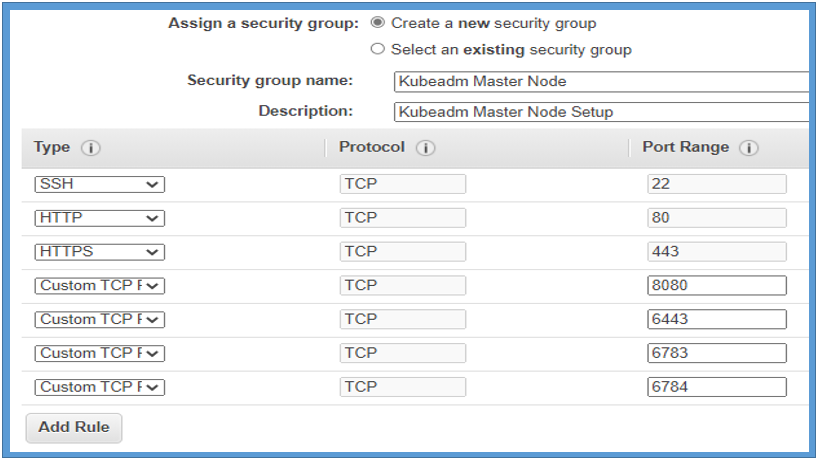
**kubernetes Configuration**

**I am creating 1 ubuntu instance in AWS Console:**

Click on Launch instances >> Ubuntu Server 20.04 LTS (HVM), SSD Volume Type >> Select t2.medium >> Next: Configuration and instance details >> Next: Add Storage >> Next: Add Tags (Give the name here) >> Next: Configure Security Groups >> Review and Launch >> Launch >> Create New Key Pair >> Download New Key Pair >> Launch Instances >> View Instances

Open Puttygen >> Click on Load >> Select downloaded .pem file >> Save Private Key (Give Name)

**Configure Security Groups :**



Open putty >> Login as ubuntu and configure server as mentioned below :

login as: ubuntu

ubuntu@ip-172-31-5-138:~$ sudo su - (To switch root user)

root@ip-172-31-5-138:~# apt list --upgradable (To see available updates)

root@ip-172-31-5-138:~# apt-get update (To update machine)

root@ip-172-31-5-138:~# apt-get upgrade (To upgrade all old software)

root@ip-172-31-5-138:~# apt-get install apt-transport-https (To install https for more security)

root@ip-172-31-5-138:~# apt-get install docker.io -y (to install the docker) by default package available

root@ip-172-31-5-138:~# docker --version

root@ip-172-31-5-138:~# systemctl start docker (To start the docker we have to run this command post installation)

root@ip-172-31-5-138:~# systemctl enable docker (To enable the docker for next time)

root@ip-172-31-5-138:~# systemctl status docker (To check docker status)

root@ip-172-31-5-138:~# sudo usermod -a -G docker ubuntu (To add ubuntu user in docker group)

root@ip-172-31-5-138:~# curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add (This is a key and required for intra cluster communications)

OK (Output)

Now we required an packages so we will create an file and then mention the link of packages inside of file)

root@ip-172-31-5-138:~# vi /etc/apt/sources.list.d/kubernetes.list (Creating file to import packges dabian familty as mentioned upper)

deb http://apt.kubernetes.io/ kubernetes-xenial main

root@ip-172-31-5-138:~# apt-get update (as we written the link inside of file so we need to run this command) (Unzip)

root@ip-172-31-5-138:~# sudo apt-get install policykit-1 (Packages required to restart nay service)

root@ip-172-31-5-138:~# apt-get install -y kubelet kubeadm kubectl kubernetes-cni (installation of kubelet kubeadm etc.)

