Lesson 03 Demo 05

Launching the Kubernetes Dashboard

Objective: To deploy the Kubernetes dashboard to facilitate the management and troubleshooting of cluster resources and applications

Tools required: kubeadm, kubectl, kubelet, and containerd

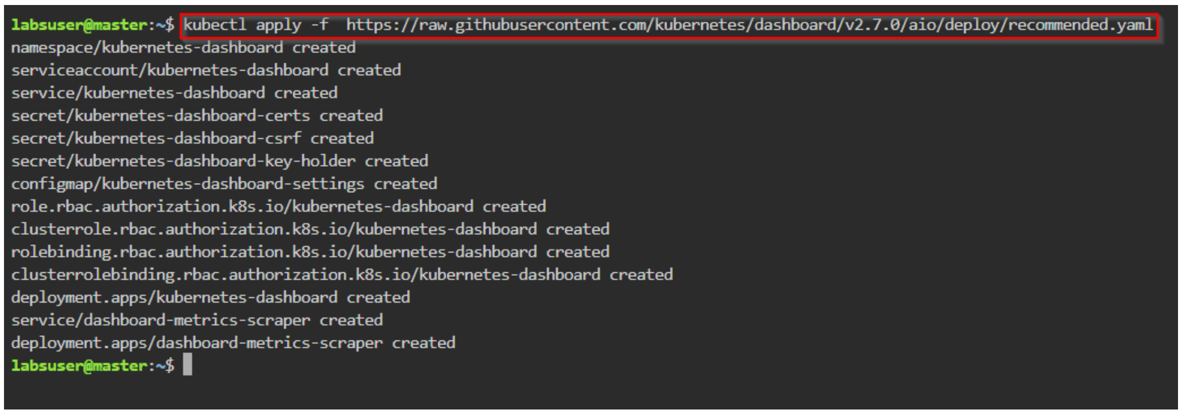
Prerequisites: A Kubernetes cluster should already be set up (refer to the steps provided in Lesson 02, Demo 01 for guidance).

Steps to be followed:

1. Implement the dashboard deployment
2. Validate the pod, service, and deployment creation
3. Confirm the dashboard service type
4. Access the master node IP
5. Log into the service dashboard
6. Access the Kubernetes dashboard

Step 1: Implement the dashboard deployment

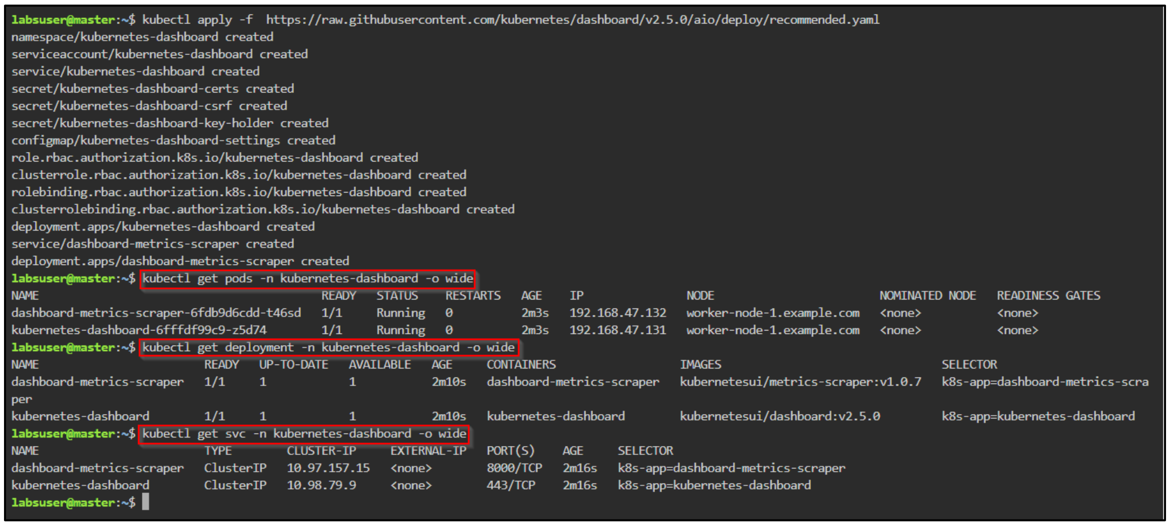
1.1 The dashboard user interface is not deployed by default. To deploy it, run the following command:
kubectl apply -f https://raw.githubusercontent.com/kubernetes/dashboard/v2.7.0/aio/deploy/recommended.yaml



```
labsuser@master:~$ kubectl apply -f  https://raw.githubusercontent.com/kubernetes/dashboard/v2.7.0/aio/deploy/recommended.yaml
namespace/kubernetes-dashboard created
serviceaccount/kubernetes-dashboard created
service/kubernetes-dashboard created
secret/kubernetes-dashboard-certs created
secret/kubernetes-dashboard-csrf created
secret/kubernetes-dashboard-key-holder created
configmap/kubernetes-dashboard-settings created
role.rbac.authorization.k8s.io/kubernetes-dashboard created
clusterrole.rbac.authorization.k8s.io/kubernetes-dashboard created
rolebinding.rbac.authorization.k8s.io/kubernetes-dashboard created
clusterrolebinding.rbac.authorization.k8s.io/kubernetes-dashboard created
deployment.apps/kubernetes-dashboard created
service/dashboard-metrics-scraper created
deployment.apps/dashboard-metrics-scraper created
labsuser@master:~$
```

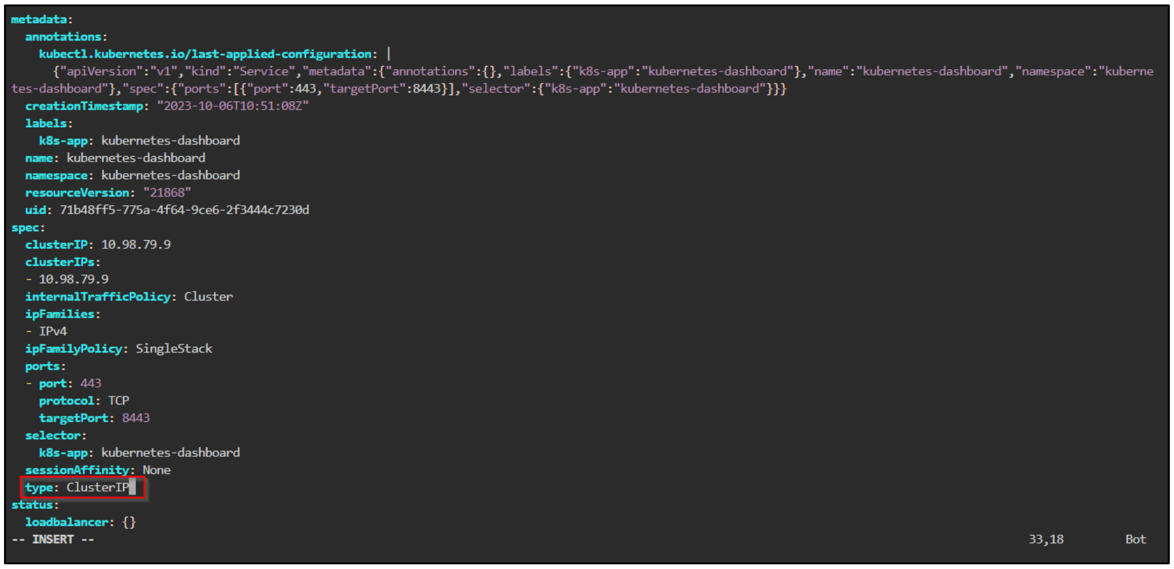
Step 2: Validate the pod, service, and deployment creation

