 17h
 192.168.0.165   node1   <none>    <none>
test2-744cdc6c89-dvpgb       1/1     Running   1 (16h ago) 17h
 192.168.0.104   node4   <none>    <none>
videploy-756559bdf6-nzc8     1/1     Running   0          82s
 192.168.0.207   node5   <none>    <none>
```

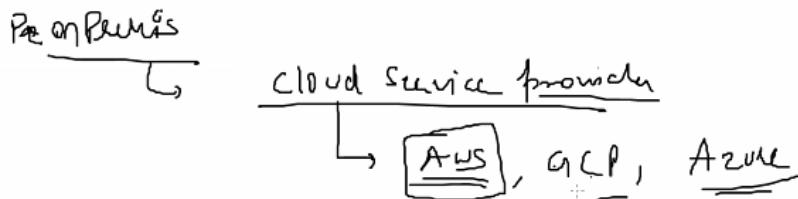
The problem why it is not working was – might be due to overload. So, I have exposed it into 8080 port.

```
[root@vishali task3]# kubectl delete service videploy
service "videploy" deleted
[root@vishali task3]# kubectl get service
NAME      TYPE      CLUSTER-IP      EXTERNAL-IP      PORT(S)      AGE
vishalideploy  NodePort  10.105.86.135  <none>        80:32420/TCP  14m
vishdeploy   NodePort  10.111.217.91   <none>        80:30427/TCP  11m
[root@vishali task3]# kubectl expose deployment videploy --port 8080 --type NodePort
service/videploy exposed
[root@vishali task3]# kubectl get service
NAME      TYPE      CLUSTER-IP      EXTERNAL-IP      PORT(S)      AGE
videploy   NodePort  10.100.218.22  <none>        8080:30342/TCP  4s
vishalideploy  NodePort  10.105.86.135  <none>        80:32420/TCP  14m
vishdeploy   NodePort  10.111.217.91   <none>        80:30427/TCP  11m
[root@vishali task3]#
```

Check the site: <http://3.137.37.209:30342>



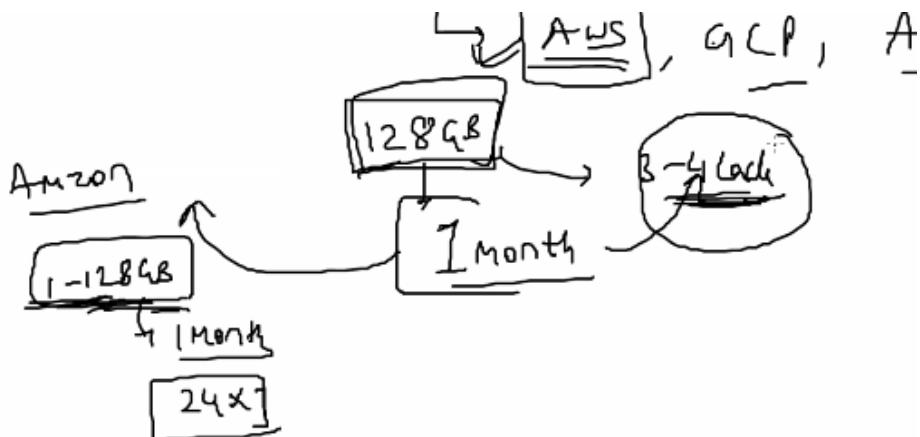
Cloud Service Provider:



AWS is recommended – as it contains everything

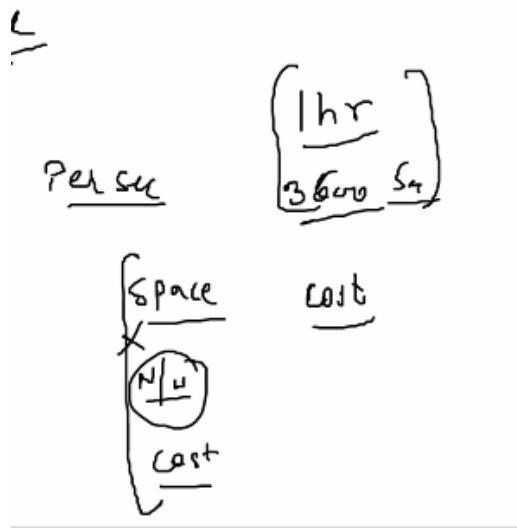
What is the cost of 128GB RAM Laptop → Approx. 3–4 Lakh cost

But if you need only for one month, then its waste of money.



Cloud is very flexible. Whenever we need it, we can create it and we can terminate it. Bill will come – whatever the time you used it.

We will save the space money; cost is saved in network and server.



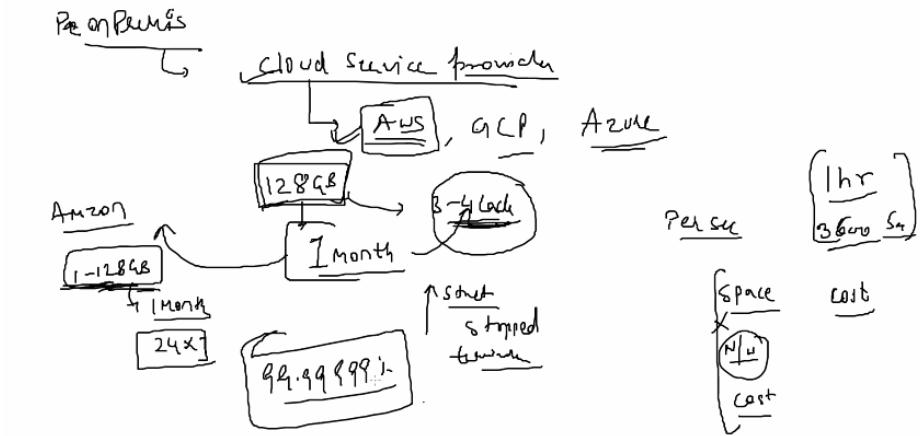
AWS Data Center:

AWS Data Center Locations

The AWS Cloud Data Center locations are spread across the globe with 22 Regions and 69 AZ (Availability Zones). AWS has broadcast plans for 9 new Availability Zones and 3 additional Regions in Cape Town, Jakarta, and Milan.



In every region, we can see the datacenter. Try to give the guarantee of data – 99.99%



Services:

- ELB: Elastic Load Balancing
- EC2: Amazon Elastic Compute Cloud

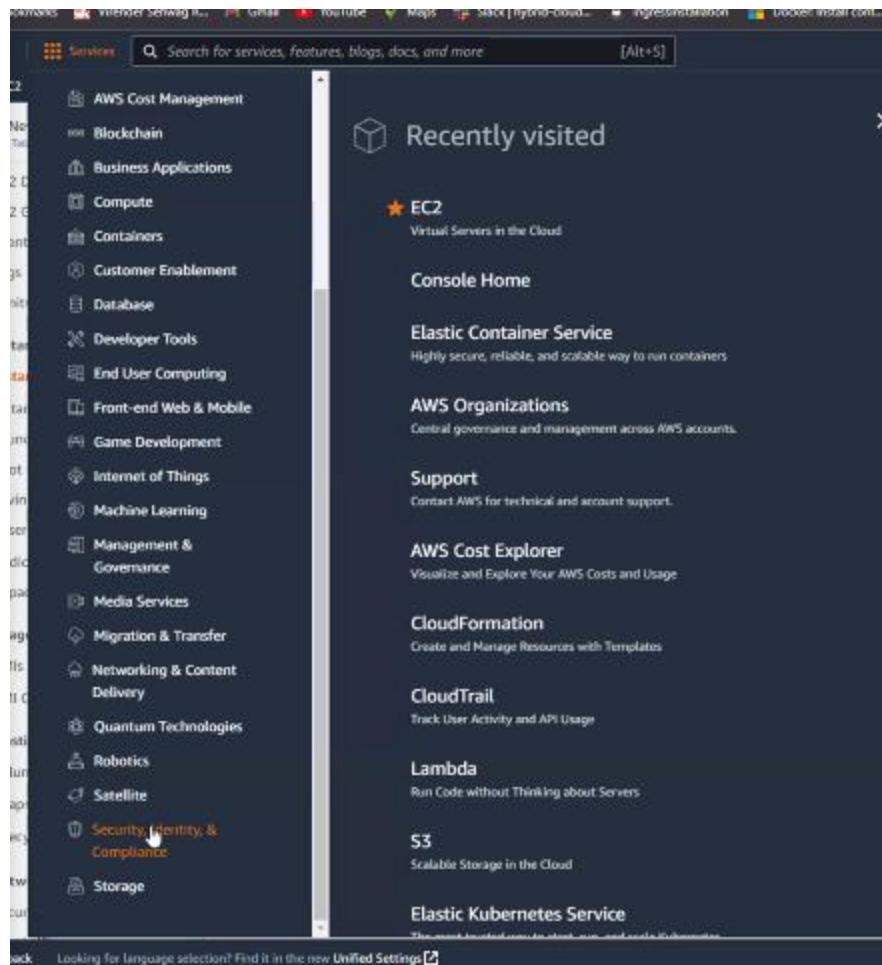


Storage:

- S3: Simple storage
- EBS: Elastic Block Storage



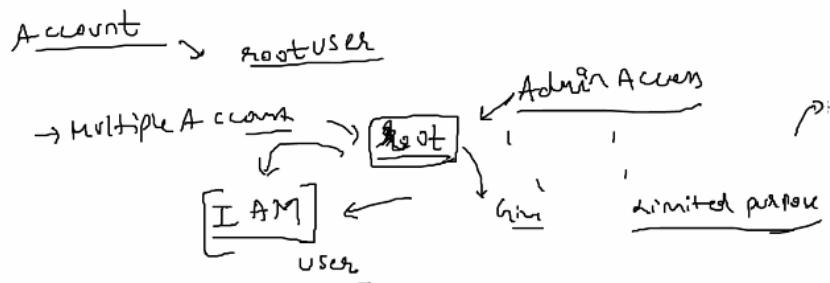
So many services:



The problem is whenever you create the account, the environment can have multiple accounts.

Whether they will give you the root access? (admin)? NO!

They will give the access of limited participants. (group the participant)

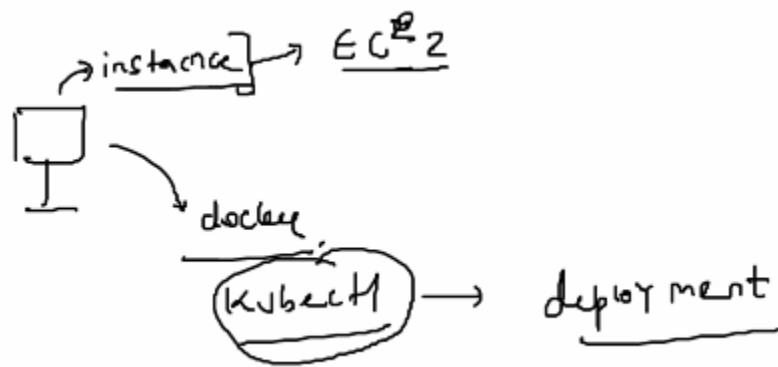


Different group get different access – that is called IAM entity – Identity Admin Management

What will you do inside?

We need a machine(instance) -> In AWS we will get by EC2.

For docker, we will get it by kubectl -> Deployment.



Terraform tool:

IAC – Infrastructure anywhere in Code tool – Automatically create infrastructure anywhere!

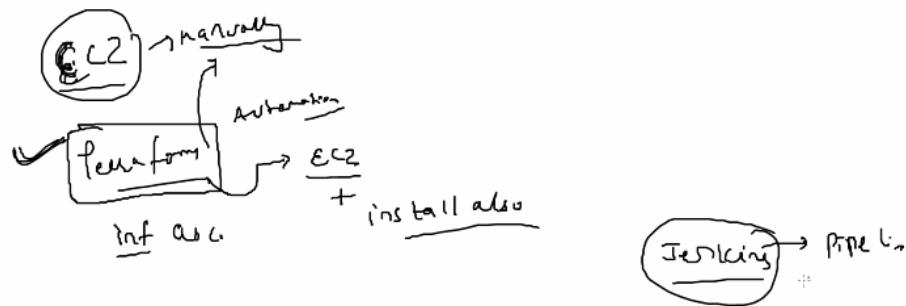
If you are good in this tool, can you use this terraform tool to create automation?

Logically no!

(Without having a knowledge of EC2 & AWS)

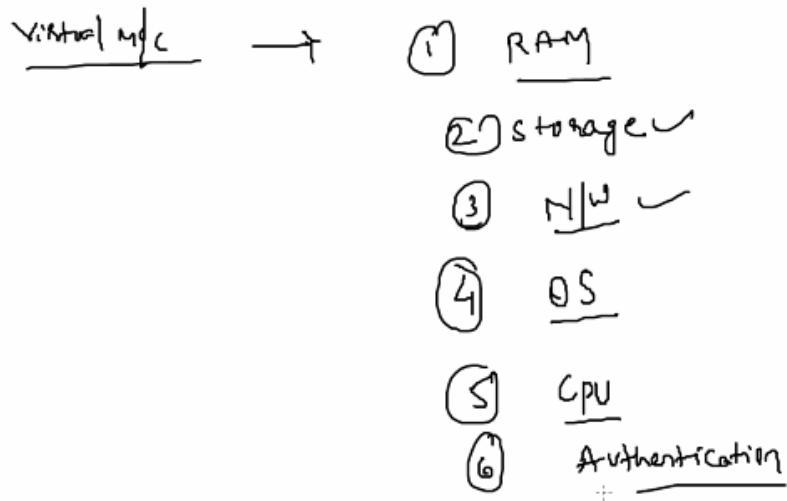
How to use EC2 manually?

Using terraform, we can automate it. - using this we can create instance and install anything automatically.

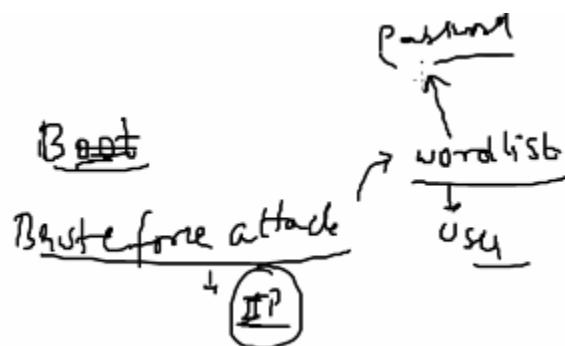


Simple virtual machine – if you want to create – what are the basic things you need?

1. RAM
2. Storage
3. Networking
4. OS
5. How much CPU
6. Authentication



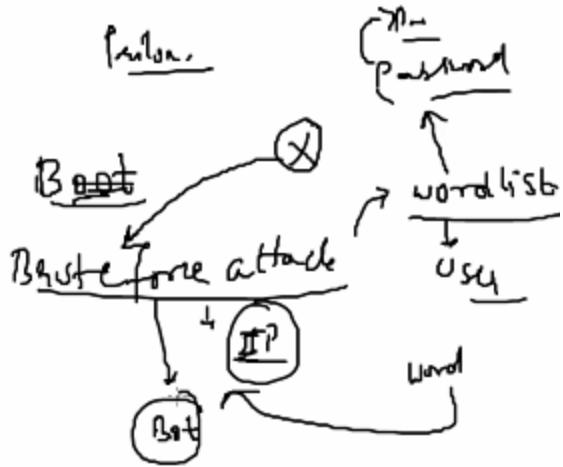
Tool to hack a thing using multiple passwords very second using a bot – this thing is called what? **Brute Force Attack**



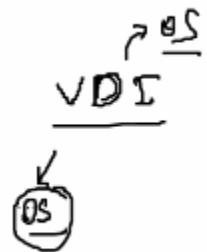
Trillions of users and passwords.

Will the hacker take one by one and use it? No!

Using bot, they can automatically use user and password together.



Do the company provide VDI OS? Yes, our company uses the same



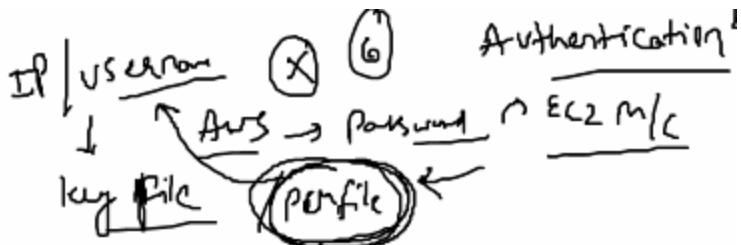
A different OS, that is created in cloud on premises – you can connect this OS/VDI using VPN. When you try to access this machine – you use user and password.



If I get the VPN by other means, then how it is secure?

Security in these times we can enhance it. But we can break it.

AWS – won't provide password to access EC2. Here, they provide key –pair file. Brute Force



But the key file can be shared? - Attack can't be done here. Brute Force can't happen here.

Fishing might happen – we need to be careful while someone is accessing emails and Pen drive.

EC2 Machine:

We will do it manually now.

The lab which we are using right now is enabled by password.

Do you know how many attempts one has done?

W command:

w command in Linux is **used to show who is logged on and what they are doing**. This command shows the information about the users currently on the machine and their processes

```
[root@trainer ~]# w
 05:36:32 up 1:58, 1 user,  load average: 0.05, 0.01, 0.00
USER   TTY   FROM           LOGIN@   IDLE   JCPU   PCPU WHAT
root   pts/0  59.91.129.50    05:35   0.00s  0.03s  0.00s w
[root@trainer ~]#
```

Last command:

```
[root@trainer ~]# last
root  pts/0      59.91.129.50      Wed Apr 27 05:35      still logged in
root  pts/0      223.178.58.150     Wed Apr 27 04:43 - 04:56 (00:13)
reboot  system boot  5.10.109-104.500  Wed Apr 27 03:37 - 05:36 (01:59)
root  pts/0      117.199.170.131     Tue Apr 26 06:13 - down (06:32)
root  pts/0      117.199.170.131     Tue Apr 26 04:34 - 06:12 (01:38)
reboot  system boot  5.10.109-104.500  Tue Apr 26 03:34 - 12:45 (09:10)
root  pts/1      117.199.171.96      Mon Apr 25 06:55 - 12:55 (05:59)
root  pts/0      1.39.212.61       Mon Apr 25 04:48 - 06:56 (02:08)
reboot  system boot  5.10.109-104.500  Mon Apr 25 04:36 - 12:45 (1+08:09)

wtmp begins Mon Apr 25 04:36:17 2022
[root@trainer ~]#
```

Lastb command:

```
[root@trainer ~]# lastb
[root@trainer ~]# lastb
pi    ssh:notty  82-65-211-222.su Wed Apr 27 04:44 - 04:44 (00:00)
pi    ssh:notty  82-65-211-222.su Wed Apr 27 04:44 - 04:44 (00:00)
pi    ssh:notty  82-65-211-222.su Wed Apr 27 04:44 - 04:44 (00:00)
pi    ssh:notty  82-65-211-222.su Wed Apr 27 04:44 - 04:44 (00:00)
user  ssh:notty  68.183.77.86   Tue Apr 26 07:14 - 07:14 (00:00)
user  ssh:notty  68.183.77.86   Tue Apr 26 07:14 - 07:14 (00:00)
user  ssh:notty  68.183.77.86   Tue Apr 26 07:13 - 07:13 (00:00)
user  ssh:notty  68.183.77.86   Tue Apr 26 07:13 - 07:13 (00:00)
user  ssh:notty  68.183.77.86   Tue Apr 26 07:13 - 07:13 (00:00)
user  ssh:notty  68.183.77.86   Tue Apr 26 07:13 - 07:13 (00:00)
user  ssh:notty  68.183.77.86   Tue Apr 26 07:13 - 07:13 (00:00)
user  ssh:notty  68.183.77.86   Tue Apr 26 07:13 - 07:13 (00:00)
user  ssh:notty  68.183.77.86   Tue Apr 26 07:13 - 07:13 (00:00)
user  ssh:notty  68.183.77.86   Tue Apr 26 07:13 - 07:13 (00:00)
user  ssh:notty  68.183.77.86   Tue Apr 26 07:13 - 07:13 (00:00)
user  ssh:notty  68.183.77.86   Tue Apr 26 07:12 - 07:12 (00:00)
user  ssh:notty  68.183.77.86   Tue Apr 26 07:12 - 07:12 (00:00)
user  ssh:notty  68.183.77.86   Tue Apr 26 07:12 - 07:12 (00:00)
user  ssh:notty  68.183.77.86   Tue Apr 26 07:12 - 07:12 (00:00)
user  ssh:notty  68.183.77.86   Tue Apr 26 07:12 - 07:12 (00:00)
user  ssh:notty  68.183.77.86   Tue Apr 26 07:12 - 07:12 (00:00)
user  ssh:notty  68.183.77.86   Tue Apr 26 07:12 - 07:12 (00:00)
user  ssh:notty  68.183.77.86   Tue Apr 26 07:12 - 07:12 (00:00)
user  ssh:notty  68.183.77.86   Tue Apr 26 07:11 - 07:11 (00:00)
user  ssh:notty  68.183.77.86   Tue Apr 26 07:11 - 07:11 (00:00)
user  ssh:notty  68.183.77.86   Tue Apr 26 07:11 - 07:11 (00:00)
user  ssh:notty  68.183.77.86   Tue Apr 26 07:11 - 07:11 (00:00)
user  ssh:notty  68.183.77.86   Tue Apr 26 07:11 - 07:11 (00:00)
user  ssh:notty  68.183.77.86   Tue Apr 26 07:10 - 07:10 (00:00)
user  ssh:notty  68.183.77.86   Tue Apr 26 07:10 - 07:10 (00:00)
user  ssh:notty  68.183.77.86   Tue Apr 26 07:10 - 07:10 (00:00)
user  ssh:notty  68.183.77.86   Tue Apr 26 07:10 - 07:10 (00:00)
user  ssh:notty  68.183.77.86   Tue Apr 26 07:10 - 07:10 (00:00)
user  ssh:notty  68.183.77.86   Tue Apr 26 07:09 - 07:09 (00:00)
user  ssh:notty  68.183.77.86   Tue Apr 26 07:09 - 07:09 (00:00)
root  ssh:notty  185.28.39.119  Tue Apr 26 06:27 - 06:27 (00:00)
```

To see the count of how many times they have tried. This is the Brute Force Attack

```
Usage: lastb [-num | -n num] [-f file] [-t Y
[root@trainer ~]# lastb | wc -l
90
[root@trainer ~]#
```

Password authentication error will be seen when tried with different username

```
[root@master docker]# lastb
deploy ssh:notty 164.92.215.30 Tue Apr 26 12:45 - 12:45 (00:00)
deploy ssh:notty 164.92.215.30 Tue Apr 26 12:44 - 12:44 (00:00)
deploy ssh:notty 164.92.215.30 Tue Apr 26 12:44 - 12:44 (00:00)
demo ssh:notty 164.92.215.30 Tue Apr 26 12:43 - 12:43 (00:00)
data ssh:notty 164.92.215.30 Tue Apr 26 12:42 - 12:42 (00:00)
centos ssh:notty 164.92.215.30 Tue Apr 26 12:41 - 12:41 (00:00)
bot ssh:notty 164.92.215.30 Tue Apr 26 12:41 - 12:41 (00:00)
ansible ssh:notty 64.227.175.194 Tue Apr 26 12:40 - 12:40 (00:00)
bigdata ssh:notty 164.92.215.30 Tue Apr 26 12:40 - 12:40 (00:00)
appuser ssh:notty 164.92.215.30 Tue Apr 26 12:39 - 12:39 (00:00)
app ssh:notty 164.92.215.30 Tue Apr 26 12:38 - 12:38 (00:00)
app ssh:notty 164.92.215.30 Tue Apr 26 12:37 - 12:37 (00:00)
app ssh:notty 164.92.215.30 Tue Apr 26 12:37 - 12:37 (00:00)
app ssh:notty 164.92.215.30 Tue Apr 26 12:36 - 12:36 (00:00)
admin ssh:notty 164.92.215.30 Tue Apr 26 12:34 - 12:34 (00:00)
admin ssh:notty 164.92.215.30 Tue Apr 26 12:33 - 12:33 (00:00)
admin ssh:notty 164.92.215.30 Tue Apr 26 12:32 - 12:32 (00:00)
awsgui ssh:notty 164.92.215.30 Tue Apr 26 12:31 - 12:31 (00:00)
user ssh:notty 137.184.214.165 Tue Apr 26 11:26 - 11:26 (00:00)
admin ssh:notty 137.184.214.165 Tue Apr 26 11:26 - 11:26 (00:00)
test ssh:notty 137.184.214.165 Tue Apr 26 11:26 - 11:26 (00:00)
admin ssh:notty 137.184.214.165 Tue Apr 26 11:26 - 11:26 (00:00)
oracle ssh:notty 137.184.214.165 Tue Apr 26 11:26 - 11:26 (00:00)
admin ssh:notty 137.184.214.165 Tue Apr 26 11:26 - 11:26 (00:00)
admin ssh:notty 137.184.214.165 Tue Apr 26 11:26 - 11:26 (00:00)
support ssh:notty 137.184.214.165 Tue Apr 26 11:26 - 11:26 (00:00)
usuario ssh:notty 137.184.214.165 Tue Apr 26 11:26 - 11:26 (00:00)
admin ssh:notty 137.184.214.165 Tue Apr 26 11:26 - 11:26 (00:00)
admin ssh:notty 137.184.214.165 Tue Apr 26 11:26 - 11:26 (00:00)
admin ssh:notty 137.184.214.165 Tue Apr 26 11:26 - 11:26 (00:00)
1234 ssh:notty 137.184.214.165 Tue Apr 26 11:26 - 11:26 (00:00)
admin ssh:notty 137.184.214.165 Tue Apr 26 11:26 - 11:26 (00:00)
ubnt ssh:notty 137.184.214.165 Tue Apr 26 11:26 - 11:26 (00:00)
username ssh:notty 111.127.99.186 Tue Apr 26 11:05 - 11:05 (00:00)
```

To get Logs:

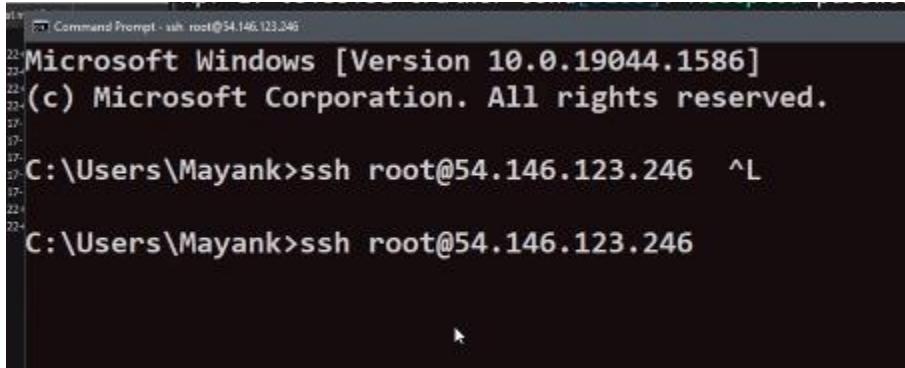
```
[root@master docker]# ls -l /var/log/secure
-rw----- 1 root root 25296 Apr 27 04:42 /var/log/secure
[root@master docker]#
```

Live logs:

```
[root@master docker]# tail -f /var/log/secure
Apr 27 04:04:24 master sshd[3459]: Accepted publickey for ec2-user from 59.91.129.50 port 50219 ssh2: RSA SHA256:VnA
p1NKhB0/0PucMV2b5RZS4sLPTUpM91Djem7P3SC8
Apr 27 04:04:24 master sshd[3459]: pam_unix(sshd:session): session opened for user ec2-user by (uid=0)
Apr 27 04:04:24 master sudo: ec2-user : TTY:pts/0 ; PWD=/home/ec2-user ; USER=root ; COMMAND=/bin/bash
Apr 27 04:04:24 master sudo: pam_unix(sudo:session): session opened for user root by ec2-user(uid=0)
Apr 27 04:15:05 master sshd[9952]: Connection closed by 167.248.133.117 port 41830 [preauth]
Apr 27 04:33:56 master sshd[29280]: Bad protocol version identification '026.003.001' from 202.95.12.44 port 50864
Apr 27 04:34:36 master sshd[29293]: Did not receive identification string from 202.95.12.44 port 51084
Apr 27 04:37:55 master sshd[10047]: Connection closed by 202.95.12.44 port 34790 [preauth]
Apr 27 04:37:56 master sshd[1025]: Protocol major versions differ for 202.95.12.44 port 34858: SSH-2.0-OpenSSH_7.4 v
s. SSH-1.5-Server
Apr 27 04:42:14 master sshd[5049]: Connection closed by 162.142.125.220 port 44488 [preauth]
```

```
[root@trainer ~]# tail -f /var/log/secure
Apr 27 05:35:51 trainer sshd[5283]: Accepted password for root from 59.91.129.50 port 50883 ssh2
Apr 27 05:35:51 trainer sshd[5283]: pam_unix(sshd:session): session opened for user root by (uid=0)
Apr 27 05:35:53 trainer sshd[5285]: error: AuthorizedKeysCommand /opt/aws/bin/eic_run_authorized_ke
ys root SHA256:VnApINKhB0/OPuCMV2b5RZS4sLPTupM91Djem7P3SC8 failed, status 22
Apr 27 05:35:54 trainer sshd[5285]: error: AuthorizedKeysCommand /opt/aws/bin/eic_run_authorized_ke
ys root SHA256:VnApINKhB0/OPuCMV2b5RZS4sLPTupM91Djem7P3SC8 failed, status 22
Apr 27 05:35:54 trainer sshd[5285]: error: AuthorizedKeysCommand /opt/aws/bin/eic_run_authorized_ke
ys root SHA256:VnApINKhB0/OPuCMV2b5RZS4sLPTupM91Djem7P3SC8 failed, status 22
Apr 27 05:35:54 trainer sshd[5285]: error: AuthorizedKeysCommand /opt/aws/bin/eic_run_authorized_ke
ys root SHA256:VnApINKhB0/OPuCMV2b5RZS4sLPTupM91Djem7P3SC8 failed, status 22
Apr 27 05:35:54 trainer sshd[5285]: error: AuthorizedKeysCommand /opt/aws/bin/eic_run_authorized_ke
ys root SHA256:VnApINKhB0/OPuCMV2b5RZS4sLPTupM91Djem7P3SC8 failed, status 22
Apr 27 05:35:54 trainer sshd[5285]: error: AuthorizedKeysCommand /opt/aws/bin/eic_run_authorized_ke
ys root SHA256:VnApINKhB0/OPuCMV2b5RZS4sLPTupM91Djem7P3SC8 failed, status 22
Apr 27 05:35:55 trainer sshd[5285]: error: maximum authentication attempts exceeded for root from 5
9.91.129.50 port 50884 ssh2 [preauth]
Apr 27 05:35:55 trainer sshd[5285]: Disconnecting: Too many authentication failures [preauth]
```

When I try to access it from the local machine? (Command prompt)



You can see the message – password wrong! Permission denied.

```
C:\Users\Mayank>ssh root@54.146.123.246 ^L
[root@trainer ~]# tail -f /var/log/auth.log
Apr 27 05:35:51 trainer sshd[5283]: C:\Users\Mayank>ssh root@54.146.123.246
Apr 27 05:35:51 trainer sshd[5283]: ssh: connect to host 54.146.123.246 port 22: Connecti
Apr 27 05:35:53 trainer sshd[5283]: timed out
Apr 27 05:35:54 root SHA256:VnApINKhB0/OpUcMv2C:\Users\Mayank>ssh root@54.145123.246
Apr 27 05:35:54 root SHA256:VnApINKhB0/OpUcMv2: ssh: Could not resolve hostname 54.145123.246: No suc
Apr 27 05:35:54 root SHA256:VnApINKhB0/OpUcMv2host is known.
Apr 27 05:35:54 root SHA256:VnApINKhB0/OpUcMv2C:\Users\Mayank>ssh root@54.145.123.246
Apr 27 05:35:54 root SHA256:VnApINKhB0/OpUcMv2: The authenticity of host '54.145.123.246 (54.145.123.
Apr 27 05:35:55 root SHA256:VnApINKhB0/OpUcMv2:6)' can't be established.
Apr 27 05:35:55 root sshd[5283]: Warning: Permanently added '54.145.123.246' (ECDSA) to the
Apr 27 05:35:55 root sshd[5283]: list of known hosts.
Apr 27 05:41:21 root@54.145.123.246's password:
[4.18.222.1] 100% 0.00s 0.00s 0.00s
```

Accepted authentication:

```

[root@trainer ~]# tail -f /var/log/secure
Apr 27 05:35:53 trainer sshd[5285]: error: AuthorizedKeysCommand /opt/aws/bin/eic_run_authorized_keys root SHA256:VnApINKhB0/0PucMV2b5RZ54sLPTupM91Djem7P3SC8 failed, status 22
Apr 27 05:35:54 trainer sshd[5285]: error: AuthorizedKeysCommand /opt/aws/bin/eic_run_authorized_keys root SHA256:VnApINKhB0/0PucMV2b5RZ54sLPTupM91Djem7P3SC8 failed, status 22
Apr 27 05:35:54 trainer sshd[5285]: error: AuthorizedKeysCommand /opt/aws/bin/eic_run_authorized_keys root SHA256:VnApINKhB0/0PucMV2b5RZ54sLPTupM91Djem7P3SC8 failed, status 22
Apr 27 05:35:54 trainer sshd[5285]: error: AuthorizedKeysCommand /opt/aws/bin/eic_run_authorized_keys root SHA256:VnApINKhB0/0PucMV2b5RZ54sLPTupM91Djem7P3SC8 failed, status 22
Apr 27 05:35:54 trainer sshd[5285]: error: AuthorizedKeysCommand /opt/aws/bin/eic_run_authorized_keys root SHA256:VnApINKhB0/0PucMV2b5RZ54sLPTupM91Djem7P3SC8 failed, status 22
Apr 27 05:35:54 trainer sshd[5285]: error: AuthorizedKeysCommand /opt/aws/bin/eic_run_authorized_keys root SHA256:VnApINKhB0/0PucMV2b5RZ54sLPTupM91Djem7P3SC8 failed, status 22
Apr 27 05:35:54 trainer sshd[5285]: error: AuthorizedKeysCommand /opt/aws/bin/eic_run_authorized_keys root SHA256:VnApINKhB0/0PucMV2b5RZ54sLPTupM91Djem7P3SC8 failed, status 22
Apr 27 05:35:54 trainer sshd[5285]: error: AuthorizedKeysCommand /opt/aws/bin/eic_run_authorized_keys root SHA256:VnApINKhB0/0PucMV2b5RZ54sLPTupM91Djem7P3SC8 failed, status 22
Apr 27 05:35:54 trainer sshd[5285]: error: maximum authentication attempts exceeded for root from 59.91.129.50 port 50884 ssh2 [preauth]
Apr 27 05:35:55 trainer sshd[5285]: Disconnecting: Too many authentication failures [preauth]
Apr 27 05:41:21 trainer sshd[6902]: pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=0 tty=ssh ruser= rhost=59.91.129.50 user=root
Apr 27 05:41:23 trainer sshd[6902]: Failed password for root from 59.91.129.50 port 50909 ssh2
Apr 27 05:41:42 trainer sshd[6902]: Accepted password for root from 59.91.129.50 port 50909 ssh2
Apr 27 05:41:42 trainer sshd[6902]: pam_unix(sshd:session): session opened for user root by (uid=0)

```

Sources are same – but two different ways of access.

```

^C
[root@trainer ~]# w
 05:41:49 up  2:04,  2 users,  load average:  0.02,  0.02,  0.00
USER   TTY      FROM          LOGIN@    IDLE   JCPU   PCPU WHAT
root   pts/0    59.91.129.50   05:35   0.00s  0.04s  0.00s w
root   pts/1    59.91.129.50   05:41   6.00s  0.02s  0.02s -bash
[root@trainer ~]#

```

Everywhere you have to secure your platform.

– what platform? – Suppose, the machine which you currently use it.

How our platform is secure? Let's check it!

```

[root@trainer ~]# <<X
>
>
> Secure your platform also
>
>
> ssh secure -- permission

```

Access the ssh using default port number – It works!

```

[root@trainer ~]# ssh localhost -v
OpenSSH_7.4p1, OpenSSL 1.0.2k-fips  26 Jan 2017
debug1: Reading configuration data /etc/ssh/ssh_config
debug1: /etc/ssh/ssh_config line 58: Applying options for *
debug1: Connecting to localhost [127.0.0.1] port 22.
debug1: Connection established.
debug1: permanently_set_uid: 0/0
debug1: key_load_public: No such file or directory
debug1: identity file /root/.ssh/id_rsa type -1
debug1: key_load_public: No such file or directory
debug1: identity file /root/.ssh/id_rsa-cert type -1
debug1: key_load_public: No such file or directory
debug1: identity file /root/.ssh/id_dsa type -1
debug1: key_load_public: No such file or directory

```

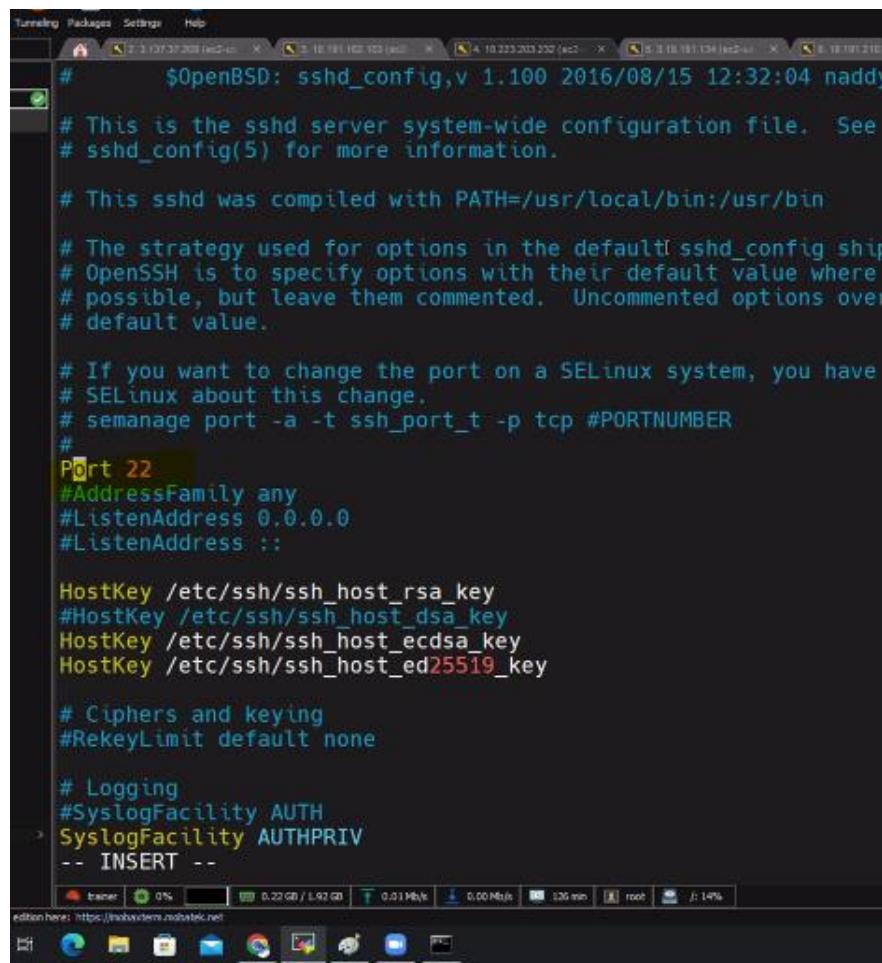
Now, try to use it with different port number

```
[root@trainer ~]# ssh localhost -v -p 3434
OpenSSH_7.4p1, OpenSSL 1.0.2k-fips 26 Jan 2017
debug1: Reading configuration data /etc/ssh/ssh_config
debug1: /etc/ssh/ssh_config line 58: Applying options for *
debug1: Connecting to localhost [127.0.0.1] port 3434.
debug1: connect to address 127.0.0.1 port 3434: Connection refused
ssh: connect to host localhost port 3434: Connection refused
[root@trainer ~]#
```

Edit the configuration file:

```
ssh: connect to host localhost port 3434: Connection refused
[root@trainer ~]# vi /etc/ssh/sshd_config
```

Port 22 is default:



```
# $OpenBSD: sshd_config,v 1.100 2016/08/15 12:32:04 naddy
#
# This is the sshd server system-wide configuration file. See
# sshd_config(5) for more information.