For driver we have to create and write and file as mentioned below :

root@ip-172-31-5-138:~# vi /etc/docker/daemon.json

Press i

{

"exec-opts": ["native.cgroupdriver=systemd"]

}

:wq

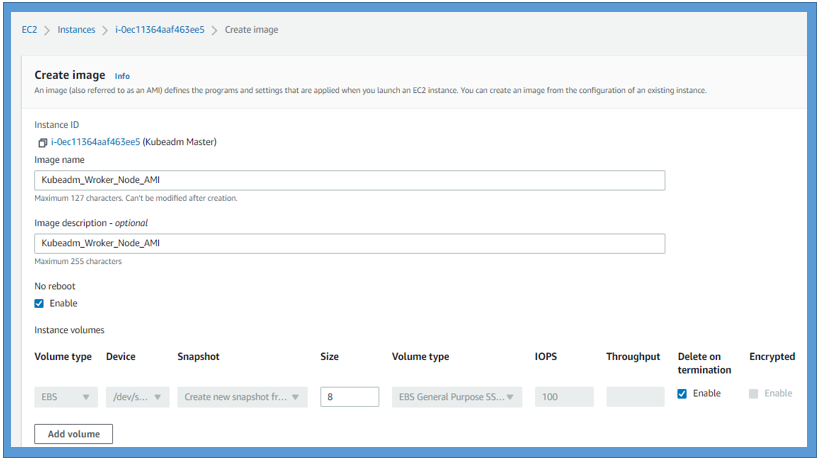
root@ip-172-31-5-138:~# systemctl daemon-reload

root@ip-172-31-5-138:~# systemctl restart docker

root@ip-172-31-5-138:~# systemctl restart kubelet

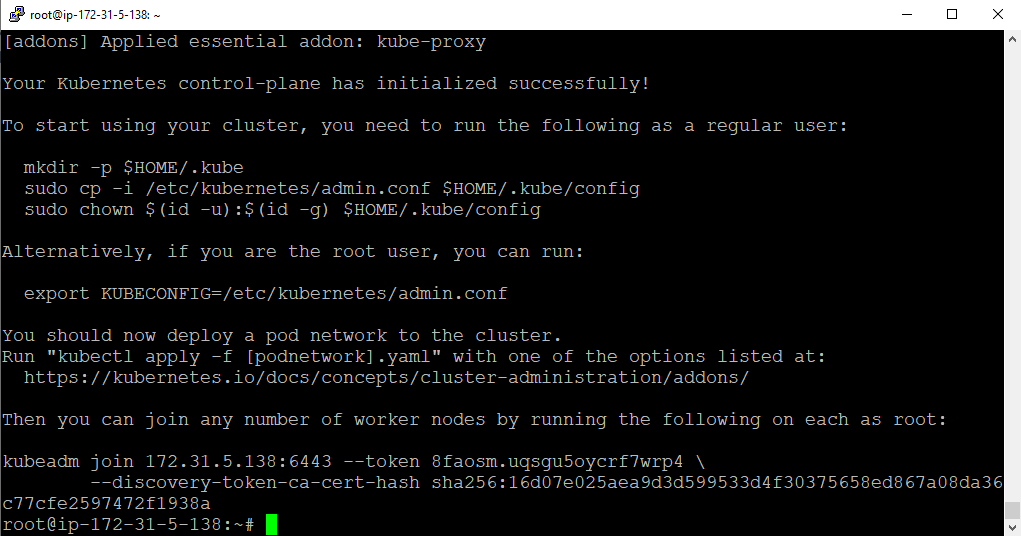
Now create an AMI of master node and we will use that AMI to create worker nodes…

Action >> Image and Template > Create Image >> Give name>> Create It.



**BOOTSTRAPPING THE MASTER NODE (IN MASTER)**

kubeadm init (to initiate the kubeadm)



root@ip-172-31-5-138:~# kubectl get nodes

The connection to the server localhost:8080 was refused - did you specify the right host or port?

**To get nodes we have to run below mentioned 3 commands one by one which we got after kubeadm init**

To start using your cluster, you need to run the following as a regular user:

mkdir -p $HOME/.kube

sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config

sudo chown $(id -u):$(id -g) $HOME/.kube/config

Now need to switch normal user to run above commands:

root@ip-172-31-5-138:~# sudo su - ubuntu (To switch to normal user)

ubuntu@ip-172-31-5-138:~$ kubectl get nodes

The connection to the server localhost:8080 was refused - did you specify the right host or port?