2.1 Enter the following commands to verify if the pods, services, and deployments have been created:
kubectl get pods -n kubernetes-dashboard -o wide
kubectl get deployment -n kubernetes-dashboard -o wide
kubectl get svc -n kubernetes-dashboard -o wide

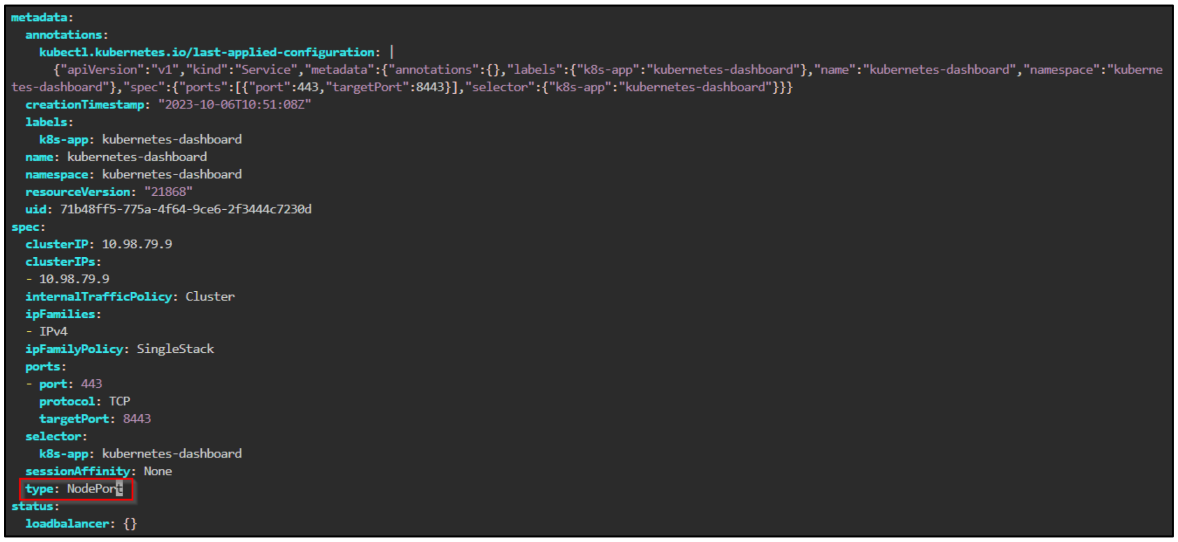


```
labsuser@master:~$ kubectl apply -f  https://raw.githubusercontent.com/kubernetes/dashboard/v2.5.0/aio/deploy/recommended.yaml
namespace/kubernetes-dashboard created
serviceaccount/kubernetes-dashboard created
service/kubernetes-dashboard created
secret/kubernetes-dashboard-certs created
secret/kubernetes-dashboard-csrf created
secret/kubernetes-dashboard-key-holder created
configmap/kubernetes-dashboard-settings created
role.rbac.authorization.k8s.io/kubernetes-dashboard created
clusterrole.rbac.authorization.k8s.io/kubernetes-dashboard created
rolebinding.rbac.authorization.k8s.io/kubernetes-dashboard created
clusterrolebinding.rbac.authorization.k8s.io/kubernetes-dashboard created
deployment.apps/kubernetes-dashboard created
service/dashboard-metrics-scraper created
deployment.apps/dashboard-metrics-scraper created
labsuser@master:~$ kubectl get pods -n kubernetes-dashboard -o wide
NAME                                         READY   STATUS    RESTARTS   AGE    IP               NODE                       NOMINATED NODE   READINESS GATES
dashboard-metrics-scraper-6fdb9d6cdd-t46sd   1/1     Running   0          2m3s   192.168.47.132   worker-node-1.example.com   <none>           <none>
kubernetes-dashboard-6fffdf99c9-z5d74        1/1     Running   0          2m3s   192.168.47.131   worker-node-1.example.com   <none>           <none>
labsuser@master:~$ kubectl get deployment -n kubernetes-dashboard -o wide
NAME                        READY   UP-TO-DATE   AVAILABLE   AGE     CONTAINERS                  IMAGES                                 SELECTOR
dashboard-metrics-scraper   1/1     1            1           2m10s   dashboard-metrics-scraper   kubernetesui/metrics-scraper:v1.0.7   k8s-app=dashboard-metrics-scra
per
kubernetes-dashboard        1/1     1            1           2m10s   kubernetes-dashboard        kubernetesui/dashboard:v2.5.0         k8s-app=kubernetes-dashboard
labsuser@master:~$ kubectl get svc -n kubernetes-dashboard -o wide
NAME                        TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)    AGE     SELECTOR
dashboard-metrics-scraper   ClusterIP   10.97.157.15   <none>        8000/TCP   2m16s   k8s-app=dashboard-metrics-scraper
kubernetes-dashboard        ClusterIP   10.98.79.9     <none>        443/TCP    2m16s   k8s-app=kubernetes-dashboard
labsuser@master:~$
```

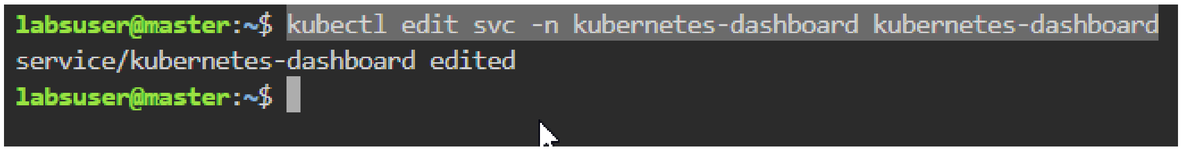
2.2 To access the service outside the cluster, edit the service type from **ClusterIP** to **NodePort** using the following command:
kubectl edit svc -n kubernetes-dashboard kubernetes-dashboard



```
metadata:
  annotations:
    kubectl.kubernetes.io/last-applied-configuration: |
      {"apiVersion":"v1","kind":"Service","metadata":{"annotations":{},"labels":{"k8s-app":"kubernetes-dashboard"},"name":"kubernetes-dashboard","namespace":"kuberne
tes-dashboard"},"spec":{"ports":[{"port":443,"targetPort":8443}],"selector":{"k8s-app":"kubernetes-dashboard"}}}
  creationTimestamp: "2023-10-06T10:51:08Z"
  labels:
    k8s-app: kubernetes-dashboard
  name: kubernetes-dashboard
  namespace: kubernetes-dashboard
  resourceVersion: "21868"
  uid: 71b48ff5-775a-4f64-9ce6-2f3444c7230d
spec:
  clusterIP: 10.98.79.9
  clusterIPs:
  - 10.98.79.9
  internalTrafficPolicy: Cluster
  ipFamilies:
  - IPv4
  ipFamilyPolicy: SingleStack
  ports:
  - port: 443
    protocol: TCP
    targetPort: 8443
  selector:
    k8s-app: kubernetes-dashboard
  sessionAffinity: None
  type: ClusterIP
status:
  loadbalancer: {}
-- INSERT --                                                                                                                                  33,18         Bot
```