# This sshd was compiled with PATH=/usr/local/bin:/usr/bin

# The strategy used for options in the default sshd_config shipped
# with OpenSSH is to specify options with their default value where
# possible, but leave them commented. Uncommented options over-
# ride default value.

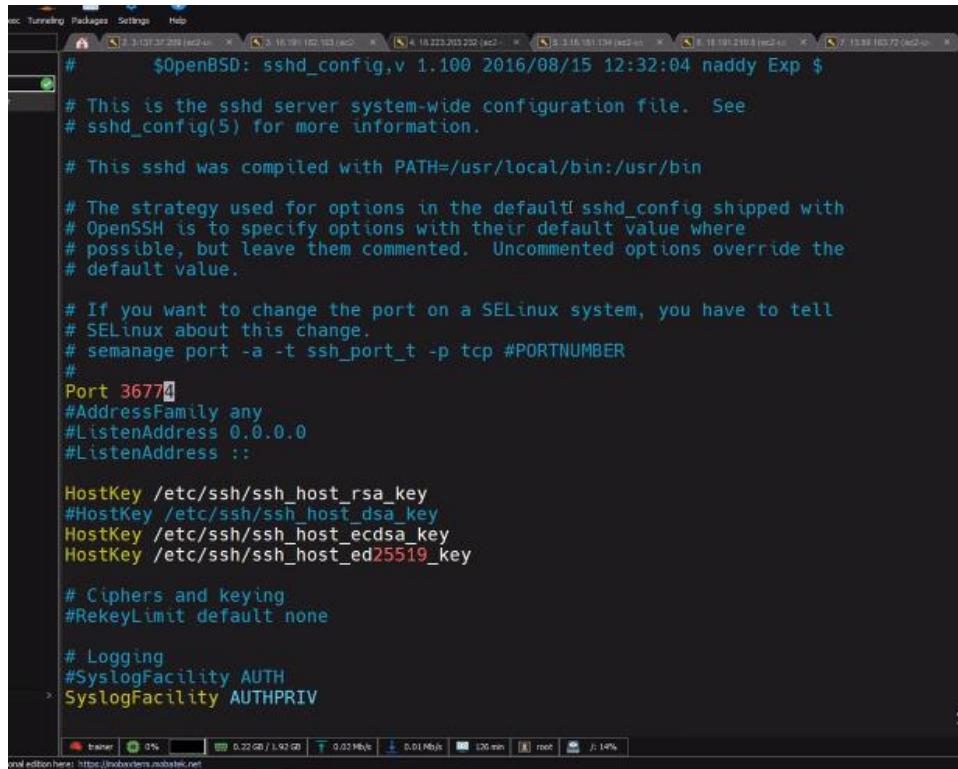
# If you want to change the port on a SELinux system, you have
# to change SELinux about this change.
# semanage port -a -t ssh_port_t -p tcp #PORTNUMBER
#
Port 22
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_dsa_key
HostKey /etc/ssh/ssh_host_ecdsa_key
HostKey /etc/ssh/ssh_host_ed25519_key

# Ciphers and keying
#RekeyLimit default none

# Logging
#SyslogFacility AUTH
SyslogFacility AUTHPRIV
-- INSERT --
```

Change the port number – 36774:



```
# $OpenBSD: sshd_config,v 1.100 2016/08/15 12:32:04 naddy Exp $
#
# This is the sshd server system-wide configuration file. See
# sshd_config(5) for more information.
#
# This sshd was compiled with PATH=/usr/local/bin:/usr/bin
#
# The strategy used for options in the default sshd_config shipped with
# OpenSSH is to specify options with their default value where
# possible, but leave them commented. Uncommented options override the
# default value.
#
# If you want to change the port on a SELinux system, you have to tell
# SELinux about this change.
# semanage port -a -t ssh_port_t -p tcp #PORTNUMBER
#
# Port 36774
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_dsa_key
HostKey /etc/ssh/ssh_host_ecdsa_key
HostKey /etc/ssh/ssh_host_ed25519_key

# Ciphers and keying
#RekeyLimit default none

# Logging
#SyslogFacility AUTH
> SyslogFacility AUTHPRIV
```

Restart the service:

```
[root@trainer ~]# vi /etc/ssh/sshd_config
[root@trainer ~]# systemctl restart sshd
[root@trainer ~]#
```

Using with port 22:

```
[root@trainer ~]# ssh localhost -v
OpenSSH_7.4p1, OpenSSL 1.0.2k-fips 26 Jan 2017
debug1: Reading configuration data /etc/ssh/ssh_config
debug1: /etc/ssh/ssh_config line 58: Applying options for *
debug1: Connecting to localhost [127.0.0.1] port 22.
debug1: connect to address 127.0.0.1 port 22: Connection refused
ssh: connect to host localhost port 22: Connection refused
[root@trainer ~]#
```

Using the wrong password:

```
[root@trainer ~]# ssh localhost -v -p 3434
OpenSSH_7.4p1, OpenSSL 1.0.2k-fips 26 Jan 2017
debug1: Reading configuration data /etc/ssh/ssh_config
debug1: /etc/ssh/ssh_config line 58: Applying options for *
debug1: Connecting to localhost [127.0.0.1] port 3434.
debug1: connect to address 127.0.0.1 port 3434: Connection refused
ssh: connect to host localhost port 3434: Connection refused
[root@trainer ~]# ssh localhost -v -p 3434
```

Using the right port number:

```
debug1: Next authentication method: publickey
debug1: Trying private key: /root/.ssh/id_rsa
debug1: Trying private key: /root/.ssh/id_dsa
debug1: Trying private key: /root/.ssh/id_ecdsa
debug1: Trying private key: /root/.ssh/id_ed25519
debug1: Next authentication method: password
root@localhost's password:
debug1: Authentications that can continue: publickey,gssapi-keyex,gssapi
Permission denied, please try again.
root@localhost's password:
debug1: Authentications that can continue: publickey,gssapi-keyex,gssapi
Permission denied, please try again.
root@localhost's password:
debug1: Authentication succeeded (password).
Authenticated to localhost ([127.0.0.1]:36774).
debug1: channel 0: new [client-session]
debug1: Requesting no-more-sessions@openssh.com
debug1: Entering interactive session.
debug1: pledge: network
debug1: client_input_global_request: rtype hostkeys-00@openssh.com war
debug1: Sending environment.
debug1: Sending env LANG = en_US.UTF-8
Last failed login: Wed Apr 27 05:44:51 UTC 2022 from localhost on ssh:
There were 2 failed login attempts since the last successful login.
Last login: Wed Apr 27 05:43:23 2022 from localhost

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

<https://aws.amazon.com/amazon-linux-2/>

This can be brute force as the port number is limited

```
[root@trainer ~]# << 36774^C
[root@trainer ~]# w
 05:45:50 up 2:08, 1 user,  load average: 0.00, 0.01, 0.00
USER   TTY      FROM          LOGIN@    IDLE   JCPU   PCPU WHAT
root    pts/0    59.91.129.50    05:35   6.00s  0.07s  0.00s w
[root@trainer ~]#
```

Do you know how to create user in Linux?

```
[root@trainer ~]# useradd harry
[root@trainer ~]#
```

How to login with harry?

```
[root@trainer ~]# su - harry
[harry@trainer ~]$
```

Verify:

```
[root@trainer ~]# jh dd harry
[harry@trainer ~]$ whoami
harry
[harry@trainer ~]$
```

Allow user: (by name or ip address we can use it)

Edit in ssh:

```
[harry@trainer ~]$ logout
[root@trainer ~]# vi /etc/ssh/sshd_config
[root@trainer ~]# vi /etc/ssh/sshd_config
[harry@trainer ~]# cat /etc/ssh/sshd_config
#UseDNS yes
#PidFile /var/run/sshd.pid
#MaxStartups 10:30:100
#PermitTunnel no
#ChrootDirectory none
#VersionAddendum none

# no default banner path
#Banner none

# Accept locale-related environment variables
AcceptEnv LANG LC_CTYPE LC_NUMERIC LC_TIME LC_COLLATE LC_MONETARY LC_MESSAGES
AcceptEnv LC_PAPER LC_NAME LC_ADDRESS LC_TELEPHONE LC_MEASUREMENT
AcceptEnv LC_IDENTIFICATION LC_ALL LANGUAGE
AcceptEnv XMODIFIERS

# override default of no subsystems
Subsystem sftp /usr/libexec/openssh/sftp-server

# Example of overriding settings on a per-user basis
#Match User anoncvs
#    X11Forwarding no
#    AllowTcpForwarding no
#    PermitTTY no
#    ForceCommand cvs server

AuthorizedKeysCommand /opt/aws/bin/eic_runAuthorizedKeys %u %f
AuthorizedKeysCommandUser ec2-instance-connect

AllowUsers harry@54.146.194.73
-- INSERT --
```

Restart the sshd and assign the password:

```
[root@trainer ~]# systemctl restart sshd
[root@trainer ~]# passwd harry
Changing password for user harry.
New password:
BAD PASSWORD: The password is a palindrome
Retype new password:
passwd: all authentication tokens updated successfully.
[root@trainer ~]#
```

IP address of Mayank – Sashi is trying to access it.

```
[root@trainer ~]# ssh harry@54.145.123.246 -p 36774
The authenticity of host '[54.145.123.246]:36774 ([54.145.123.246]:36774)' can't be established.
ECDSA key fingerprint is SHA256:TeUIr3ew9wv2cmxirC1Pojq26prWIOdXAhSwTCdHBh8.
ECDSA key fingerprint is MD5:f7:36:f1:96:96:fd:85:ca:5d:d2:04:43:46:f4:3b:c3.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '[54.145.123.246]:36774' (ECDSA) to the list of known hosts.
harry@54.145.123.246's password:
```

Everyone else can't access it as in the configuration file, Sashi IP is only given. For others, it will be accessing denied.

```
[root@trainer ~]# vim /etc/ssh/sshd_config
[root@trainer ~]# systemctl stop sshd
[root@trainer ~]#
```

If someone took the Sashi machine, then in that case stop the service. Now no one can access it.

So, this is the security in our platform.

Top 20: (first 20 lines of that file)

```
[root@trainer ~]# cat /etc/ssh/sshd_config | head -20
#      $OpenBSD: sshd_config,v 1.100 2016/08/15 12:32:04 naddy Exp $
# This is the sshd server system-wide configuration file. See
# sshd_config(5) for more information.
#
# This sshd was compiled with PATH=/usr/local/bin:/usr/bin
#
# The strategy used for options in the default sshd_config shipped with
# OpenSSH is to specify options with their default value where
# possible, but leave them commented. Uncommented options override the
# default value.
#
# If you want to change the port on a SELinux system, you have to tell
# SELinux about this change.
# semanage port -a -t ssh_port_t -p tcp #PORTNUMBER
#
Port 36774
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

[root@trainer ~]#
```

Tail 2: Last two lines

```
#[root@trainer ~]# cat /etc/ssh/sshd_config | tail -2
AllowUsers harry@54.146.194.73
[root@trainer ~]#
```

Commands shared by the trainer:

```
[root@trainer ~]# cat /etc/ssh/sshd_config | tail -2
AllowUsers harry@54.146.194.73
300 systemctl start docker
301 docker images
302 kubectl create deployment alp --image Alpana87/collinsimage
303 systemctl start kubelet
```

```
304 kubectl get all
305 systemctl start kubelet
306 yum install wget -y ; wget 3.137.37.209/config ; mkdir $HOME/.kube/ ; cp config $HOME/.kube/ ; kubectl get nodes
307 kubectl get nodes
308 kubectl get service
309 kubectl create deployment example1 --image nginx
310 kubectl create deployment mayankimage --image nginx
311 clear
312 docker images
313 kubectl create deployment example1 --image nginx
314 w
315 last
316 lastb
317 lastb \ wc -l
318 lastb | wc -l
319 tail -f /var/log/secure
320 q
321 w
322 lastb
323 tail -f /var/log/secure
324 w
325 <<X
```

Secure your platform also

```
326 ssh localhost -v
```

```
327 ssh localhost -v -p 3434
328 vi /etc/ssh/sshd_config
329 systemctl restart sshd
330 ssh localhost -v
331 ssh localhost -v -p 3434
332 vi /etc/ssh/sshd_config
333 ssh localhost -v -p 36774
334 w
335 useradd harry
336 whoami
337 su - harry
338 vi /etc/ssh/sshd_config
339 systemctl restart sshd
340 passwd harry
341 vi /etc/ssh/sshd_config
342 ssh harry@54.145.123.246
343 ssh harry@54.145.123.246 -p 36774
344 vim /etc/ssh/sshd_config
345 systemctl stop sshd
346 cat /etc/ssh/sshd_config | head -20
347 cat /etc/ssh/sshd_config | tail -2
348 history
[root@trainer ~]#
```

How to create a EC2 instance using terraform?

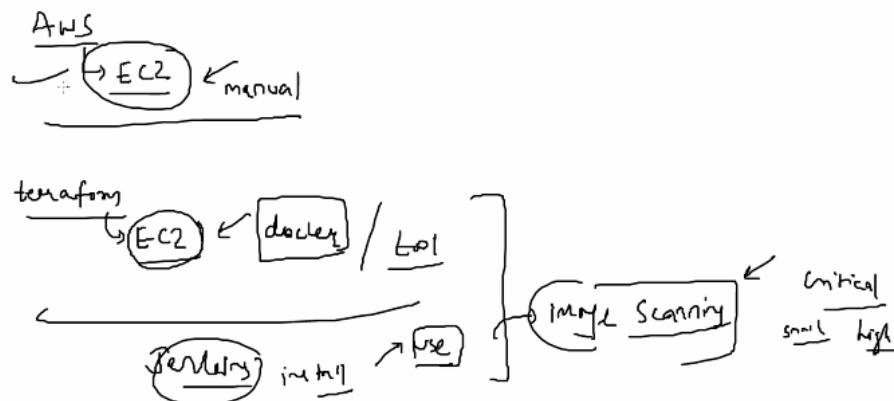
AWS service:

EC2 is created manually

Terraform -> EC2 -> Install docker or another tool.

Jenkins

Image scanning - criticality



Credentials of AWS account:

S.No	Username	Password	Access Key	Secret Key	URL
1	Mayank	User@One2022	oUlbawel/2emPNZAd0QQL0vVLxqr5copr92yUTC		https://sso.cloudsesignin.aws.amazon.com/console
2	user	User@One2022	AKIATM6APKBJ554B3NE	up2m8f0agC8CuDNSE9MdIIryf52sG6fACpV72	https://sso.cloudsesignin.aws.amazon.com/console

Login to AWS account:

The screenshot shows the AWS sign-in interface for an IAM user. The URL is https://us-west-2.login.aws.amazon.com/sso/client_id=amzn1.iam.3A1J3A%3Aconsole&challenge=3HgkW9LnxLctOrWuYdgMvmbqj7dx1E7hQOMw&code_challenge_method=SHA_256&response_type=code&redirect_uri=https://sso.cloudsesignin.aws.amazon.com/console. The page includes fields for 'Account ID (12 digits) or account alias', 'IAM user name', 'Password', and 'Remember me'. It also features a 'Manage passwords...' link and links for 'Sign in using root user email' and 'Forgot password?'. A banner on the right promotes 'Amazon Relational Database Service Free Tier'.

AWS home page:

EC2 service:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Elastic IP
master	i-097279494330194eds	Running	t2.large	✓	✓	us-east-2a	ec2-3-137-32-209.us-east-2	
node1	i-067d9214d146d7179	Running	t2.large	✓	✓	us-east-2a	ec2-18-191-162-103.us...	18.191.162.103
node2	i-0b694f9404b8e4519t	Running	t2.large	✓	✓	us-east-2a	ec2-18-223-203-232.us...	18.223.203.232
node3	i-087eb653a7c16698	Running	t2.large	✓	✓	us-east-2a	ec2-3-151-154.us-east-2	3.15.151.154
node4	i-0463d8bc78afedda	Running	t2.large	✓	✓	us-east-2a	ec2-18-191-210-8us-e...	18.191.210.8
node5	i-058695ab4429d1c5	Running	t2.large	✓	✓	us-east-2a	ec2-13-59-165-72.us-east-2	13.59.165.72

Check the region:

Launch instance:

OS:

You've been opted into the new launch experience. Find out more about this experience or send us feedback. You can still return to the previous version by opting-out.

EC2 > Instances > Launch an instance

Launch an instance info

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags

Name Add additional tags

Application and OS Images (Amazon Machine Image) info

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below.

Search our full catalog including 1000s of application and OS images

Quick Start

Amazon Linux aws Ubuntu Windows Red Hat SUSE Linux Q Browse more AMIs

Number of instances info
1

Software Image (AMI)
Amazon Linux 2 Kernel 5.10 AMI... read more
ami-09625adacc474a7b4

Virtual server type (Instance type)
t2.micro

Firewall (security group)
New security group

Storage (volumes)
1 volume(s) - 8 GiB

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 30 GiB of EBS storage, 2 million I/Os, 1 GiB of snapshots, and 100 GiB of bandwidth to the internet

Cancel **Launch Instance**

Name:

Name and tags

Name Add additional tags

Choose - Amazon AMI:

Application and OS Images (Amazon Machine Image) info

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below.