Still getting no nodes. Now entering above 3 commands one by one…

ubuntu@ip-172-31-5-138:~$ mkdir -p $HOME/.kube

ubuntu@ip-172-31-5-138:~$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config

ubuntu@ip-172-31-5-138:~$ sudo chown $(id -u):$(id -g) $HOME/.kube/config

ubuntu@ip-172-31-5-138:~$ kubectl get nodes

NAME STATUS ROLES AGE VERSION

ip-172-31-5-138 NotReady control-plane,master 19m v1.22.2

ubuntu@ip-172-31-5-138:~$ kubectl get nodes --all-namespaces

NAME STATUS ROLES AGE VERSION

ip-172-31-5-138 NotReady control-plane,master 21m v1.22.2

Since there is no network so we are not able to see any thing apart from NotReady.

ubuntu@ip-172-31-5-138:~$ kubectl get pods --all-namespaces

NAMESPACE NAME READY STATUS RESTARTS AGE

kube-system coredns-78fcd69978-9tg4v 0/1 Pending 0 22m

kube-system coredns-78fcd69978-pj5fx 0/1 Pending 0 22m

kube-system etcd-ip-172-31-5-138 1/1 Running 0 22m

kube-system kube-apiserver-ip-172-31-5-138 1/1 Running 0 22m

kube-system kube-controller-manager-ip-172-31-5-138 1/1 Running 0 22m

kube-system kube-proxy-2nmdn 1/1 Running 0 22m

kube-system kube-scheduler-ip-172-31-5-138 1/1 Running 0 22m

Still since there is no network so 2 core dns not working and in pending status.

**NOW we will install CNI network(Container Network Interface (CNI) )**

To get done above we will switch to root user and allow the IP Table

ubuntu@ip-172-31-5-138:~$ sudo su - (For switch user)

root@ip-172-31-5-138:~# sysctl net.bridge.bridge-nf-call-iptables=1 (Allow the IP Table)

net.bridge.bridge-nf-call-iptables = 1 (output of upper command)

ubuntu@ip-172-31-5-138:/root$ sudo su - ubuntu (For switch normal user)

ubuntu@ip-172-31-5-138:~$ cd /root

ubuntu@ip-172-31-5-138:/root$ kubectl version | base64 | tr -d '\n'

ubuntu@ip-172-31-5-138:/root$

**Now we will export this key to kubever:**

(Full key as above)==ubuntu@ip-172-31-5-138:/root$ export kubever=$(kubectl version | base64 | tr -d '\n')

**Now we will apply the changes to CNI weave network.**

ubuntu@ip-172-31-5-138:/root$ kubectl apply -f "https://cloud.weave.works/k8s/net?k8s-version=$kubever"

serviceaccount/weave-net created

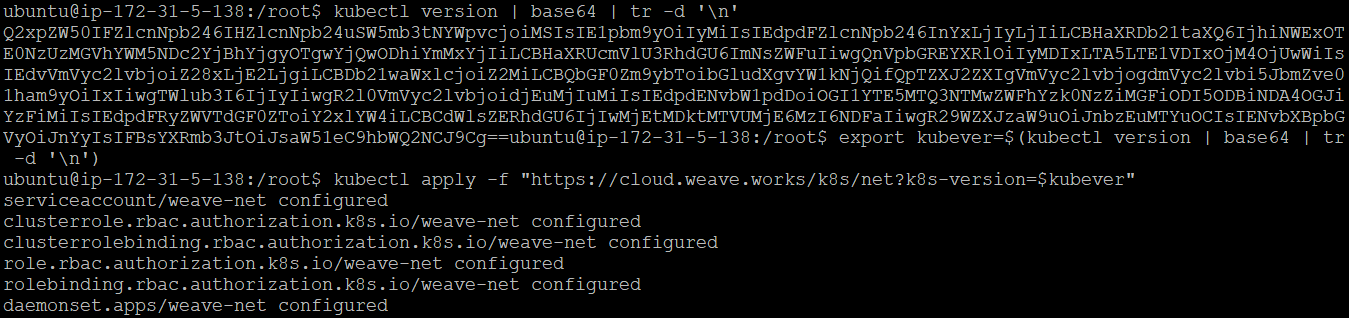
clusterrole.rbac.authorization.k8s.io/weave-net created