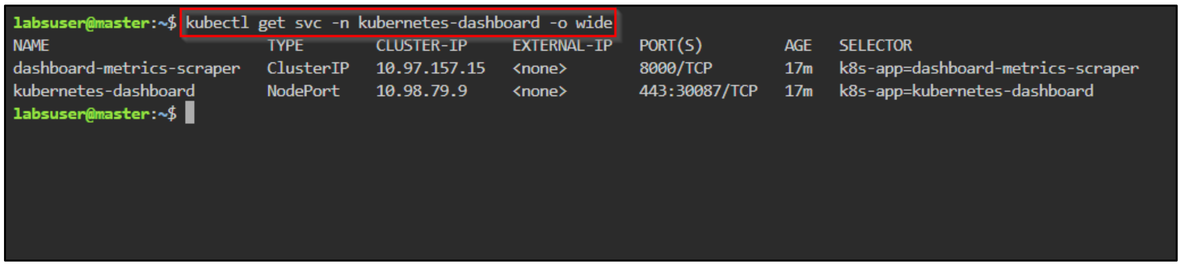
```
metadata:
  annotations:
    kubectl.kubernetes.io/last-applied-configuration: |
      {"apiVersion":"v1","kind":"Service","metadata":{"annotations":{},"labels":{"k8s-app":"kubernetes-dashboard"},"name":"kubernetes-dashboard","namespace":"kuberne
tes-dashboard"},"spec":{"ports":[{"port":443,"targetPort":8443}],"selector":{"k8s-app":"kubernetes-dashboard"}}}
  creationTimestamp: "2023-10-06T10:51:08Z"
  labels:
    k8s-app: kubernetes-dashboard
  name: kubernetes-dashboard
  namespace: kubernetes-dashboard
  resourceVersion: "21868"
  uid: 71b48ff5-775a-4f64-9ce6-2f3444c7230d
spec:
  clusterIP: 10.98.79.9
  clusterIPs:
  - 10.98.79.9
  internalTrafficPolicy: Cluster
  ipFamilies:
  - IPv4
  ipFamilyPolicy: SingleStack
  ports:
  - port: 443
    protocol: TCP
    targetPort: 8443
  selector:
    k8s-app: kubernetes-dashboard
  sessionAffinity: None
  type: NodePort
status:
  loadbalancer: {}
```



```
labsuser@master:~$ kubectl edit svc -n kubernetes-dashboard kubernetes-dashboard
service/kubernetes-dashboard edited
labsuser@master:~$
```

Step 3: Confirm the dashboard service type

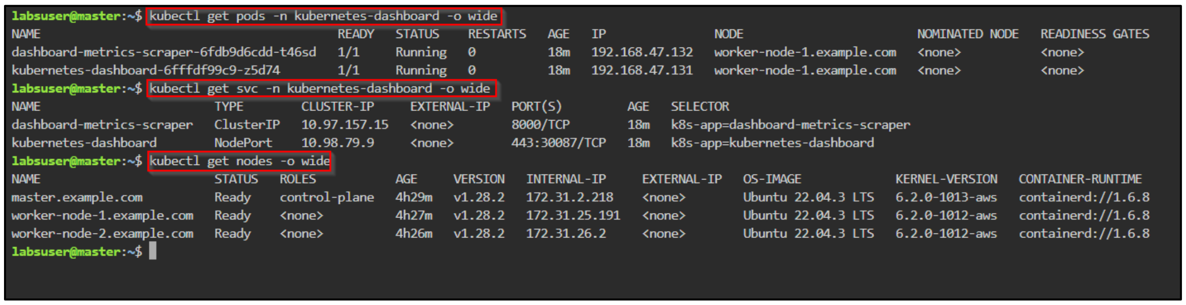
3.1 To confirm that the service type has been changed to NodePort, use the command:
kubectl get svc -n kubernetes-dashboard -o wide



```
labsuser@master:~$ kubectl get svc -n kubernetes-dashboard -o wide
NAME                        TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)         AGE   SELECTOR
dashboard-metrics-scraper   ClusterIP   10.97.157.15   <none>        8000/TCP        17m   k8s-app=dashboard-metrics-scraper
kubernetes-dashboard        NodePort    10.98.79.9     <none>        443:30087/TCP   17m   k8s-app=kubernetes-dashboard
labsuser@master:~$
```

3.2 To determine the location of the pod, run the following commands:

kubectl get pods -n kubernetes-dashboard -o wide
kubectl get svc -n kubernetes-dashboard -o wide
kubectl get nodes -o wide

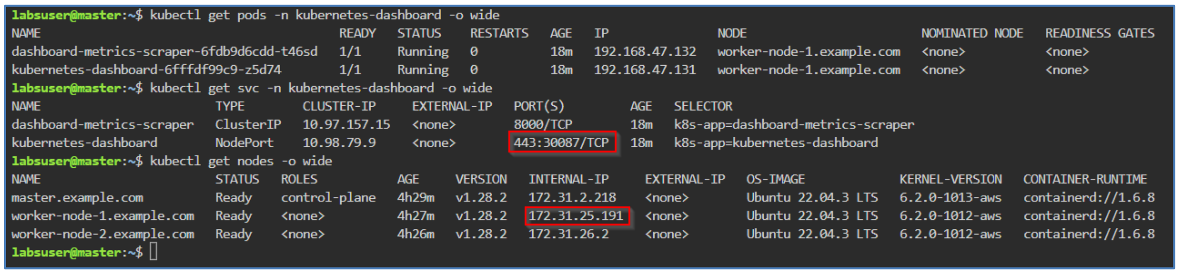


```
labsuser@master:~$ kubectl get pods -n kubernetes-dashboard -o wide
NAME                                         READY   STATUS    RESTARTS   AGE   IP               NODE                       NOMINATED NODE   READINESS GATES
dashboard-metrics-scraper-6fdb9d6cdd-t46sd   1/1     Running   0          18m   192.168.47.132   worker-node-1.example.com  <none>           <none>
kubernetes-dashboard-6fffdf99c9-z5d74        1/1     Running   0          18m   192.168.47.131   worker-node-1.example.com  <none>           <none>
labsuser@master:~$ kubectl get svc -n kubernetes-dashboard -o wide
NAME                        TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)         AGE   SELECTOR
dashboard-metrics-scraper   ClusterIP   10.97.157.15   <none>        8000/TCP        18m   k8s-app=dashboard-metrics-scraper
kubernetes-dashboard        NodePort    10.98.79.9     <none>        443:30087/TCP   18m   k8s-app=kubernetes-dashboard
labsuser@master:~$ kubectl get nodes -o wide
NAME                        STATUS   ROLES           AGE     VERSION   INTERNAL-IP     EXTERNAL-IP   OS-IMAGE             KERNEL-VERSION    CONTAINER-RUNTIME
master.example.com          Ready    control-plane   4h29m   v1.28.2   172.31.2.218    <none>        Ubuntu 22.04.3 LTS   6.2.0-1013-aws    containerd://1.6.8
worker-node-1.example.com   Ready    <none>          4h27m   v1.28.2   172.31.25.191   <none>        Ubuntu 22.04.3 LTS   6.2.0-1012-aws    containerd://1.6.8
worker-node-2.example.com   Ready    <none>          4h26m   v1.28.2   172.31.26.2     <none>        Ubuntu 22.04.3 LTS   6.2.0-1012-aws    containerd://1.6.8
labsuser@master:~$
```

Note: In this case, the Pod is running on **worker-node1**. Note down the **IP** and **NodePort** of node1.

