# Lesson 07 Demo 09

# Troubleshooting Networking Issues

**Objective:** To create, troubleshoot, and modify an httpd-pod and its associated service in a Kubernetes environment

**Tools required:** kubeadm, kubectl, kubelet, and containerd

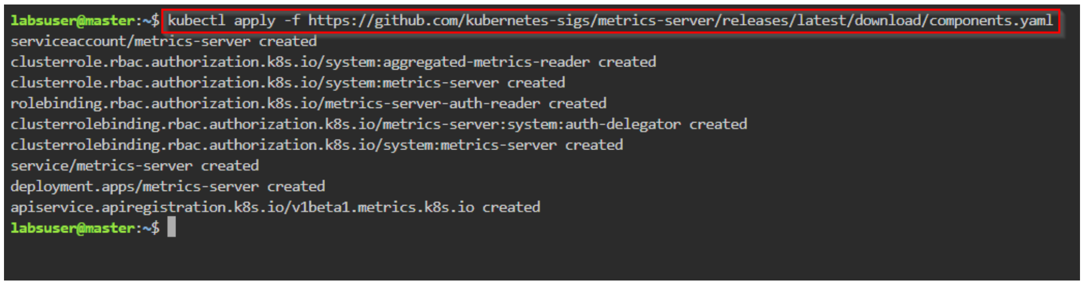
**Prerequisites:** A Kubernetes cluster (refer to Demo 01 from Lesson 01 for setting up a cluster)

Steps to be followed:

1. Create an httpd-pod
2. Create an httpd-service
3. Check the labels for all the pods

## Step 1: Create an httpd-pod

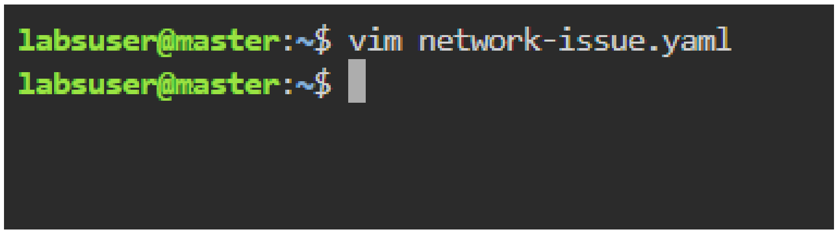
1.1 Install the metrics API using the following command:
**Kubectl apply -f https://github.com/kubernetes-sigs/metrics-server/releases/latest/download/components.yaml**



```
labsuser@master:~$ kubectl apply -f https://github.com/kubernetes-sigs/metrics-server/releases/latest/download/components.yaml
serviceaccount/metrics-server created
clusterrole.rbac.authorization.k8s.io/system:aggregated-metrics-reader created
clusterrole.rbac.authorization.k8s.io/system:metrics-server created
rolebinding.rbac.authorization.k8s.io/metrics-server-auth-reader created
clusterrolebinding.rbac.authorization.k8s.io/metrics-server:system:auth-delegator created
clusterrolebinding.rbac.authorization.k8s.io/system:metrics-server created
service/metrics-server created
deployment.apps/metrics-server created
apiservice.apiregistration.k8s.io/v1beta1.metrics.k8s.io created
labsuser@master:~$
```

