Hands-On: Kubernetes Dashboard

Step 1:

Launch 2 instances with the following configuration: ubuntu 20.04 ami, t2.medium, sg: all traffic. ubuntu 20.04 ami, t2.micro , sg: all traffic
To Install Kubernetes use the following commands:

On Master and Worker node:

```
sudo su
apt-get update
apt-get install docker.io -y
service docker restart
curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | apt-key
add -
echo "deb http://apt.kubernetes.io/ kubernetes-xenial main"
>/etc/apt/sources.list.d/kubernetes.list
apt-get update
apt install kubeadm=1.20.0-00 kubectl=1.20.0-00 kubelet=1.20.0-00 –y
```

Step 2: On both master and worker nodes run the above command:

2.1. sudo su
2.2. create a script file kubernetes.sh
2.3. to execute the script file: bash kubernetes.sh

On Master:

Step 3:

Creating cluster:

Initializing kubeadm on master using:

```
kubeadm init --pod-network-cidr=192.168.0.0/16
```

```
[bootstrap-token] configured RBAC rules to allow the csrapprover controller automatically approve CSRs from a Node Bootstrap Token
[bootstrap-token] configured RBAC rules to allow certificate rotation for all node client certificates in the cluster
[bootstrap-token] Creating the "cluster-info" ConfigMap in the "kube-public" namespace
[kubelet-finalize] Updating "/etc/kubernetes/kubelet.conf" to point to a rotatable kubelet client certificate and key
[addons] Applied essential addon: CoreDNS
[addons] Applied essential addon: kube-proxy

Your Kubernetes control-plane has initialized successfully!

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

Alternatively, if you are the root user, you can run:

  export KUBECONFIG=/etc/kubernetes/admin.conf

You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
  https://kubernetes.io/docs/concepts/cluster-administration/addons/

Then you can join any number of worker nodes by running the following on each as root:

kubeadm join 172.31.18.147:6443 --token rb42wf.rli4srhyw2cark8s \
    --discovery-token-ca-cert-hash sha256:bc23c6c2902c4c5f514ad86075b474e9c5b7060cd42e275499297c959aa8bd41
root@ip-172-31-18-147:/home/ubuntu#
```

Copy the token and paste it into the worker node

```
2 master    3 worker
root@ip-172-31-13-193:/home/ubuntu# kubeadm join 172.31.18.147:6443 --token rb42wf.rli4srhyw2cark8s \
>     --discovery-token-ca-cert-hash sha256:bc23c6c2902c4c5f514ad86075b474e9c5b7060cd42e275499297c959aa8bd41
[preflight] Running pre-flight checks
        [WARNING IsDockerSystemdCheck]: detected "cgroupfs" as the Docker cgroup driver. The recommended driver is "systemd". Pleas
e follow the guide at https://kubernetes.io/docs/setup/cri/
        [WARNING SystemVerification]: this Docker version is not on the list of validated versions: 20.10.7. Latest validated versi
on: 19.03
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'
[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-172-31-13-193:/home/ubuntu#
```

Step 4:

On Master:

We need to run the below commands:

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

Note: In case we want to retrieve the join token use the below-mentioned command.

kubeadm token create --print-join-command

Step 5:

On Master:

kubectl apply -f https://docs.projectcalico.org/manifests/calico.yaml
kubectl apply –f
https://raw.githubusercontent.com/kubernetes/ingress-nginx/controller-v0.
49.0/deploy/static/provider/baremetal/deploy.yaml

To list all nodes :

kubectl get nodes

Our Kubernetes installation and configuration is complete.

It shows the nodes but the status is not ready
because we have not installed the network plugin.

To install network plugin, run the below commands:

kubectl apply -f https://docs.projectcalico.org/manifests/calico.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/controllerv0.49.0/deploy/static/provider/baremetal/deploy.yaml

Check the nodes for its state after installing network plugins.

kubectl get nodes

Thus, we have successfully installed Kubernetes.

