SCHOOL OF COMPUTER SCIENCE

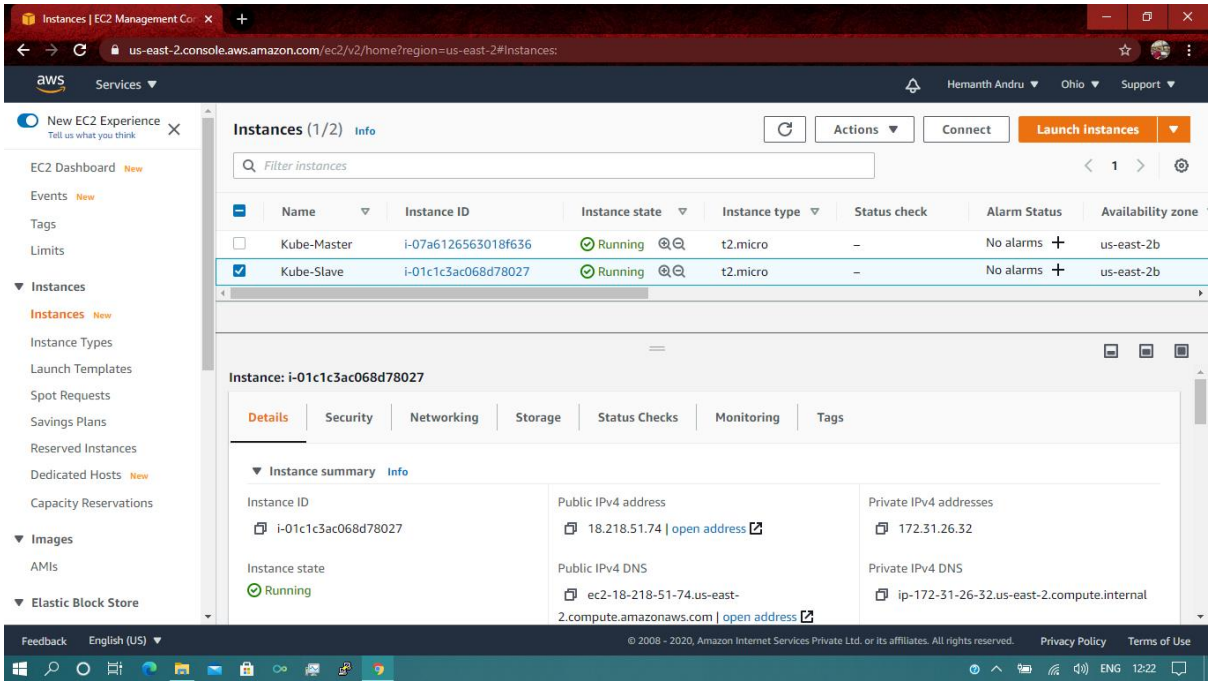
LOVELY PROFESSIONAL UNIVERSITY

PHAGWARA, PUNJAB(INDIA)-144402

KUBERNETES

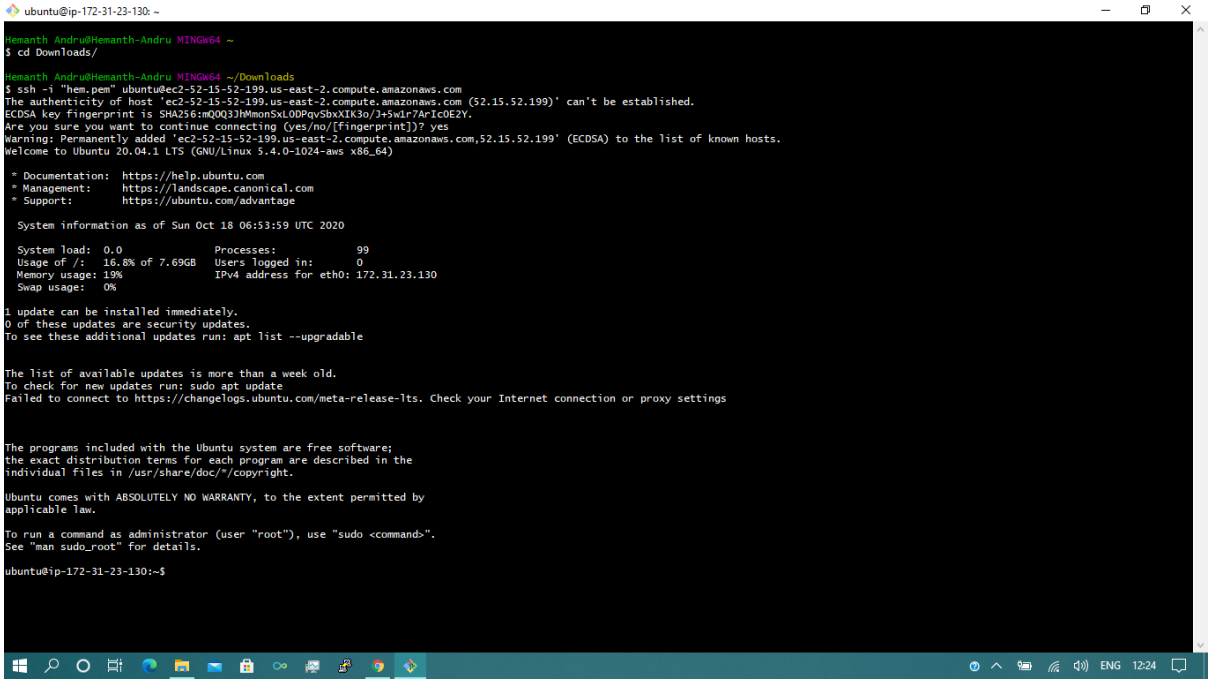
Practical :- 1 Kubernetes installation.

Step 1:- Create a two aws ec2 instance in aws using ubuntu operating system. And then name them as Kube-Master and kube-Slave. As shown in the figure below.

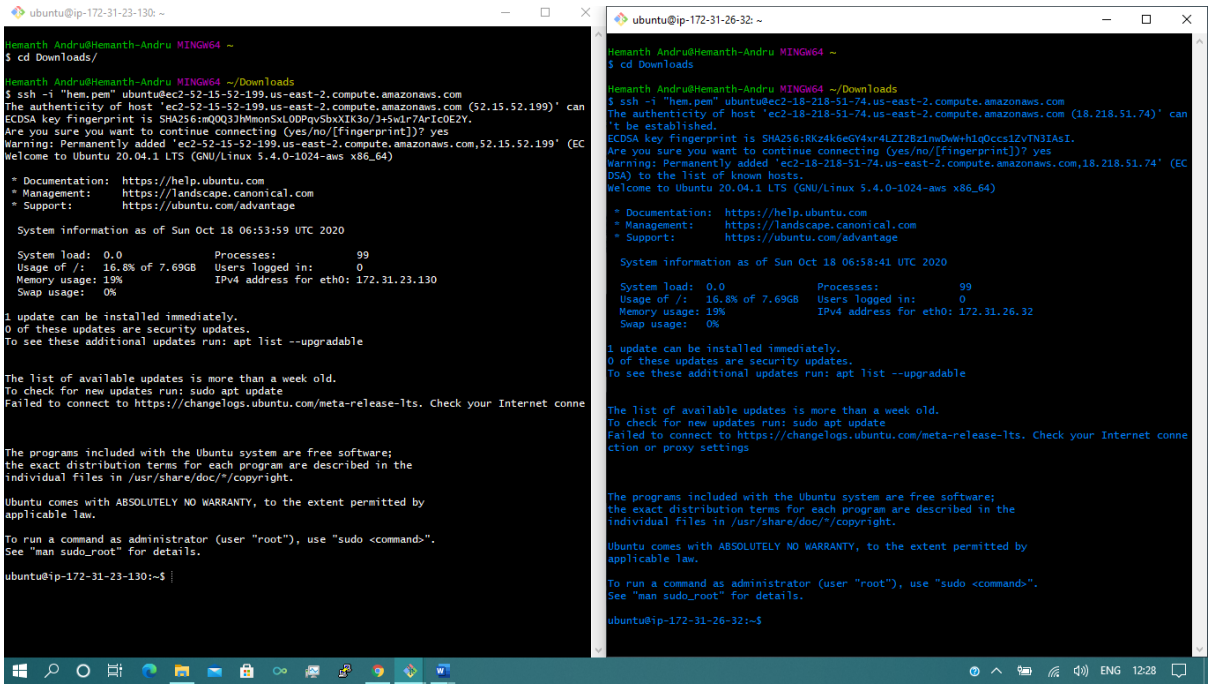


Note :- Here kube-Master is my master and kube-slave is my slave.

Step 2 :- Now start two ec2 instances and open gitbash .And then connect it with ssh connection.

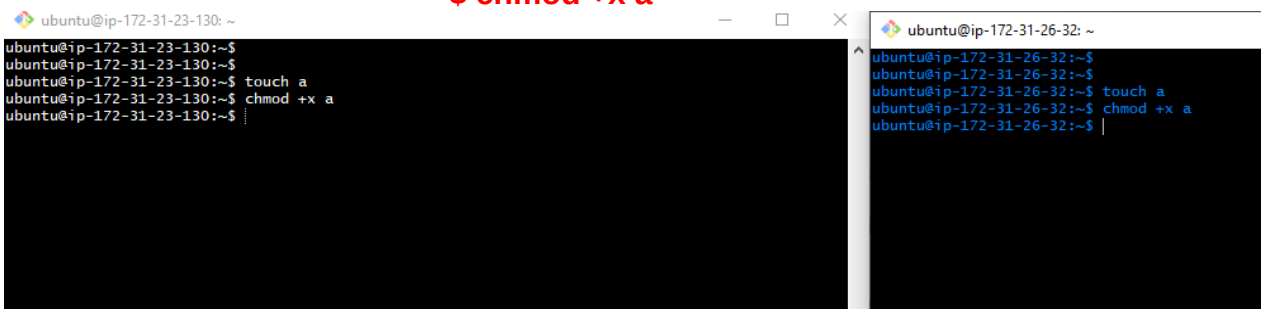


Note :- For master my gitbash in white colour and for slave my gitbash is in blue colour.



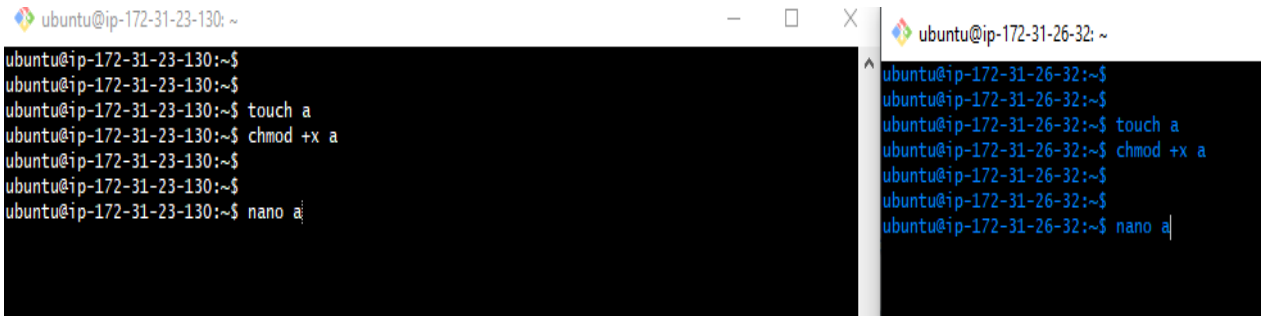
Step 3 :- Now create file with any name and give execute permission to it in both master and slave.

$ sudo su
$ touch a
$ chmod +x a



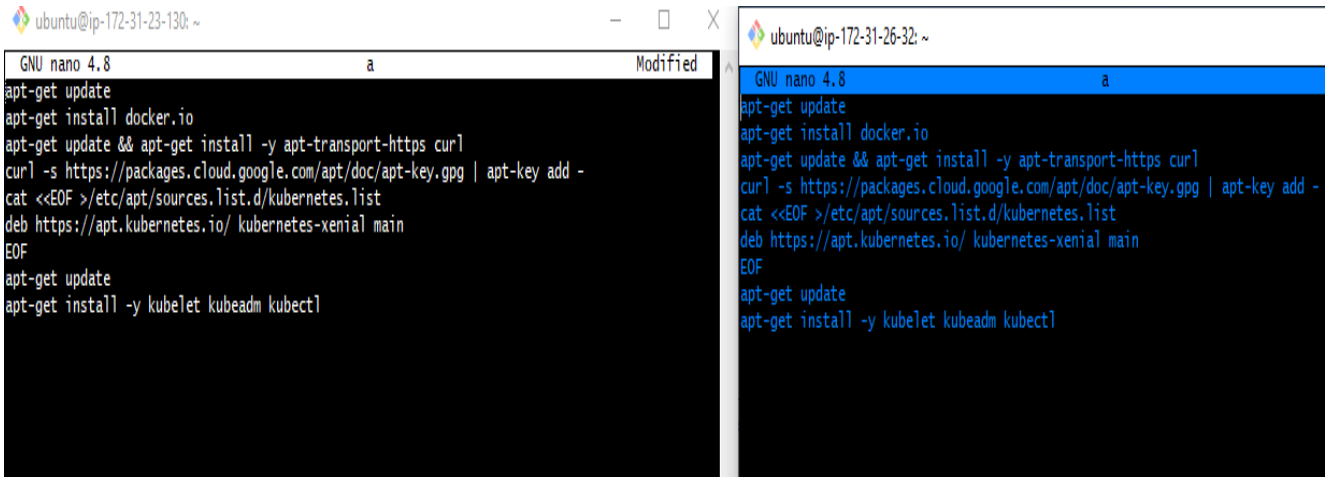
Now open the file using the command nano **.**

$ nano a



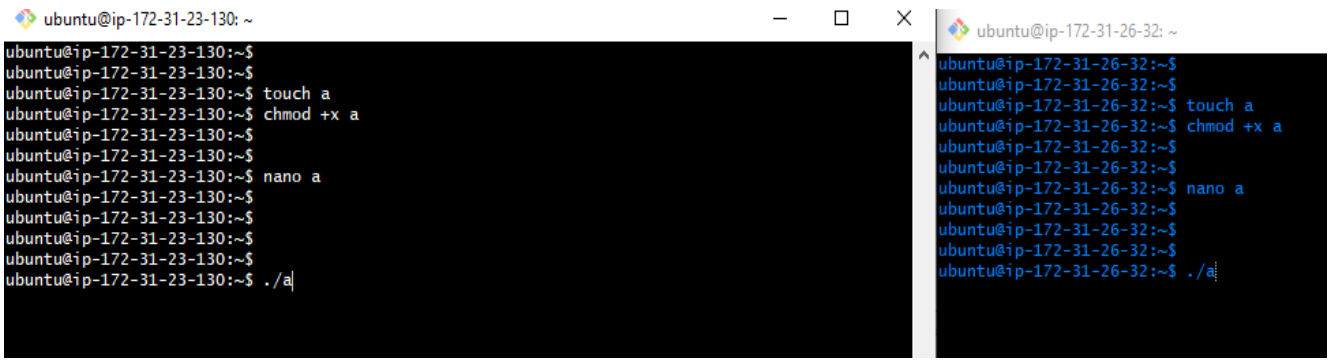
Step 4 :- After opening the files now copy the commands and paste in that file and save it and close and then run the file.