1.2 Run the following command to create a YAML file for an httpd-pod:

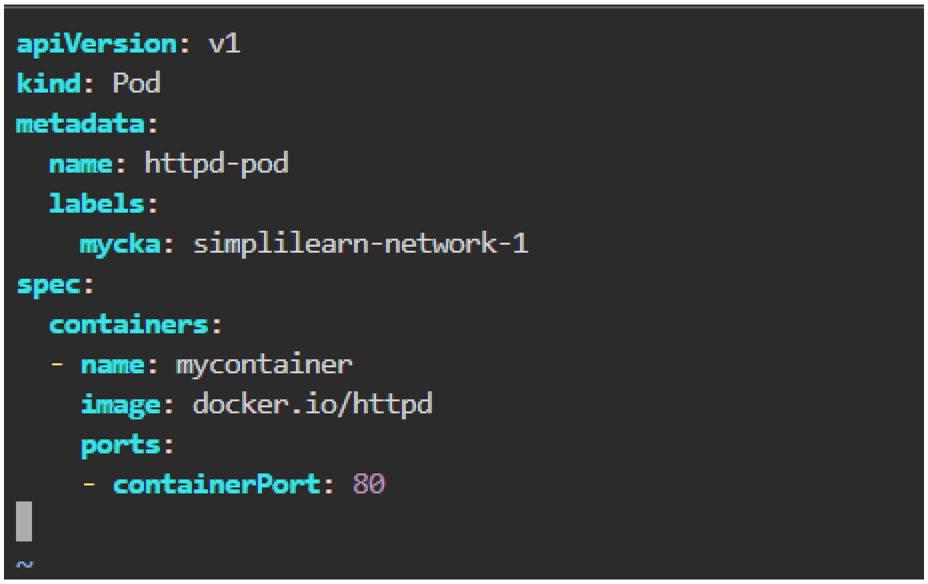
**vim network-issue.yaml**



```
labsuser@master:~$ vim network-issue.yaml
labsuser@master:~$
```

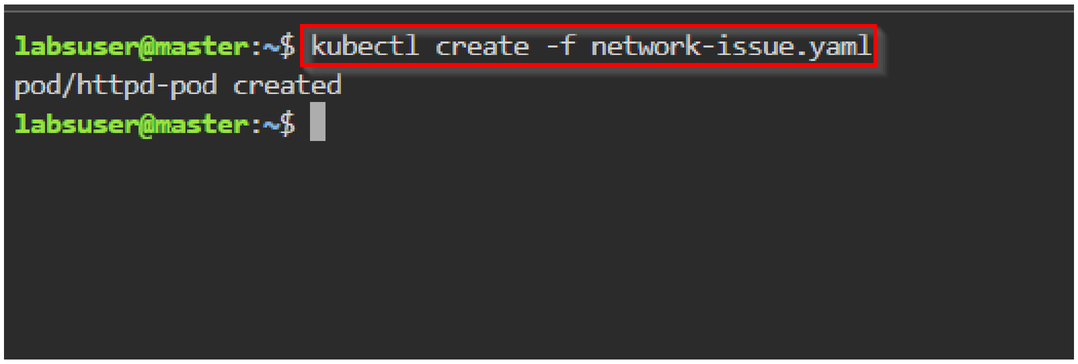
1.3 Add the following network access configuration for the pod within a container:

```
apiVersion: v1
kind: Pod
metadata:
  name: httpd-pod
  labels:
    mycka: simplilearn-network-1
spec:
  containers:
  - name: mycontainer

    image: docker.io/httpd
    ports:
    - containerPort: 80
```



```
apiVersion: v1
kind: Pod
metadata:
  name: httpd-pod
  labels:
    mycka: simplilearn-network-1
spec:
  containers:
  - name: mycontainer
    image: docker.io/httpd
    ports:
    - containerPort: 80

~
```

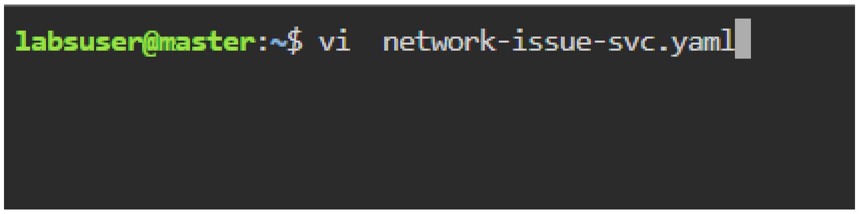
1.4 Execute the following command to create the pod:
**kubectl create -f network-issue.yaml**



```
labsuser@master:~$ kubectl create -f network-issue.yaml
pod/httpd-pod created
labsuser@master:~$
```