Search our full catalog including 1000s of application and OS images

Quick Start

Amazon Linux aws Ubuntu Windows Red Hat SUSE Linux Q Browse more AMIs

Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Amazon Linux 2 AMI (HVM) - Kernel 5.10, SSD Volume Type

ami-09625adacc474a7b4 (64-bit (x86)) / ami-0964c8302065908 (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs

Free tier eligible

Description

Amazon Linux 2 Kernel 5.10 AMI 2.0.20220419.0 x86_64 HVM gp2

Architecture AMI ID

64-bit (x86) ami-09625adacc474a7b4

Based on the use case you can choose:

The screenshot shows the AWS Machine Types (AAAMI) page. It lists several instance types under the t2 family:

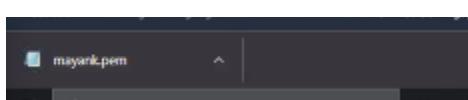
- t1.micro: Family: t1 1 vCPU 0.612 GiB Memory. On-Demand Linux pricing: 0.025 USD per Hour. On-Demand Windows pricing: 0.035 USD per Hour. Status: Free tier eligible.
- t2.nano: Family: t2 1 vCPU 0.5 GiB Memory. On-Demand Linux pricing: 0.0069 USD per Hour. On-Demand Windows pricing: 0.0092 USD per Hour. Status: Free tier eligible.
- t2.micro: Family: t2 1 vCPU 1 GiB Memory. On-Demand Linux pricing: 0.0158 USD per Hour. On-Demand Windows pricing: 0.0215 USD per Hour. Status: Free tier eligible.
- t2.small: Family: t2 1 vCPU 2 GiB Memory. On-Demand Linux pricing: 0.0276 USD per Hour. On-Demand Windows pricing: 0.0368 USD per Hour.
- t2.medium: Family: t2 2 vCPU 4 GiB Memory. On-Demand Linux pricing: 0.0552 USD per Hour. On-Demand Windows pricing: 0.0732 USD per Hour.
- t2.large: Family: t2 4 vCPU 8 GiB Memory. On-Demand Linux pricing: 0.1104 USD per Hour. On-Demand Windows pricing: 0.1464 USD per Hour. Status: Free tier eligible.

Below the list, there is a dropdown menu set to "t2.medium". To the right, there is a "Compare instance types" link.

Download the key pair:

The screenshot shows the "Create key pair" dialog box. It includes the following fields and instructions:

- Key pair name:** A text input field containing "mayank".
- Key pair type:** A radio button group where "RSA" is selected. Below it, it says "RSA encrypted private and public key pair".
- Private key file format:** A radio button group where ".pem" is selected. Below it, it says "For use with OpenSSH".
- Create key pair**: A prominent orange button at the bottom right.



Network setting:

▼ Network settings

Network
vpc-0f0be240468159a79

Subnet
No preference (Default subnet in any availability zone)

Auto-assign public IP
Enable

Security groups (Firewall) [Info](#)
We'll create a new security group called 'launch-wizard-1' with the following rules:

- Allow SSH traffic from Anywhere
Helps you connect to your instance
- Allow HTTPS traffic from the internet
To set up an endpoint, for example when creating a web server
- Allow HTTP traffic from the internet
To set up an endpoint, for example when creating a web server

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

▼ Configure storage [Info](#)

Advanced

Security group:

Auto-assign public IP [Info](#)
 Enable

Firewall (security groups)
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

Create security group Select existing security group

Security group name - required

This security group will be added to all network interfaces. The name can't be edited after the security group is created. Max length is 255 characters. Valid characters: a-z, A-Z, 0-9, spaces, and _-./[{}]=%;]\$/^*

Description - required [Info](#)

Inbound security groups rules

▼ Security group rule 1 (TCP, 22, 0.0.0.0/0)

Type [Info](#): ssh **Protocol [Info](#)**: TCP **Port range [Info](#)**: 22

Source type [Info](#): Anywhere **Source [Info](#)**:

Description - optional [Info](#): e.g. SSH for admin desktop

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

Select the type of the security group as All Traffic instead of ssh.

Security group:

Security group name - required

This security group will be added to all network interfaces. The name can't be edited after the security group is created. Max length is 255 characters. Valid characters: a-z, A-Z, 0-9, spaces, and _<"/\@{}+=&.;\\$^

Description - required [Info](#)

Inbound security groups rules

▼ Security group rule 1 (All, All, 0.0.0.0/0)

[Remove](#)

Type Info	Protocol Info	Port range Info
All traffic	TCP	All
Source type Info	Source Info	Description - optional Info
Anywhere	<input type="text" value="0.0.0.0/0"/> X	e.g. SSH for admin desktop

⚠ Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only. [X](#)

[Add security group rule](#)

Storage:

▼ Configure storage [Info](#)

Advanced

1x GiB [gp2](#) [▼](#) Root volume

ⓘ Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage [X](#)

[Add new volume](#)

0 x File systems [Edit](#)

Launch Instance:

Launching instance

Please wait while we launch your instance.
Do not close your browser while this is loading.

ⓘ Creating security group rules [▼](#) 20% [▲](#)

ⓘ Details [▼](#)

Pending state:

Instances (1/1) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP
Test Machine	i-00336c179aba3d993	Pending	t2.medium	-	No alarms	us-west-1c	ec2-104-72-27-0.us-west...	104.72.27.0	-

Instance: i-00336c179aba3d993 (Test Machine)

Details Security Networking Storage Status checks Monitoring Tags

Instance summary Info

Instance ID: i-00336c179aba3d993 (Test Machine)

Public IPv4 address: 104.72.27.0 [\[open address\]](#)

Private IPv4 addresses: 172.31.29.26

Instance state: Pending

Public IPv4 DNS: ec2-104-72-27-0.us-west-1.compute.amazonaws.com [\[open address\]](#)

Hostname type: IP name: ip-172-31-29-26.us-west-1.compute.internal

Private IP DNS name (IPv4 only): 172.31.29.26 [\[open address\]](#)

Ami ID: ami-00336c179aba3d993

AMI Catalog

Session settings

Basic SSH settings

Remote host: 104.72.27.0 Specify username: ec2-user Port: 22

Advanced SSH settings

X11-Forwarding Compression Remote environment: Interactive shell

Execute command: Do not exit after command ends

SSH-browser type: SFTP protocol Follow SSH path (experimental)

Use private key: C:\Users\Mayank\Downloads\may Adapt locales on remote server

Execute macro at session start:

Terminal settings **Network settings** **Bookmark settings**

Cancel

Storage: 4GB

```
[root@ip-172-31-29-26 ~]# free -G
free: invalid option -- 'G'

Usage:
  free [options]

Options:
  -b, --bytes      show output in bytes
  -k, --kilo       show output in kilobytes
  -m, --mega       show output in megabytes
  -g, --giga       show output in gigabytes
  -t, --tera       show output in terabytes
  -p, --peta       show output in petabytes
  -h, --human      show human-readable output
  --si             use powers of 1000 not 1024
  -l, --lohi       show detailed low and high memory statistics
  -t, --total      show total for RAM + swap
  -s N, --seconds N repeat printing every N seconds
  -c N, --count N  repeat printing N times, then exit
  -w, --wide        wide output

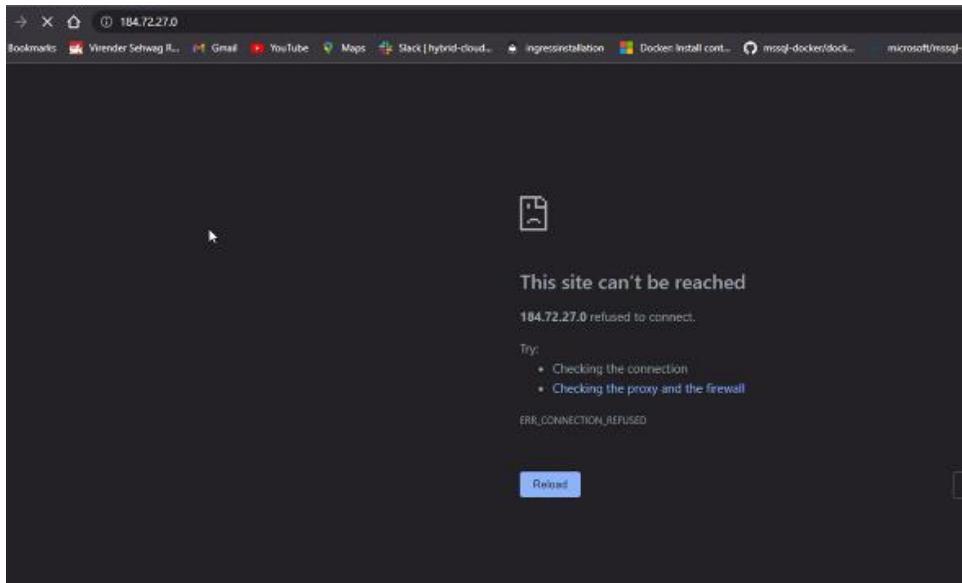
  --help      display this help and exit
  -V, --version   output version information and exit

For more details see free(1).
[root@ip-172-31-29-26 ~]# free -g
              total        used        free      shared  buff/cache   available
Mem:          3           0           3           0           0           0           3
Swap:         0           0           0
[root@ip-172-31-29-26 ~]#
```

CPU: 2

```
[root@ip-172-31-29-26 ~]# lscpu
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Little Endian
CPU(s):                2
On-line CPU(s) list:  0,1
Thread(s) per core:    1
Core(s) per socket:    2
Socket(s):             1
NUMA node(s):          1
Vendor ID:             GenuineIntel
CPU family:            6
Model:                 79
Model name:            Intel(R) Xeon(R) CPU E5-2686 v4 @ 2.30GHz
Stepping:               1
CPU MHz:               2300.099
BogoMIPS:              4600.12
Hypervisor vendor:     Xen
Virtualization type:   full
L1d cache:             32K
L1i cache:             32K
L2 cache:              256K
L3 cache:              46080K
NUMA node0 CPU(s):     0,1
Flags:      fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca
2 ht syscall nx rdtscp lm constant_tsc rep_good nopl xtopology cpuid tsc_knows
sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave avx f16c rdrand
vpcid_single pti fsgsbase bmi1 bmi2 erms invpcid xsaveopt
[root@ip-172-31-29-26 ~]#
```

Web server: Now it is not accessible.



Install Apache service:

```
[root@ip-172-31-29-26 ~]# yum install httpd -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Resolving Dependencies
--> Running transaction check
--> Package httpd.x86_64 0:2.4.52-1.amzn2 will be installed
--> Processing Dependency: httpd-tools = 2.4.52-1.amzn2 for package: httpd-2.4.52-1.amzn2
--> Processing Dependency: httpd-filesystem = 2.4.52-1.amzn2 for package: httpd-2.4.52-1.a
--> Processing Dependency: system-logos-httpd for package: httpd-2.4.52-1.amzn2.x86_64
--> Processing Dependency: mod_http2 for package: httpd-2.4.52-1.amzn2.x86_64
--> Processing Dependency: httpd-filesystem for package: httpd-2.4.52-1.amzn2.x86_64
--> Processing Dependency: /etc/mime.types for package: httpd-2.4.52-1.amzn2.x86_64
--> Processing Dependency: libaprutil-1.so.0()(64bit) for package: httpd-2.4.52-1.amzn2.x8
--> Processing Dependency: libapr-1.so.0()(64bit) for package: httpd-2.4.52-1.amzn2.x86_64
--> Running transaction check
--> Package apr.x86_64 0:1.7.0-9.amzn2 will be installed
--> Package apr-util.x86_64 0:1.6.1-5.amzn2.0.2 will be installed
--> Processing Dependency: apr-util-bdb(x86-64) = 1.6.1-5.amzn2.0.2 for package: apr-util-
--> Package generic-logos-htpd.noarch 0:18.0.0-4.amzn2 will be installed
--> Package httpd-filesystem.noarch 0:2.4.52-1.amzn2 will be installed
--> Package httpd-tools.x86_64 0:2.4.52-1.amzn2 will be installed
--> Package mailcap.noarch 0:2.1.41-2.amzn2 will be installed
--> Package mod_http2.x86_64 0:1.15.19-1.amzn2.0.1 will be installed
--> Running transaction check
--> Package apr-util-bdb.x86_64 0:1.6.1-5.amzn2.0.2 will be installed
```

Configuration:

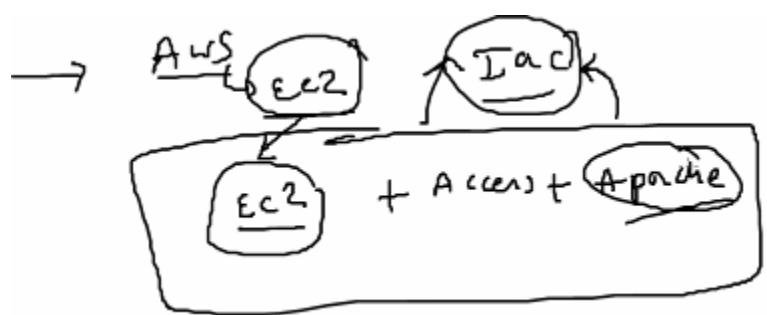
Start the service:

```
[root@ip-172-31-29-26 html]# ls  
[root@ip-172-31-29-26 html]# vim index.html  
[root@ip-172-31-29-26 html]# systemctl start
```

Web service accessible now:

(The website showing the message in the index.html file)

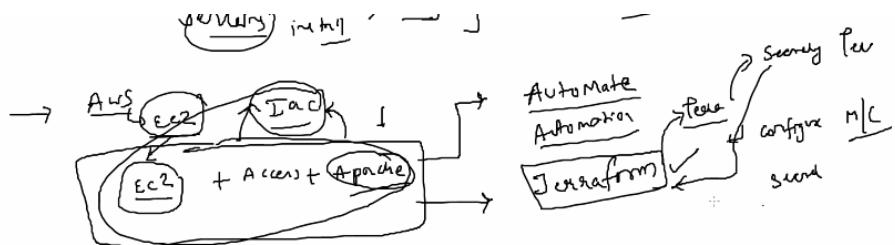
This is a kind of infrastructure.



Automate:



Tool used to automate: Terraform



Task:

Link: <https://sscloudtse.signin.aws.amazon.com/console>

Credentials:

Username: user

Password: User@tse2022

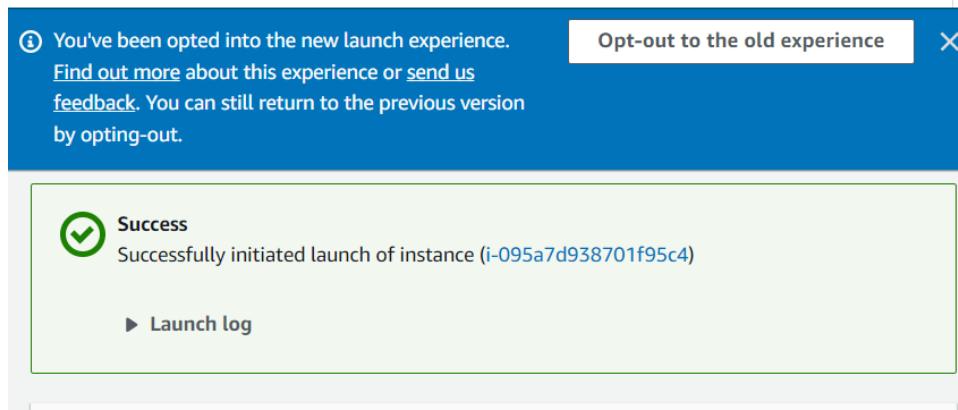
Use key pair for authentication.

1. Login into ur account:

The screenshot shows the AWS Management Console login page. At the top, there's a banner about the new AWS Console Home. Below it, the AWS services menu is visible, with EC2 selected. There are also sections for "Build a solution" (with options like "Launch a virtual machine" and "Build a web app") and "Stay connected to your AWS resources on-the-go" (with information about the AWS Console Mobile App).

2. Create an EC2 machine – access it by the pem file. Make sure the region should be North California.

EC2 > Instances > Launch an instance



EC2 instance:

The screenshot shows the AWS EC2 Instances page with one instance listed. The instance details are as follows:

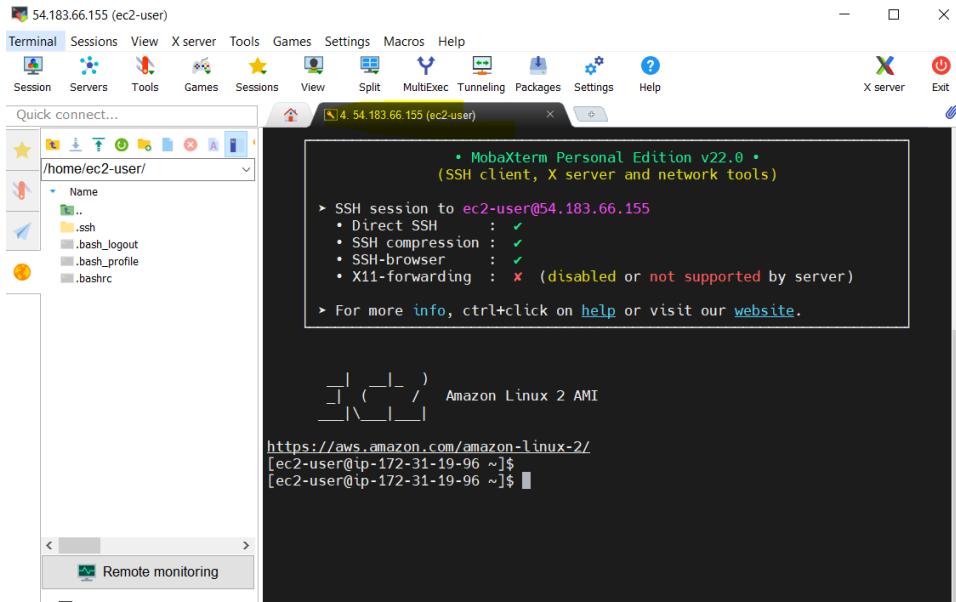
Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
Test Vishali	i-095a7d938701f95c4	Running	t2.micro	Initializing	No alarms	us-west-1c	ec2-54-183-66-155.us-west-1.amazonaws.com

IP address: 54.183.66.155 // 13.56.160.23

The screenshot shows the AWS EC2 Instances page with the Public IP address (54.183.66.155) highlighted in yellow.

Username: ec2-user

Access the machine:



3. Configure apache server into the ec2 machine.

Install the apache:

Make sure you change the user to root using → sudo -i

```
[ec2-user@ip-172-31-19-96 ~]$  
[ec2-user@ip-172-31-19-96 ~]$ yum install httpd -y  
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd  
You need to be root to perform this command.  
[ec2-user@ip-172-31-19-96 ~]$ sudo -i  
[root@ip-172-31-19-96 ~]#
```

Configuration and restart the service:

```
Complete!  
[root@ip-172-31-21-69 ~]# cd /var/www/html/  
[root@ip-172-31-21-69 html]# vim index.html  
[root@ip-172-31-21-69 html]# systemctl start httpd  
[root@ip-172-31-21-69 html]#
```

4. After that, Install docker — create a container that should be exposed on 3434 port no. using quay.io/mayank123modi/simple-webapp

Install the docker:

```
[root@ip-172-31-21-69 ~]# yum install docker -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-mot
amzn2-core
Resolving Dependencies
--> Running transaction check
--> Package docker.x86_64 0:20.10.7-5.amzn2 will be installed
--> Processing Dependency: runc >= 1.0.0 for package: docker-20.10.7-
64
--> Processing Dependency: libcgROUP >= 0.40.rc1-5.15 for package: do
7-5.amzn2.x86_64
--> Processing Dependency: containerd >= 1.3.2 for package: docker-20
n2.x86_64
--> Processing Dependency: pigz for package: docker-20.10.7-5.amzn2.x
> Running transaction check
```

Start the docker service:

```
[root@ip-172-31-21-69 ~]# systemctl start docker
```

```
docker run -itd --name newcont2 -p 3435:8080 quay.io/mayank123modi/simple-webapp
```

```
[root@ip-172-31-21-69 ~]# docker run -itd --name newcont1 -p 3434:80 quay.io/
mayank123modi/simple-webapp
Unable to find image 'quay.io/mayank123modi/simple-webapp:latest' locally
latest: Pulling from mayank123modi/simple-webapp
7d0b3176f146: Pull complete
656769b42b54: Pull complete
46877b8f32e2: Pull complete
f5e790313325: Pull complete
03b307d006b6: Pull complete
a7b78dcf172f: Pull complete
cc19739c4acc: Pull complete
Digest: sha256:9a22693032c8fc8d93a4bb30b8e8d36804736c1a06d3419fc83e4a7ecb85c5f9
Status: Downloaded newer image for quay.io/mayank123modi/simple-webapp:latest
e64095b098134df726be281f1c7582f7cdf42887c062a4543f1259efb0866146
[root@ip-172-31-21-69 ~]#
```

The error here is don't use port 80. It should be port 8080.

```
[root@ip-172-31-21-69 ~]# docker run -itd --name newcont3 -p 3435:8080 quay.io/
mayank123modi/simple-webapp
b7e4c4c096c06b3e21593006cc461c4df333fb65ef06c4aa93b206aa1b9624b
[root@ip-172-31-21-69 ~]#
```

5. Share me the URL

<http://13.56.160.23:3435/>



Terminate the EC2 instance once the task is completed.