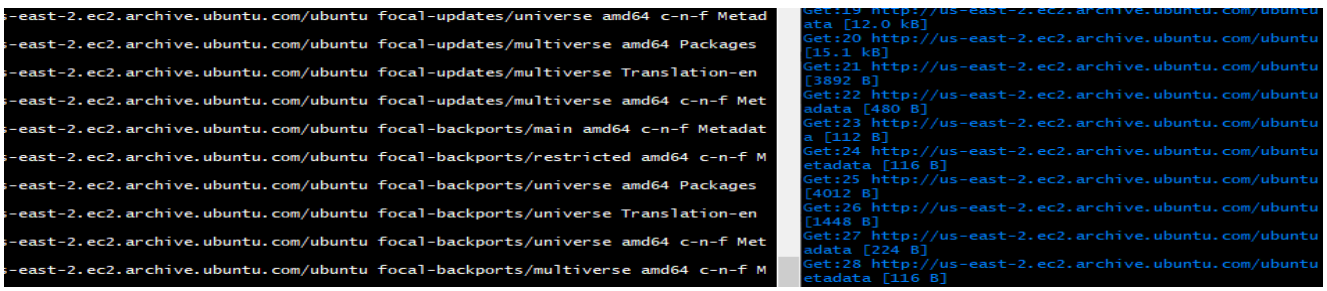
$ apt-get update
$ apt-get install docker.io
$ apt-get update && apt-get install -y apt-transport-https curl
$ curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | apt-key add -
$ cat <<EOF >/etc/apt/sources.list.d/kubernetes.list
deb https://apt.kubernetes.io/ kubernetes-xenial main
EOF
$ apt-get update
$ apt-get install -y kubelet kubeadm kubectl

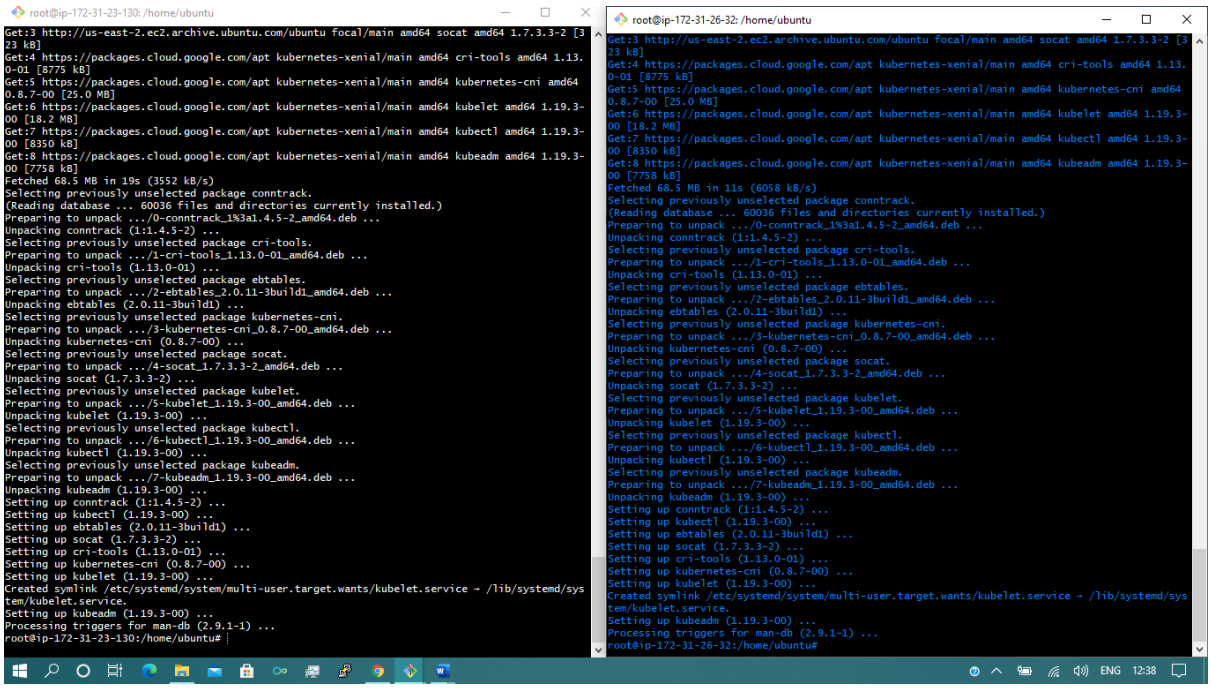


Now save the file using crtl+s and exit using crtl+x. And then run it with following command.

$ ./a



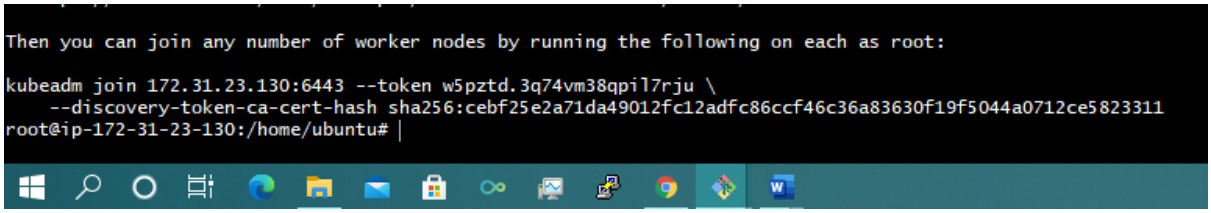




Step 5 :-Now we have to run kubeadm init command in master to generate a token to connect with slave.Here my master private ip is 172.31.23.130 .

$ kubeadm init --apiserver-advertise-address=<private ip-address-of-master> --pod-network-cidr=192.168.0.0/16 --ignore-preflight-errors=all

It will generate a token. copy it and paste in slave to connect the slave in cluster.



In slave ,

root@ip-172-31-26-32: /home/ubuntu

```
root@ip-172-31-26-32:/home/ubuntu#
root@ip-172-31-26-32:/home/ubuntu#
root@ip-172-31-26-32:/home/ubuntu#
root@ip-172-31-26-32:/home/ubuntu# kubeadm join 172.31.23.130:6443 --token w5pztd.3q74vm38qpil7rju \
        --discovery-token-ca-cert-hash sha256:cebf25e2a71da49012fc12adfc86ccf46c36a83630f19f5044a0712ce5823311
[preflight] Running pre-flight checks
        [WARNING Service-Docker]: docker service is not enabled, please run 'systemctl enable docker.service'
        [WARNING IsDockerSystemdCheck]: detected "cgroupfs" as the Docker cgroup driver. The recommended driver i
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -oyaml'
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap...

This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.

root@ip-172-31-26-32:/home/ubuntu#
```

Step 6 :- Now we have to create a directory for kuberenets cluster to start. In master.

$ mkdir -p $HOME/.kube
$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
$ sudo chown $(id -u):$(id -g) $HOME/.kube/config

ubuntu@ip-172-31-23-130: ~

```
root@ip-172-31-23-130:/home/ubuntu#
root@ip-172-31-23-130:/home/ubuntu#
root@ip-172-31-23-130:/home/ubuntu# exit
exit
ubuntu@ip-172-31-23-130:~$ mkdir -p $HOME/.kube
ubuntu@ip-172-31-23-130:~$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
ubuntu@ip-172-31-23-130:~$ sudo chown $(id -u):$(id -g) $HOME/.kube/config
ubuntu@ip-172-31-23-130:~$
```

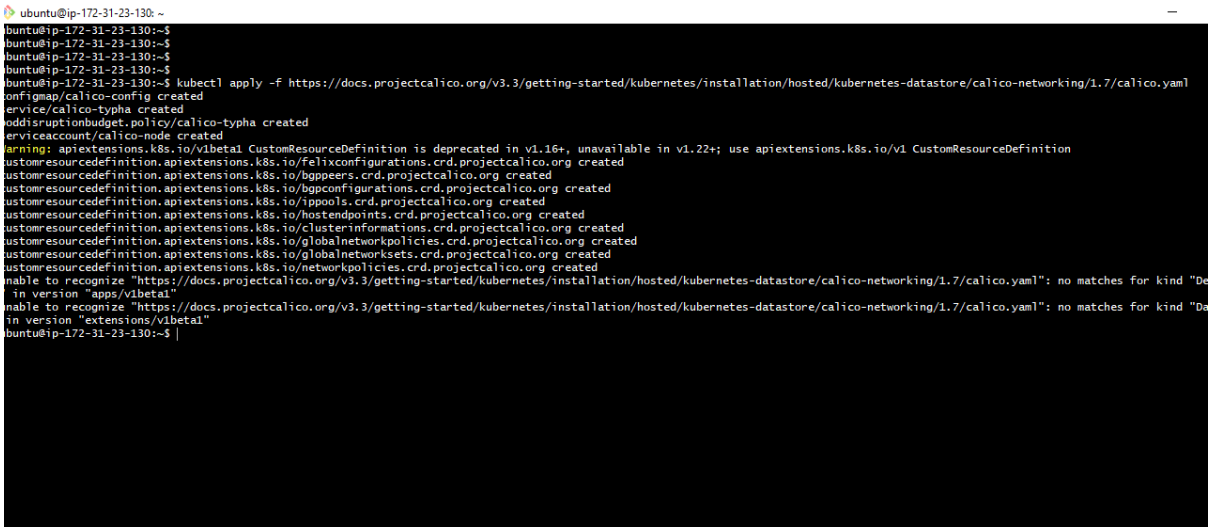
Step 7 :- Now we can use the cluster. But if we write kubectl get nodes on master it shows the nodes are not ready in state. Because there is no network plugin in cluster.To install network plugin the following commands are

$ kubectl apply -f https://docs.projectcalico.org/v3.3/getting-started/kubernetes/installation/hosted/rbac-kdd.yaml

ubuntu@ip-172-31-23-130: ~

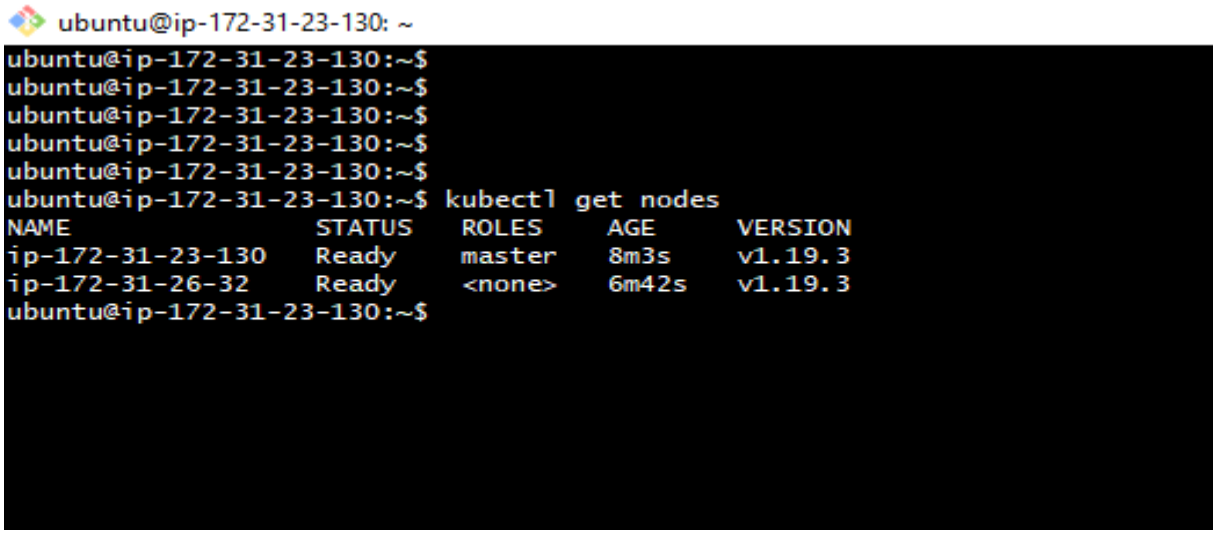
```
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$ kubectl apply -f https://docs.projectcalico.org/v3.3/getting-started/kubernetes/installation/hosted/rbac-kdd.yaml
Warning: rbac.authorization.k8s.io/v1beta1 ClusterRole is deprecated in v1.17+, unavailable in v1.22+; use rbac.authorization.k8s.io/v1 ClusterRole
clusterrole.rbac.authorization.k8s.io/calico-node created
Warning: rbac.authorization.k8s.io/v1beta1 ClusterRoleBinding is deprecated in v1.17+, unavailable in v1.22+; use rbac.authorization.k8s.io/v1 ClusterRoleBinding
clusterrolebinding.rbac.authorization.k8s.io/calico-node created
ubuntu@ip-172-31-23-130:~$
```

$ kubectl apply -f https://docs.projectcalico.org/v3.3/getting-started/kubernetes/installation/hosted/kubernetes-datastore/calico-networking/1.7/calico.yaml



Step 8 :- The previous command will take some time to take effect. After 4-5 mins, try the following command, if both the nodes are in the ready state, Installation is successful!

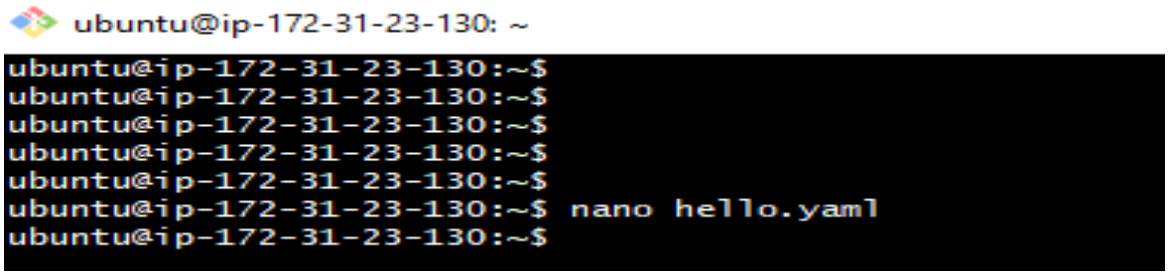
$ kubectl get nodes



Practical :- 2 Kubernetes Depolyment.

Step 1 :- Now for creating deployment we have to create a yaml file.The command for creating yaml file.

$ nano hello.yaml



Step 2 :- Now the yaml file is created and it will be opened. In the file you have paste the following content.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: hello-depo
  labels:
    app: nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.14.2
        ports:
        - containerPort: 80
```

```
ubuntu@ip-172-31-23-130: ~
  GNU nano 4.8
apiVersion: apps/v1
kind: Deployment
metadata:
  name: hello-depo
  labels:
    app: nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.14.2
        ports:
        - containerPort: 80
```

Step 3 :- For create deployment ,we have to run kubectl create of yaml file.

$ kubectl create -f hello.yaml

```
ubuntu@ip-172-31-23-130: ~
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$ kubectl create -f hello.yaml
deployment.apps/hello-depo created
ubuntu@ip-172-31-23-130:~$
```

Deployment is created .

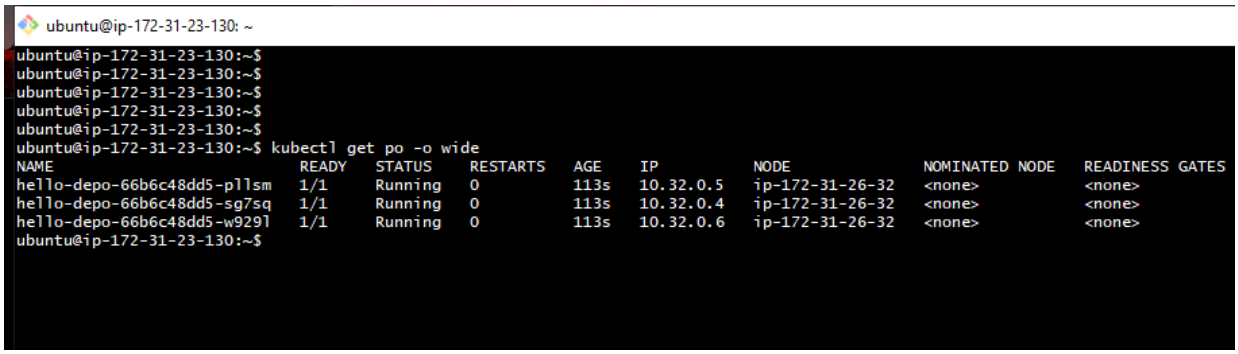
Step 4 :- we can check the replicas where created or not and they are running in running state or not.

$ kubectl get po