## Step 2: Create an httpd-service

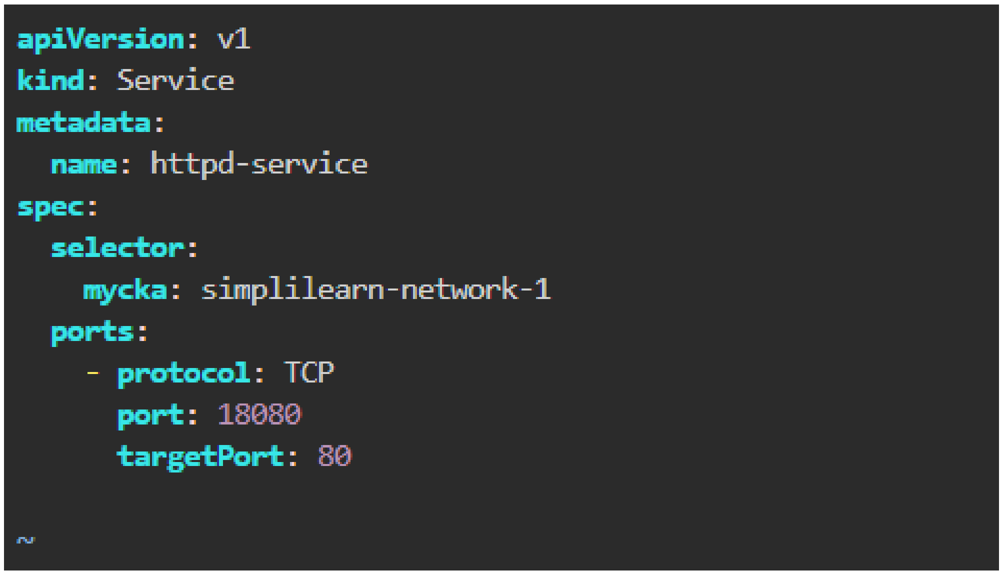
2.1 Execute the following command to create **network-issue-svc.yaml** file:
**vi  network-issue-svc.yaml**



```
labsuser@master:~$ vi  network-issue-svc.yaml
```

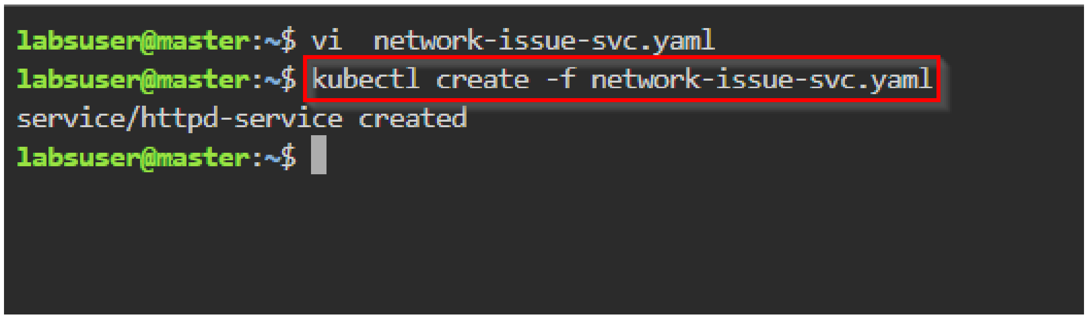
2.2 Add the following code to **network-issue-svc.yaml**:

```
apiVersion: v1
kind: Service
metadata:
  name: httpd-service
spec:
  selector:
    mycka: simplilearn-network-1
  ports:
    - protocol: TCP
      port: 18080
      targetPort: 80
```


```
apiVersion: v1
kind: Service
metadata:
  name: httpd-service
spec:
  selector:
    mycka: simplilearn-network-1
  ports:
    - protocol: TCP
      port: 18080
      targetPort: 80

~
```