3.3 Use the **INTERNAL-IP** as **172.31.25.191**, and PORT(S) as **30087,** and copy the link:

https://172.31.25.191:30087

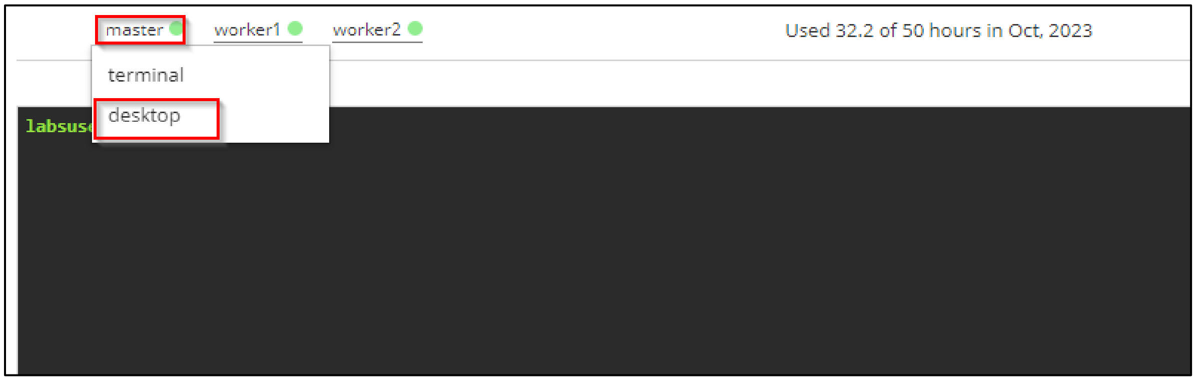


```
labsuser@master:~$ kubectl get pods -n kubernetes-dashboard -o wide
NAME                                         READY   STATUS    RESTARTS   AGE   IP               NODE                       NOMINATED NODE   READINESS GATES
dashboard-metrics-scraper-6fdb9d6cdd-t46sd   1/1     Running   0          18m   192.168.47.132   worker-node-1.example.com  <none>           <none>
kubernetes-dashboard-6fffdf99c9-z5d74        1/1     Running   0          18m   192.168.47.131   worker-node-1.example.com  <none>           <none>
labsuser@master:~$ kubectl get svc -n kubernetes-dashboard -o wide
NAME                        TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)         AGE   SELECTOR
dashboard-metrics-scraper   ClusterIP   10.97.157.15   <none>        8000/TCP        18m   k8s-app=dashboard-metrics-scraper
kubernetes-dashboard        NodePort    10.98.79.9     <none>        443:30087/TCP   18m   k8s-app=kubernetes-dashboard
labsuser@master:~$ kubectl get nodes -o wide
NAME                        STATUS   ROLES           AGE     VERSION   INTERNAL-IP     EXTERNAL-IP   OS-IMAGE             KERNEL-VERSION    CONTAINER-RUNTIME
master.example.com          Ready    control-plane   4h29m   v1.28.2   172.31.2.218    <none>        Ubuntu 22.04.3 LTS   6.2.0-1013-aws    containerd://1.6.8
worker-node-1.example.com   Ready    <none>          4h27m   v1.28.2   172.31.25.191   <none>        Ubuntu 22.04.3 LTS   6.2.0-1012-aws    containerd://1.6.8
worker-node-2.example.com   Ready    <none>          4h26m   v1.28.2   172.31.26.2     <none>        Ubuntu 22.04.3 LTS   6.2.0-1012-aws    containerd://1.6.8
labsuser@master:~$
```

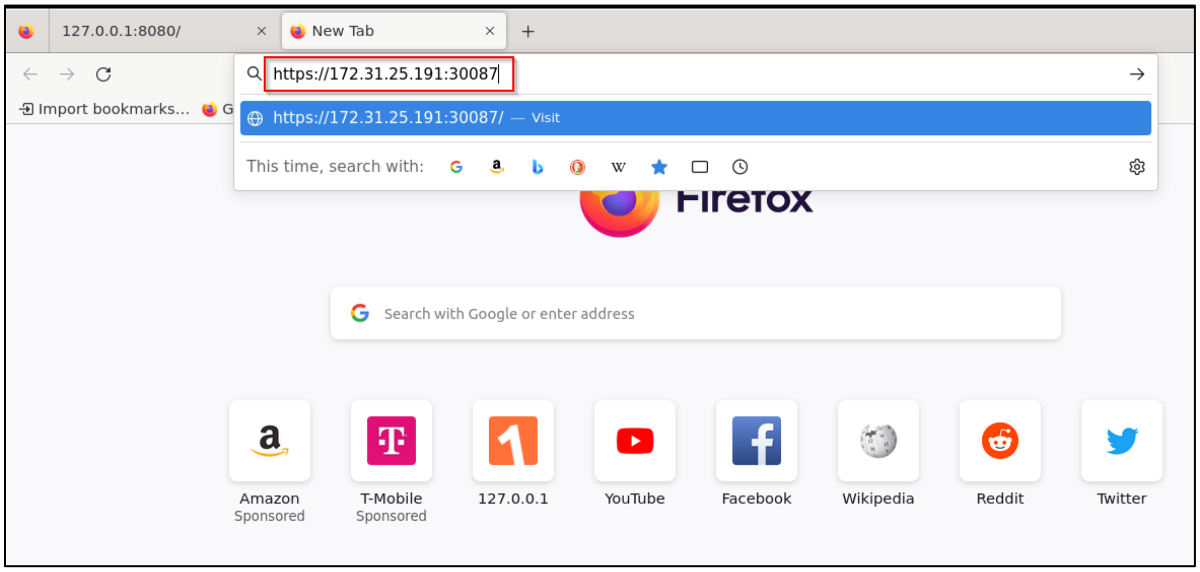
Note: In your case, the IP and NodePort will be different. Change the IP and NodePort accordingly:
https:// <<your worker-node-1>>:<<NodePort>>

Step 4: Access the master node IP

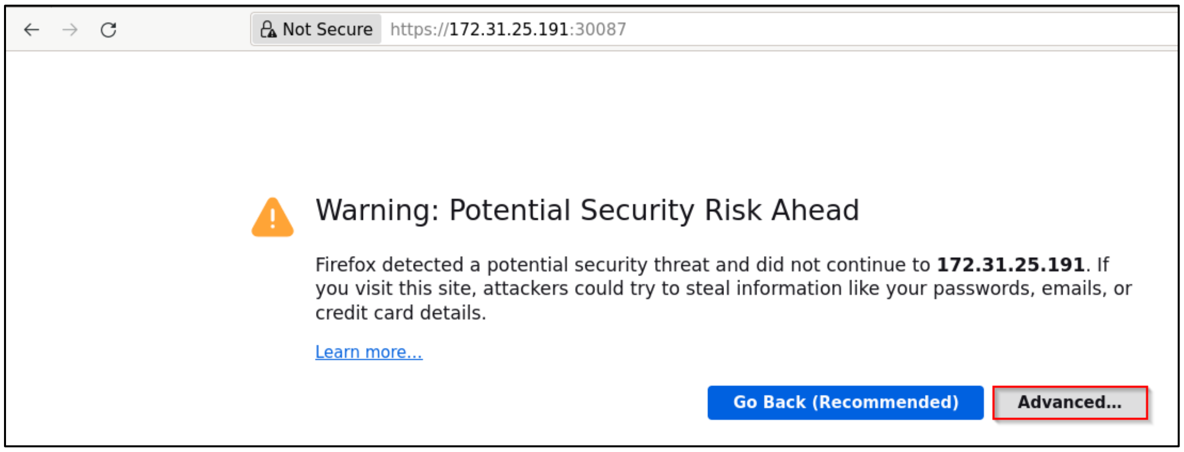
4.1 Navigate to the LMS dashboard and click on **master** then **desktop**