Understand about terraform:

Can you automate the above process using terraform? Yes!

What is terraform?

Tool which help to automate the infrastructure – it can be cloud/ on premises/ private cloud.

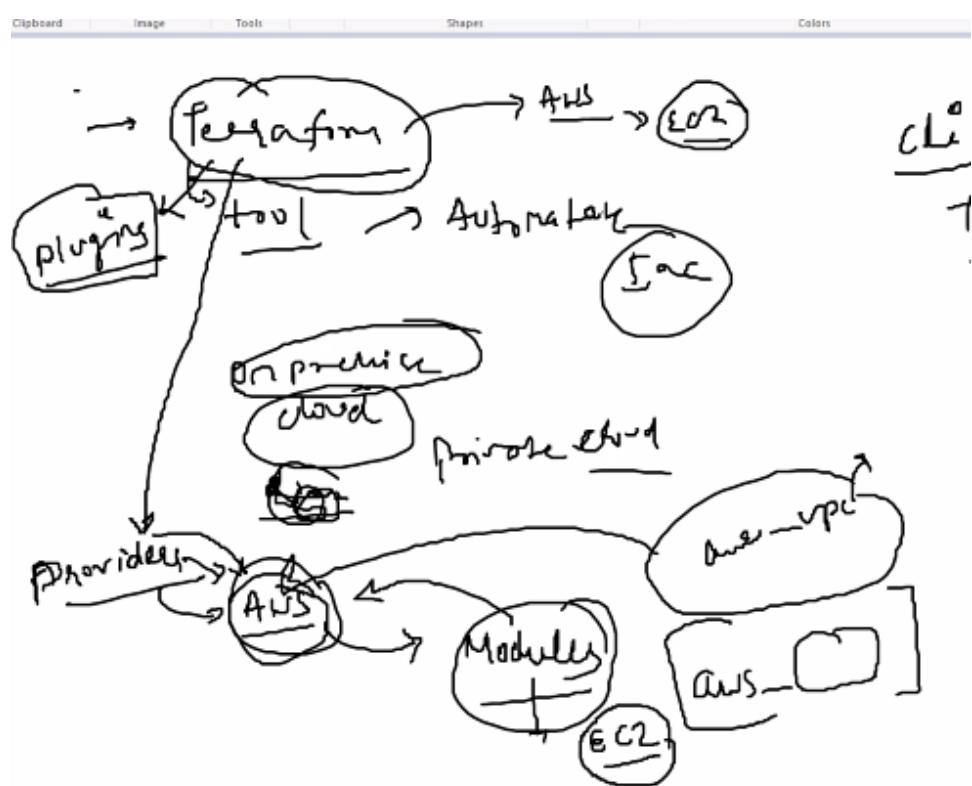
How terraform will automate the ec2 instance using AWS? Using plugins in the tools

Who is the provider here? AWS

Terraform will be using this plugins

After that, there is a certain type of modules – ec2 services (aws_<****>) - This is the module name. All the module is coming from AWS.

From CLI only we will access it.



How to provide the credentials of AWS to the terraform?

First, we need to configure the AWS to give the authentication



Config aws ch^a
↑ credentials



Terraform file extension is .tf files

terraform
file
- tf

Terraform providers:

The screenshot shows the Terraform Registry interface. At the top, there's a search bar and navigation links for 'Browse', 'Publish', and 'Sign-in'. Below the header, there's a section titled 'Providers' with a sub-section 'Providers' and a brief description: 'Providers are a logical abstraction of an upstream API. They are responsible for understanding API interactions and exposing resources.' The main area displays a grid of provider icons and names. The visible providers include AWS (orange), Azure (blue), Google Cloud Platform (blue), Kubernetes (blue), Alibaba Cloud (black), Active Directory (by hashicorp), Archive (by hashicorp), AWS Cloud Control (by hashicorp), Azure Active Directory (by hashicorp), Azure Stack (by hashicorp), and Boundary (by hashicorp). On the left side, there are filters for 'Tier' (Official, Verified, Community) and 'Category' (HashiCorp Platform, Public Cloud, Asset Management, Cloud Automation, Communication & Messaging, Container Orchestration, Continuous Integration/Deployment (CI/CD), Data Management, Database, Infrastructure (IaaS), Logging & Monitoring, Networking, Platform (PaaS), Security & Authentication, Utility, VCS (Version Control)).

These are the terraform providers. It can automate AWS, kubernetes, Alibaba Cloud

Create ec2 instance by using terraform:

Use this command:

The screenshot shows a browser window with the URL registry.terraform.io/providers/hashicorp/aws/latest/doc/resources/instance. The page displays a Terraform configuration snippet for an AWS instance. The code uses a data block to find the latest Ubuntu AMI, filters it by name and virtualization type, and then creates a resource block for an AWS instance named 'web' using the found AMI. The configuration includes tags for 'HelloWorld'.

```

data "aws_ami" "ubuntu" {
  most_recent = true

  filter {
    name   = "name"
    values = ["ubuntu/images/hvm-ssd/ubuntu-focal-20-04-amd64-server-*"]
  }

  filter {
    name   = "virtualization-type"
    values = ["hvm"]
  }

  owners = ["099720109477"] # Canonical
}

resource "aws_instance" "web" {
  ami           = data.aws_ami.ubuntu.id
  instance_type = "t3.micro"

  tags = [
    { name = "HelloWorld" }
  ]
}

```

Terraform language: HashiCorp Configuration Language

The screenshot shows a Google search results page for the query 'terraform language'. The top result is a link to the HashiCorp Configuration Language documentation on the Terraform website. The snippet from the page describes HCL as a balance between human readability and machine-friendliness, mentioning its JSON compatibility.

[Syntax - 0.11 Configuration Language | Terraform by HashiCorp](https://www.terraform.io/language/configuration-0-11)

Access the machine

The screenshot shows a terminal window with three tabs. The active tab shows a user running a command on an EC2 instance. The command 'ls' lists files like 'calico.yaml', 'code', and 'docker'. When the user tries to run 'terraform', they receive an error message stating 'command not found'.

```

[root@master ~]# ls
calico.yaml  code  docker
[root@master ~]# terraform
[bash: terraform: command not found
[root@master ~]#

```

HashiCorp Learn | Browse products | Sign in

Jump to section | Show Terminal | Search | Docs | Forum | Bookmark

Terraform

AWS

- What is Infrastructure as Code with Terraform?
- Install Terraform**
- Build Infrastructure
- Change Infrastructure
- Destroy Infrastructure
- Define Input Variables
- Query Data with Outputs
- Store Remote State

Install Terraform

Reference this often? [Create an account](#) to bookmark tutorials.

8 MIN | PRODUCTS USED: Terraform

This tutorial also appears in: OCI, GCP, Docker, Associate Tutorials and Azure.

[Download this video](#)

HashiCorp | Browse Products | About HashiCorp

Terraform Overview | Editors | Registry | Tutorials | Docs | Community | Status | Download CLI

Downloads **Download Terraform**

Below are the available downloads for the latest version of Terraform (0.15.5). Please download the proper package for your operating system and architecture.

RHEL/Fedora Yum Packages

Terraform is distributed as a single binary. Install Terraform by unzipping it and moving it to.

Hashicorp currently maintains the stable packages for the following Linux distributions:

Ubuntu/Debian CentOS/RHEL Fedora Amazon Linux

Install `yum-config-manager` to manage your repositories.

```
$ sudo yum install -y yum-utils
```

Copy

Use `yum-config-manager` to add the official HashiCorp Linux repository.

```
$ sudo yum-config-manager --add-repo https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.i
```

Copy

Install.

```
$ sudo yum -y install terraform
```

Copy

TIP: Now that you have added the HashiCorp repository, you can install Vault, Consul, Nomad and Packer with the same command.

Link: <https://learn.hashicorp.com/tutorials/terraform/install-cli>

```
-bash: terraform: command not found
[root@master ~]# sudo yum install -y yum-utils
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Repository 'kube' is missing name in configuration, using id
amzn2-core
Package yum-utils-1.1.31-46.amzn2.0.1.noarch already installed and latest version
Nothing to do
[root@master ~]# sudo yum-config-manager --add-repo https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.repo
```

Verify:

```

Complete:
[root@master ~]# terraform --help
[root@master ~]# terraform --help
Usage: terraform [global options] <subcommand> [args]

The available commands for execution are listed below.
The primary workflow commands are given first, followed by
less common or more advanced commands.

Main commands:
  init      Prepare your working directory for other commands
  validate   Check whether the configuration is valid
  plan       Show changes required by the current configuration
  apply      Create or update infrastructure
  destroy    Destroy previously-created infrastructure

All other commands:
  console    Try Terraform expressions at an interactive command prompt
  fmt        Reformatted your configuration in the standard style
  force-unlock Release a stuck lock on the current workspace
  get        Install or upgrade remote Terraform modules
  graph      Generate a Graphviz graph of the steps in an operation
  import     Associate existing infrastructure with a Terraform resource
  login      Obtain and save credentials for a remote host

```

Check whether terraform command is working or not:

```

[root@master ~]# terraform
.bash_history .bash_profile calico.yaml .cshrc .docker/ .ssh/ .viminfo
.bash_logout .bashrc code/ docker/ .kube/ .tcshrc
[root@master ~]# terraform terraform -install-autocomplete
Terraform has no command named "terraform".

To see all of Terraform's top-level commands, run:
  terraform -help

[root@master ~]#

```

You can see the command which can be used in terraform:

```

[root@master ~]# terraform -install-autocomplete
[root@master ~]# terraform
.bash_history .bash_profile calico.yaml .cshrc .docker/ .ssh/ .viminfo
.bash_logout .bashrc code/ docker/ .kube/ .tcshrc
[root@master ~]# bash
[root@master ~]# terraform

```

Create a folder:

```

Positions View Split MultiExec Tunneling Packages Settings Help
[2, 3, 137, 37, 203 (ec2-user) X [3, 18, 181, 102, 103] (ec2-... X [4, 18, 233, 203, 200] (ec2-... X [5, 3, 18, 181]
[root@master ~]# mkdir terracode
[root@master ~]# ls
calico.yaml code docker terracode/
[root@master ~]# cd terracode/
[root@master terracode]# ls
[root@master terracode]# ls -la
total 0
drwxr-xr-x 2 root root 6 Apr 27 08:59 .
dr-xr-x--- 9 root root 250 Apr 27 08:59 ..
[root@master terracode]#

```

Create a tf file:

```
[root@master ~]# mkdir terracode
[root@master ~]# ls
calico.yaml code docker terracode
[root@master ~]# cd terracode/
[root@master terracode]# ls
[root@master terracode]# ls -la
total 0
drwxr-xr-x 2 root root 6 Apr 27 08:59 .
dr-xr-x--- 9 root root 250 Apr 27 08:59 ..
[root@master terracode]# vim ec2.tf
```

How to find the credentials of provider? Use aws command

```
[root@master terracode]# vim ec2.tf
[root@master terracode]# aws
Note: AWS CLI version 2, the latest major version of the AWS CLI, is now stable and recommended for general use. See https://docs.aws.amazon.com/cli/latest/userguide/install叮嘱
usage: aws [options] <command> <subcommand> [<subcommand> ...] [parameters]
To see help text, you can run:
aws help
aws <command> help
aws <command> <subcommand> help
aws: error: too few arguments
[root@master terracode]#
```

Whether this command is working or not – check it!

Can you find any file .aws?

```
[root@master ~]# ls -la
total 260
dr-xr-x--- 9 root root 250 Apr 27 09:00 .
dr-xr-xr-x 18 root root 257 Apr 26 06:49 ..
-rw----- 1 root root 4270 Apr 26 12:45 .bash_history
-rw-r--r-- 1 root root 18 Oct 18 2017 .bash_logout
-rw-r--r-- 1 root root 176 Oct 18 2017 .bash_profile
-rw-r--r-- 1 root root 218 Apr 27 08:57 .bashrc
-rw-r--r-- 1 root root 222270 Apr 26 07:16 calico.yaml
drwxr-xr-x 2 root root 45 Apr 26 10:58 code
-rw-r--r-- 1 root root 100 Oct 18 2017 .cshrc
drwxr-xr-x 2 root root 42 Apr 27 04:59 docker
drwx----- 2 root root 25 Apr 27 04:16 .docker
drwxr-xr-x 3 root root 33 Apr 27 04:09 .kube
drwx----- 2 root root 29 Apr 26 06:49 .ssh
-rw-r--r-- 1 root root 129 Oct 18 2017 .tcshrc
drwxr-xr-x 2 root root 20 Apr 27 09:00 terracode
drwxr-xr-x 2 root root 34 Apr 27 08:57 .terraform.d
-rw----- 1 root root 9827 Apr 27 09:00 .viminfo
[root@master ~]#
```

You need to tell you need to configure:

```
[root@master ~]# aws configure --profile mayank
AWS Access Key ID [None]:
```

Go to security:

Search for services, features, blogs, docs, and more [Alt+5] Global

My security credentials

Account details

User name	mayank (created on 2022-04-25 08:16 UTC+0530)
User ARN	arn:aws:iam::232221966626:user/mayank
AWS account ID	232221966626
Account canonical user ID	05849bb6b685a40c54f49ab064436e4901c72760a801342031da245a6cda962

AWS IAM credentials AWS CodeCommit credentials Amazon MCS credentials

Password for console access

As an IAM user, you need a password to access the AWS Management Console. We recommend changing your password on a regular basis. Your current password is 2 days old. Learn more.

[Change password](#)

Access keys for CLI, SDK, & API access

Use access keys to make programmatic calls to AWS from the AWS CLI, Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time.

For your protection, you should never share your secret keys with anyone. As a best practice, we recommend frequent key rotation. If you lose or forget your secret key, you cannot retrieve it. Instead, create a new access key and make the old key inactive. Learn more.

[Create access key](#)

Access key ID	Status	Created	Last used	Actions
AKIAATMEMAPKBPCIMDI3E	Active	2022-04-25 08:17 UTC+0530	N/A	Make inactive Delete

Multi-factor authentication (MFA)

For increased security, we recommend configuring MFA to help protect your AWS resources. MFA requires users to type a unique authentication code from an approved authentication device when they sign in to AWS. Learn more.

Security key:

S No	Username	Password	Access key	Secret Key	URL
1	Mayank	User@one2022		oUJbxweI/2emPNZAd0QQLtoVlwqr5cojr9ZYUTC	https://sscloudse.sgnin.aws.amazon.com/console
2	user	User@tsa2022	AKIAATMEMAPKBJS5A3NE	up2m8fo8gC8FCUDN9EE9MdUlhYf52sG6fACpV72	https://sscloudse.sgnin.aws.amazon.com/console

```
[root@master ~]# aws configure --profile mayank
AWS Access Key ID [None]: AKIAATMEMAPKBPCIMDI3E
AWS Secret Access Key [None]: oUJbxweI/2emPNZAd0QQLtoVlwqr5cojr9ZYUTC
Default region name [None]: us-west-1
Default output format [None]:
[root@master ~]#
```

To check: (In case you want to change it)

```
[root@master ~]# aws configure --profile mayank
AWS Access Key ID [*****DI3E]:
```

List of AWS profile:

```
[root@master ~]# aws configure list --profile mayank
      Name      Value   Type    Location
      ----      ----   ----    -----
profile           mayank     manual   --profile
access_key        *****DI3E shared-credentials-file
secret_key        *****YUTC shared-credentials-file
region           us-west-1    config-file  ~/.aws/config
[root@master ~]#
```

To check whether you have .aws file:

```
[root@master ~]# ls -la
total 260
dr-xr-x--- 10 root root 262 Apr 27 09:03 .
dr-xr-xr-x 18 root root 257 Apr 26 06:49 ..
drwxr-xr-x 2 root root 39 Apr 27 09:03 .aws [REDACTED]
-rw----- 1 root root 4270 Apr 26 12:45 .bash_history
-rw-r--r-- 1 root root 18 Oct 18 2017 .bash_logout
-rw-r--r-- 1 root root 176 Oct 18 2017 .bash_profile
-rw-r--r-- 1 root root 218 Apr 27 08:57 .bashrc
-rw-r--r-- 1 root root 222270 Apr 26 07:16 calico.yaml
drwxr-xr-x 2 root root 45 Apr 26 10:58 code
-rw-r--r-- 1 root root 100 Oct 18 2017 .cshrc
drwxr-xr-x 2 root root 42 Apr 27 04:59 docker
drwx----- 2 root root 25 Apr 27 04:16 .docker
drwxr-xr-x 3 root root 33 Apr 27 04:09 .kube
drwx----- 2 root root 29 Apr 26 06:49 .ssh
-rw-r--r-- 1 root root 129 Oct 18 2017 .tcshrc
drwxr-xr-x 2 root root 20 Apr 27 09:00 terracode
drwxr-xr-x 2 root root 34 Apr 27 08:57 .terraform.d
-rw----- 1 root root 9827 Apr 27 09:00 .viminfo
[root@master ~]# cd .aws
[root@master .aws]# ls
config credentials
[root@master .aws]# cat config
[profile mayank]
region = us-west-1
[root@master .aws]# cat cr
```

To check the credentials inside .aws folder:

```
[root@master ~]# cd .aws
[root@master .aws]# ls
config credentials
[root@master .aws]# cat config
[profile mayank]
region = us-west-1
[root@master .aws]# cat credentials
[mayank]
aws_access_key_id = AKIATMEMAPKBPCIMDI3E
aws_secret_access_key = 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
[root@master .aws]#
```

Open the terraform file:

```
[root@master ~]# cd terracode/
[root@master terracode]# ls
ec2.tf
[root@master terracode]# v
```

Usually, you don't need to write the Terraform code – copy from google – you don't need to type it.