clusterrolebinding.rbac.authorization.k8s.io/weave-net created

role.rbac.authorization.k8s.io/weave-net created

rolebinding.rbac.authorization.k8s.io/weave-net created

daemonset.apps/weave-net created



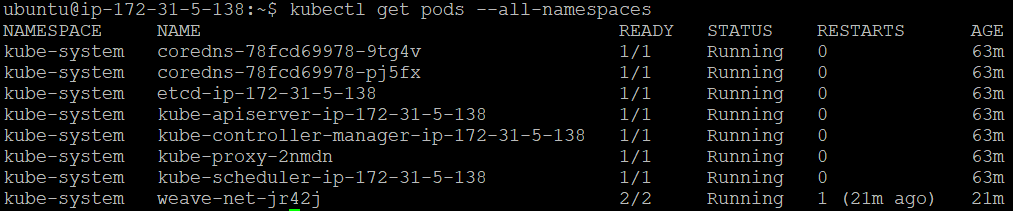
**Now we will check for nodes status again by kubectl command**

ubuntu@ip-172-31-5-138:~$ kubectl get nodes

NAME STATUS ROLES AGE VERSION

ip-172-31-5-138 Ready control-plane,master 48m v1.22.2

ubuntu@ip-172-31-5-138:~$ kubectl get pods --all-namespaces



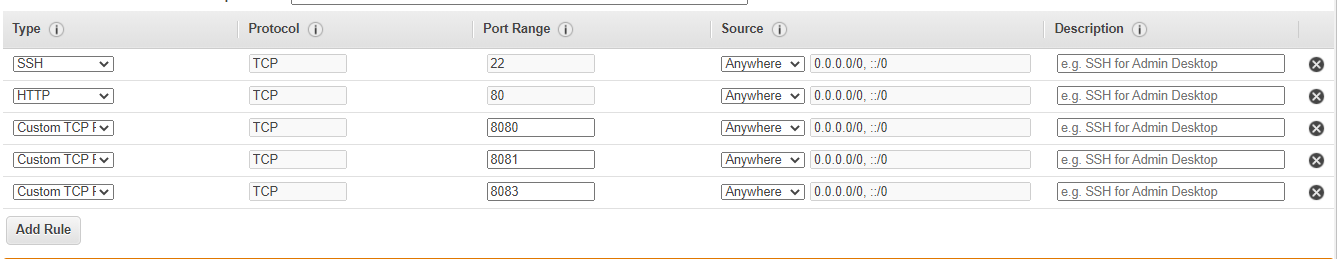
Its grate… Now everything is fine from master end.

**CONFIGURE WORKER NODES (IN NODES)**

Now we will work in slave side….

Create 2 salve with the help of created AMI :

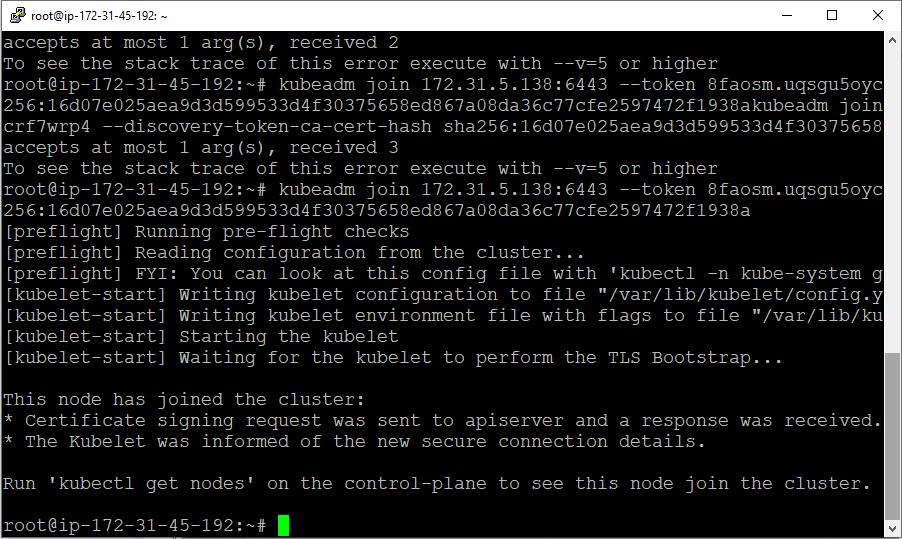
Click on AMI >> Launch > Next > configure security group >> Next >> Next etc.