4.2 Open Firefox, paste the copied link from step 3.3 into the search bar, and press Enter



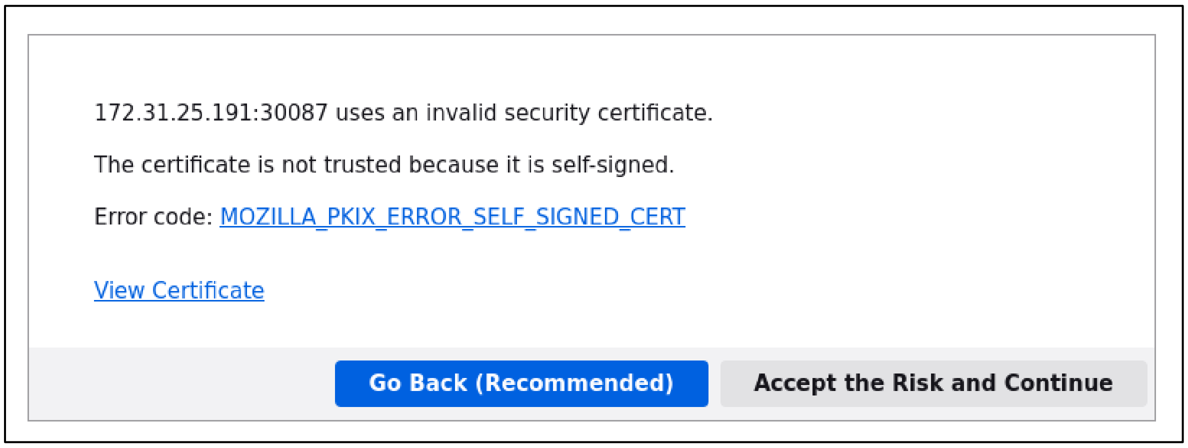
4.3 Click on the **Advanced** button


Not Secure https://172.31.25.191:30087

Warning: Potential Security Risk Ahead

Firefox detected a potential security threat and did not continue to **172.31.25.191**. If you visit this site, attackers could try to steal information like your passwords, emails, or credit card details.

4.4 Click **Accept the Risk and Continue**

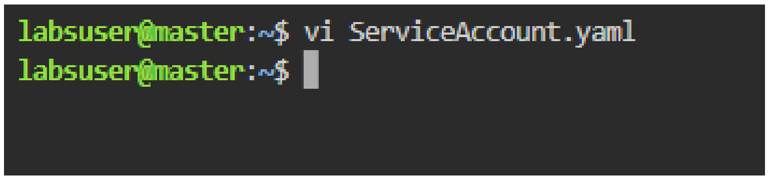

172.31.25.191:30087 uses an invalid security certificate.

The certificate is not trusted because it is self-signed.

Error code: MOZILLA_PKIX_ERROR_SELF_SIGNED_CERT

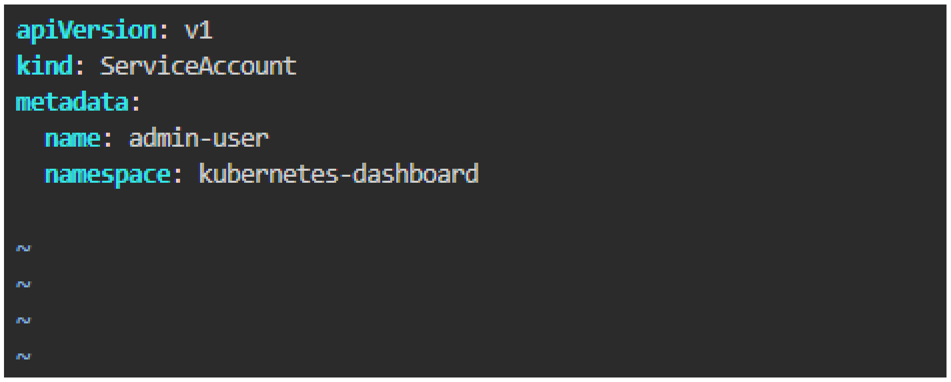
Step 5: Log into the service dashboard

5.1 Create a service account by running the following command, and then input the code in the master node:
vi ServiceAccount.yaml



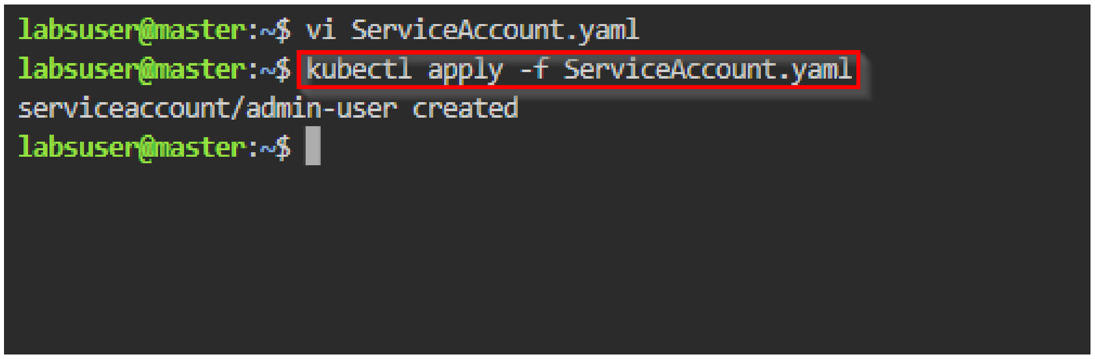
```
labsuser@master:~$ vi ServiceAccount.yaml
labsuser@master:~$
```

apiVersion: v1
kind: ServiceAccount
metadata:
  name: admin-user
  namespace: kubernetes-dashboard



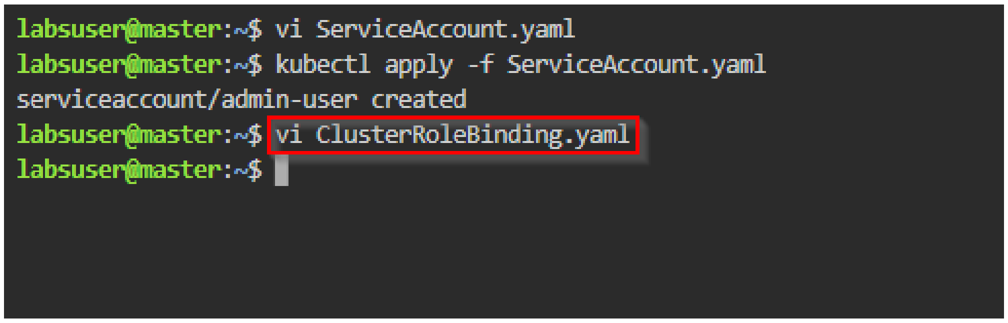
```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: admin-user
  namespace: kubernetes-dashboard

~
~
~
~
```