```
ubuntu@ip-172-31-23-130: ~
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$ kubectl get po
NAME                          READY   STATUS    RESTARTS   AGE
hello-depo-66b6c48dd5-pllsm   1/1     Running   0          50s
hello-depo-66b6c48dd5-sg7sq   1/1     Running   0          50s
hello-depo-66b6c48dd5-w929l   1/1     Running   0          50s
ubuntu@ip-172-31-23-130:~$
```

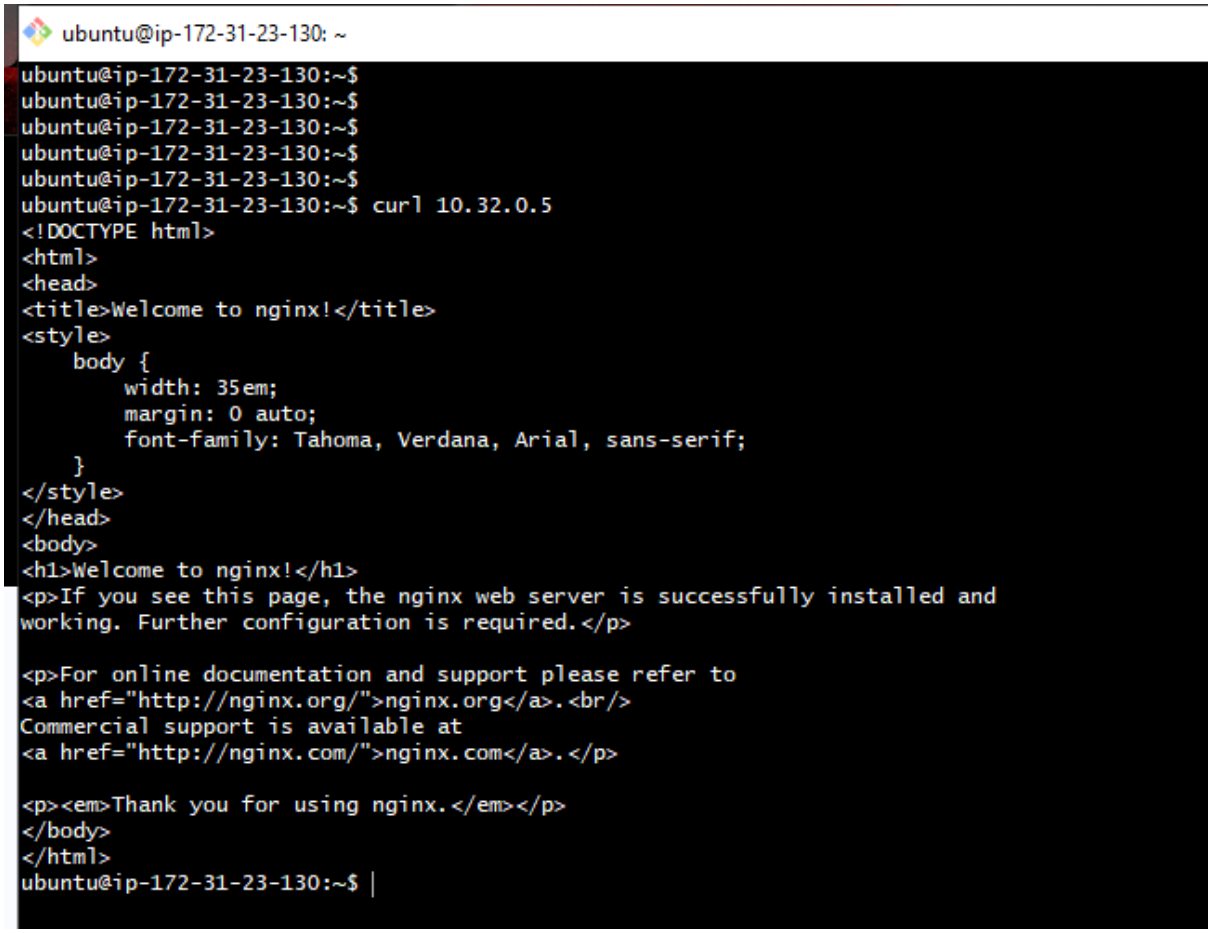
Step 5 :- we get internal cluster ip using the following command**.**

$ kubectl get po -o wide



Step 6 :- Now take the ip address and use curl command to check our depolyments are working properly.

$ curl 10.32.0.5



It seems it is working properly.

Practical :- 3 Kubernetes Services.

Why we need service in Kubernetes ? :

We need services in Kubernetes is to access our application from outside of cluster. ip address of the pods are used inside of the cluster and it cannot work outside.For getting accessible outside we have use external load balancer which is used to listen on particular port and forwards it to access.

Step 1 :- Now for create service we have use kubectl command and we have mention tcp port. It will create a new port in which we access our application from outside of our cluster.

ubuntu@ip-172-31-23-130: ~

```
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$ kubectl create service nodeport nginx --tcp=80:80
service/nginx created
ubuntu@ip-172-31-23-130:~$
```

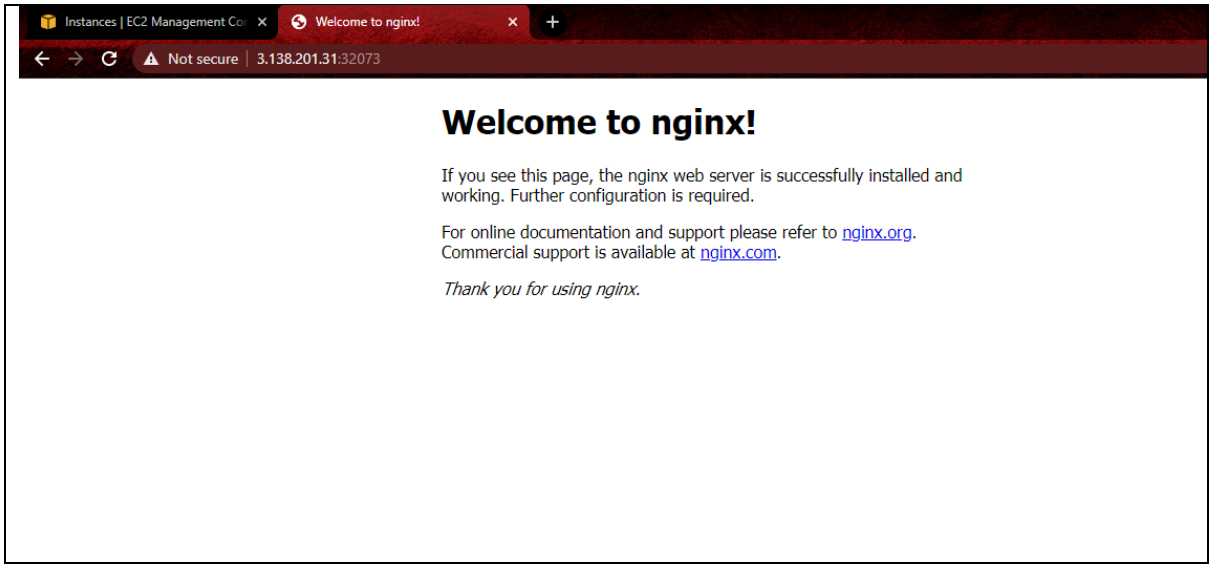
Step 2 :- Now we check the port in which we can access the application.

ubuntu@ip-172-31-23-130: ~

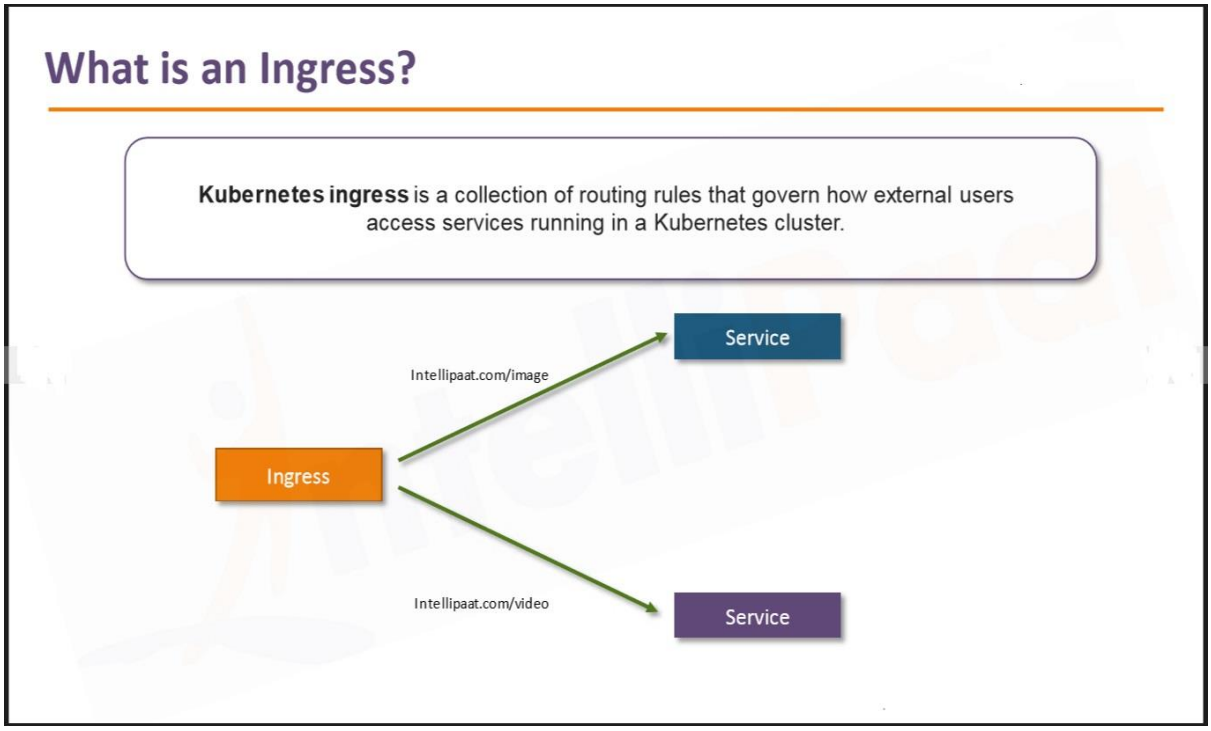
```
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$ kubectl get svc nginx
NAME    TYPE       CLUSTER-IP       EXTERNAL-IP   PORT(S)        AGE
nginx   NodePort   10.110.147.230   <none>        80:32073/TCP   38s
ubuntu@ip-172-31-23-130:~$
```

Step 3 :- Now we to take master-ip address and go ot browser ,then paste the ip address of followed by port number shown in above picture.

https://3.138.201.31:32073



Practical :- 4 Kubernetes Ingress.



Step 1 :- we have install bare-metal for ingress. Which will install ingress controller and ingress service.

$ kubectl apply -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/controller-v0.40.2/deploy/static/provider/baremetal/deploy.yaml

```
ubuntu@ip-172-31-23-130: ~
ubuntu@ip-172-31-23-130:~$ kubectl apply -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/controller-v0.40.2/deploy/sta
namespace/ingress-nginx created
serviceaccount/ingress-nginx created
configmap/ingress-nginx-controller created
clusterrole.rbac.authorization.k8s.io/ingress-nginx created
clusterrolebinding.rbac.authorization.k8s.io/ingress-nginx created
role.rbac.authorization.k8s.io/ingress-nginx created
rolebinding.rbac.authorization.k8s.io/ingress-nginx created
service/ingress-nginx-controller-admission created
service/ingress-nginx-controller created
deployment.apps/ingress-nginx-controller created
validatingwebhookconfiguration.admissionregistration.k8s.io/ingress-nginx-admission created
serviceaccount/ingress-nginx-admission created
clusterrole.rbac.authorization.k8s.io/ingress-nginx-admission created
clusterrolebinding.rbac.authorization.k8s.io/ingress-nginx-admission created
role.rbac.authorization.k8s.io/ingress-nginx-admission created
rolebinding.rbac.authorization.k8s.io/ingress-nginx-admission created
job.batch/ingress-nginx-admission-create created
job.batch/ingress-nginx-admission-patch created
ubuntu@ip-172-31-23-130:~$
```

Step 2 :- now we can check weather ingress controller is created or not .

$ kubectl get svc –all-namespaces

```
ubuntu@ip-172-31-23-130: ~
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$ kubectl get svc --all-namespaces
NAMESPACE              NAME                                TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)                       AGE
default                kubernetes                          ClusterIP   10.96.0.1        <none>        443/TCP                       7d6h
default                nginx                               NodePort    10.110.147.230   <none>        80:32073/TCP                  6d22h
ingress-nginx          ingress-nginx-controller            NodePort    10.97.234.209    <none>        80:32225/TCP,443:31012/TCP    2m45s
ingress-nginx          ingress-nginx-controller-admission  ClusterIP   10.106.211.205   <none>        443/TCP                       2m45s
kube-system            calico-typha                        ClusterIP   10.103.25.221    <none>        5473/TCP                      7d6h
kube-system            kube-dns                            ClusterIP   10.96.0.10       <none>        53/UDP,53/TCP,9153/TCP        7d6h
kubernetes-dashboard   dashboard-metrics-scraper           ClusterIP   10.98.68.208     <none>        8000/TCP                      6d22h
kubernetes-dashboard   kubernetes-dashboard                NodePort    10.108.175.87    <none>        443:31704/TCP                 6d22h
ubuntu@ip-172-31-23-130:~$
```

Step 3 :- Now we have to create our application in cluster ip. But we can see that our nginx is nodeport.So,we have to delete it and create new service with cluster ip.

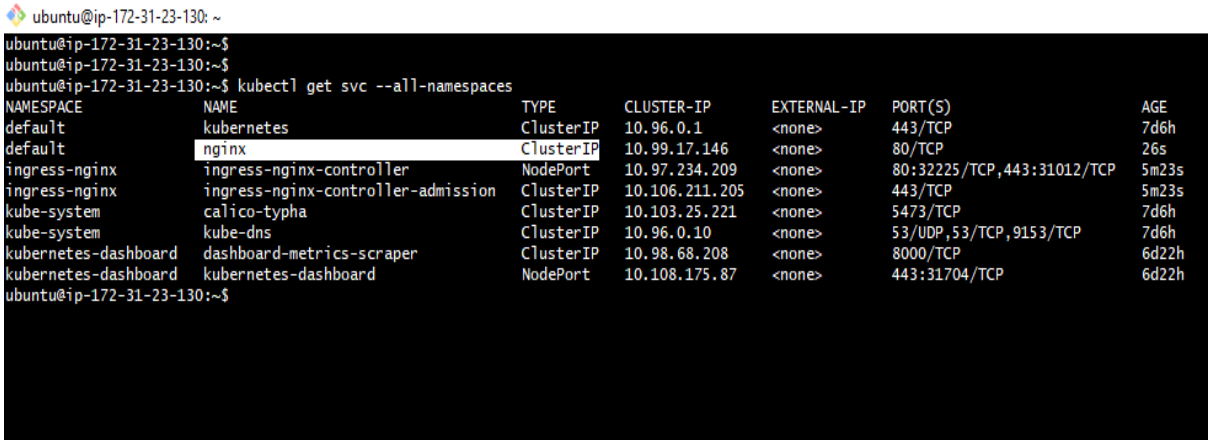
```
ubuntu@ip-172-31-23-130: ~
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$ kubectl get svc --all-namespaces
NAMESPACE              NAME                                TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)                       AGE
default                kubernetes                          ClusterIP   10.96.0.1        <none>        443/TCP                       7d6h
default                nginx                               NodePort    10.110.147.230   <none>        80:32073/TCP                  6d22h
ingress-nginx          ingress-nginx-controller            NodePort    10.97.234.209    <none>        80:32225/TCP,443:31012/TCP    2m45s
ingress-nginx          ingress-nginx-controller-admission  ClusterIP   10.106.211.205   <none>        443/TCP                       2m45s
kube-system            calico-typha                        ClusterIP   10.103.25.221    <none>        5473/TCP                      7d6h
kube-system            kube-dns                            ClusterIP   10.96.0.10       <none>        53/UDP,53/TCP,9153/TCP        7d6h
kubernetes-dashboard   dashboard-metrics-scraper           ClusterIP   10.98.68.208     <none>        8000/TCP                      6d22h
kubernetes-dashboard   kubernetes-dashboard                NodePort    10.108.175.87    <none>        443:31704/TCP                 6d22h
ubuntu@ip-172-31-23-130:~$
```

$ kubectl delete service nginx

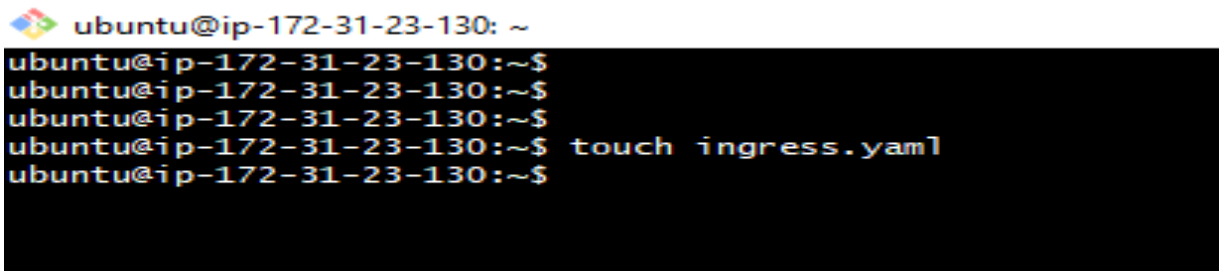
```
ubuntu@ip-172-31-23-130: ~
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$ kubectl delete service nginx
service "nginx" deleted
ubuntu@ip-172-31-23-130:~$
```

$ kubectl create service clusterip nginx –tcp=80:80

```
ubuntu@ip-172-31-23-130: ~
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$ kubectl create service clusterip nginx --tcp=80:80
service/nginx created
ubuntu@ip-172-31-23-130:~$
```



Step 4 :- Next, we will have to create an ingress rule, create an ingress.yaml file with the below code.