The screenshot shows the AWS provider documentation on the HashiCorp Terraform website. The left sidebar lists various AWS services like ACM, API Gateway, Lambda, etc. The main content area shows examples for Terraform 0.13 and later, featuring code snippets for configuring the AWS provider and creating a VPC.

```
provider "aws" {
  region      = "us-west-1"
  profile     = "mayank"
}

# Create a VPC
resource "aws_vpc" "example" {
  cidr_block = "10.0.0.0/16"
}
```

Edit the terraform file:

A terminal window showing a partially typed Terraform configuration. The provider block is defined with the region set to us-west-1 and a profile named mayank.

```
provider "aws" {
  region      = "us-west-1"
  profile     = "mayank"
```

View the terraform file:

A terminal window showing the full Terraform configuration file (ec2.tf) with the provider block defined.

```
[root@master terracode]# cat ec2.tf
provider "aws" {
  region      = "us-west-1"
  profile     = "mayank"
}

[root@master terracode]#
```

Now, you don't see the .terraform file:

```
[root@master terracode]# ls -la
total 4
drwxr-xr-x  2 root root  20 Apr 27 09:07 .
dr-xr-x--- 10 root root 262 Apr 27 09:07 ..
-rw-r--r--  1 root root  97 Apr 27 09:07 ec2.tf
[root@master terracode]#
```

Download module or plugins:

This will download the AWSmodules/plugins

```
[root@master terracode]# terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v4.11.0...
```

You will see the .terraform folder being created

```
[root@master terracode]# ls -la
total 8
drwxr-xr-x  3 root root  65 Apr 27 09:07 .
dr-xr-x--- 10 root root 262 Apr 27 09:07 ..
-rw-r--r--  1 root root  97 Apr 27 09:07 ec2.tf
drwxr-xr-x  3 root root  23 Apr 27 09:07 .terraform
-rw-r--r--  1 root root 1152 Apr 27 09:07 .terraform.lock.hcl
[root@master terracode]# cd .terraform/
[root@master .terraform]# ls
providers
[root@master .terraform]# cd providers/registry.terraform.io/hashicorp/aws/4.11.0/linux_amd64/
[root@master linux_amd64]# s
bash: s: command not found
[root@master linux_amd64]# ls
terraform-provider-aws_v4.11.0_x5
[root@master linux_amd64]#
```

Mention the provider:

```
[root@master ~]# cd terracode/ls
bash: cd: terracode/ls: No such file or directory
[root@master ~]# cd terracode/
[root@master terracode]# cat ec2.tf
provider "aws" {
    region          = "us-west-1"
    profile         = "mayank"
}

[root@master terracode]#
```

What I need to do?

Need to create the instance:

It's a resource called aws_instance

All these should be saved in a variable – collinsterraform

In {} write whatever you want? Ami, instance type, security group, tag, key_pair

To check the ID of the AMI. (open the aws account)

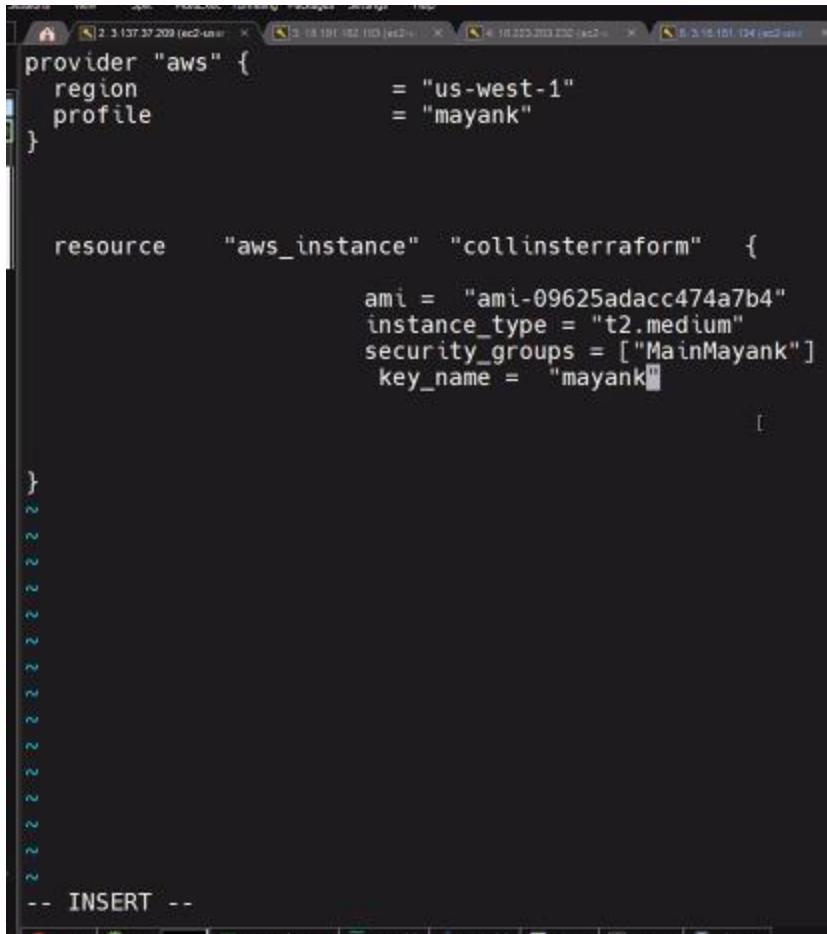
ID

The screenshot shows a list of available Amazon Machine Images (AMIs). The first item in the list is highlighted with a yellow background. The list includes:

- Amazon Linux 2 AMI (HVM) - Kernel 5.10, SSD Volume Type
- Amazon Linux 2 AMI (HVM) - Kernel 4.14, SSD Volume Type
- Deep Learning AMI GPU PyTorch 1.11.0 (Amazon Linux 2) 20220328
- Deep Learning AMI (Amazon Linux 2) Version 6.0.0
- Amazon Linux 2 LTS with SQL Server 2017 Standard
- Amazon Linux 2 with .NET 6, PowerShell, Mono, and MATE Desktop Environment
- Amazon Linux 2 AMI (HVM) - Kernel 5.10, SSD Volume Type

A blue box labeled "ID" is overlaid on the top of the screenshot.

Edit the terraform file:

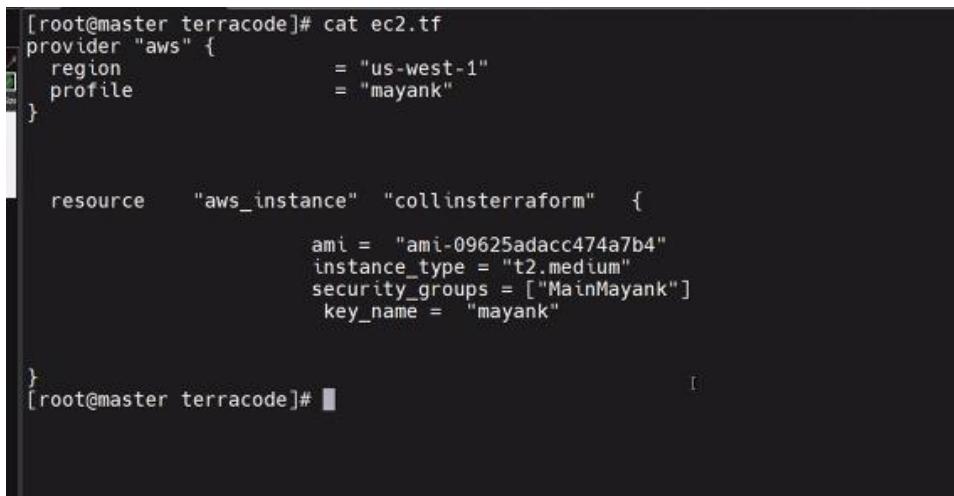


```
provider "aws" {
  region      = "us-west-1"
  profile     = "mayank"
}

resource "aws_instance" "collinsterraform" {
  ami          = "ami-09625adacc474a7b4"
  instance_type = "t2.medium"
  security_groups = ["MainMayank"]
  key_name      = "mayank"
}

-- INSERT --
```

View terraform file:



```
[root@master terracode]# cat ec2.tf
provider "aws" {
  region      = "us-west-1"
  profile     = "mayank"
}

resource "aws_instance" "collinsterraform" {
  ami          = "ami-09625adacc474a7b4"
  instance_type = "t2.medium"
  security_groups = ["MainMayank"]
  key_name      = "mayank"
}
```

Key pair: (if we need to create new): This method you can follow

Example Usage

```
resource "aws_key_pair" "deployer" {
  key_name   = "deployer-key"
  public_key = "ssh-rsa AAAAB3NzaC1yc2EAAQDQABAAQDQJF6tyPEFEzV8LX3XBBsXdm8Qz1x2i
}
```

Security group: (need to create new)

The screenshot shows the AWS Documentation search results for the term 'security'. The search bar at the top contains 'security'. Below it, there are 23 matching results listed under two main categories: 'Security Hub' and 'VPC (Virtual Private Cloud)'. Under 'VPC (Virtual Private Cloud)', the 'Resources' section is expanded, and the 'aws_security_group' resource is highlighted with a blue underline. To the right of the search results, a detailed example of creating an AWS Security Group named 'allow_tls' is shown. The code defines the security group with an ingress rule allowing TLS traffic from the VPC's CIDR block (0.0.0.0/0) to port 443, and an egress rule allowing all traffic (0 to -1). It also adds a tag 'Name: allow_tls'. A note below the code explains that by default, AWS creates an 'ALLOW ALL' egress rule when creating a new Security Group inside a VPC. Terraform will remove this rule and require you to specifically re-create it if you desire that rule.

```
resource "aws_security_group" "allow_tls" {
  name        = "allow_tls"
  description = "Allow TLS inbound traffic"
  vpc_id      = aws_vpc.main.id

  ingress {
    description = "TLS from VPC"
    from_port   = 443
    to_port     = 443
    protocol    = "tcp"
    cidr_blocks = [aws_vpc.main.cidr_block]
    ipv6_cidr_blocks = [aws_vpc.main.ipv6_cidr_block]
  }

  egress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"]
    ipv6_cidr_blocks = ["::/0"]
  }

  tags = {
    Name = "allow_tls"
  }
}
```

⚠️ NOTE on Egress rules:
By default, AWS creates an `ALLOW ALL` egress rule when creating a new Security Group inside a VPC. When creating a new Security Group inside a VPC, **Terraform will remove this rule**, and require you specifically re-create it if you desire that rule. We feel this lessens surprises in terms of controlling your egress rules. If you desire this rule to be in place, use this `:egress` block:

How to run this terraform file? Using attach command

Validate command:

```
[root@master terracode]# terraform validate
Success! The configuration is valid.
```

Plan command:

```
[root@master terracode]# terraform plan
[root@master terracode]# terraform plan

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with
the following symbols:
  + create

Terraform will perform the following actions:

# aws_instance.collinsterraform will be created
+ resource "aws_instance" "collinsterraform" {
    + ami                               = "ami-09625adacc474a7b4"
    + arn                               = "(known after apply)"
    + associate_public_ip_address      = "(known after apply)"
    + availability_zone                = "(known after apply)"
    + cpu_core_count                   = "(known after apply)"
    + cpu_threads_per_core            = "(known after apply)"
    + disable_api_termination         = "(known after apply)"
    + ebs_optimized                    = "(known after apply)"
    + get_password_data               = "false"
    + host_id                          = "(known after apply)"
    + id                               = "(known after apply)"
    + instance_initiated_shutdown_behavior = "(known after apply)"
    + instance_state                  = "(known after apply)"
    + instance_type                   = "t2.medium"
    + ipv6_address_count              = "(known after apply)"
    + ipv6_addresses                  = "(known after apply)"
    + key_name                         = "mayank"
    + monitoring                      = "(known after apply)"
```

Plan: 1 to add

Type: yes

```
Plan: 1 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

Enter a value: yes

> aws_instance.collinsterraform: Creating...
```

Apply:

```
[root@master terracode]# terraform apply
```

Now, ec2 instance is created.

	Test Instance	i-03e76f351cf1ff0ef8	Terminated	t2.micro	-	No alarms	+	us-west-1c	-	-	-
<input checked="" type="checkbox"/>	-	i-09b8516b4b5041b65	Running	@Q t2.medium	Initializing	No alarms	+	us-west-1c	ec2-54-177-72-128.us....	54.177.72.128	-

Destroy command:

```
}
```

```
[root@master terracode]# terraform destroy
```

It will ask for the permission:

```

    - enabled = false -> null
}

- metadata_options {
    - http_endpoint           = "enabled" -> null
    - http_put_response_hop_limit = 1 -> null
    - http_tokens              = "optional" -> null
    - instance_metadata_tags   = "disabled" -> null
}

- root_block_device {
    - delete_on_termination = true -> null
    - device_name           = "/dev/xvda" -> null
    - encrypted              = false -> null
    - iops                   = 100 -> null
    - tags                   = {} -> null
    - throughput              = 0 -> null
    - volume_id               = "vol-038ea8cc12fe26030" -> null
    - volume_size              = 8 -> null
    - volume_type              = "gp2" -> null
}
}

Plan: 0 to add, 0 to change, 1 to destroy.

Do you really want to destroy all resources?
Terraform will destroy all your managed infrastructure, as shown above.
There is no undo. Only 'yes' will be accepted to confirm.

Enter a value: yes

aws_instance.collinsterraform: Destroying... [id=i-09b8516b4b5041b65]

```

How to get the output from the terraform code:

```
[root@master terracode]# vim ec2.tf
```

From where?

Output name - “PublicIP” (name can be anything)

Variable – value

Module.variablename.<what output you need>

```
[root@master terracode]# cat ec2.tf
provider "aws" {
    region           = "us-west-1"
    profile          = "mayank"
}

resource "aws_instance" "collinsterraform" {
    ami      = "ami-09625adacc474a7b4"
    instance_type = "t2.medium"
    security_groups = ["MainMayank"]
    key_name = "mayank"
}

output "PublicIP" {
    value = aws_instance.collinsterraform.public_ip
}
[root@master terracode]#
```

Every time you run, these two files are created!

After the terraform init command – you will find these two files:

```
[root@master terracode]# ls
ec2.tf  terraform.tfstate  terraform.tfstate.backup
```

Verify the content of one of that file:

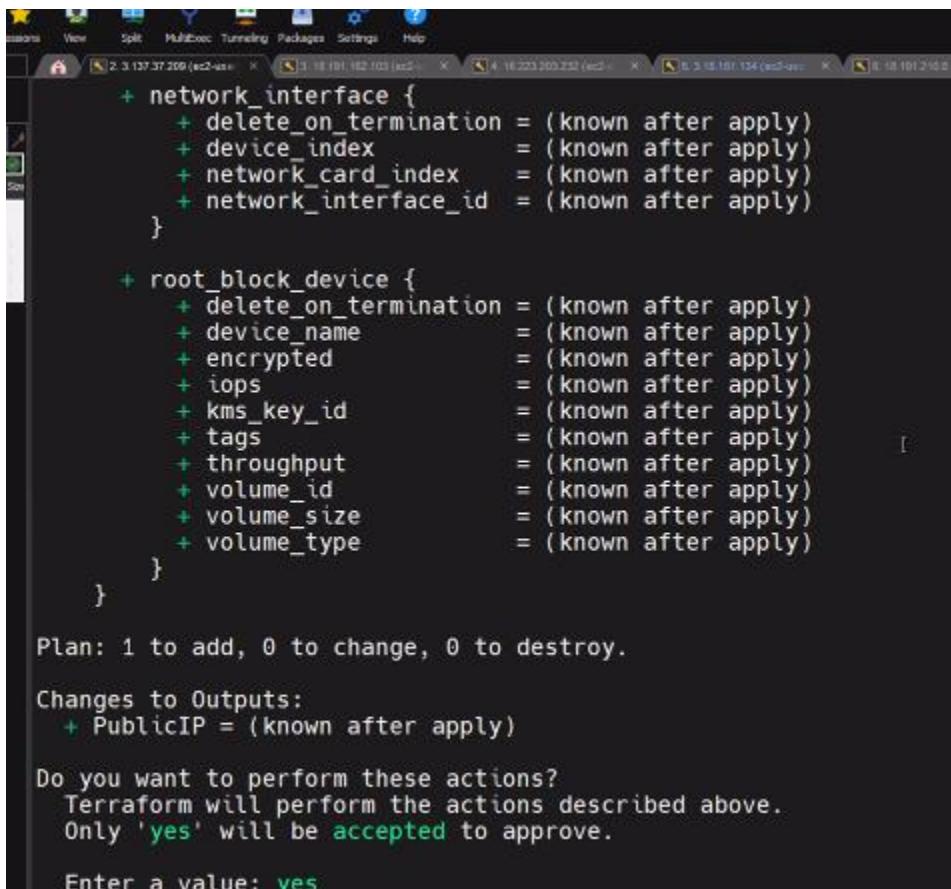
```
[root@master terracode]# cat terraform.tfstate.backup
{
  "version": 4,
  "terraform_version": "1.1.9",
  "serial": 1,
  "lineage": "8f0af8b2-3b1e-b586-5c34-b046b2343087",
  "outputs": {},
  "resources": [
    {
      "mode": "managed",
      "type": "aws_instance",
      "name": "collinsterraform",
      "provider": "provider[\"registry.terraform.io/hashicorp/aws\"]",
      "instances": [
        {
          "schema_version": 1,
          "attributes": {
            "ami": "ami-09625adacc474a7b4",
            "arn": "arn:aws:ec2:us-west-1:232221866626:instance/i-09b8516b4b5041b65",
            "associate_public_ip_address": true,
            "availability_zone": "us-west-1c",
            "capacity_reservation_specification": [
              {
                "capacity_reservation_preference": "open",
                "capacity_reservation_target": []
              }
            ]
          }
        }
      ]
    }
  ]
}
```

```

        },
        "monitoring": false,
        "network_interface": [],
        "outpost_arn": "",
        "password_data": "",
        "placement_group": "",
        "placement_partition_number": null,
        "primary_network_interface_id": "eni-050029faf8e74df",
        "private_dns": "ip-172-31-20-139.us-west-1.compute.internal",
        "private_ip": "172.31.20.139",
        "public_dns": "ec2-54-177-72-128.us-west-1.compute.amazonaws.com",
        "public_ip": "54.177.72.128",
        "root_block_device": [
            {
                "delete_on_termination": true,
                "device_name": "/dev/xvda",
                "encrypted": false,
                "iops": 100,
                "kms_key_id": "",
                "tags": {},
                "throughput": 0,
                "volume_id": "vol-038ea8cc12fe26030",
                "volume_size": 8,
                "volume_type": "gp2"
            }
        ],
        "secondary_private_ips": [],
        "security_groups": [

```

Validate:



The screenshot shows the Terraform UI interface with several tabs open at the top. The main window displays the validation output for a specific resource. The validation output includes:

- network_interface** block:
 - + delete_on_termination = (known after apply)
 - + device_index = (known after apply)
 - + network_card_index = (known after apply)
 - + network_interface_id = (known after apply)
- root_block_device** block:
 - + delete_on_termination = (known after apply)
 - + device_name = (known after apply)
 - + encrypted = (known after apply)
 - + iops = (known after apply)
 - + kms_key_id = (known after apply)
 - + tags = (known after apply)
 - + throughput = (known after apply)
 - + volume_id = (known after apply)
 - + volume_size = (known after apply)
 - + volume_type = (known after apply)

Below the validation output, the terminal shows:

```

Plan: 1 to add, 0 to change, 0 to destroy.

Changes to Outputs:
+ PublicIP = (known after apply)

Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

```

Apply:

```
Plan: 1 to add, 0 to change, 0 to destroy.

Changes to Outputs:
+ PublicIP = (known after apply)

Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

aws_instance.collinsterraform: Creating...
aws_instance.collinsterraform: Still creating... [10s elapsed]
[1]
```

Now you can see the public ip:

```
Enter a value: yes

aws_instance.collinsterraform: Creating...
aws_instance.collinsterraform: Still creating... [10s elapsed]
aws_instance.collinsterraform: Still creating... [20s elapsed]
aws_instance.collinsterraform: Creation complete after 22s [id=i-07f7304cb95100fe0]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

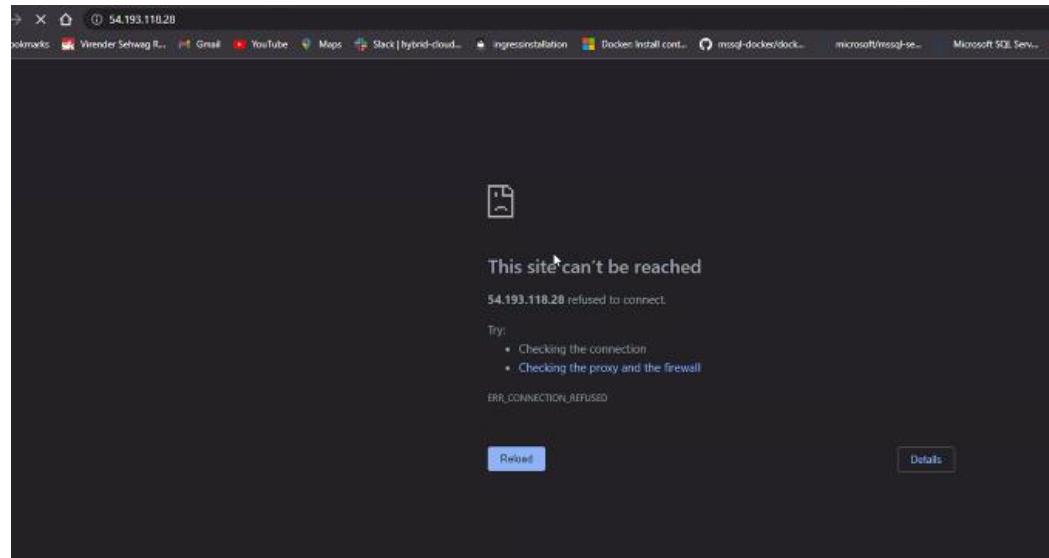
Outputs:

PublicIP = "54.193.118.28"
[root@master terracode]# [1]
```

Verify:

07f7304cb95100fe0	Running	t2.medium	us-west-1c	ec2-54-193-118-28.us...	54.193.118.28	-
0bede8bf6e67637df	Terminated	t1.micro	us-west-1b	-	-	-

Not able to do anything right now:



That should show the website:

Destroy Command:

```
> [root@master terracode]# terraform destroy
```

It will ask for the permission:

```
Changes to Outputs:
- PublicIP = "54.193.118.28" -> null

Do you really want to destroy all resources?
Terraform will destroy all your managed infrastructure, as shown above.
There is no undo. Only 'yes' will be accepted to confirm.

Enter a value: yes

aws_instance.collinsterraform: Destroying... [id=i-07f7304cb95100fe0]
```

User data in terraform:

How can we add it?