5.2 Apply the YAML file with the command:
kubectl apply -f ServiceAccount.yaml



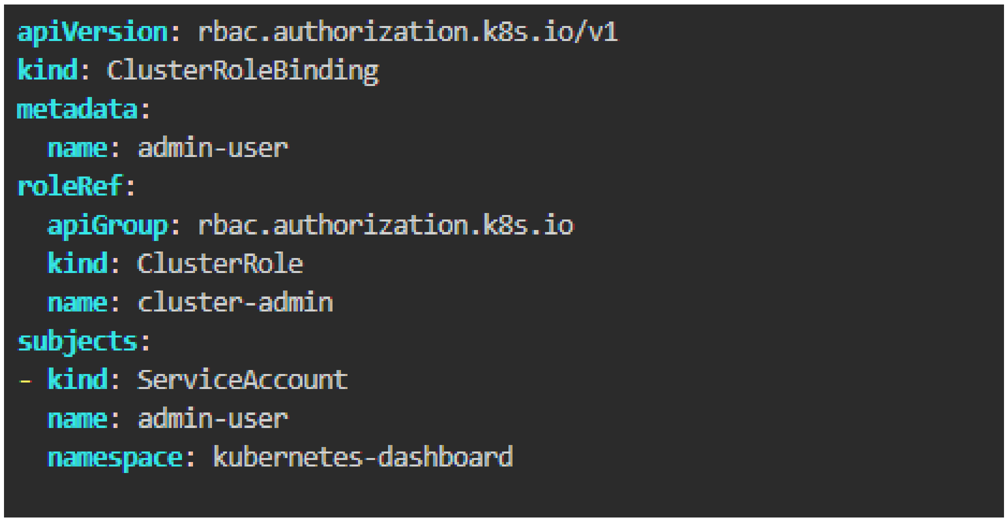
```
labsuser@master:~$ vi ServiceAccount.yaml
labsuser@master:~$ kubectl apply -f ServiceAccount.yaml
serviceaccount/admin-user created
labsuser@master:~$
```

5.3 Create a **yaml** file for cluster role binding using below command and code:
vi ClusterRoleBinding.yaml



```
labsuser@master:~$ vi ServiceAccount.yaml
labsuser@master:~$ kubectl apply -f ServiceAccount.yaml
serviceaccount/admin-user created
labsuser@master:~$ vi ClusterRoleBinding.yaml
labsuser@master:~$
```

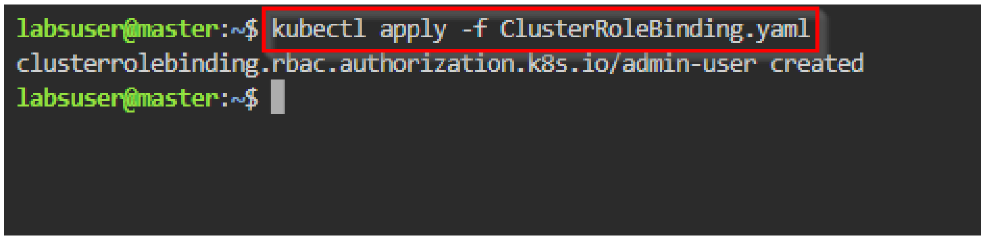
```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: admin-user
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: cluster-admin
subjects:
- kind: ServiceAccount
  name: admin-user
  namespace: kubernetes-dashboard
```



```
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
```

5.4 Run the below command to create cluster role binding:

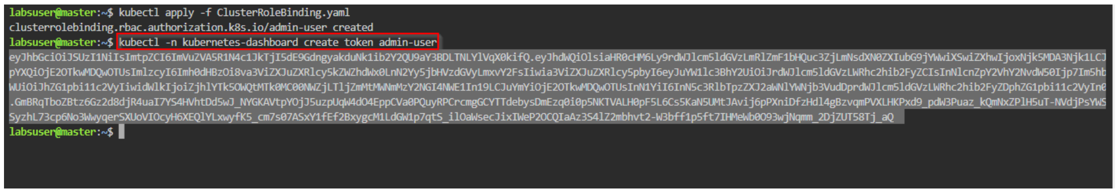
kubectl apply -f ClusterRoleBinding.yaml



```
labsuser@master:~$ kubectl apply -f ClusterRoleBinding.yaml
clusterrolebinding.rbac.authorization.k8s.io/admin-user created
labsuser@master:~$
```

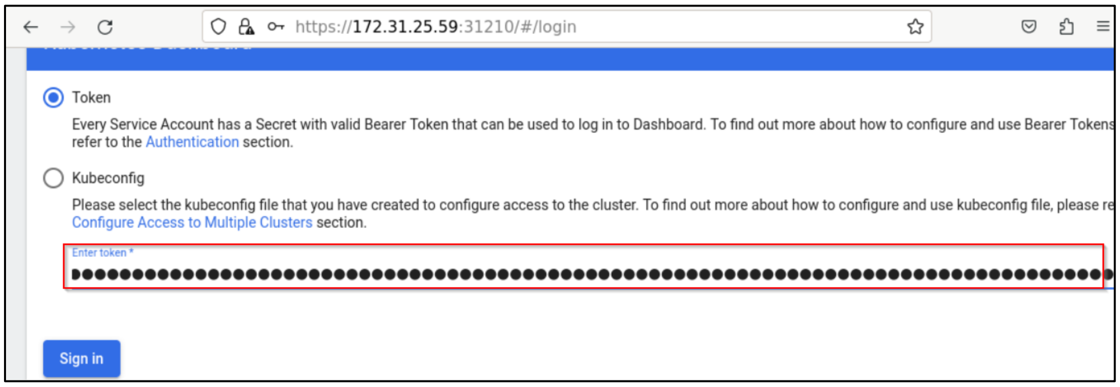
5.5 Retrieve the token to log in by running the following command:

kubectl -n kubernetes-dashboard create token admin-user



```
labsuser@master:~$ kubectl apply -f ClusterRoleBinding.yaml
clusterrolebinding.rbac.authorization.k8s.io/admin-user created
labsuser@master:~$ kubectl -n kubernetes-dashboard create token admin-user
eyJhbGciOiJSUzI1NiIsImtpZCI6ImVuZVA5R1N4c1JkTjI5dE9GdngyakduNk1ib2Y2QU9aY3BDLTNLYlVqX0kifQ.eyJhdWQiOlsiaHR0cHM6Ly9rdWJlcm5ldGVzLmRlZmF1bHQuc3ZjLmNsdXN0ZXIubG9jYWwiXSwiZXhwIjoxNjk5MDA3Njk1LCJ
pYXQiOjE2OTkwMDQwOTUsImlzcyI6Imh0dHBzOi8va3ViZXJuZXRlcy5kZWZhdWx0LnN2Yy5jbHVzdGVyLmxvY2FsIiwia3ViZXJuZXRlcy5pbyI6eyJuYW1lc3BhY2UiOiJrdWJlcm5ldGVzLWRhc2hib2FyZCIsInNlcnZpY2VhY2NvdW50Ijp7Im5hb
WUiOiJhZG1pbi11c2VyIiwidWlkIjoiZjhlYTk5OWQtMTk0MC00NWZjLTljZmMtMWNmMzY2NGI4NWE1In19LCJuYmYiOjE2OTkwMDQwOTUsInN1YiI6InN5c3RlbTpzZXJ2aWNlYWNjb3VudDprdWJlcm5ldGVzLWRhc2hib2FyZDphZG1pbi11c2VyIn0
.GmBRqTboZBtz6Gz2d8djR4uaI7YS4HVhtDd5wJ_NYGKAVtpYOjJ5uzpUqW4dO4EppCVa0PQuyRPCrcmgGCYTTdebysDmEzq0i0p5NKTVALH0pF5L6Cs5KaN5UMtJAvij6pPXniDfzHdl4gBzvqmPVXLHKPxd9_pdW3Puaz_kQmNxZPlH5uT-NVdjPsYWS
SyzhL73cp6No3WwyqerSXUoVIOcyH6XEQlYLxwyfK5_cm7s07ASxY1fEf2BxygcM1LdGW1p7qtS_ilOaWsecJixIWeP2OCQIaAz3S4lZ2mbhvt2-W3bff1p5ft7IHMeWb0O93wjNqmm_2DjZUT58Tj_aQ
labsuser@master:~$
```

5.6 Copy the token and paste it into the Kubernetes dashboard login screen, then click **Sign in**



[✓ 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 refer to the Authentication section.

Kubeconfig

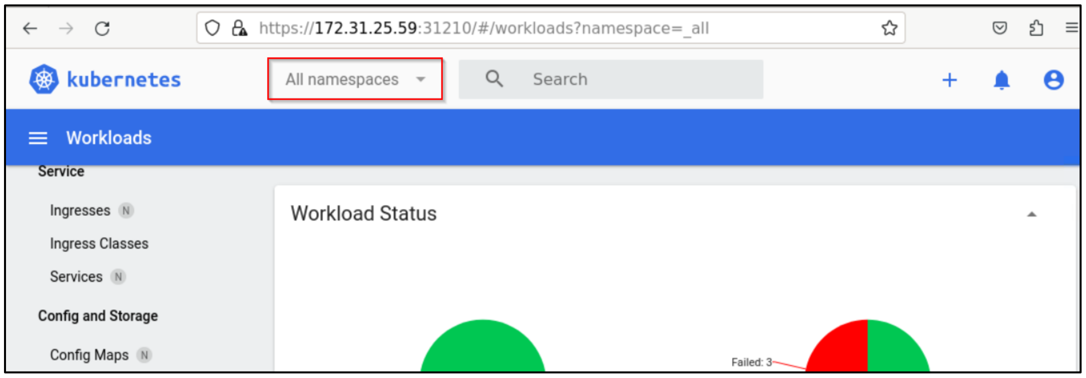
Please select the kubeconfig file that you have created to configure access to the cluster. To find out more about how to configure and use kubeconfig file, please re
Configure Access to Multiple Clusters section.

Enter token *

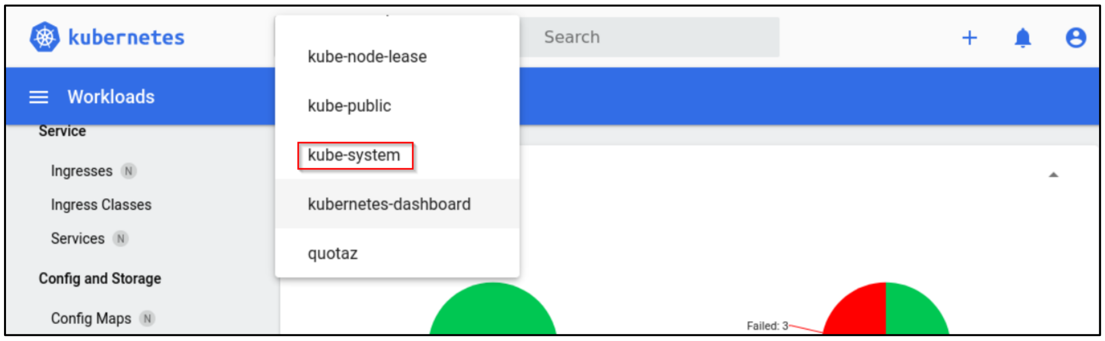
●●●

Step 6: Access the Kubernetes dashboard

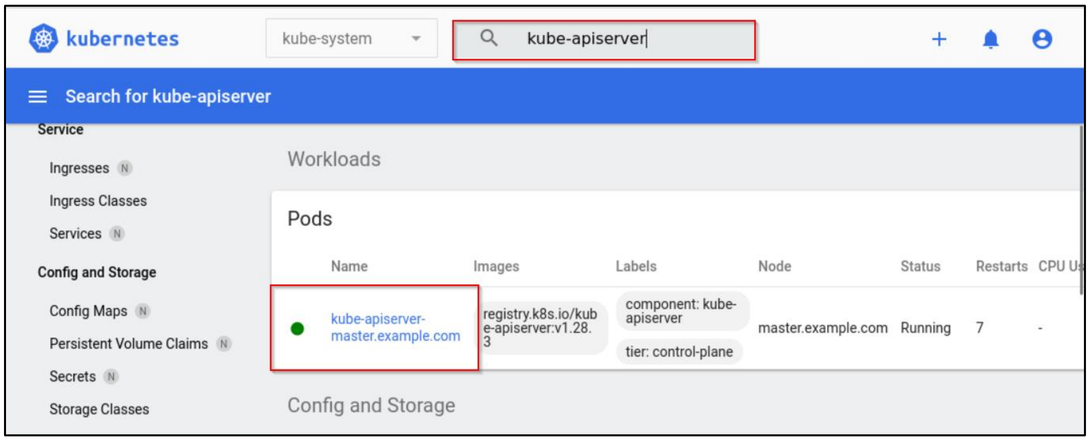
6.1 Click on the All **namespaces** drop down menu