```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: minimal-ingress
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /
spec:
  rules:
  - http:
      paths:
      - path: /hemanth
        pathType: Prefix
        backend:
          service:
            name: test
            port:
              number: 80
```

```
ubuntu@ip-172-31-23-130: ~
  GNU nano 4.8
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: minimal-ingress
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /
spec:
  rules:
  - http:
      paths:
      - path: /hemanth
        pathType: Prefix
        backend:
          service:
            name: test
            port:
              number: 80
```

Step 5 :- Finally, create the ingress rule using the following command

$ kubectl create –f ingress.yaml

```
ubuntu@ip-172-31-23-130: ~
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$ kubectl create -f ingress.yaml
ingress.networking.k8s.io/minimal-ingress created
ubuntu@ip-172-31-23-130:~$
```

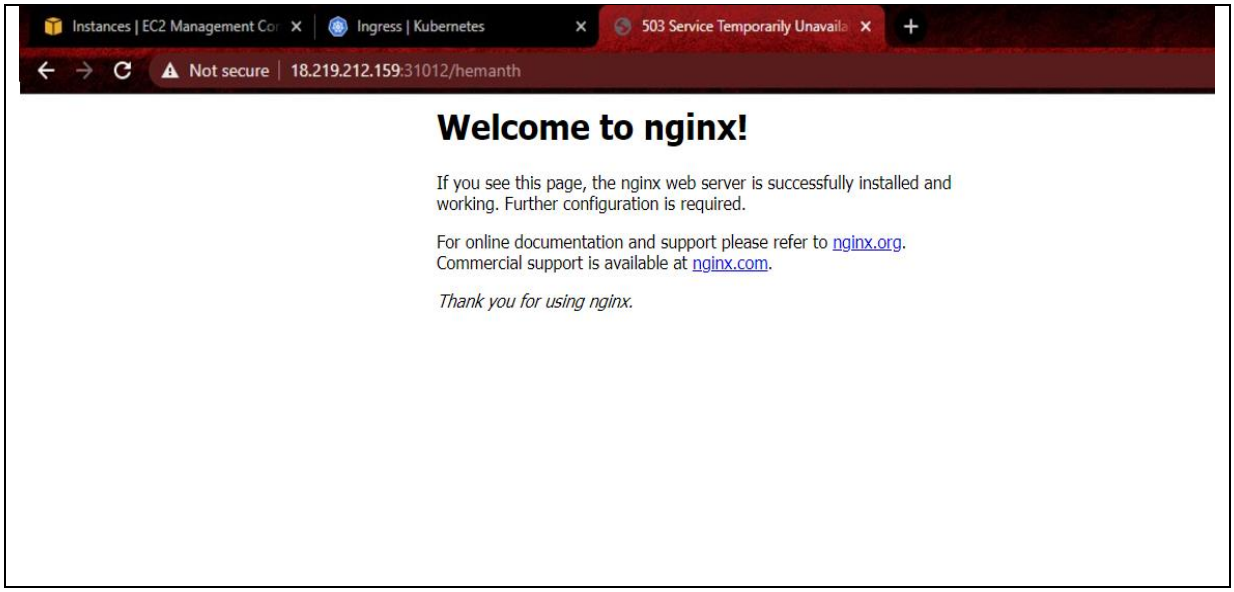
Step 6: Let's verify if ingress is working or not, by checking the nodeport of the ingress service, for checking the nodeport use the following command

$ kubectl get svc –n ingress-nginx

```
ubuntu@ip-172-31-23-130: ~
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$ kubectl get svc --all-namespaces
NAMESPACE              NAME                                 TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)                      AGE
default                kubernetes                           ClusterIP   10.96.0.1        <none>        443/TCP                      7d6h
default                nginx                                ClusterIP   10.99.17.146     <none>        80/TCP                       13m
ingress-nginx          ingress-nginx-controller             NodePort    10.97.234.209    <none>        80:32225/TCP,443:31012/TCP   17m
ingress-nginx          ingress-nginx-controller-admission   ClusterIP   10.106.211.205   <none>        443/TCP                      17m
kube-system            calico-typha                         ClusterIP   10.103.25.221    <none>        5473/TCP                     7d6h
kube-system            kube-dns                             ClusterIP   10.96.0.10       <none>        53/UDP,53/TCP,9153/TCP       7d6h
kubernetes-dashboard   dashboard-metrics-scraper            ClusterIP   10.98.68.208     <none>        8000/TCP                     6d22h
kubernetes-dashboard   kubernetes-dashboard                 NodePort    10.108.175.87    <none>        443:31704/TCP                6d22h
ubuntu@ip-172-31-23-130:~$
```

Step 7 :- Now finally we have to browse using master ip address and ingress-controller port number.

https://18.219.212.159:31012 /hemanth


Instances | EC2 Management Cor
Ingress | Kubernetes
503 Service Temporarily Unavaila
Not secure | 18.219.212.159:31012/hemanth
Welcome to nginx!
If you see this page, the nginx web server is successfully installed and working. Further configuration is required.
For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.
Thank you for using nginx.

Practical :- 5 Kubernetes UI.

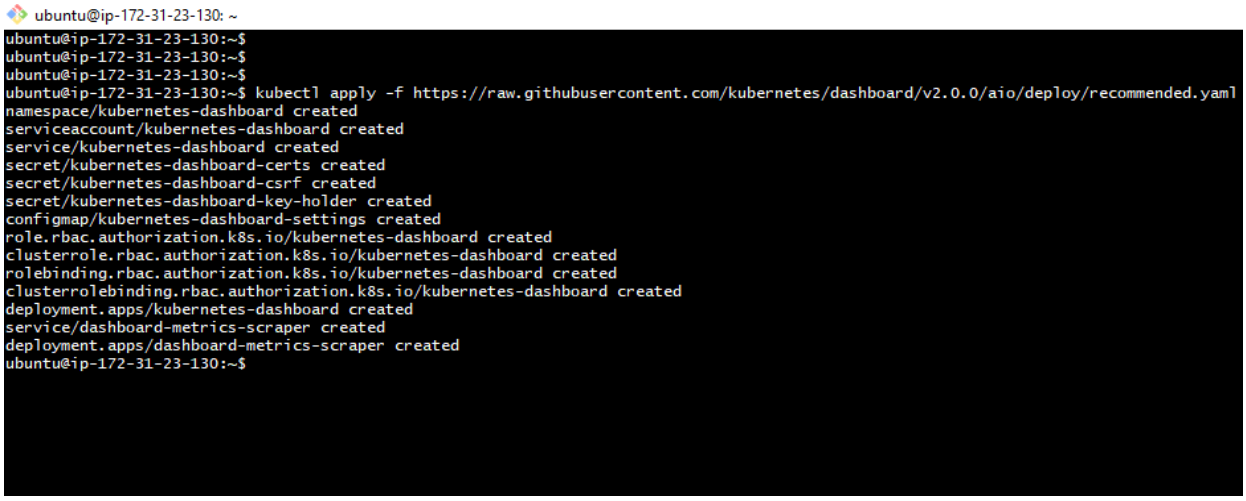
Dashboard is a web-based Kubernetes user interface. You can use Dashboard to deploy containerized applications to a Kubernetes cluster, troubleshoot your containerized application, and manage the cluster resources.

You can use Dashboard to get an overview of applications running on your cluster, as well as for creating or modifying individual Kubernetes resources (such as Deployments, Jobs, DaemonSets, etc).

Dashboard also provides information on the state of Kubernetes resources in your cluster and on any errors that may have occurred.

Step 1 :- The Dashboard UI is not deployed by default. To deploy it, run the following command.

$ kubectl apply -f https://raw.githubusercontent.com/kubernetes/dashboard/v2.0.0/aio/deploy/recommended.yaml



Step 2 :- For Accessing the Dashboard UI,
To protect your cluster data, Dashboard deploys with a minimal RBAC configuration by default. Currently, Dashboard only supports logging in with a Bearer Token
We are creating Service Account with name admin-user in namespace kubernetes-dashboard first**.**

```
cat <<EOF | kubectl apply -f -
apiVersion: v1
kind: ServiceAccount
metadata:
  name: admin-user
  namespace: kubernetes-dashboard
EOF
```

ubuntu@ip-172-31-23-130: ~

```
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$ cat <<EOF | kubectl apply -f -
> apiVersion: v1
> kind: ServiceAccount
metadata:
> metadata:
>   name: admin-user
>   namespace: kubernetes-dashboard
> EOF
serviceaccount/admin-user created
ubuntu@ip-172-31-23-130:~$
```

Step 3 :- Now we have to create a ClusterRoleBinding

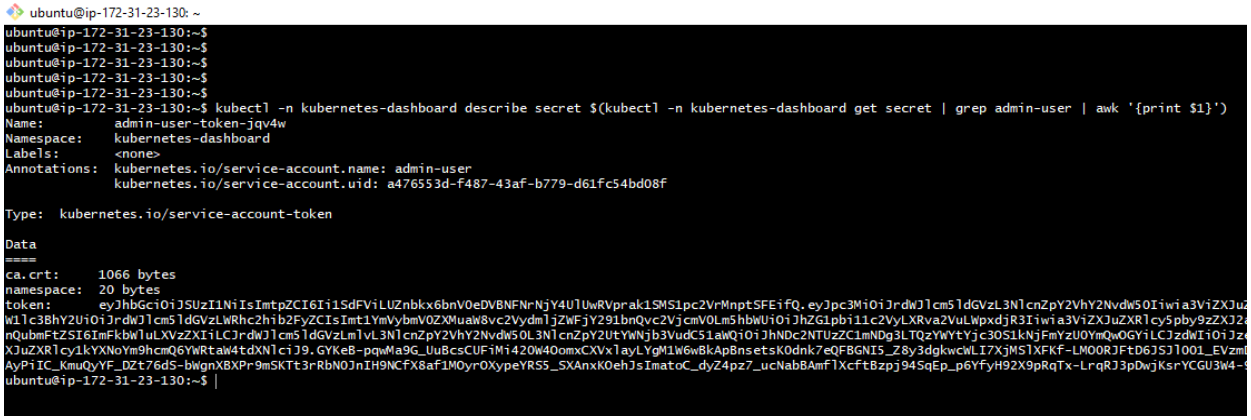
```
cat <<EOF | kubectl apply -f -
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: admin-user
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: cluster-admin
subjects:
- kind: ServiceAccount
  name: admin-user
namespace: kubernetes-dashboard
EOF
```

ubuntu@ip-172-31-23-130: ~

```
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$ cat <<EOF | kubectl apply -f -
> apiVersion: rbac.authorization.k8s.io/v1
> kind: ClusterRoleBinding
> metadata:
>   name: admin-user
> roleRef:
>   apiGroup: rbac.authorization.k8s.io
>   kind: ClusterRole
>   name: cluster-admin
> subjects:
> - kind: ServiceAccount
>   name: admin-user
>   namespace: kubernetes-dashboard
> EOF
clusterrolebinding.rbac.authorization.k8s.io/admin-user created
ubuntu@ip-172-31-23-130:~$
```

Step 4 :- Now we need to find token we can use to log in .we have to use following command.

$ kubectl -n kubernetes-dashboard describe secret $(kubectl -n kubernetes-dashboard get secret | grep admin-user | awk '{print $1}')



My generated token is :

eyJhbGciOiJSUzI1NiIsImtpZCI6Ii1SdFViLUZnbkx6bnV0eDVBNFNrNjY4UlUwRVprak1SMS1pc2VrMnptSFEifQ.eyJpc3MiOiJrdWJlcm5ldGVzL3NlcnZpY2VhY2NvdW50Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9uYW1lc3BhY2UiOiJrdWJlcm5ldGVzLWRhc2hib2FyZCIsImt1YmVybmV0ZXMuaW8vc2VydmljZWFjY291bnQvc2VjcmV0LmNhbWUiOiJhZG1pbi1c2VyLXRva2VuLWpxdjR3Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9zZXJ2aWNlLWFjY291bnQubmFtZSI6ImFkbWluLXVzZXIiLCJrdWJlcm5ldGVzLmlvL3NlcnZpY2VhY2NvdW50L3NlcnZpY2UtYWNjb3VudC51aWQiOiJhNDc2NTUzZC1mNDg3LTQzYWYtYjc3OS1kNjFmYzU0YmQwOGYiLCJzdWIiOiJzeXN0ZW06c2VydmljZWFjY291bnQ6a3ViZXJuZXRlcy1kYXNoYm9hcmQ6YWRtaW4tdXNlciJ9.GYKeB-pqwMa9G_UuBcsCUFiMi420W4OomxCXVxlayLYgM1W6wBkApBnsetsK0dnk7eQFBGNI5_Z8y3dgkwcWLI7XjMSlXFKf-LMO0RJFtD6JSJl001_EVzmDMOE5QlCM0qdszwJMLHOqOIe2FBu9RIpmAyPiIC_KmuQyYF_DZt76dS-bWgnXBXPr9mSKTt3rRbN0JnIH9NCfX8af1MOyrOXypeYRS5_SXAnxKOehJsImatoC_dyZ4pz7_ucNabBAmflXcftBzpj94SqEp_p6YfyH92X9pRqTx-LrqRJ3pDwjKsrYCGU3W4-92B1yn2zIQgofzY2mm_6D6j8-xp6lIQ

Step 5 :- Now for accessing dashboard we have to edit cluster ip to node port because it was a recommended way for accessing dashboard .

$ kubectl -n kubernetes-dashboard edit service kubernetes-dashboard

ubuntu@ip-172-31-23-130: ~

```
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$ kubectl -n kubernetes-dashboard edit service kubernetes-dashboard
service/kubernetes-dashboard edited
ubuntu@ip-172-31-23-130:~$
```

Now the new yml file is opened in it we have to change Cluster ip to NodePort**.**

ubuntu@ip-172-31-23-130: ~