2.3 Execute the following command to create the service:

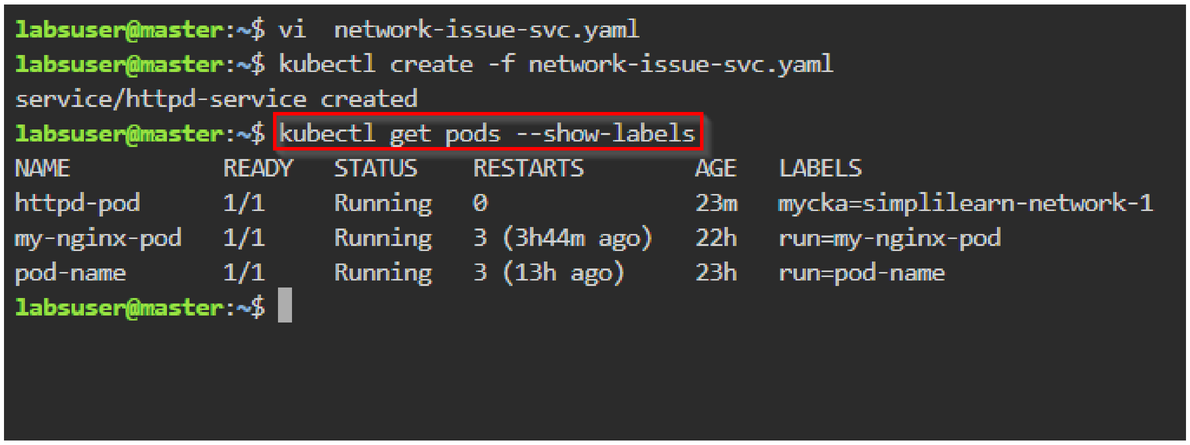
**kubectl create -f network-issue-svc.yaml**


```
labsuser@master:~$ vi  network-issue-svc.yaml
labsuser@master:~$ kubectl create -f network-issue-svc.yaml
service/httpd-service created
labsuser@master:~$ 
```

## Step 3: Check the labels for all the pods

3.1 To check the labels, execute the following command:

**kubectl get pods --show-labels**

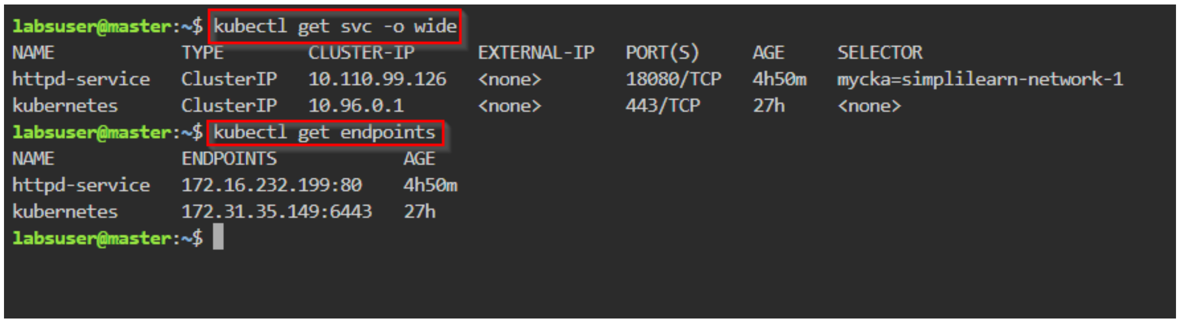


```
labsuser@master:~$ vi  network-issue-svc.yaml
labsuser@master:~$ kubectl create -f network-issue-svc.yaml
service/httpd-service created
labsuser@master:~$ kubectl get pods --show-labels
NAME           READY   STATUS    RESTARTS        AGE   LABELS
httpd-pod      1/1     Running   0               23m   mycka=simplilearn-network-1
my-nginx-pod   1/1     Running   3 (3h44m ago)   22h   run=my-nginx-pod
pod-name       1/1     Running   3 (13h ago)     23h   run=pod-name
labsuser@master:~$
```