6.2 Select **kube-system**

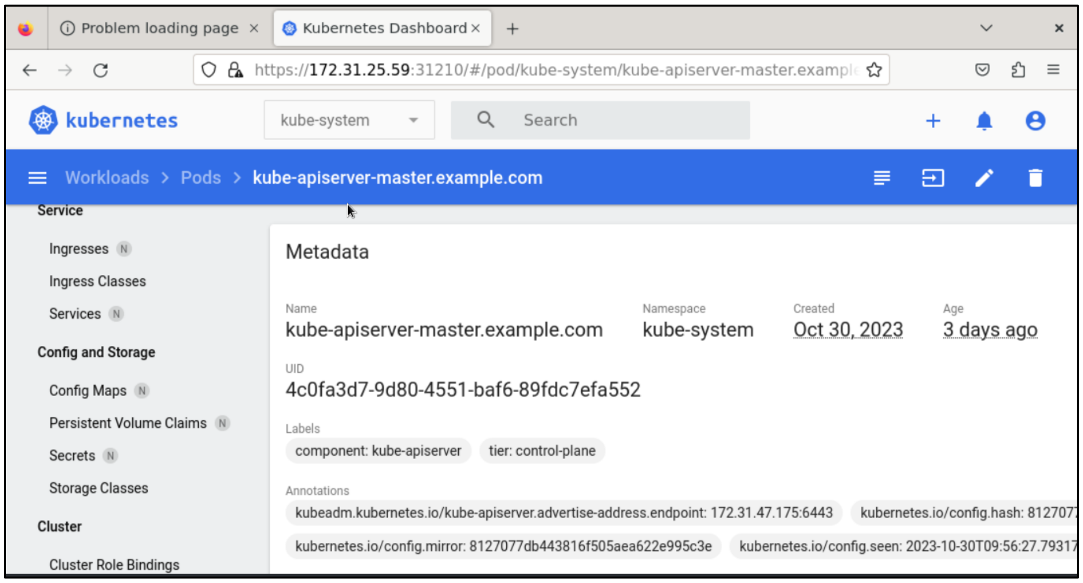


6.3 Use the search bar to find and select **kube-apiserver**



Name	Images	Labels	Node	Status	Restarts	CPU U
● kube-apiserver-master.example.com	registry.k8s.io/kube-apiserver:v1.28.3	component: kube-apiserver; tier: control-plane	master.example.com	Running	7	-

View the logs of the kube-apiserver.



Problem loading page × | Kubernetes Dashboard × | +

https://172.31.25.59:31210/#/pod/kube-system/kube-apiserver-master.exampl

kubernetes | kube-system | Search

Workloads > Pods > kube-apiserver-master.example.com

Metadata

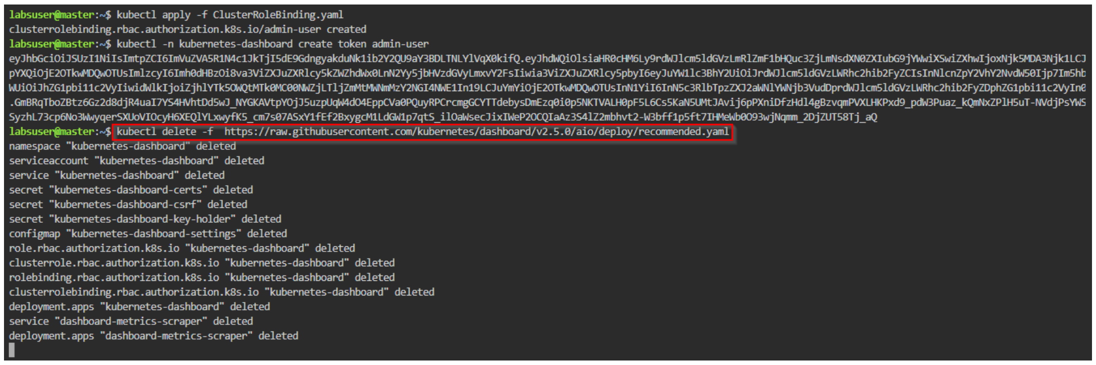
Name: kube-apiserver-master.example.com
Namespace: kube-system
Created: Oct 30, 2023
Age: 3 days ago

UID: 4c0fa3d7-9d80-4551-baf6-89fdc7efa552

Labels: component: kube-apiserver | tier: control-plane

Annotations: kubeadm.kubernetes.io/kube-apiserver.advertise-address.endpoint: 172.31.47.175:6443 | kubernetes.io/config.hash: 812707 | kubernetes.io/config.mirror: 8127077db443816f505aea622e995c3e | kubernetes.io/config.seen: 2023-10-30T09:56:27.79317

6.4 [OPTIONAL] Cleanup: To delete the Kubernetes dashboard version 2.5, use the following command in the master node:
kubectl delete -f https://raw.githubusercontent.com/kubernetes/dashboard/v2.5.0/aio/deploy/recommended.yaml



```
labsuser@master:~$ kubectl apply -f ClusterRoleBinding.yaml
clusterrolebinding.rbac.authorization.k8s.io/admin-user created
labsuser@master:~$ kubectl -n kubernetes-dashboard create token admin-user
eyJhbGciOiJSUzI1NiIsImtpZCI6ImVuZVA5R1N4c1JkTjI5dE9GdngyakduNk1ib2Y2QU9aY3BDLTNLYlVqX0kifQ.eyJhdWQiOlsiaHR0cHM6Ly9rdWJlcm5ldGVzLmRlZmF1bHQuc3ZjLmNsdXN0ZXIubG9jYWwiXSwiZXhwIjoxNjk5MDA3Njk1LCJ
pYXQiOjE2OTkwMDQwOTUsImlzcyI6Imh0dHBzOi8va3ViZXJuZXRlcy5kZWZhdWx0LnN2Yy5jbHVzdGVyLmxvY2FsIiwia3ViZXJuZXRlcy5pbyI6eyJuYW1lc3BhY2UiOiJrdWJlcm5ldGVzLWRhc2hib2FyZCIsInNlcnZpY2VhY2NvdW50Ijp7Im5hb
WUiOiJhZG1pbi11c2VyIiwidWlkIjoiZjhlYTk5OWQtMTk0MC00NWZjLTljZmMtMWNmMzY2NGI4NWE1In19LCJuYmYiOjE2OTkwMDQwOTUsInN1YiI6InN5c3RlbTpzZXJ2aWNlYWNjb3VudDprdWJlcm5ldGVzLWRhc2hib2FyZDphZG1pbi11c2VyIn0
.GmBRqTboZBtz6Gz2d8djR4uaI7YS4HVhtDd5wJ_NYGKAVtpYOjJ5uzpUqW4dO4EppCVa0PQuyRPCrcmgGCYTTdebysDmEzq0i0p5NKTVALH0pF5L6Cs5KaN5UMtJAvij6pPXniDfzHdl4gBzvqmPVXLHKPxd9_pdW3Puaz_kQmNxZPlH5uT-NVdjPsYWS
SyzhL73cp6No3WwyqerSXUoVIOcyH6XEQlYLxwyfK5_cm7s07ASxY1fEf2BxygcM1LdGW1p7qtS_ilOaWsecJixIWeP2OCQIaAz3S4lZ2mbhvt2-W3bff1p5ft7IHMeWb0O93wjNqmm_2DjZUT58Tj_aQ
labsuser@master:~$ kubectl delete -f  https://raw.githubusercontent.com/kubernetes/dashboard/v2.5.0/aio/deploy/recommended.yaml
namespace "kubernetes-dashboard" deleted
serviceaccount "kubernetes-dashboard" deleted
service "kubernetes-dashboard" deleted
secret "kubernetes-dashboard-certs" deleted
secret "kubernetes-dashboard-csrf" deleted
secret "kubernetes-dashboard-key-holder" deleted
configmap "kubernetes-dashboard-settings" deleted
role.rbac.authorization.k8s.io "kubernetes-dashboard" deleted
clusterrole.rbac.authorization.k8s.io "kubernetes-dashboard" deleted
rolebinding.rbac.authorization.k8s.io "kubernetes-dashboard" deleted
clusterrolebinding.rbac.authorization.k8s.io "kubernetes-dashboard" deleted
deployment.apps "kubernetes-dashboard" deleted
service "dashboard-metrics-scraper" deleted
deployment.apps "dashboard-metrics-scraper" deleted
```

By following these steps, you will be able to deploy the Kubernetes Dashboard, establish secure access, and navigate the interface to monitor and manage your Kubernetes cluster.