```
# Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this file will be
# reopened with the relevant failures.
#
apiVersion: v1
kind: Service
metadata:
  annotations:
    kubectl.kubernetes.io/last-applied-configuration: |
      {"apiVersion":"v1","kind":"Service","metadata":{"annotations":{},"labels":{"k8s-app":"kubernetes-dashboard"},"name":"kubernetes-dashboard","namespace":"kubernete
rts":[{"port":443,"targetPort":8443}],"selector":{"k8s-app":"kubernetes-dashboard"}}}
  creationTimestamp: "2020-10-18T15:27:32Z"
  labels:
    k8s-app: kubernetes-dashboard
  name: kubernetes-dashboard
  namespace: kubernetes-dashboard
  resourceVersion: "14584"
  selfLink: /api/v1/namespaces/kubernetes-dashboard/services/kubernetes-dashboard
  uid: 0879b458-1283-40f5-831b-fabbc25952f6
spec:
  clusterIP: 10.108.175.87
  externalTrafficPolicy: Cluster
  ports:
  - nodePort: 31704
    port: 443
    protocol: TCP
    targetPort: 8443
  selector:
    k8s-app: kubernetes-dashboard
  sessionAffinity: None
  type: NodePort
status:
  loadBalancer: {}
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"/tmp/kubectl-edit-e9joe.yaml" 32L, 1162C
```

Step 5 :- Next we need to check port on which Dashboard was exposed.

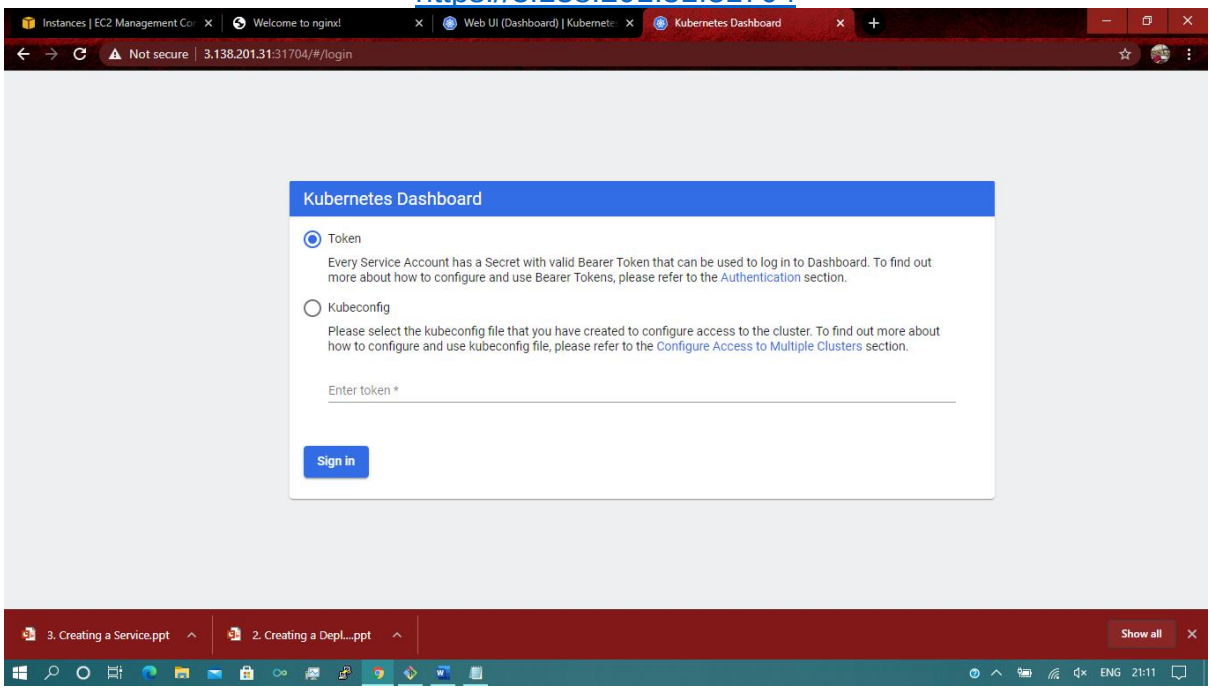
$ kubectl -n kubernetes-dashboard get service kubernetes-dashboard

ubuntu@ip-172-31-23-130: ~

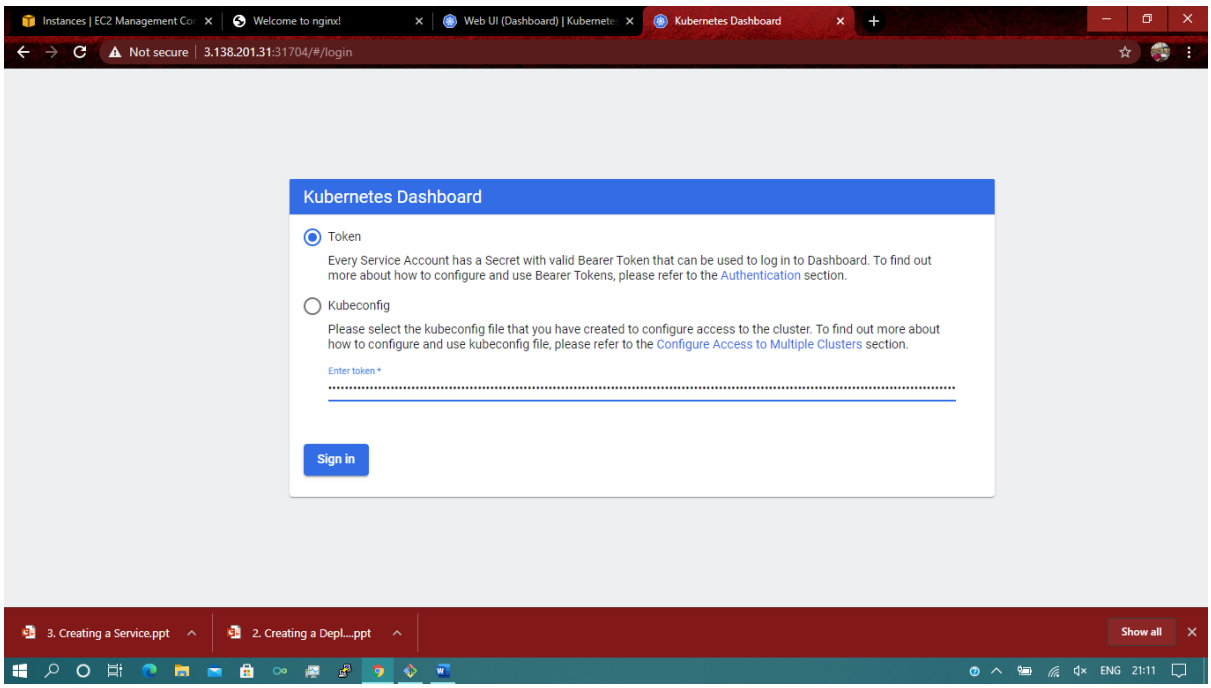
```
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$
ubuntu@ip-172-31-23-130:~$ kubectl -n kubernetes-dashboard get service kubernetes-dashboard
NAME                   TYPE       CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
kubernetes-dashboard   NodePort   10.108.175.87   <none>        443:31704/TCP    11m
ubuntu@ip-172-31-23-130:~$
```

Step 6 :- Now copy the ip-address. Paste it in browser along with port number shown in above picture .

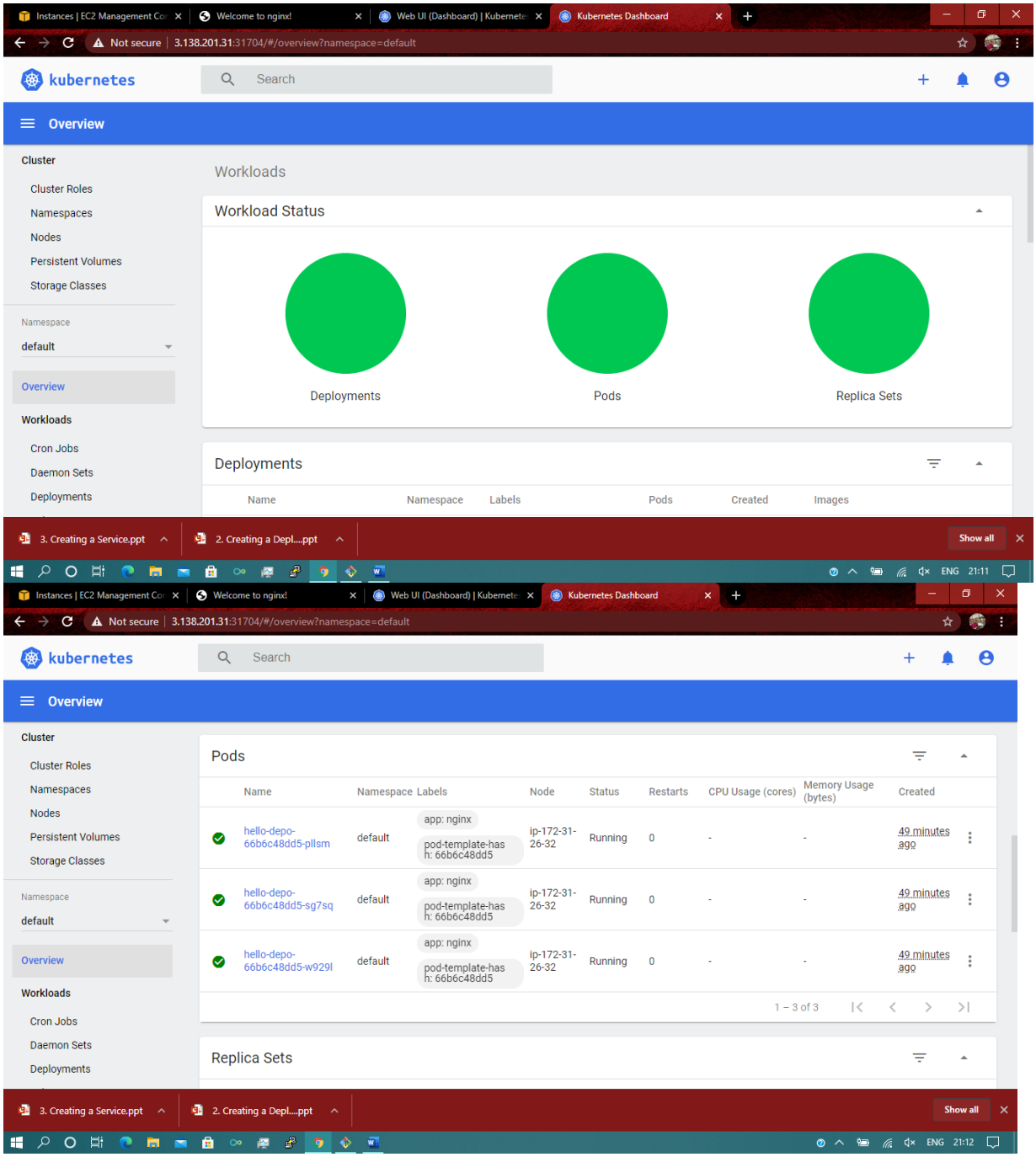
https://3.138.201.31:31704



Step 7 :- Now paste the token generated by cluster binding in the login page to enter into the bash-board ui .



After pasting token just press sign in.


Instances | EC2 Management Co
Welcome to nginx!
Web UI (Dashboard) | Kubernete
Kubernetes Dashboard
Not secure | 3.138.201.31:31704/#/overview?namespace=default
kubernetes
Search
Overview
Cluster
Cluster Roles
Namespaces
Nodes
Persistent Volumes
Storage Classes
Namespace
default
Overview
Workloads
Cron Jobs
Daemon Sets
Deployments
Workloads
Workload Status
Deployments
Pods
Replica Sets
Deployments
Name
Namespace
Labels
Pods
Created
Images
3. Creating a Service.ppt
2. Creating a Depl....ppt
Show all
ENG 21:11
Instances | EC2 Management Co
Welcome to nginx!
Web UI (Dashboard) | Kubernete
Kubernetes Dashboard
Not secure | 3.138.201.31:31704/#/overview?namespace=default
kubernetes
Search
Overview
Cluster
Cluster Roles
Namespaces
Nodes
Persistent Volumes
Storage Classes
Namespace
default
Overview
Workloads
Cron Jobs
Daemon Sets
Deployments
Pods
Name
Namespace
Labels
Node
Status
Restarts
CPU Usage (cores)
Memory Usage (bytes)
Created
hello-depo-66b6c48dd5-pllsm
default
app: nginx
pod-template-hash: 66b6c48dd5
ip-172-31-26-32
Running
0
-
-
49 minutes ago
hello-depo-66b6c48dd5-sg7sq
default
app: nginx
pod-template-hash: 66b6c48dd5
ip-172-31-26-32
Running
0
-
-
49 minutes ago
hello-depo-66b6c48dd5-w929l
default
app: nginx
pod-template-hash: 66b6c48dd5
ip-172-31-26-32
Running
0
-
-
49 minutes ago
1 – 3 of 3
Replica Sets
3. Creating a Service.ppt
2. Creating a Depl....ppt
Show all
ENG 21:12

Practical 6: Splunk

Step 1 :- Now create two instances, name them one as master and other as universal forwarder. And connect master using ssh command in gitbash.

ubuntu@ip-172-31-46-43: ~

```
Hemanth Andru@Hemanth-Andru MINGW64 ~
$ cd Downloads/

Hemanth Andru@Hemanth-Andru MINGW64 ~/Downloads
$ ssh -i "hem1.pem" ubuntu@ec2-13-126-194-168.ap-south-1.compute.amazonaws.com
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-1024-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Mon Oct 26 04:15:05 UTC 2020

  System load:  0.21              Processes:             102
  Usage of /:   16.8% of 7.69GB   Users logged in:       0
  Memory usage: 20%               IPv4 address for eth0: 172.31.46.43
  Swap usage:   0%


1 update can be installed immediately.
0 of these updates are security updates.
To see these additional updates run: apt list --upgradable


The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Mon Oct 26 04:14:06 2020 from 157.48.140.191
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-46-43:~$
```

Step 2 :- Now just enter into super user mode using the following command :

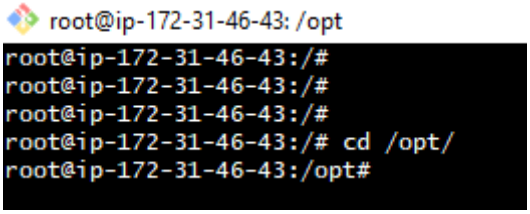
$ sudo su

root@ip-172-31-46-43: /home/ubuntu