**COPY JOIN CODE PROVIDED FROM MASTER NODE TO WORKER NODE:**

kubeadm join 172.31.5.138:6443 --token 8faosm.uqsgu5oycrf7wrp4 --discovery-token-ca-cert-hash sha256:16d07e025aea9d3d599533d4f30375658ed867a08da36c77cfe2597472f1938a

**(kubeadm join 172.31.5.138:6443 --token 8faosm.uqsgu5oycrf7wrp4 --discovery-token-ca-cert-hash sha256:16d07e025aea9d3d599533d4f30375658ed867a08da36c77cfe2597472f1938a)**



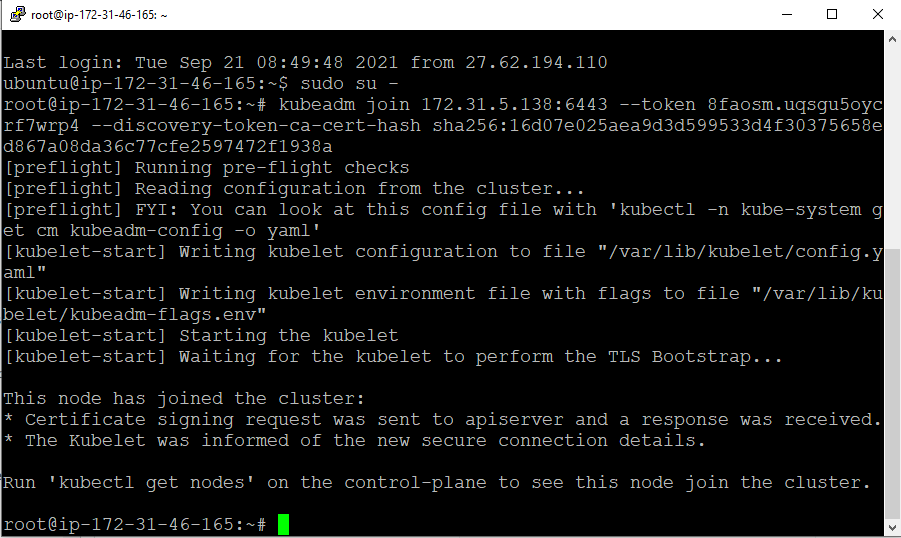
ubuntu@ip-172-31-5-138:~$ kubectl get nodes

NAME STATUS ROLES AGE VERSION

ip-172-31-45-192 Ready <none> 2m34s v1.22.2

ip-172-31-5-138 Ready control-plane,master 87m v1.22.2

Do the same for 2nd worker Node..



ubuntu@ip-172-31-5-138:~$ kubectl get nodes

NAME STATUS ROLES AGE VERSION

ip-172-31-45-192 Ready <none> 10m v1.22.2

ip-172-31-46-165 Ready <none> 5m13s v1.22.2

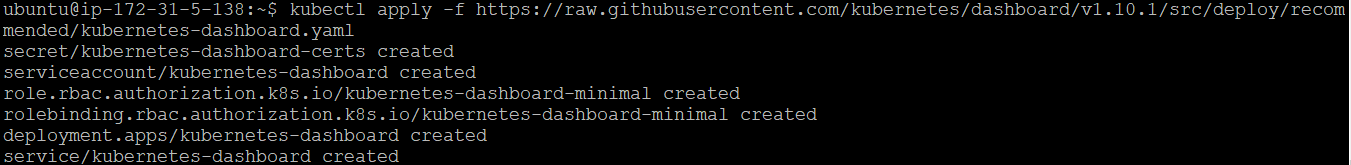
ip-172-31-5-138 Ready control-plane,master 96m v1.22.2

We can add n number of node here by above procedure..