Step 6:

Run the below command tocreate a dashboard:

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

Then edit the service:

kubectl edit service kubernetes-dashboard -n kubernetes-dashboard

```
# 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":
s-dashboard"},"spec":{"ports":[{"port":443,"targetPort":8443}],"selector":{"k8s-app":"kuber
  creationTimestamp: "2022-01-13T07:16:45Z"
  labels:
    k8s-app: kubernetes-dashboard
  name: kubernetes-dashboard
  namespace: kubernetes-dashboard
  resourceVersion: "1539"
  uid: 01f6f13a-d1bf-45b0-afc4-ca9713168cac
spec:
  clusterIP: 10.105.231.218
  clusterIPs:
  - 10.105.231.218
  ipFamilies:
  - IPv4
  ipFamilyPolicy: SingleStack
  ports:
  - port: 443
    protocol: TCP
    targetPort: 8443
  selector:
    k8s-app: kubernetes-dashboard
  sessionAffinity: None
  type: ClusterIP
status:
```

Change ClusterIP to NodePort

Note: We need to change the type from ClusterIP to NodePort. The below given image contains the modified service file.

```
apiVersion: v1
kind: Service
metadata:
  annotations:
    kubectl.kubernetes.io/last-applied-configuration: |
      {"apiVersion":"v1","kind":"Service","metadata":{"annotations":{},"labels":{"k8s-app":
s-dashboard"},"spec":{"ports":[{"port":443,"targetPort":8443}],"selector":{"k8s-app":"kuber
  creationTimestamp: "2022-01-13T07:16:45Z"
  labels:
    k8s-app: kubernetes-dashboard
  name: kubernetes-dashboard
  namespace: kubernetes-dashboard
  resourceVersion: "1997"
  uid: 01f6f13a-d1bf-45b0-afc4-ca9713168cac
spec:
  clusterIP: 10.105.231.218
  clusterIPs:
  - 10.105.231.218
  externalTrafficPolicy: Cluster
  ipFamilies:
  - IPv4
  ipFamilyPolicy: SingleStack
  ports:
  - nodePort: 31707
    port: 443
    protocol: TCP
    targetPort: 8443
  selector:
    k8s-app: kubernetes-dashboard
  sessionAffinity: None
  type: NodePort
status:
  loadBalancer: {}
```

Note: The editor used is vim. Therefore to save and exit we need to press ESC and then: wq

```
root@ip-172-31-23-197:/home/ubuntu# kubectl edit service kubernetes-dashboard -n kubernetes-dashboard
service/kubernetes-dashboard edited
```