```
ubuntu@ip-172-31-46-43:~$
ubuntu@ip-172-31-46-43:~$
ubuntu@ip-172-31-46-43:~$ sudo su
root@ip-172-31-46-43:/home/ubuntu#
```

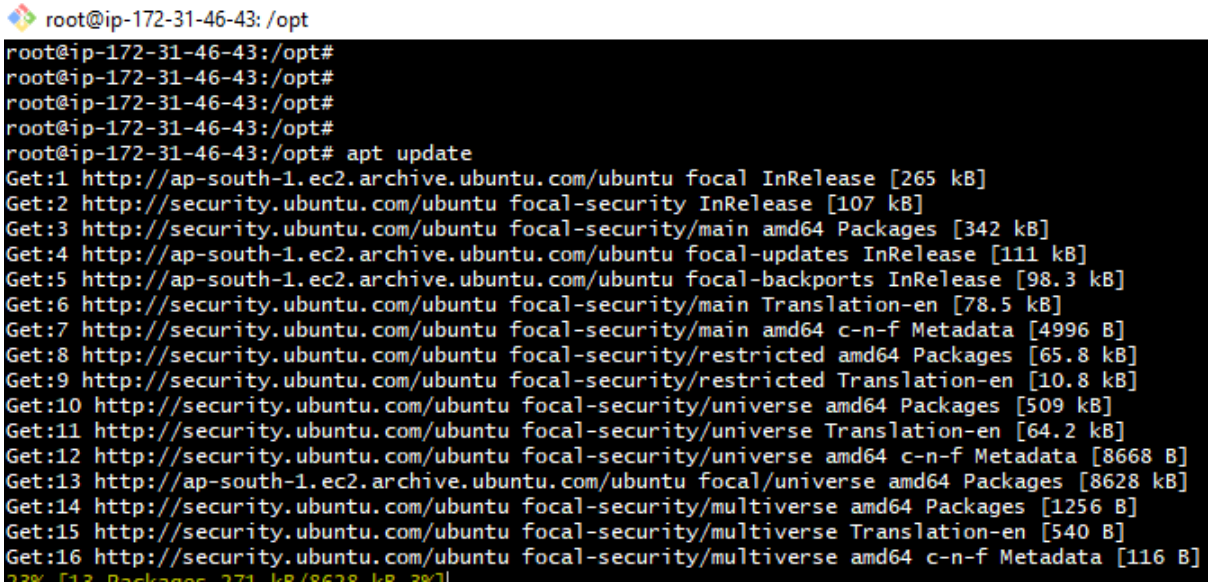
Step 3 :- Now just enter into opt folder using following command :

$ cd /opt



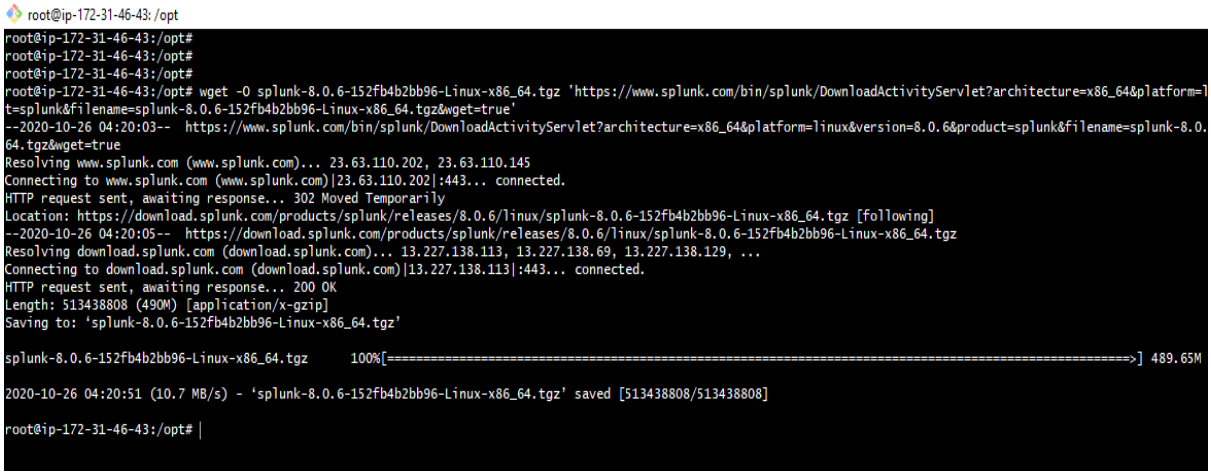
Step 4 :- Now just update ubuntu using the following command :

$ apt update



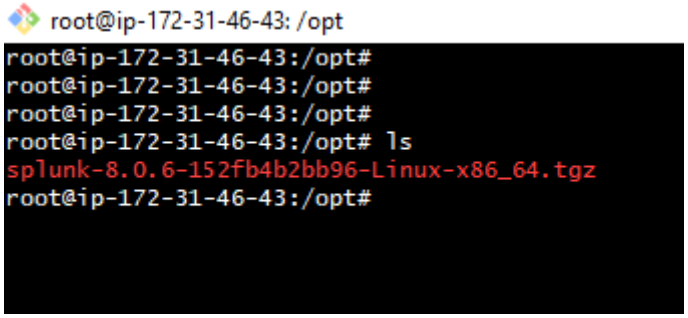
Step 5 :- Now we have to download the splunk package from online using the following command :

$ wget -O splunk-8.0.6-152fb4b2bb96-Linux-x86_64.tgz 'https://www.splunk.com/bin/splunk/DownloadActivityServlet?architecture=x86_64&platform=linux&version=8.0.6&product=splunk&filename=splunk-8.0.6-152fb4b2bb96-Linux-x86_64.tgz&wget=true'



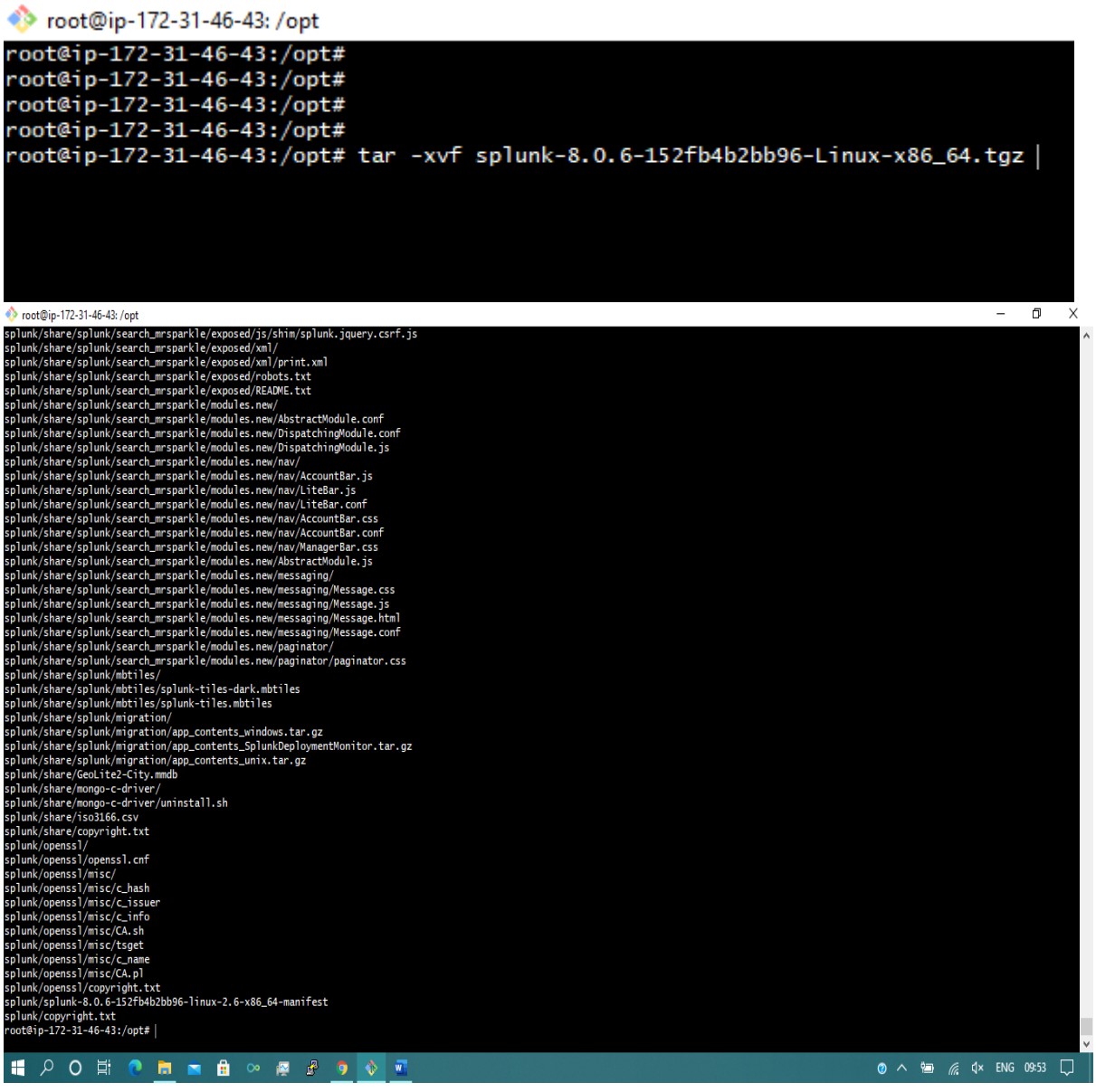
Step 6 :- Now just check wheater the splunk zip file is downloaded or not.

$ ls



Step 7 :- Now unzip the file to get splunk documentation.

$ tar –xvf (file name after ls)

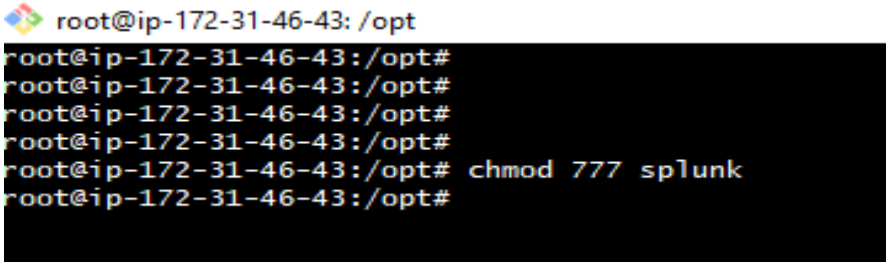


Step 8 :- Now check the folder is created or not using ls command.And the give

permission to the file.

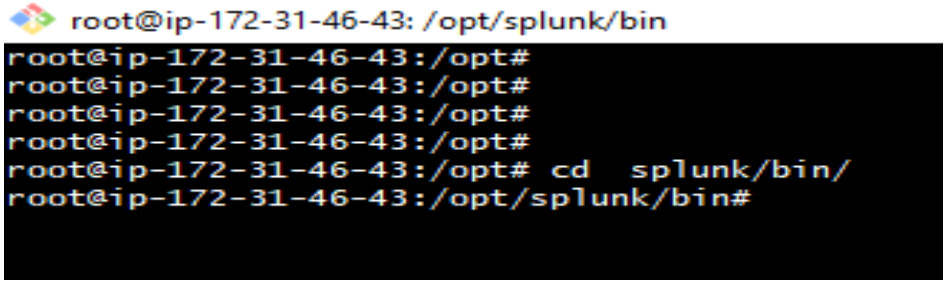
$ ls

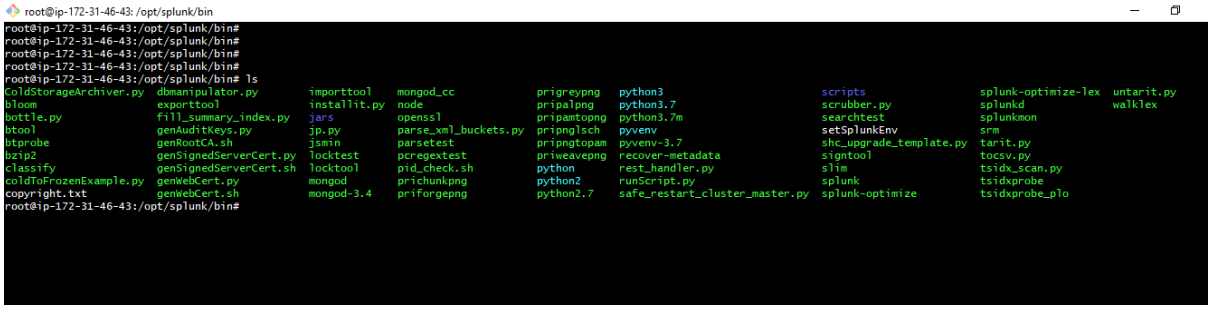
$ chmod 777 splunk



Step 9 :- now move to splunk bin folder where all applications of splunk can be worked.

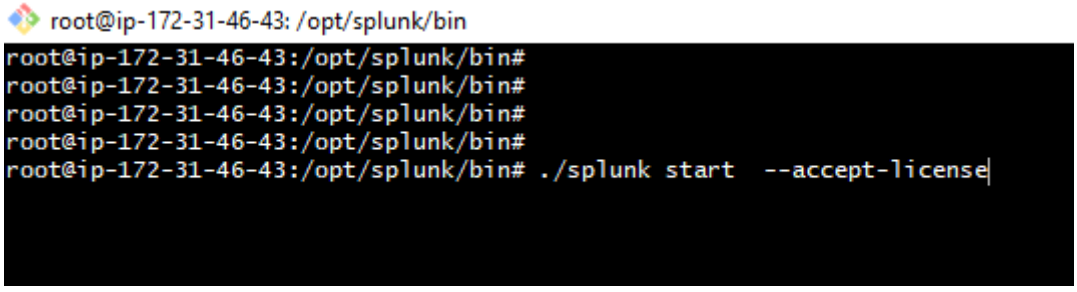
$ cd splunk/bin



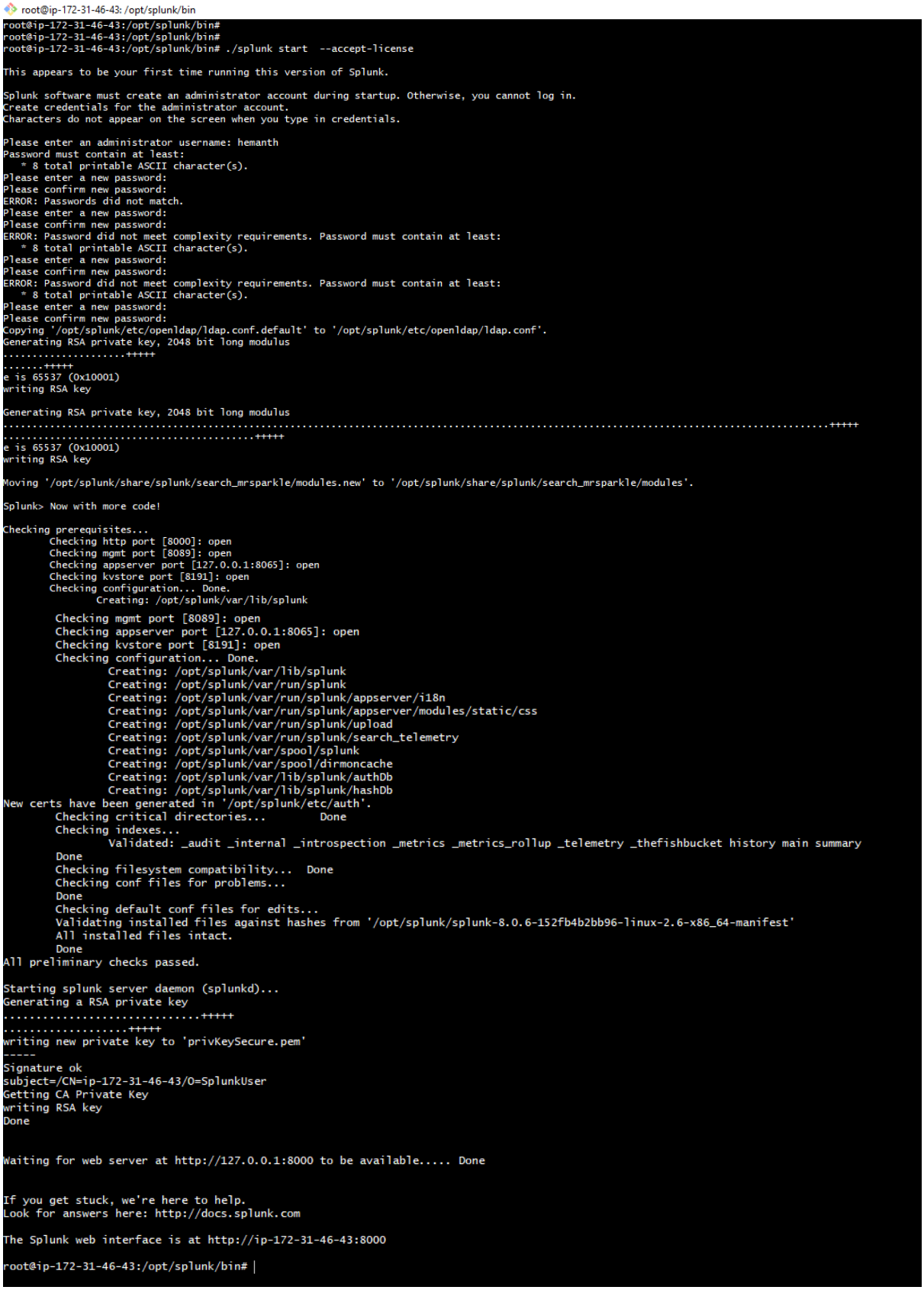


Step 10 :- for starting splunk ,we have to accept license of the splunk.

$ ./splunk start –accept-license



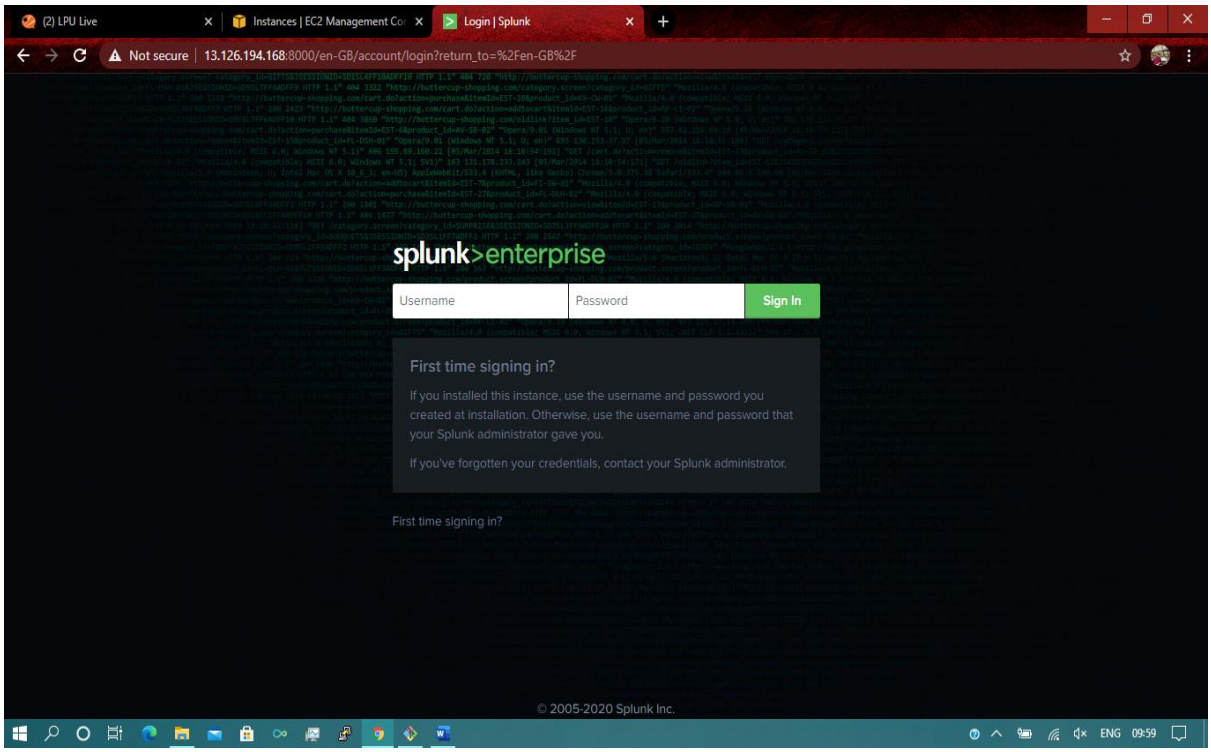
Now it will ask us to create user name and password to create account.



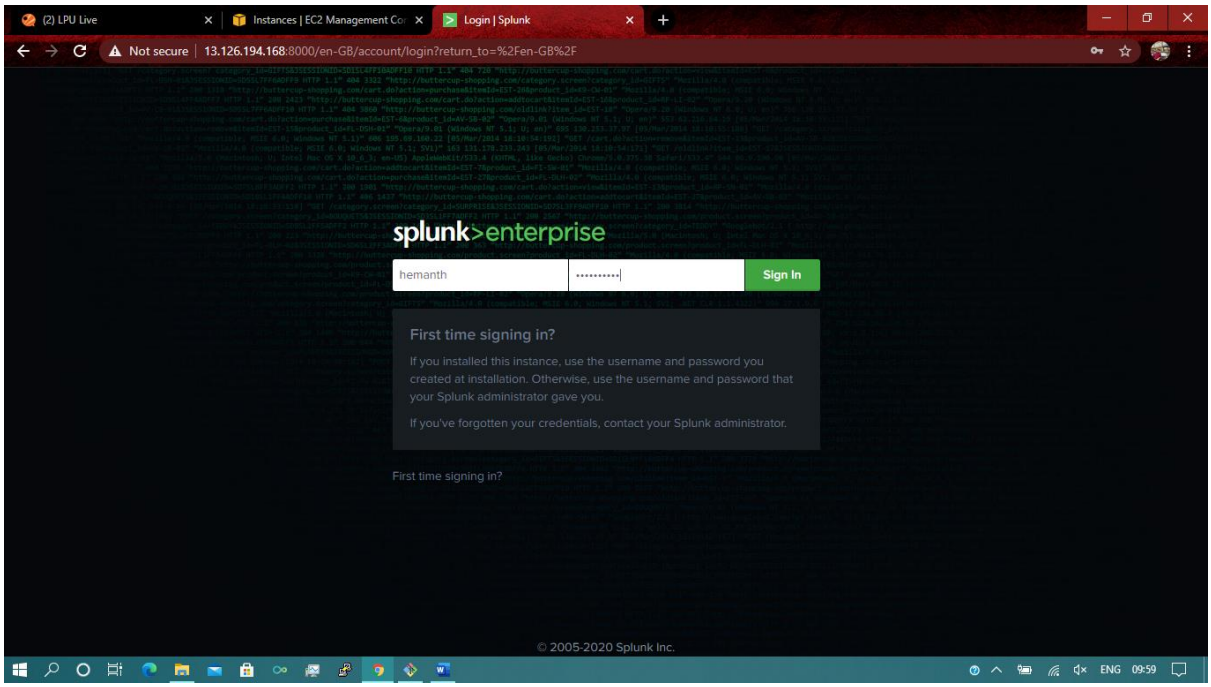
Step 11 :- now copy the ip-address of master from ec2 instance and paste it in

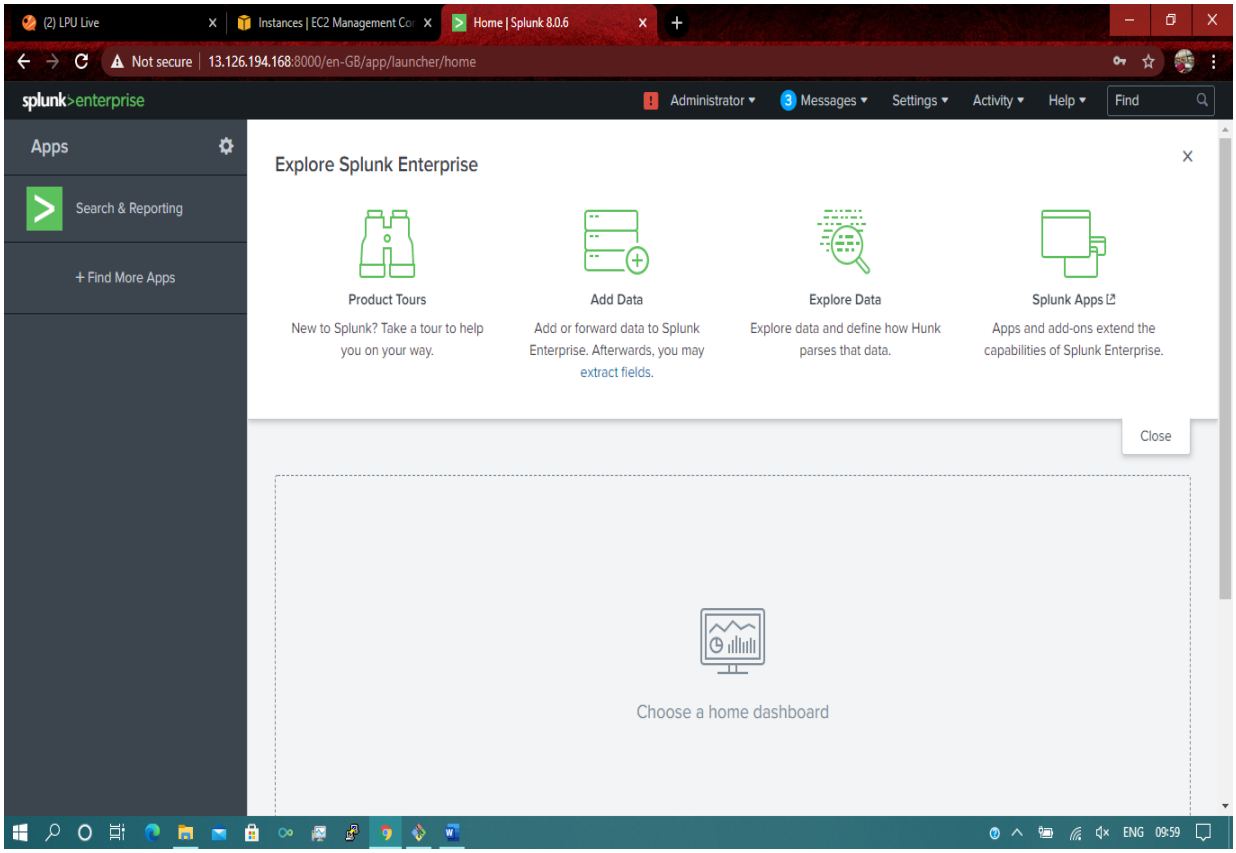
browser following with port 8000.

https://13.126.194.168:8000



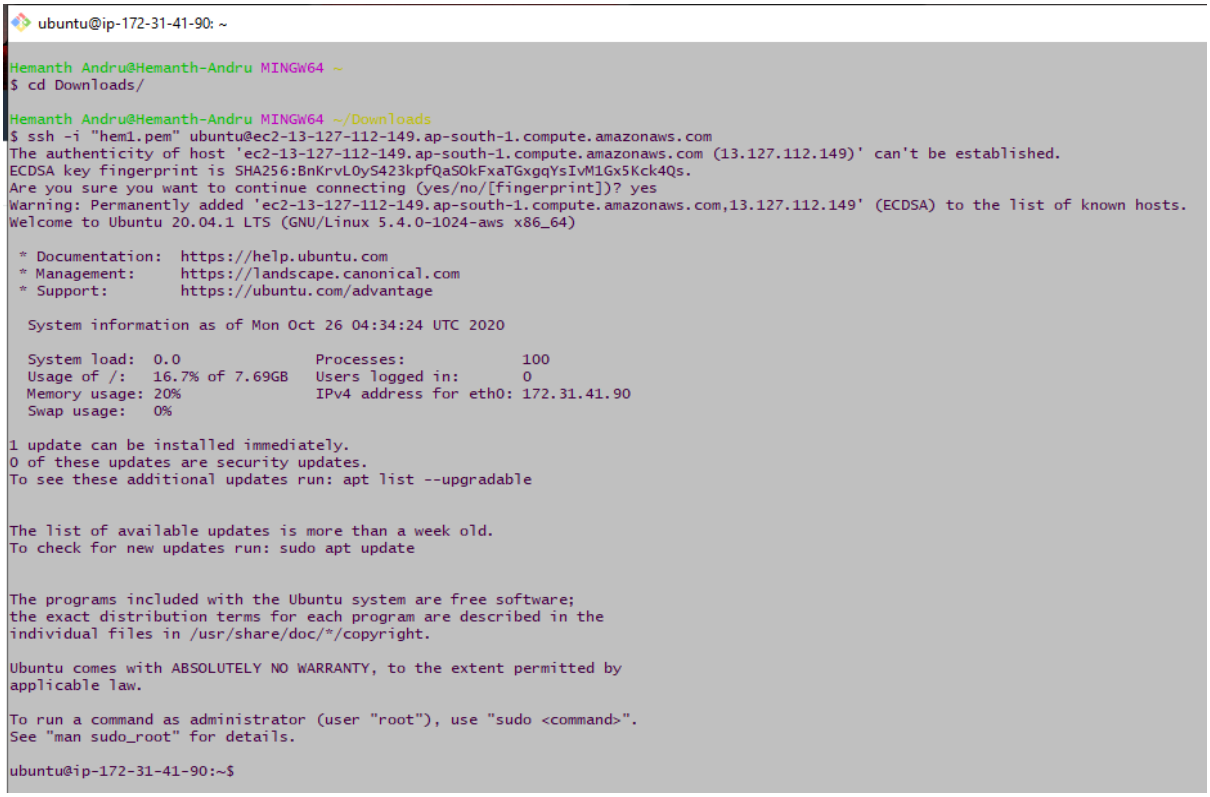
Now enter the details of username and password created before.





Now lets start with installing splunk frowder..

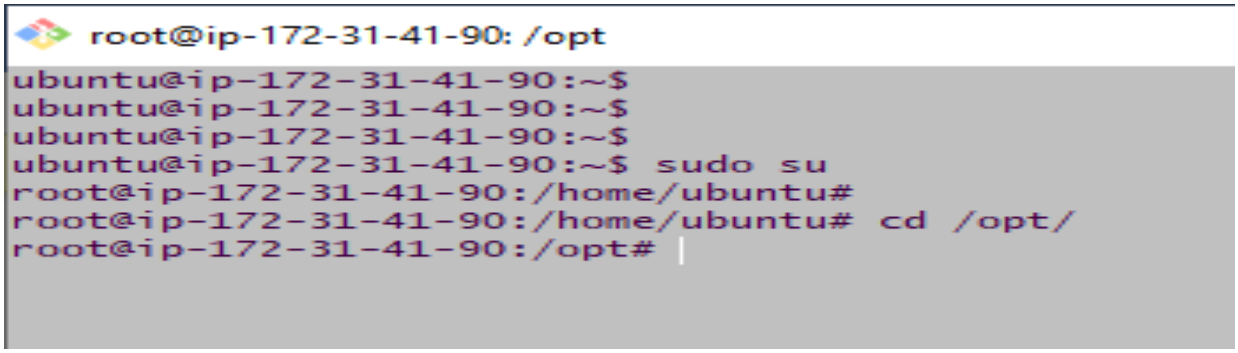
Step 1 :- Now open universal forwarder ec2 instance that named before and connect with ssh command.



Step 2 :- Enter into super using sudo su and enter into opt folder using following command :

$ sudo su

$cd /opt



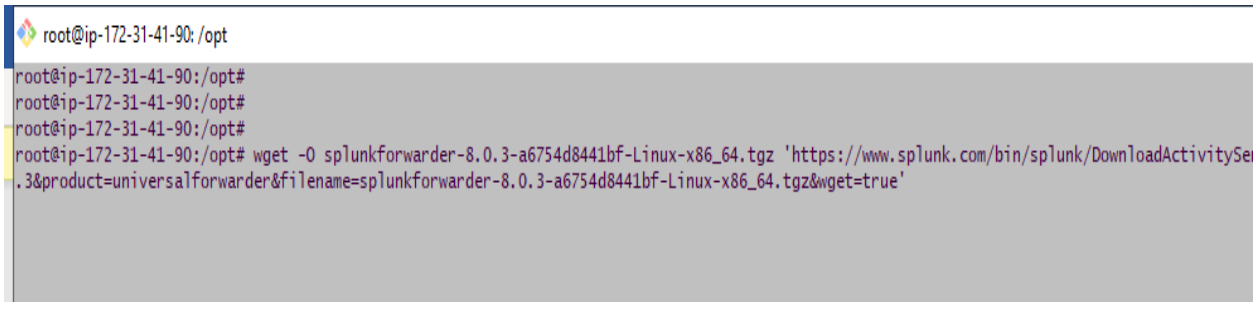
Step 3 :- update the cli using the following command :

$ apt update