If we want to run a shell script

EOF – End of file. (we need to create a end of file inside a bash script – inside that install, configure, and start)

```
[root@node1 ~]# systemctl start kubelet
[root@node1 ~]# sudo su
[root@node1 ~]# exit
[root@node1 ~]# logout
[ec2-user@node1 ~]$ sudo su
[root@node1 ec2-user]#
```

Add user data:

```
provider "aws" {
    region      = "us-west-1"
    profile     = "mayank"
}

resource "aws_instance" "collinsterraform" {
    ami          = "ami-09625adacc474a7b4"
    instance_type = "t2.medium"
    security_groups = ["MainMayank"]
    user_data = <<EOF
#!/bin/bash
sudo su
yun install httpd -y && echo "Hi there !! , Using terraform My server is up and running" >> /var/www/html/index.html && systemctl start httpd
EOF
    key_name = "mayank"
}

output "PublicIP" {
    value = aws_instance.collinsterraform.public_ip
}
```

If we want private IP:

```

provider "aws" {
  region      = "us-west-1"
  profile     = "mayank"
}

resource "aws_instance" "collinsterraform" {
  ami          = "ami-09625adacc474a7b4"
  instance_type = "t2.medium"
  security_groups = ["MainMayank"]
  user_data = <>EOF
    #!/bin/bash
    sudo su
    yum install httpd -y && echo "Hi there !! , Using terraform My server is up and RUNNING" >> /var/www/html/index.html && systemctl start httpd
  EOF
  key_name = "mayank"
}

output "PublicIP" {
  value = aws_instance.collinsterraform.public_ip
}

output "PrivateIP" {
  value = aws_instance.collinsterraform.private_ip
}

-- INSERT --

```

Cat command to check the content of terraform file:

```

[root@master terracode]# vim ec2.tf
[root@master terracode]# cat ec2.tf
provider "aws" {
  region      = "us-west-1"
  profile     = "mayank"
}

resource "aws_instance" "collinsterraform" {
  ami          = "ami-09625adacc474a7b4"
  instance_type = "t2.medium"
  security_groups = ["MainMayank"]
  user_data = <>EOF
    #!/bin/bash
    sudo su
    yum install httpd -y && echo "Hi there !! , Using terraform My server is up and RUNNING" >> /var/www/html/index.html && systemctl start httpd
  EOF
  key_name = "mayank"
}

output "PublicIP" {
  value = aws_instance.collinsterraform.public_ip
  value = aws_instance.collinsterraform.private_ip
}

```

To know what is inside .aws folder:

```

}
[root@master terracode]# echo $HOME/.aws
/root/.aws
[root@master terracode]# 

>

/ /root/.aws
[root@master terracode]# ls $( echo $HOME/.aws )
config credentials
[root@master terracode]# 

```

```
[root@vishali terracode]# ls -la ../
total 60
dr-xr-x--- 10 root root 299 Apr 27 10:35 .
dr-xr-xr-x 22 root root 321 Apr 26 05:51 ..
drwxr-xr-x  2 root root 39 Apr 27 09:49 .aws
-rw-----  1 root root 11289 Apr 27 07:21 .bash_history
-rw-r--r--  1 root root 18 Oct 18 2017 .bash_logout
-rw-r--r--  1 root root 176 Oct 18 2017 .bash_profile
-rw-r--r--  1 root root 218 Apr 27 09:47 .bashrc
drwxr-xr-x  2 root root 39 Apr 26 10:09 code
-rw-r--r--  1 root root 30 Apr 25 07:22 collins
-rw-r--r--  1 root root 5640 Apr 26 08:59 config
-rw-r--r--  1 root root 100 Oct 18 2017 .cshrc
-rw-r--r--  1 root root 397 Apr 26 11:17 deployment.yaml
drwx----- 2 root root 25 Apr 26 11:46 .docker
drwxr-xr-x  5 root root 98 Apr 26 05:26 dockerfile
drwxr-xr-x  3 root root 33 Apr 26 09:48 .kube
drwx----- 2 root root 29 Apr 25 04:52 .ssh
-rw-r--r--  1 root root 129 Oct 18 2017 .tcshrc
drwxr-xr-x  3 root root 141 Apr 27 10:36 terracode
drwxr-xr-x  2 root root 58 Apr 27 09:52 .terraform.d
-rw-----  1 root root 9575 Apr 27 10:35 .viminfo
[root@vishali terracode]#
```

Apply command:

```
[root@master terracode]# terraform apply
```

If you don't want to ask question (for confirmation): use this --auto-approve

```
Options:
  -auto-approve      Skip interactive approval of plan before applying.
  -backup=path       Path to backup the existing state file before
                     modifying. Defaults to the "-state-out" path with
                     "terraform state backup" if no path is specified.
```

```
[root@master terracode]# terraform apply --auto-approve
```

```
[root@master terracode]# terraform validate
Success! The configuration is valid.
```

```
[root@master terracode]# terraform apply --auto-approve
```

What will it do?

We will get the ip – that will directly show the website:

```

Plan: 1 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + PrivateIP = (known after apply)
  + PublicIP  = (known after apply)

aws_instance.collinsterraform: Creating...
aws_instance.collinsterraform: Still creating... [10s elapsed]
aws_instance.collinsterraform: Still creating... [20s elapsed]
aws_instance.collinsterraform: Still creating... [30s elapsed]
aws_instance.collinsterraform: Still creating... [40s elapsed]
aws_instance.collinsterraform: Still creating... [50s elapsed]
aws_instance.collinsterraform: Still creating... [1m0s elapsed]
aws_instance.collinsterraform: Still creating... [1m10s elapsed]
aws_instance.collinsterraform: Creation complete after 1m12s [id=i-01b37ff743cb6eeb1]

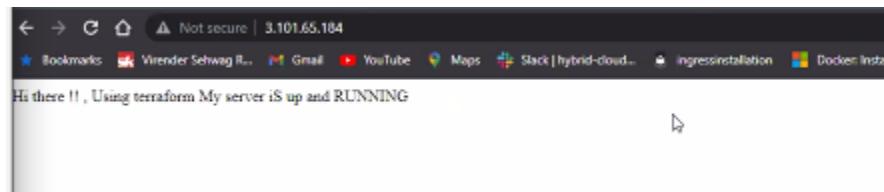
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

Outputs:

PrivateIP = "172.31.17.124"
PublicIP  = "3.101.65.184"
[root@master terracode]#

```

Website:



Without going to AWS account, our website is up and running!

Can we create a docker container and expose it somewhere else?

Can we do that? Yes

```

provider "aws" {
  region      = "us-west-1"
  profile     = "mayank"
}

resource "aws_instance" "collinsterraform" {
  ami          = "ami-09625adacc474a7b4"
  instance_type = "t2.medium"
  security_groups = ["MainMayank"]
  user_data = <><EOF
    #!/bin/bash
    sudo su
    yum install httpd -y && echo "Hi there !! , Using terraform My server is up and RUNNING" >> /var/www/html/index.html && systemctl start httpd
    yum install docker -y ; systemctl start docker ; docker run -itd -p 3434:8080 quay.io/mayank123modi/simple-webapp<
    EOF
  key_name = "mayank"
}

output "PublicIP" {
  value = aws_instance.collinsterraform.public_ip
}

output "PrivateIP" {
  value = aws_instance.collinsterraform.private_ip
}

-- INSERT --

```

Destroy it:

```

[root@master terracode]# vim ec2.tf
[root@master terracode]# terraform destroy --auto-approve

```

Terraform appy command: You will get the public id and private ip

```
        }
    + root_block_device {
        + delete_on_termination = (known after apply)
        + device_name          = (known after apply)
        + encrypted             = (known after apply)
        + iops                  = (known after apply)
        + kms_key_id            = (known after apply)
        + tags                  = (known after apply)
        + throughput             = (known after apply)
        + volume_id              = (known after apply)
        + volume_size            = (known after apply)
        + volume_type             = (known after apply)
    }
}

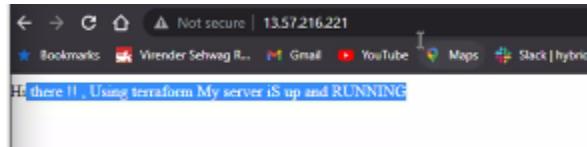
Plan: 1 to add, 0 to change, 0 to destroy.

Changes to Outputs:
+ PrivateIP = (known after apply)
+ PublicIP  = (known after apply)
aws_instance.collinsterraform: Creating...
aws_instance.collinsterraform: Still creating... [10s elapsed]
aws_instance.collinsterraform: Creation complete after 16s [id=i-0bac099c1b0b2be6]

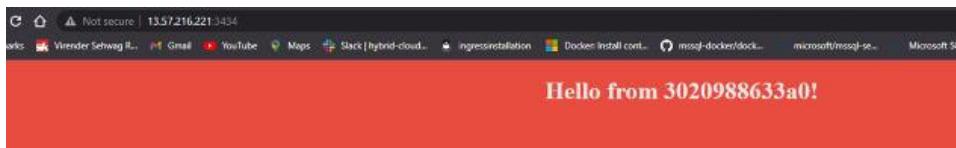
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

Outputs:
PrivateIP = "172.31.28.10"
PublicIP  = "13.57.216.221"
[root@master terracode]#
```

Default port:



Docker container port:



We can create an infrastructure automatically.

Commands are shared by the Trainer:

185 mkdir terracode

186 ls

187 cd terracode/

188 ls

189 ls -la

190 vim ec2.tf

191 aws

192 aws

```
193 cd
194 ls -la
195 aws configure --profile mayank
196 aws configure --list
197 aws configure list
198 aws configure --list-profile
199 aws configure --list-profiles
200 aws configure list
201 aws configure --list profile
202 aws configure --profile mayank
203 aws configure list --profile mayank
204 ls -la
205 cd .aws
206 ls
207 cat config
208 cat credentials
209 cd
210 cd terracode/
211 ls
212 vim ec2.tf
213 ls -la
214 cat ec2.tf
215 ls -la
216 terraform init
217 ls -la
218 cd .terraform/
219 ls
220 cd providers/registry.terraform.io/hashicorp/aws/4.11.0/linux_amd64/
221 s
222 ls
223 ccd
```

```
224 cd terr
225 cd
226 cd terracode/ls
227 cd terracode/
228 cat ec2.tf
229 viim ec2.tf
230 vim ec2.tf
231 terraform validate
232 vim ec2.tf
233 terraform validate
234 vim ec2.tf
235 terraform validate
236 terraform apply
237 terraform plan
238 terraform apply
239 cat ec2.tf
240 terraform destroy
241 vim ec2.tf
242 cat ec2.tf
243 ls
244 cat terraform.tfstate
245 cat terraform.tfstate.backup
246 ls
247 terraform apply
248 terraform destroy
249 vim ec2.tf
250 cat ec2.tf
251 echo $HOME/.aws
252 ls $( echo $HOME/.aws)
253 terraform apply --help
254 ls $( echo $HOME/.aws)
```

```
255 terraform apply --auto-approve  
256 vim ec2.tf  
257 terraform validate  
258 terraform apply --auto-approve  
259 vim ec2.tf  
260 terraform destroy --auto-approve  
261 cat ec2.tf  
262 terraform apply --auto-approve  
263 cat ec2.tf  
264 history
```

TASK:

Q.1 Create ec2 instance using terraform code

Q.2 Create ec2 instance with apache server using

Q.3 Docker containers

AcessKEY AKIATMEMAPKBJ554B3NE

Secret access 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

Authetication of AWS:

```
[root@vishali terracode]# aws configure --profile vishali  
AWS Access Key ID [*****B3NE]:  
AWS Secret Access Key [*****pV72]:  
Default region name [us-west-1]:  
Default output format [None]:  
[root@vishali terracode]#
```

Terraform init:

```
[root@vishali terracode]# terraform init  
Initializing the backend...  
  
Initializing provider plugins...  
- Reusing previous version of hashicorp/aws from the dependency lock file  
- Using previously-installed hashicorp/aws v4.11.0  
  
Terraform has been successfully initialized!  
  
You may now begin working with Terraform. Try running "terraform plan" to see  
any changes that are required for your infrastructure. All Terraform commands  
should now work.  
  
If you ever set or change modules or backend configuration for Terraform,  
rerun this command to reinitialize your working directory. If you forget, other  
commands will detect it and remind you to do so if necessary.  
[root@vishali terracode]# terraform validate  
Success! The configuration is valid.
```

Terraform file:



```
provider "aws" {
  region      = "us-west-1"
  profile     = "vishali"
}

resource "aws_instance" "collins" {
  ami          = "ami-09625adacc474a7b4"
  instance_type = "t2.micro"
  security_groups = ["vishali"]
  user_data = <EOF>
    sudo su
    yum install httpd -y && echo "Hi there!! This is vishali using terraform my server is up and running" >> /var/www/html/index.html &
<EOF>
  key_name = "vishali"
}

output "PublicIP" {
  value = aws_instance.collins.public_ip
}
```

Terraform apply command:

```
[root@vishali terracode]# terraform apply --auto-approve
Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
+ create

Terraform will perform the following actions:

# aws_instance.collins will be created
+ resource "aws_instance" "collins" {
  + ami                         = "ami-09625adacc474a7b4"
  + arn                         = (known after apply)
  + associate_public_ip_address = (known after apply)
  + availability_zone            = (known after apply)
```

IP Address is seen now:

```
2. 3.88.196.199 (root) x
+ http_put_response_hop_limit = (known after apply)
+ http_tokens                 = (known after apply)
+ instance_metadata_tags     = (known after apply)
}

+ network_interface {
  + delete_on_termination = (known after apply)
  + device_index          = (known after apply)
  + network_card_index    = (known after apply)
  + network_interface_id   = (known after apply)
}

+ root_block_device {
  + delete_on_termination = (known after apply)
  + device_name           = (known after apply)
  + encrypted              = (known after apply)
  + iops                   = (known after apply)
  + kms_key_id             = (known after apply)
  + tags                   = (known after apply)
  + throughput              = (known after apply)
  + volume_id               = (known after apply)
  + volume_size              = (known after apply)
  + volume_type              = (known after apply)
}
}

Plan: 1 to add, 0 to change, 0 to destroy.

Changes to Outputs:
+ PublicIP = (known after apply)
aws_instance.collins: Creating...
aws_instance.collins: Still creating... [10s elapsed]
aws_instance.collins: Still creating... [20s elapsed]
aws_instance.collins: Creation complete after 23s [id=i-0320a91a8bc218eec]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

Outputs:
PublicIP = "13.57.28.133"
[root@vishali ~]# ^C
```

Verify:

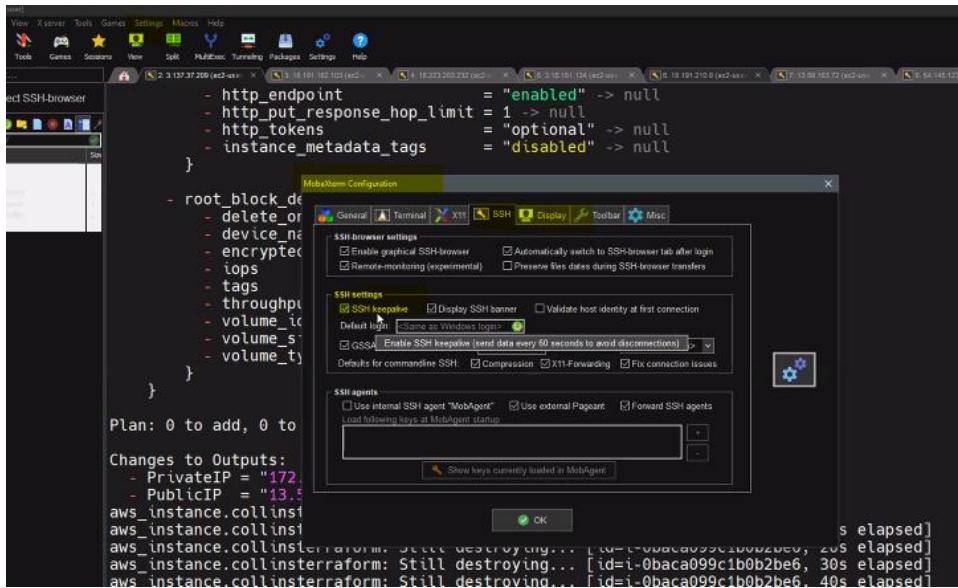
Default page:



Docker container page:



To keep the session alive:



Terraform file - Shared by the trainer:

```

provider "aws" {
    region      = "us-west-1"
    profile     = "mayank"
}

resource "aws_instance" "collinsterraform" {
    ami          = "ami-09625adacc474a7b4"
    instance_type = "t2.medium"
    security_groups = ["MainMayank"]
    user_data = <<EOF
        #!/bin/bash
        sudo su
        yum install httpd -y && echo "Hi there !! , Using terraform My server iS
        up and RUNNING" >> /var/www/html/index.html && systemctl start httpd
        yum install docker -y ; systemctl start docker ; docker run -itd -p
        3434:8080 quay.io/mayank123modi/simple-webapp
    EOF
    key_name = "mayank"
}

output "PublicIP" {

```

```

    value = aws_instance.collinsterraform.public_ip
}

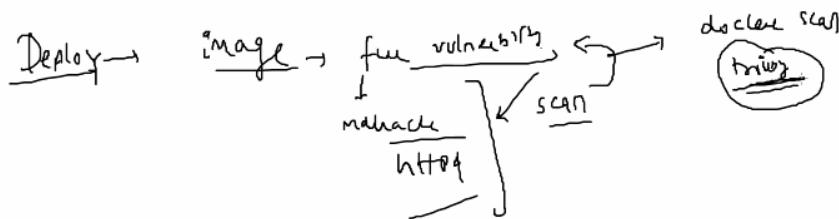
output "PrivateIP" {
    value = aws_instance.collinsterraform.private_ip
}

```

Image scanning:

Purpose: For checking vulnerability

Which tool we are going to use? Trivy



Install Trivy tool:

The first one won't work. The second command will work.