3.2 Use the following commands to get endpoints:

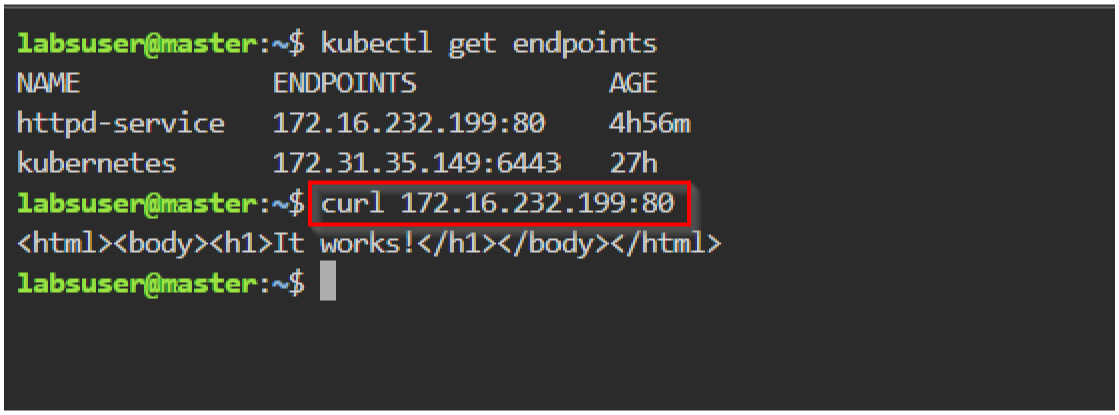
**kubectl get svc -o wide**

**kubectl get endpoints**



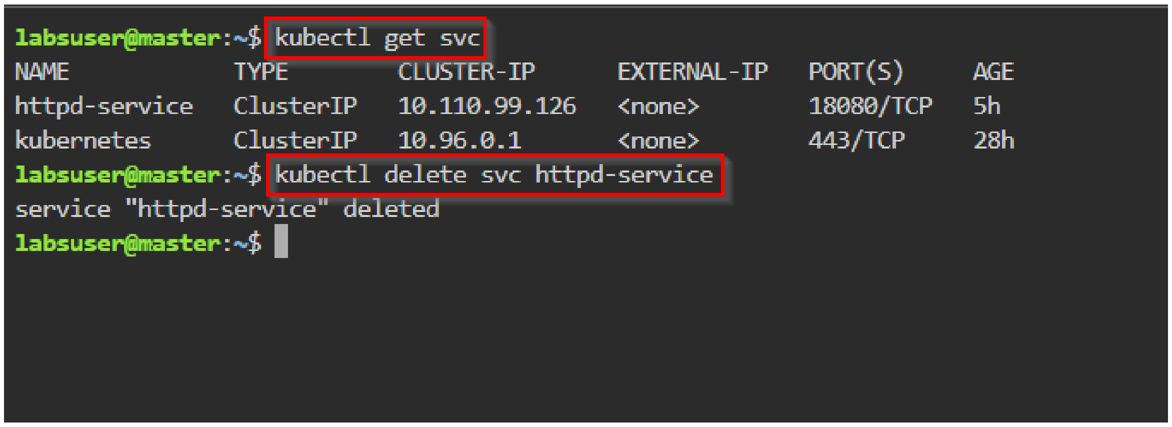
```
labsuser@master:~$ kubectl get svc -o wide
NAME            TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)     AGE     SELECTOR
httpd-service   ClusterIP   10.110.99.126   <none>        18080/TCP   4h50m   mycka=simplilearn-network-1
kubernetes      ClusterIP   10.96.0.1       <none>        443/TCP     27h     <none>
labsuser@master:~$ kubectl get endpoints
NAME            ENDPOINTS            AGE
httpd-service   172.16.232.199:80    4h50m
kubernetes      172.31.35.149:6443   27h
labsuser@master:~$
```

3.3 Verify the service with the **curl** command followed by the cluster IP of the httpd-service
Example: **curl 172.16.232.199:80**



```
labsuser@master:~$ kubectl get endpoints
NAME            ENDPOINTS             AGE
httpd-service   172.16.232.199:80     4h56m
kubernetes      172.31.35.149:6443    27h
labsuser@master:~$ curl 172.16.232.199:80
<html><body><h1>It works!</h1></body></html>
labsuser@master:~$
```