**Install the Dashboard :**

Its always good to have the dashboard running as you can get a good visual view of your deployment. We can install the dashboard by below command.

ubuntu@ip-172-31-5-138:~$ kubectl apply -f https://raw.githubusercontent.com/kubernetes/dashboard/v1.10.1/src/deploy/recommended/kubernetes-dashboard.yaml



To access the Dashboard , Lets create new user using Service Account mechanism , grant admin permission to the user and log in to dashboard using bearer token tied to this user.

**Create Service Account & Cluster Role Binding:**

**Creating Service Account:**

ubuntu@ip-172-31-5-138:~$ vi service.yaml

Press i button

apiVersion: v1

kind: ServiceAccount

metadata:

name: admin-user

namespace: kube-system

:wq

ubuntu@ip-172-31-5-138:~$ kubectl apply -f service.yaml (Creating service account by applying)

serviceaccount/admin-user created

**Creating Cluster Role Binding:**

ubuntu@ip-172-31-5-138:~$ vi rolebinding.yaml

Press i

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: kube-system

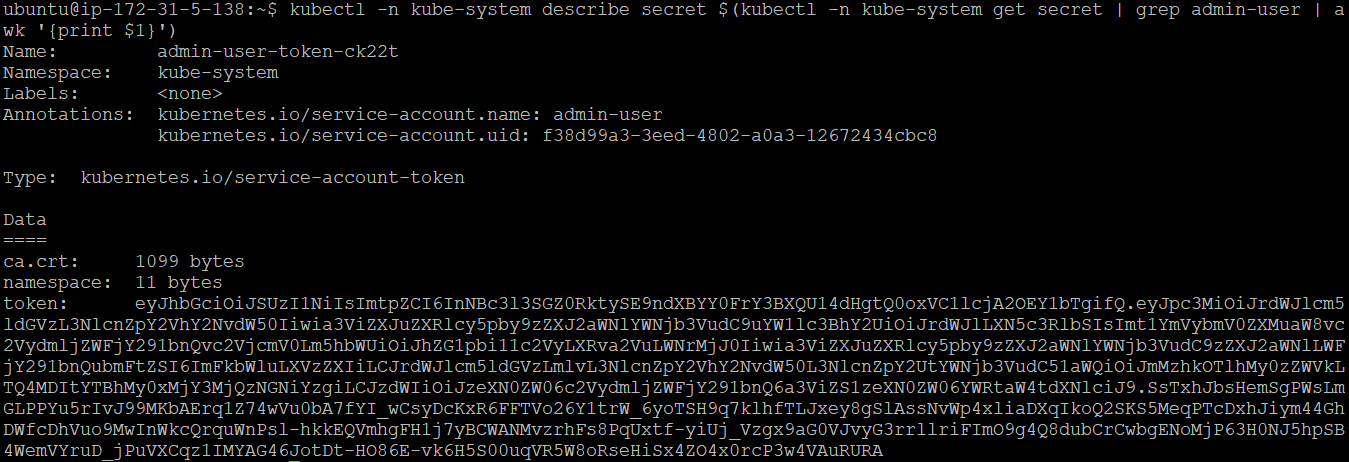
:wq

ubuntu@ip-172-31-5-138:~$ kubectl apply -f rolebinding.yaml (Creating rolebinding by applying)

clusterrolebinding.rbac.authorization.k8s.io/admin-user created

**Now we will create a user if we want to login as a admin user**

ubuntu@ip-172-31-5-138:~$ kubectl -n kube-system describe secret $(kubectl -n kube-system get secret | grep admin-user | awk '{print $1}')



**Once get the output copy the token paste into Token section in the UI page.**

**Dashboard Config:** By default Dashboard won’t be visible. We have to follow the below process :

We should enable to allow access from external network.

ubuntu@ip-172-31-5-138:~$ nohup kubectl proxy --address 0.0.0.0 --acceot-host '.\*' &