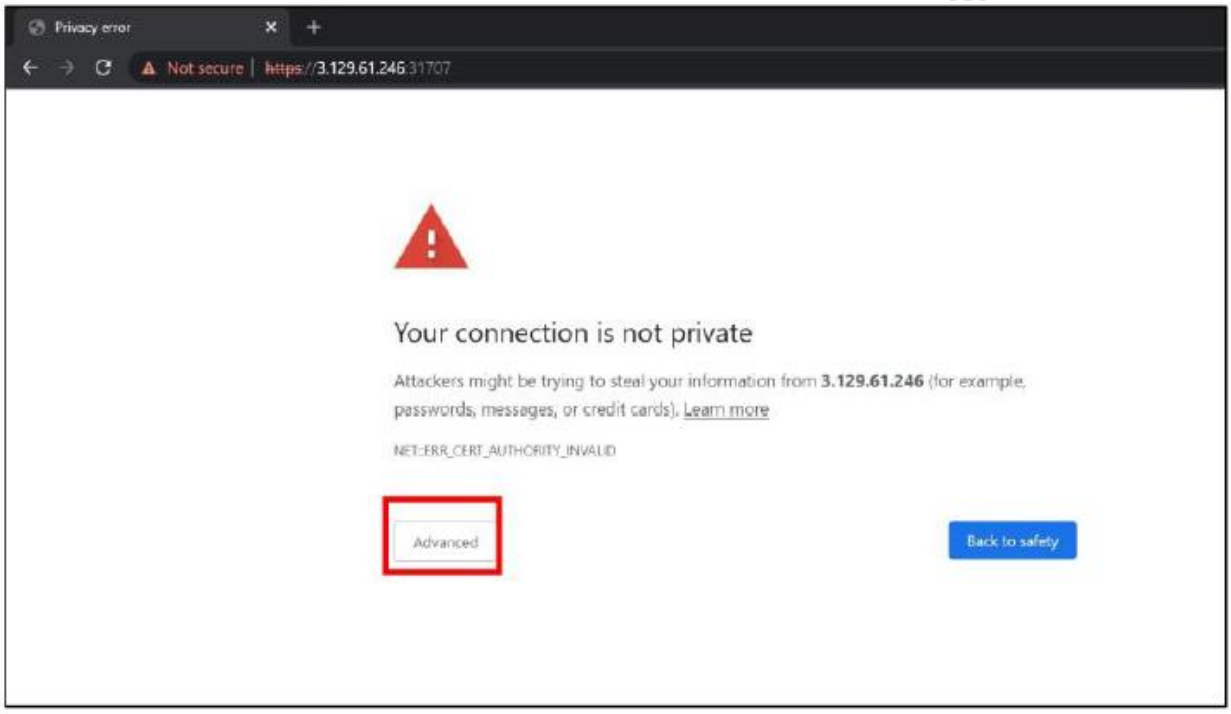
kubectl get svc -n kubernetes-dashboard

Step 7:

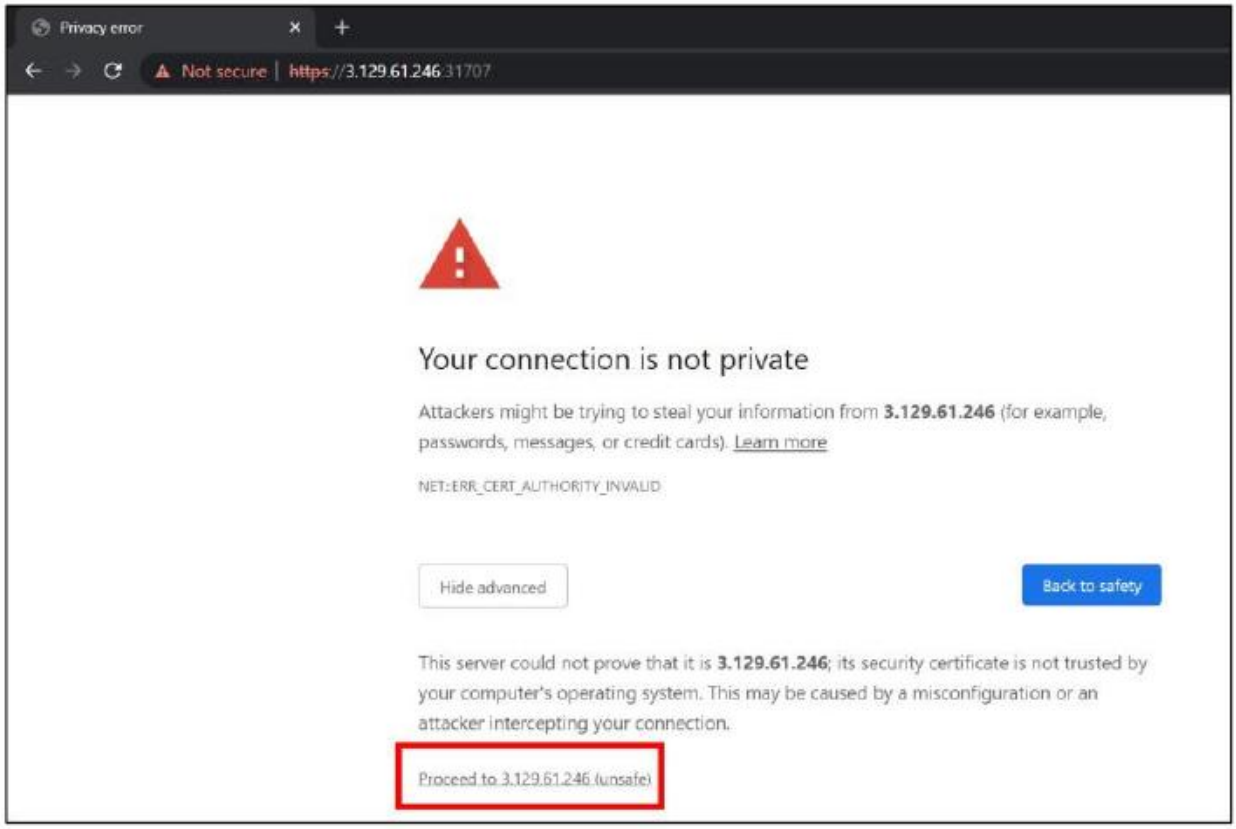
Now go to a browser and paste the ip along with the port