The screenshot shows a browser window with the URL <https://aquasecurity.github.io/trivy/v0.18.3/installation/>. The page title is "Installation". On the left, there's a sidebar with navigation links like Overview, Installation, Quick Start, Scanning, Modes, Examples, Integrations, Private Docker Registries, Vulnerability Detection, Usage, Plugins, Air-Gapped Environment, Comparison with Other Scanners, Further Reading, FAQ, Maintainer, and Credits. The main content area has a section titled "RHEL/CentOS" with a sub-section "Add repository setting to /etc/yum.repos.d.". It contains a code snippet:

```
$ sudo vim /etc/yum.repos.d/trivy.repo  
[trivy]  
name=Trivy repository  
baseurl=https://aquasecurity.github.io/trivy-repo/rpm/releases/$releasever/$basearch  
gpgcheck=0  
enabled=1  
$ sudo yum -y update  
$ sudo yum -y install trivy
```

Below this, there's an "or" option with another code snippet:

```
rpm -ivh https://github.com/aquasecurity/trivy/releases/download/v0.18.3/trivy_0.11
```

On the right side, there's a "Table of contents" sidebar with links to RHEL/CentOS, Debian/Ubuntu, Arch Linux, Homebrew, Nix/NixOS, Install Script, Binary, From source, Docker, Docker Hub, GitHub CI, Amazon EI, Helm, Installing f, Chart Repo, and Installing t.

Edit the content at this location:

```
> ^C
[root@master terracode]# sudo vim /etc/yum.repos.d/trivy.repo
[root@master terracode]# yum
```

Remove this as it doesn't work

```
[root@master terracode]# rm /etc/yum.repos.d/  
ec2.tf .terraform.lock.hcl terraform.tfstate  
.terraform/ terraform.tfstate.backup  
[root@master terracode]# rm /etc/yum.repos.d/
```

Trivy install: (This works)

```
[root@master terracode]# rm -rf /etc/yum.repos.d/trivy.repo
[root@master terracode]# rpm -vh https://github.com/aquasecurity/trivy/releases/download/v0.18.3/trivy_0.18.3_Linux-64bit.rpm
Retrieving https://github.com/aquasecurity/trivy/releases/download/v0.18.3/trivy_0.18.3_Linux-64bit.rpm
Preparing.. #####
Updating / installing... 1:trivy-0.18.3-1 #####
[root@master terracode]#
```

Now, try trivy command:

```
[root@master terracode]# trivy
bash: trivy: command not found
[root@master terracode]#
```

You can find the trivy folder here:

```
[root@master terracode]# cd /usr/local/bin/
[root@master bin]# ls
trivy
[root@master bin]#
```

We need to move the trivy folder into usr/bin

```
[root@master bin]# cp trivy /usr/bin/
[root@master bin]# trivy
NAME:
    trivy - A simple and comprehensive vulnerability scanner for containers

USAGE:
    trivy command [command options] target

COMMANDS:
    image, i      scan an image
    filesystem, fs  scan local filesystem
    repository, repo  scan remote repository
    client, c     client mode
    server, s     server mode
    plugin, p     manage plugins
    help, h       Shows a list of commands or help for one command
```

Let's check whether image is secure

Docker pull the image:

```
[root@master bin]# docker pull mdhack/myserver
Using default tag: latest
latest: Pulling from mdhack/myserver
74f0853ba93b: Pull complete
7aa70b934c32: Pull complete
2d68deff9aaf: Pull complete
31ea4127808d: Pull complete
Digest: sha256:70adde817b3b8c8029ed3eb6d8542cce47a90087517ec05813149f0c7a292d1
Status: Downloaded newer image for mdhack/myserver:latest
docker.io/mdhack/myserver:latest
```

Docker pull the image:

```
[root@master bin]# docker pull quay.io/mayank123modi/simple-webapp
Using default tag: latest
latest: Pulling from mayank123modi/simple-webapp
7d00b3176f146: Pull complete
656769b42b54: Pull complete
46877b8bf32e2: Pull complete
f5e790313325: Pull complete
03b307d006b6: Pull complete
a7b78dcf172f: Pull complete
cc19739c4acc: Pull complete
Digest: sha256:9a22693032c8fc8d93a4bb30b8e8d36804736c1a06d3419fc83e4a7ecb85c5f9
Status: Downloaded newer image for quay.io/mayank123modi/simple-webapp:latest
quay.io/mayank123modi/simple-webapp:latest
[root@master bin]#
```

Trivy image:

```
[root@master ~]# trivy image mdhack/myserver
2022-04-27T11:44:13.335Z    INFO  Need to update DB
2022-04-27T11:44:13.335Z    INFO  Downloading DB...
23.07 MiB / 27.67 MiB [=====] 83.36% 19.71 MiB p/s ETA 0s
```

	CVE-2022-0351			
Access of memory location				
start of buffer				
aquasec.com/nvd/cve-2022-0351				
	CVE-2022-1154			
Leak after free in utf_ptr2char				
aquasec.com/nvd/cve-2022-1154				
yum-plugin-fastestmirror	CVE-2018-10897	HIGH	1.1.31-40.el7	1.1.31-46.el7
ls: reposync: improper validation may lead				
directory traversal				
aquasec.com/nvd/cve-2018-10897				
yum-plugin-ovl				

To count:

```
[root@master ~]# trivy image mdhack/myserver | head -10
2022-04-27T11:44:56.654Z    INFO  Detected OS: centos
2022-04-27T11:44:56.654Z    INFO  Detecting RHEL/CentOS vulnerabilities...
2022-04-27T11:44:56.671Z    INFO  Number of PL dependency files: 0

mdhack/myserver (centos 7.3.1611)
=====
Total: 176 (UNKNOWN: 0, LOW: 693, MEDIUM: 987, HIGH: 73, CRITICAL: 9)

+-----+-----+-----+-----+
| LIBRARY | VULNERABILITY ID | SEVERITY | INSTALLED VERSION | FIXED VERSION |
| TITLE   |                   |          |                  |                 |
+-----+-----+-----+-----+
[root@master ~]#
```

Will you use this in production environment? NO

It shows Low risk, medium risk, high risk.....

Count the vulnerability:

```

[trivy]# trivy image quay.io/mayank123modi/simple-webapp | head -10
2022-04-27T11:45:39.804Z    INFO  Detected OS: alpine
2022-04-27T11:45:39.804Z    INFO  Detecting Alpine vulnerabilities...
2022-04-27T11:45:39.805Z    INFO  Number of PL dependency files: 0
2022-04-27T11:45:39.805Z    WARN  This OS version is no longer supported by the distribution: alpine 3.8.1
2022-04-27T11:45:39.805Z    WARN  The vulnerability detection may be insufficient because security updates are
not provided
quay.io/mayank123modi/simple-webapp (alpine 3.8.1)
=====
Total: 14 (UNKNOWN: 0, LOW: 0, MEDIUM: 5, HIGH: 5, CRITICAL: 4)

[root@master ~]#

```

But still, will you use this image in production? NO

Then how to find out what images are secure.

Get nodes:

```

[root@master ~]# kubectl get nodes
NAME      STATUS   ROLES          AGE     VERSION
master    Ready    control-plane,master   28h    v1.23.6
node1     Ready    <none>        28h    v1.23.6
node2     Ready    <none>        28h    v1.23.6
node3     Ready    <none>        28h    v1.23.6
node4     Ready    <none>        28h    v1.23.6
node5     Ready    <none>        28h    v1.23.6
[root@master ~]#

```

Docker images:

```

[root@master ~]# docker images
REPOSITORY          TAG      IMAGE ID      CREATED       SIZE
mayanktest          latest   373987aa2ad7  8 hours ago  393MB
mdhack/collinsimage latest   373987aa2ad7  8 hours ago  393MB
calico/cni          v3.22.2  be7dfc21ba2e  12 days ago  236MB
calico/pod2daemon-flexvol  v3.22.2  d6660bf471e1  12 days ago  19.7MB
calico/node          v3.22.2  fd1608dbbc19  12 days ago  198MB
k8s.gcr.io/kube-apiserver  v1.23.6  8fa62c12256d  13 days ago  135MB
k8s.gcr.io/kube-proxy        v1.23.6  4c0375452406  13 days ago  112MB
k8s.gcr.io/kube-scheduler  v1.23.6  595f327f224a  13 days ago  53.5MB
k8s.gcr.io/kube-controller-manager  v1.23.6  df7b72818ad2  13 days ago  125MB
k8s.gcr.io/etcd         3.5.1-0   25f8c7f3da61  5 months ago  293MB
k8s.gcr.io/coredns      v1.8.6   a4ca41631cc7  6 months ago  46.8MB
k8s.gcr.io/pause         3.6     6270bb605e12  8 months ago  683kB
oraclelinux           8.3     816d99f0bbe8  12 months ago  224MB
mdhack/myserver        latest   e8ce7504414a  20 months ago  350MB
quay.io/mayank123modi/simple-webapp  latest   c6e3cd9aae36  3 years ago  84.8MB
[root@master ~]#

```

These images are responsible to run your containers in the cluster.

Is the vulnerability okay here? NO

There should be no vulnerability.

```
[root@master ~]# docker history k8s.gcr.io/kube-apiserver:v1.23.6
IMAGE           CREATED      CREATED BY
8fa62c12256d  13 days ago   COPY /kube-apiserver /usr/local/bin/kube-api...  131MB  buildkit.dockerfile.v0
<missing>       13 days ago   ARG BINARY
<missing>       2 weeks ago    ENTRYPOINT [ "/go-runner"]
<missing>       2 weeks ago    COPY ./workspace/go-runner . # buildkit
<missing>       2 weeks ago    WORKDIR /
<missing>       2 weeks ago    LABEL description=go based runner for distro...
<missing>       2 weeks ago    LABEL maintainers=Kubernetes Authors
<missing>       52 years ago   bazel build ...
[root@master ~]#
```

Check for one of this image:

```
[root@master ~]# trivy image k8s.gcr.io/kube-apiserver:v1.23.6
^C
```

Check for the count:

```
[root@master ~]# trivy image k8s.gcr.io/kube-apiserver:v1.23.6 | head -10
2022-04-27T11:49:22.313Z      INFO   Detected OS: debian
2022-04-27T11:49:22.313Z      INFO   Detecting Debian vulnerabilities...
2022-04-27T11:49:22.313Z      INFO   Number of PL dependency files: 0

k8s.gcr.io/kube-apiserver:v1.23.6 (debian 11.3)
=====
Total: 0 (UNKNOWN: 0, LOW: 0, MEDIUM: 0, HIGH: 0, CRITICAL: 0)
```

We can use these kind of images in production? Yes

Check for another image:

Vulnerability is okay for cmd. You can see 2.

```
[root@master ~]# trivy image k8s.gcr.io/etcd:3.5.1-0
bash: trivy: command not found
[root@master ~]# trivy image k8s.gcr.io/etcd:3.5.1-0
[root@master ~]# trivy image k8s.gcr.io/etcd:3.5.1-0
2022-04-27T11:49:59.780Z      INFO   Detected OS: debian
2022-04-27T11:49:59.780Z      INFO   Detecting Debian vulnerabilities...
2022-04-27T11:49:59.780Z      INFO   Number of PL dependency files: 4
2022-04-27T11:49:59.780Z      INFO   Detecting gobinary vulnerabilities...

k8s.gcr.io/etcd:3.5.1-0 (debian 11.1)
=====
Total: 0 (UNKNOWN: 0, LOW: 0, MEDIUM: 0, HIGH: 0, CRITICAL: 0)

usr/local/bin/etcd
=====
Total: 2 (UNKNOWN: 1, LOW: 0, MEDIUM: 0, HIGH: 1, CRITICAL: 0)
```

Check for the count of vulnerability:

```
[root@master ~]# trivy image k8s.gcr.io/etcd:3.5.1-0 | head -10
2022-04-27T11:50:04.092Z      INFO   Detected OS: debian
2022-04-27T11:50:04.092Z      INFO   Detecting Debian vulnerabilities...
2022-04-27T11:50:04.092Z      INFO   Number of PL dependency files: 4
2022-04-27T11:50:04.092Z      INFO   Detecting gobinary vulnerabilities...

k8s.gcr.io/etcd:3.5.1-0 (debian 11.1)
=====
Total: 0 (UNKNOWN: 0, LOW: 0, MEDIUM: 0, HIGH: 0, CRITICAL: 0)

[root@master ~]#
```

How to fix the vulnerability? We will see later.

Commands shared by the Trainer:

```
277 rpm -ivh  
https://github.com/aquasecurity/trivy/releases/download/v0.18.3/trivy\_0.18.3\_Linux-64bit.rpm  
278 trivy  
279 cd /usr/local/bin/  
280 ls  
281 cp trivy /usr/bin/  
282 trivy  
283 docker images  
284 docker pull mdhack/myserver  
285 docker pull quay.io/mayank123modi/simple-webapp  
286 cd  
287 docker images  
288 trivy  
289 trivy image mdhack/myserver  
290 trivy image mdhack/myserver | head -190  
291 trivy image mdhack/myserver | head -10  
292 trivy image quay.io/mayank123modi/simple-webapp | head -10  
293 docker images  
294 kubectl get nodes  
295 <<X  
kube- scheduler  
api server  
296 docker images  
297 trivy image k8s.gcr.io/kube-apiserver | head -10  
298 trivy image k8s.gcr.io/kube-apiserver  
299 docker image  
300 docker images  
301 trivy image k8s.gcr.io/kube-apiserver  
302 systemctl start docker
```

```
303 trivy image k8s.gcr.io/kube-apiserver
304 trivy image k8s.gcr.io/kube-apiserver:latest
305 docker history k8s.gcr.io/kube-apiserver
306 docker images
307 docker history k8s.gcr.io/kube-apiserver:v1.23.6
308 trivy image k8s.gcr.io/kube-apiserver:v1.23.6
309 trivy image k8s.gcr.io/kube-apiserver:v1.23.6 | head -10
310 docker images
311 trivy image k8s.gcr.io/etcd:3.5.1-0
312 trivy image k8s.gcr.io/etcd:3.5.1-0
313 trivy image k8s.gcr.io/etcd:3.5.1-0 | head -10
314 <<X
<<X
315 history
```

TASK:

Try the above command:

```
[root@vishali ~]# cd /usr/local/bin/
[root@vishali bin]# cp trivy /usr/bin/
cp: cannot stat 'trivy': No such file or directory
[root@vishali bin]# ls
trivy
[root@vishali bin]# cp trivy /usr/bin/
```

Vulnerability for myserver image:

```
[root@vishali bin]# trivy image mdhack/myserver | head -10
2022-04-27T12:08:23.839Z      INFO    Detected OS: centos
2022-04-27T12:08:23.839Z      INFO    Detecting RHEL/CentOS vulnerabilities...
2022-04-27T12:08:23.953Z      INFO    Number of PL dependency files: 0

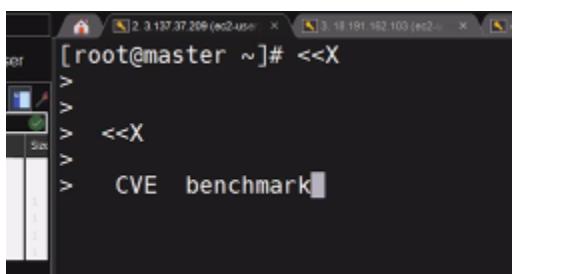
mdhack/myserver (centos 7.3.1611)
=====
Total: 1762 (UNKNOWN: 0, LOW: 693, MEDIUM: 987, HIGH: 73, CRITICAL: 9)

+-----+-----+-----+
|     LIBRARY      | VULNERABILITY ID | SEVERITY | INSTALLED VERSION
|     FIXED VERSION |                 |          | TITLE
|                 |
[root@vishali bin]#
```

Vulnerability for webapp image:

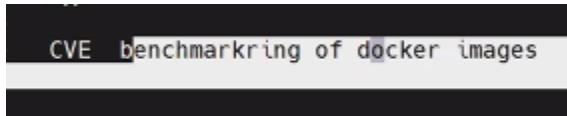
```
[root@vishali bin]# docker pull quay.io/mayank123modi/simple-webapp
Using default tag: latest
latest: Pulling from mayank123modi/simple-webapp
7d0b3176f146: Pull complete
656769b42b54: Pull complete
46877b8f32e2: Pull complete
f5e790313325: Pull complete
03b307d006b6: Pull complete
a7b78dcf172f: Pull complete
cc19739c4acc: Pull complete
Digest: sha256:9a22693032c8fc8d93a4bb30b8e8d36804736c1a06d3419fc83e4a7ecb85c5f9
Status: Downloaded newer image for quay.io/mayank123modi/simple-webapp:latest
quay.io/mayank123modi/simple-webapp:latest
[root@vishali bin]# trivy image quay.io/mayank123modi/simple-webapp | head -10
2022-04-27T12:06:03.875Z      INFO    Need to update DB
2022-04-27T12:06:03.875Z      INFO    Downloading DB...
27.67 MiB / 27.67 MiB [=====] 100.00% 20.50 MiB p/s 1s
2022-04-27T12:06:10.435Z      INFO    Detected OS: alpine
2022-04-27T12:06:10.436Z      INFO    Detecting Alpine vulnerabilities...
2022-04-27T12:06:10.436Z      INFO    Number of PL dependency files: 0
2022-04-27T12:06:10.437Z      WARN    This OS version is no longer supported b
y the distribution: alpine 3.8.1
2022-04-27T12:06:10.437Z      WARN    The vulnerability detection may be insuf
ficient because security updates are not provided
=====
quay.io/mayank123modi/simple-webapp (alpine 3.8.1)
=====
[root@vishali bin]#
```

Home work: Learn about this:



The screenshot shows a terminal window with a black background and white text. The command 'CVE benchmark' is being typed in. The terminal has a title bar with the text 'root@master ~#'. There are two tabs open in the background: one titled '2.3.137.37.209 (ec2-user)' and another titled '3.18.191.162.103 (ec2-...)'.

```
[root@master ~]# <<X
>
>
> <<X
>
> CVE benchmark
```

The screenshot shows the output of the 'CVE benchmark' command. It consists of a single line of text: 'CVE benchmarking of docker images'.

```
CVE benchmarking of docker images
```