```
root@ip-172-31-41-90:/opt# apt update
Hit:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal InRelease
Get:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-updates InRelease [111 kB]
Get:3 http://security.ubuntu.com/ubuntu focal-security InRelease [107 kB]
Get:4 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-backports InRelease [98.3 kB]
Get:5 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 Packages [8628 kB]
Get:6 http://security.ubuntu.com/ubuntu focal-security/main amd64 Packages [342 kB]
Get:7 http://security.ubuntu.com/ubuntu focal-security/main Translation-en [78.5 kB]
Get:8 http://security.ubuntu.com/ubuntu focal-security/main amd64 c-n-f Metadata [4996 B]
Get:9 http://security.ubuntu.com/ubuntu focal-security/restricted amd64 Packages [65.8 kB]
Get:10 http://security.ubuntu.com/ubuntu focal-security/restricted Translation-en [10.8 kB]
Get:11 http://security.ubuntu.com/ubuntu focal-security/universe amd64 Packages [509 kB]
Get:12 http://security.ubuntu.com/ubuntu focal-security/universe Translation-en [64.2 kB]
Get:13 http://security.ubuntu.com/ubuntu focal-security/universe amd64 c-n-f Metadata [8668 B]
Get:14 http://security.ubuntu.com/ubuntu focal-security/multiverse amd64 Packages [1256 B]
Get:15 http://security.ubuntu.com/ubuntu focal-security/multiverse Translation-en [540 B]
Get:16 http://security.ubuntu.com/ubuntu focal-security/multiverse amd64 c-n-f Metadata [116 B]
Get:17 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal/universe Translation-en [5124 kB]
Get:18 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 c-n-f Metadata [265 kB]
Get:19 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal/multiverse amd64 Packages [144 kB]
Get:20 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal/multiverse Translation-en [104 kB]
Get:21 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal/multiverse amd64 c-n-f Metadata [9136 B]
Get:22 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages [618 kB]
Get:23 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-updates/main Translation-en [156 kB]
Get:24 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-updates/main amd64 c-n-f Metadata [10.4 kB]
Get:25 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-updates/restricted amd64 Packages [78.2 kB]
Get:26 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-updates/restricted Translation-en [12.4 kB]
Get:27 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-updates/restricted amd64 c-n-f Metadata [348 B]
Get:28 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-updates/universe amd64 Packages [673 kB]
Get:29 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-updates/universe Translation-en [127 kB]
Get:30 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-updates/universe amd64 c-n-f Metadata [12.3 kB]
Get:31 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-updates/multiverse amd64 Packages [15.1 kB]
Get:32 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-updates/multiverse Translation-en [3892 B]
Get:33 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-updates/multiverse amd64 c-n-f Metadata [480 B]
Get:34 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-backports/main amd64 c-n-f Metadata [112 B]
Get:35 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-backports/restricted amd64 c-n-f Metadata [116 B]
Get:36 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-backports/universe amd64 Packages [4012 B]
Get:37 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-backports/universe Translation-en [1448 B]
Get:38 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-backports/universe amd64 c-n-f Metadata [224 B]
Get:39 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu focal-backports/multiverse amd64 c-n-f Metadata [116 B]
Fetched 17.4 MB in 6s (2854 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
68 packages can be upgraded. Run 'apt list --upgradable' to see them.
root@ip-172-31-41-90:/opt#
```

Step 4 :- now we have download the splunk-forwarder zip file from online using wget command :

$ wget -O splunkforwarder-8.0.3-a6754d8441bf-Linux-x86_64.tgz 'https://www.splunk.com/bin/splunk/DownloadActivityServlet?architecture=x86_64&platform=linux&version=8.0.3&product=universalforwarder&filename=splunkforwarder-8.0.3-a6754d8441bf-Linux-x86_64.tgz&wget=true'



```
root@ip-172-31-41-90: /opt
root@ip-172-31-41-90:/opt#
root@ip-172-31-41-90:/opt#
root@ip-172-31-41-90:/opt#
root@ip-172-31-41-90:/opt# wget -O splunkforwarder-8.0.3-a6754d8441bf-Linux-x86_64.tgz 'https://www.splunk.com/bin/
.3&product=universalforwarder&filename=splunkforwarder-8.0.3-a6754d8441bf-Linux-x86_64.tgz&wget=true'
--2020-10-26 04:36:34--  https://www.splunk.com/bin/splunk/DownloadActivityServlet?architecture=x86_64&platform=li
754d8441bf-Linux-x86_64.tgz&wget=true
Resolving www.splunk.com (www.splunk.com)... 23.63.110.202, 23.63.110.145
Connecting to www.splunk.com (www.splunk.com)|23.63.110.202|:443... connected.
HTTP request sent, awaiting response... 302 Moved Temporarily
Location: https://download.splunk.com/products/universalforwarder/releases/8.0.3/linux/splunkforwarder-8.0.3-a6754
--2020-10-26 04:36:35--  https://download.splunk.com/products/universalforwarder/releases/8.0.3/linux/splunkforwar
Resolving download.splunk.com (download.splunk.com)... 13.227.138.114, 13.227.138.113, 13.227.138.69, ...
Connecting to download.splunk.com (download.splunk.com)|13.227.138.114|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 27023654 (26M) [application/x-gzip]
Saving to: 'splunkforwarder-8.0.3-a6754d8441bf-Linux-x86_64.tgz'

splunkforwarder-8.0.3-a6754d8441bf-Linux-x86_64 100%[=============================================================