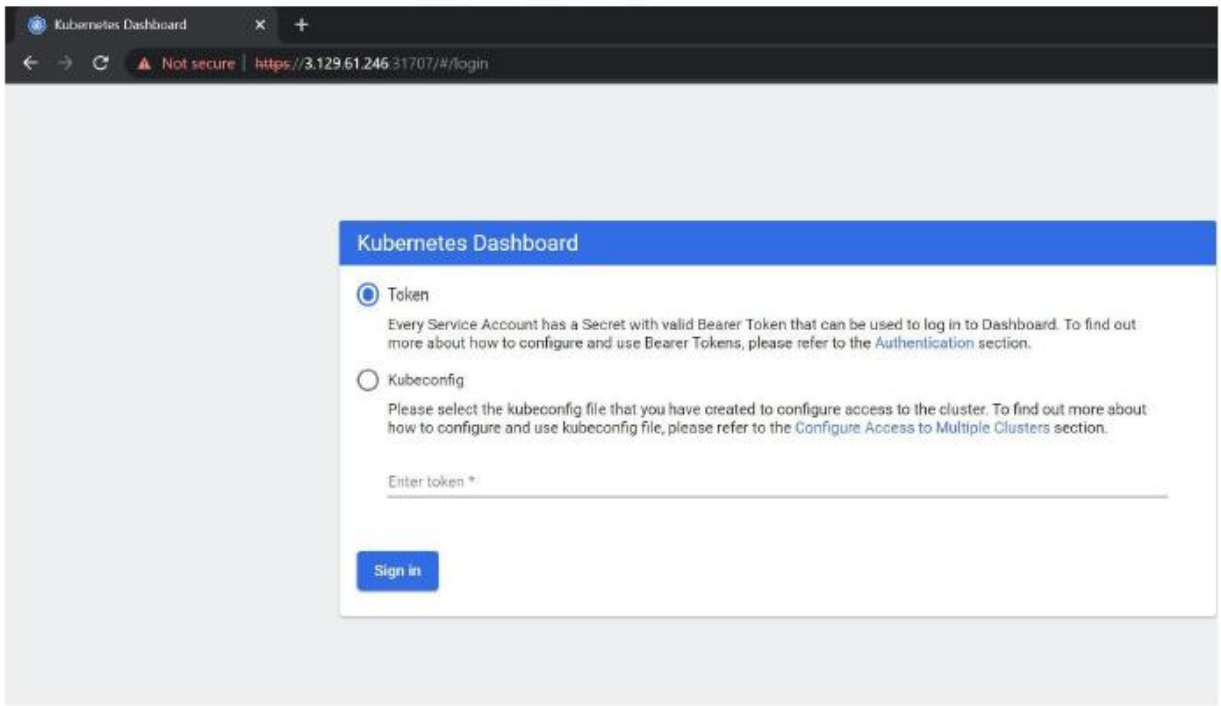
https://<ip-of-master>

We need to click on Advance option to access the webpage



Click on the Proceed option to open the dashboard


Privacy error
Not secure | https://3.129.61.246:31707
Your connection is not private
Attackers might be trying to steal your information from 3.129.61.246 (for example, passwords, messages, or credit cards). Learn more
NET::ERR_CERT_AUTHORITY_INVALID
Hide advanced
Back to safety
This server could not prove that it is 3.129.61.246; its security certificate is not trusted by your computer's operating system. This may be caused by a misconfiguration or an attacker intercepting your connection.
Proceed to 3.129.61.246 (unsafe)


Kubernetes Dashboard
Not secure | https://3.129.61.246:31707/#/login
Kubernetes Dashboard
Token
Every Service Account has a Secret with valid Bearer Token that can be used to log in to Dashboard. To find out more about how to configure and use Bearer Tokens, please refer to the Authentication section.