[1] 51211

ubuntu@ip-172-31-5-138:~$ nohup: ignoring input and appending output to 'nohup.out'

ubuntu@ip-172-31-5-138:~$ kubectl -n kube-system get service kubernetes-dashboard

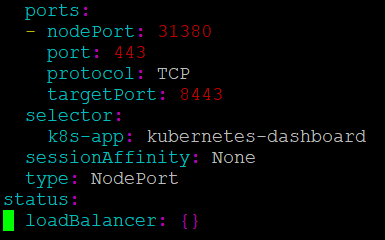
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE

kubernetes-dashboard ClusterIP 10.101.173.159 <none> 443/TCP 46m

**Here we need to edit ClusterIP to NodePort see below**

ubuntu@ip-172-31-5-138:~$ kubectl -n kube-system edit service kubernetes-dashboard

Press i



ClusterIp replaced by NodePort

**Change above ClusterIP to NodePort**

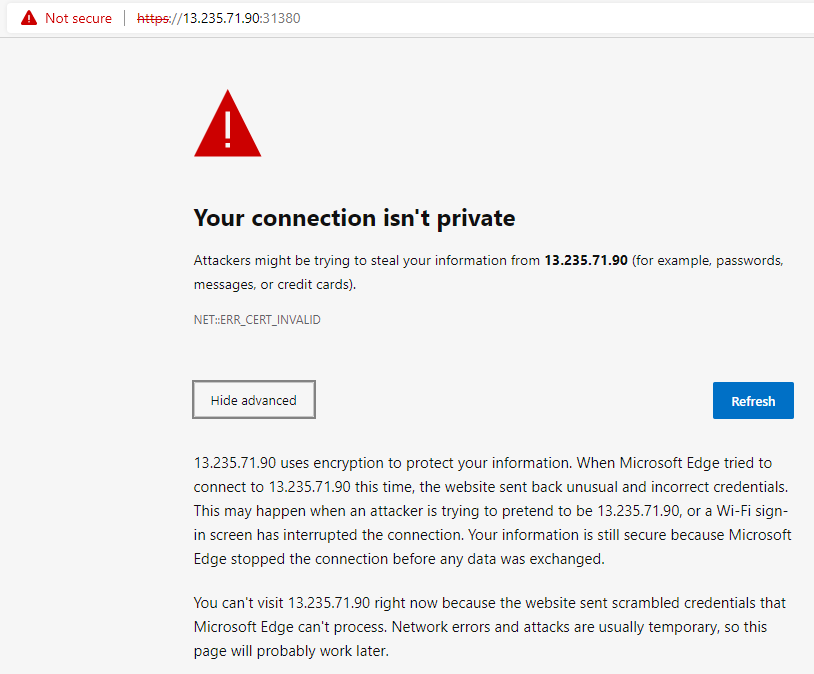
service/kubernetes-dashboard edited

ubuntu@ip-172-31-5-138:~$ kubectl -n kube-system get service kubernetes-dashboard (to see the port to access outer)

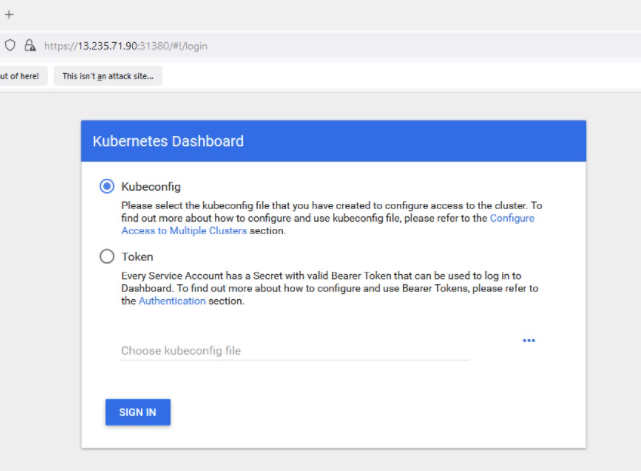
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE

kubernetes-dashboard NodePort 10.101.173.159 <none> 443:**31380**/TCP 56m

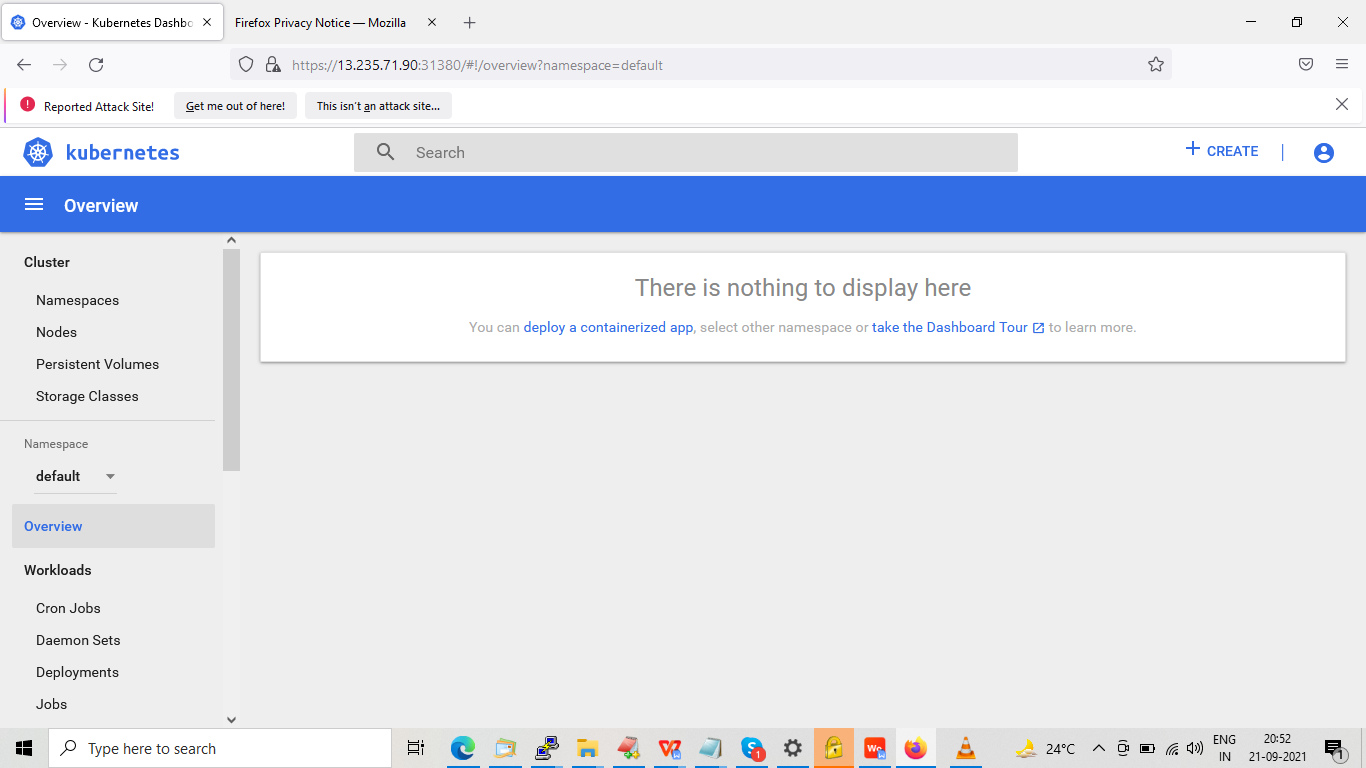
Now enter [https://13.235.71.90:](https://13.235.71.90:31380/)**[31380](https://13.235.71.90:31380/)**[/](https://13.235.71.90:31380/) in Edge/Google Chrome.



**Due to security reason we can’t do login so we are ready to use the same link in mozilla firefox and accept risk.**



Copy token and paste here and save for next time…



ubuntu@ip-172-31-5-138:~$ kubectl run test --image=nginx

pod/test created

ubuntu@ip-172-31-5-138:~$ kubectl get pods

NAME READY STATUS RESTARTS AGE

test 1/1 Running 0 119s

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*END\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***