2020-10-26 04:36:40 (7.11 MB/s) - 'splunkforwarder-8.0.3-a6754d8441bf-Linux-x86_64.tgz' saved [27023654/27023654]
```

Step 5 :- Now just check wheater the splunk-forwarder zip file is downloaded **or not.**

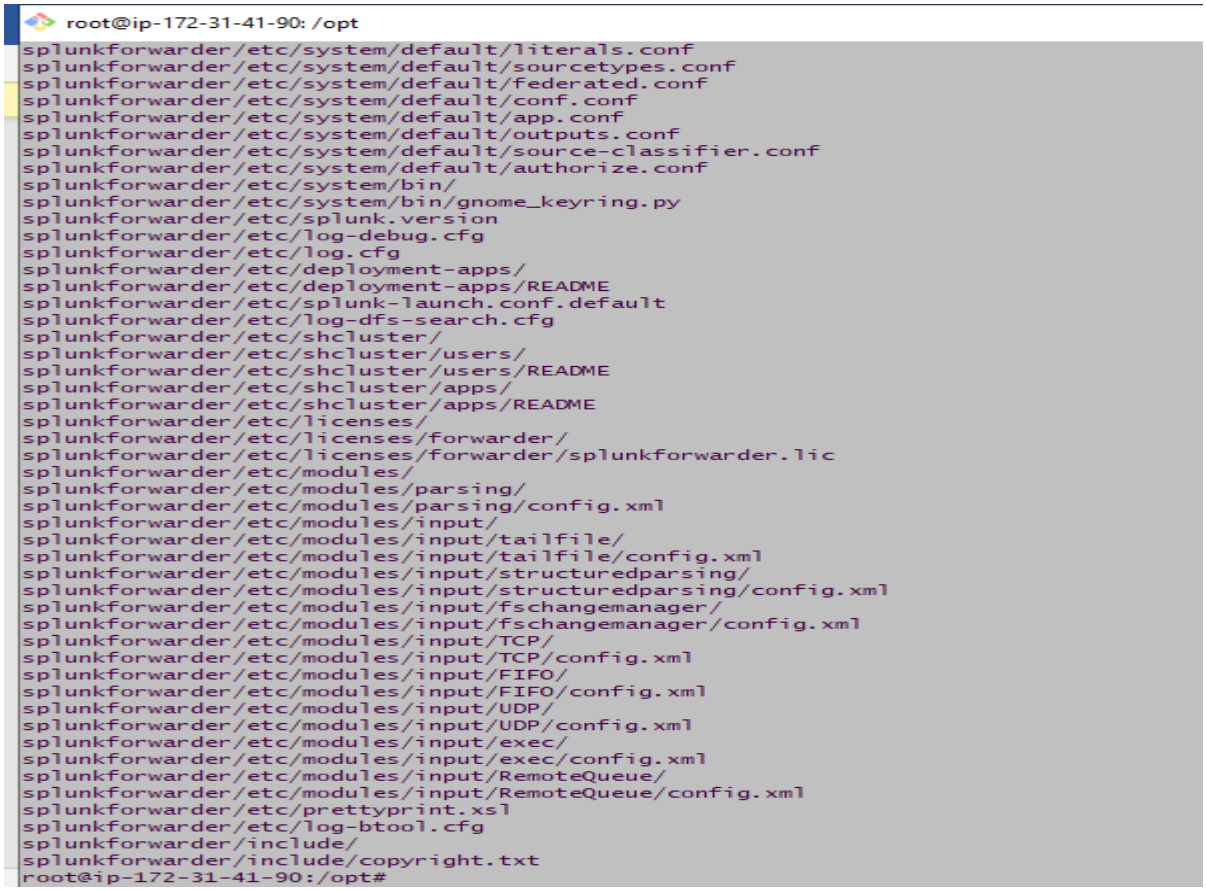
$ ls

```
root@ip-172-31-41-90: /opt
root@ip-172-31-41-90:/opt#
root@ip-172-31-41-90:/opt#
root@ip-172-31-41-90:/opt# ls
splunkforwarder-8.0.3-a6754d8441bf-Linux-x86_64.tgz
root@ip-172-31-41-90:/opt#
```

Step 6 :- Now just check wheater the splunk-forwader zip file is downloaded or not.

$ tar –xvf (file name)

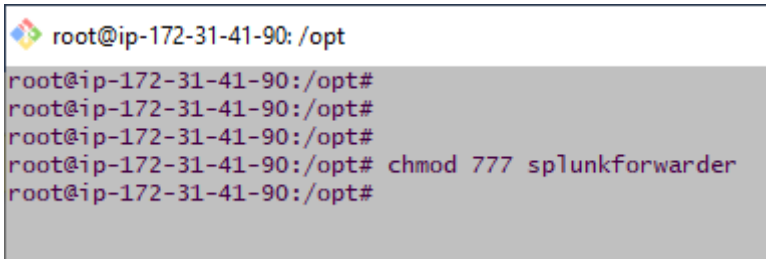
```
root@ip-172-31-41-90: /opt
root@ip-172-31-41-90:/opt#
root@ip-172-31-41-90:/opt#
root@ip-172-31-41-90:/opt#
root@ip-172-31-41-90:/opt# tar -xvf splunkforwarder-8.0.3-a6754d8441bf-Linux-x86_64.tgz
```



Step 7 :- Now check the folder is created or not using ls command .And the give permission to the folder.

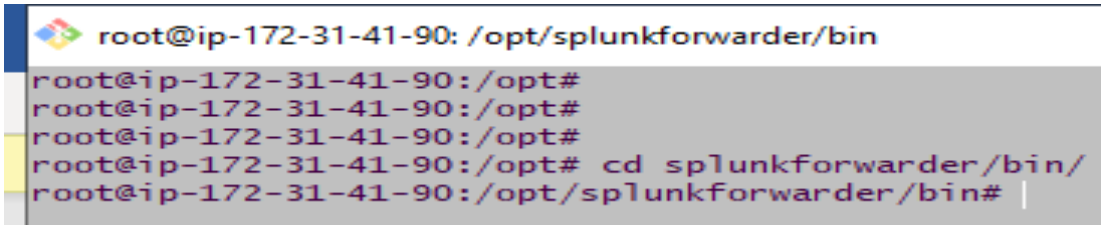
$ ls

$ chmod 777 splunkforwarder



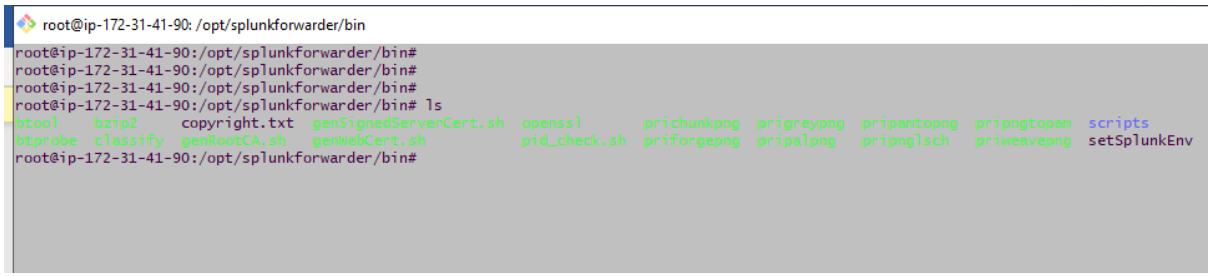
Step 8 :- now move to splunk-forwarder bin folder where all applications of splunkforwarder can be worked**.**

$ cd splunkforwarder/bin/



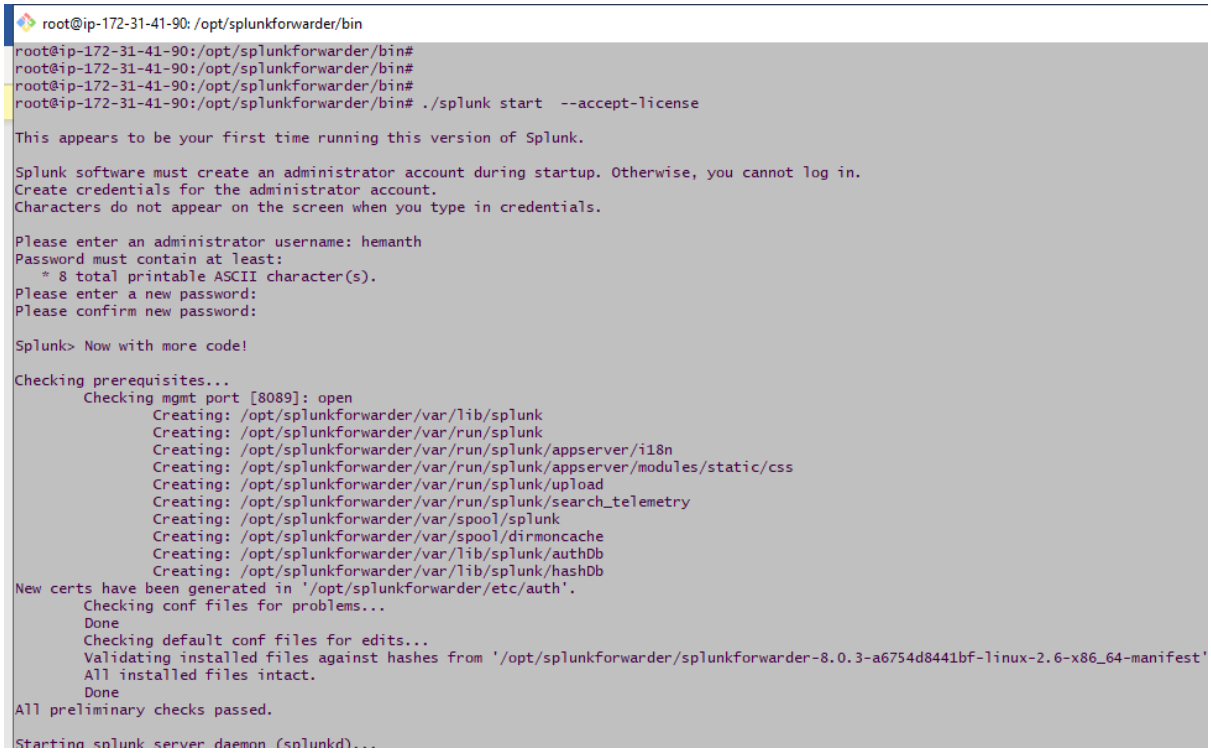
Step 9 :- we check the files that are included in bin folder .

$ ls



Step 10 :- for starting splunk-forwarder ,we have to accept license of the splunk. In it we have to create username and password**.**

$ ./splunk start --accept-license



Step 11 :- now we add our universal forwarder to your master using master-ip address in following command :

$./splunk add forward-server 13.127.149:8000

```
root@ip-172-31-41-90: /opt/splunkforwarder/bin
root@ip-172-31-41-90:/opt/splunkforwarder/bin#
root@ip-172-31-41-90:/opt/splunkforwarder/bin#
root@ip-172-31-41-90:/opt/splunkforwarder/bin#
root@ip-172-31-41-90:/opt/splunkforwarder/bin# ./splunk add forward-server 13.127.112.149:8000
Splunk username: hemanth
Password:
Added forwarding to: 13.127.112.149:8000.
root@ip-172-31-41-90:/opt/splunkforwarder/bin#
```

Step 12 :- now we have to add the log that has to be observed and indexed by splunk.

$ ./splunk add monitor /var/log/syslog -index main -sourcetype MyUFServerLogs

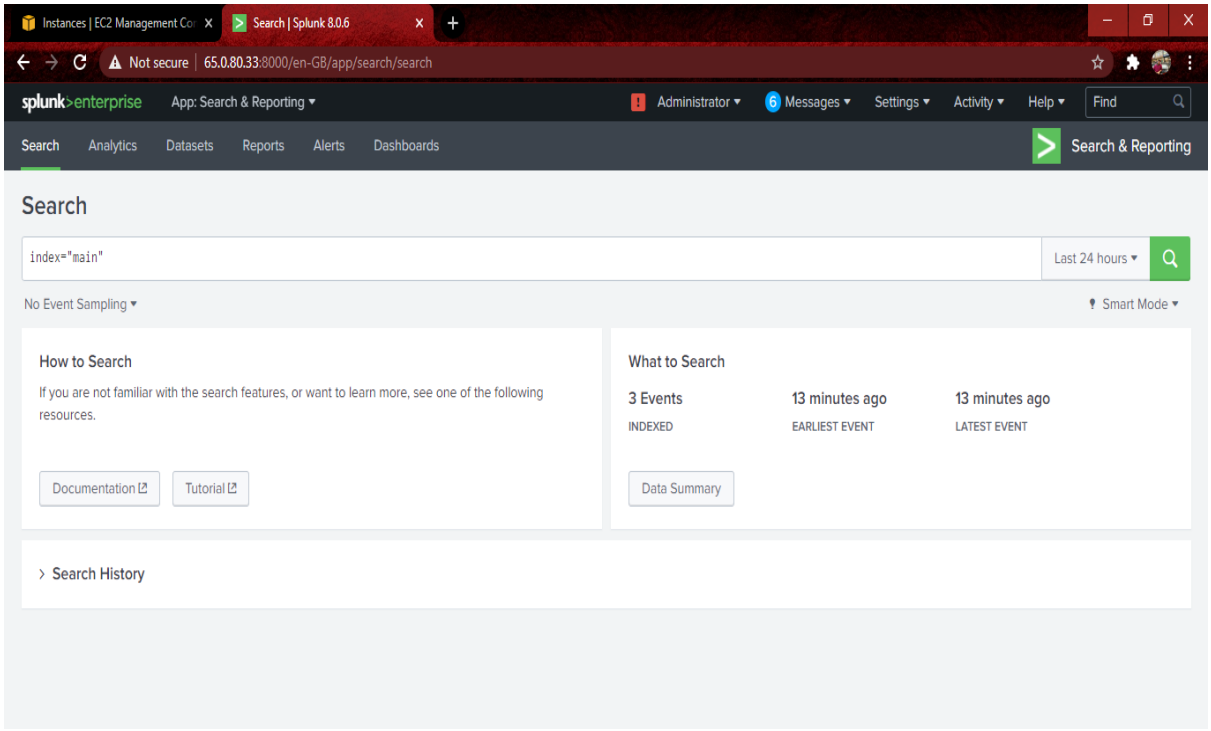
```
root@ip-172-31-41-90: /opt/splunkforwarder/bin
root@ip-172-31-41-90:/opt/splunkforwarder/bin#
root@ip-172-31-41-90:/opt/splunkforwarder/bin#
root@ip-172-31-41-90:/opt/splunkforwarder/bin#
root@ip-172-31-41-90:/opt/splunkforwarder/bin#
root@ip-172-31-41-90:/opt/splunkforwarder/bin# ./splunk add monitor /var/log/syslog -index main  -sourcetype MyUFServerLogs
Added monitor of '/var/log/syslog'.
root@ip-172-31-41-90:/opt/splunkforwarder/bin#
```

Step 13 :- Now open the master instance and type the enble command and listen on particular port.it will ask username and password.

$ ./splunk enable listen 9997

```
root@ip-172-31-46-43: /opt/splunk/bin
root@ip-172-31-46-43:/opt/splunk/bin#
root@ip-172-31-46-43:/opt/splunk/bin#
root@ip-172-31-46-43:/opt/splunk/bin#
root@ip-172-31-46-43:/opt/splunk/bin#
root@ip-172-31-46-43:/opt/splunk/bin# ./splunk enable listen 8000
Splunk username: hemanth
Password:
Parameter name: TCP port 8000 is not available.
root@ip-172-31-46-43:/opt/splunk/bin# ./splunk enable listen 9997
Listening for Splunk data on TCP port 9997.
root@ip-172-31-46-43:/opt/splunk/bin#
```

Step 14 :- Now open the browser and and open the login splunk using master ip and then go to search and type index=”main”.


Instances | EC2 Management Co
Search | Splunk 8.0.6
Not secure | 65.0.80.33:8000/en-GB/app/search/search
splunk>enterprise
App: Search & Reporting
Administrator
6 Messages
Settings
Activity
Help
Find
Search
Analytics
Datasets
Reports
Alerts
Dashboards
Search & Reporting
Search
index="main"
Last 24 hours
No Event Sampling
Smart Mode
How to Search
If you are not familiar with the search features, or want to learn more, see one of the following resources.
Documentation
Tutorial
What to Search
3 Events
INDEXED
13 minutes ago
EARLIEST EVENT
13 minutes ago
LATEST EVENT
Data Summary
Search History