Kubeconfig
Please select the kubeconfig file that you have created to configure access to the cluster. To find out more about how to configure and use kubeconfig file, please refer to the Configure Access to Multiple Clusters section.
Enter token *
Sign in

Step 8:

We need to create a service account:

To create service account run the below command:

```
kubectl create serviceaccount cluster-admin-dashboard-sa
```

To bind clusterAdmin role to the service account use the below command:

```
kubectl create clusterrolebinding cluster-admin-dashboard-sa \
--clusterrole=cluster-admin \
--serviceaccount=default:cluster-admin-dashboard-sa
```

To parse the token run the below command:

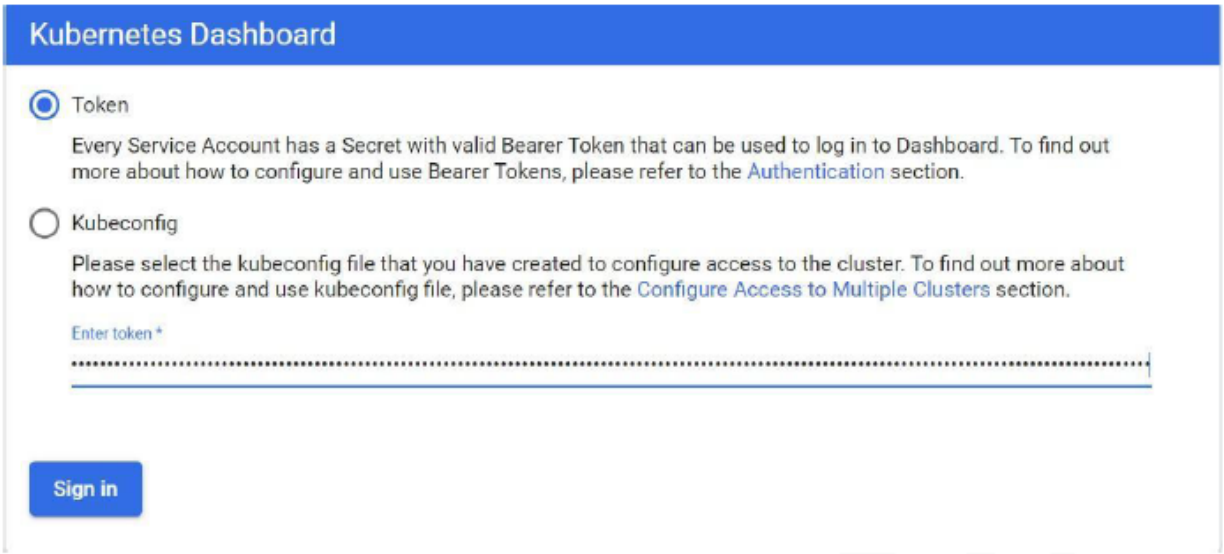
```
TOKEN=$(kubectl describe secret $(kubectl -n kube-system get secret | awk '/^clusteradmin-dashboardsa-token-/{print $1}') | awk '$1=="token:"{print $2}')
```

Then we need to run the below command:

```
echo $TOKEN
```

```
root@ip-172-31-23-197:/home/ubuntu# echo $TOKEN
eyJhbGciOiJSUzI1NiIsImtpZCI6Ikx0MzdxclRxZG1zcWhoNnNMTm5yTXN6MEZ5ZGRyT3dCc0dSVk1uSS1UVkEifQ.eyJpc3MiOiJrdWJlcm5ldGVzL3NlcnZ
ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9uYW1lc3BhY2UiOiJkZWZhdWx0Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9zZWNyZXQubmFtZSI6
tZGFzaGJvYXJkLXNhLXRva2VuLWc2cmp6Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9zZXJ2aWNlLWFjY291bnQudWlkIjoiYWM2MmMzNTItMzA3NC00ZWExLThlMzUtODM1Njg0YWZkZTQ1Ii
nNlcnZpY2VhY2NvdW50OmRlZmF1bHQ6Y2x1c3Rlci1hZG1pbi1kYXNoYm9hcmQtc2EifQ.m-m9wtaRNgJb74CN-d20RD7BsCUvlh8KeAq-H_TobJf-oGvpPJxd
CFIJQRM6CywQvH3yIoiw0kUrMXUwkZbT9LO30ap-_kpTiQcu28SsKPhb8JOltFIBdAvC6Zxo_2WUlYirtM1pscjlwQQNURhht8vU3YKO06vCC6yK8TNOC8gjym
xDSO8ORTDmm4JQ8_Tzy17ZhlDzsWWBARB_3m7kas2m_ZXgpJ-vUgL4XdOPg5qIei6BleE8UDDbOV-PKY4hfJZ17t483piQm3CxQ_-uVS_DsS9P_o5h3DevhSRI
JSUzI1NiIsImtpZCI6Ikx0MzdxclRxZG1zcWhoNnNMTm5yTXN6MEZ5ZGRyT3dCc0dSVk1uSS1UVkEifQ.eyJpc3MiOiJrdWJlcm5ldGVzL3NlcnZpY2VhY2Nvd
cy5pby9zZXJ2aWNlYWNjb3VudC9uYW1lc3BhY2UiOiJkZWZhdWx0Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9zZWNyZXQubmFtZSI6ImRlZmF1bH
CJrdWJlcm5ldGVzLmlvL3NlcnZpY2VhY2NvdW50L3NlcnZpY2UtYWNjb3VudC5uYW1lIjoiZGVmYXVsdCIsImt1YmVybmV0ZXMuaW8vc2Vydmljj ZWFjY291bnQ
50LnVpZCI6IjZjOWUyMDY4LWI4MzItNGMwOS04YWYyLWM5MWRiYjE1MDk4ZiIsInN1YiI6InN5c3RlbTpzZXJ2aWNlYWNjb3VudDpkZWZhdWx0OmRlZmF1bHQi
XCrZBli14ztVArzK3Ymo3CsOCIH0rKtrQrxQoxHH8-k3Tj1_WYabVsm-ND-r8ewW398IghTZ1Mu_K9kXwCZ3XT--P6OeYZFYvCzkuuzqJ8cdji1i9dNXFMa6lb
FnV-0Xx7krrao_jiviIb7fUXw7cNp8rosVaH2u4tfbNXWNyTneG_KV1pY5G_nVY7l3iS3uXBfywR_8DkI8cEUaUNN1OePskKnJLGMRERnxPdH2EDUaaZ-8Kx4a
4blgHIQtjLwjwq0fywPR5r6RdCjh8X7c55DfYg5rG4Eg
root@ip-172-31-23-197:/home/ubuntu#
```

Copy the token and paste in the Kubernetes dashboard.



Then click on Sign in. The dashboard is created.