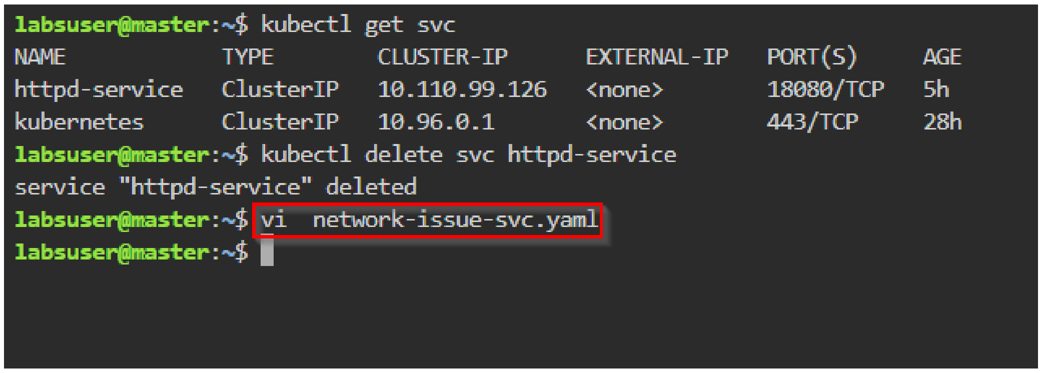
3.4 Delete the httpd-service with these commands:
**kubectl get svc**
**kubectl delete svc httpd-service**



```
labsuser@master:~$ kubectl get svc
NAME            TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)     AGE
httpd-service   ClusterIP   10.110.99.126   <none>        18080/TCP   5h
kubernetes      ClusterIP   10.96.0.1       <none>        443/TCP     28h
labsuser@master:~$ kubectl delete svc httpd-service
service "httpd-service" deleted
labsuser@master:~$
```

3.5 To modify the service file for using a different network name, execute the following command to open the **network-issue-svc.yaml** file:

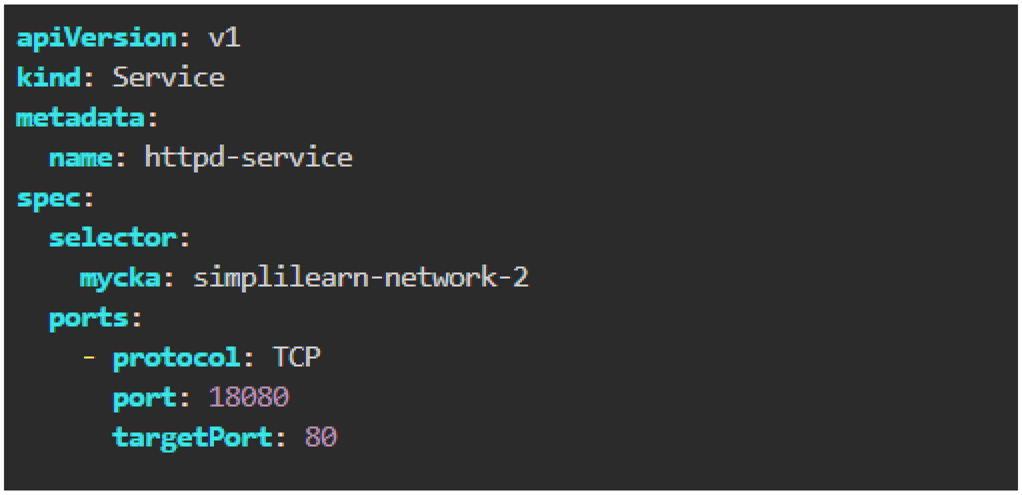
**vi network-issue-svc.yaml**



```
labsuser@master:~$ kubectl get svc
NAME            TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)     AGE
httpd-service   ClusterIP   10.110.99.126   <none>        18080/TCP   5h
kubernetes      ClusterIP   10.96.0.1       <none>        443/TCP     28h
labsuser@master:~$ kubectl delete svc httpd-service
service "httpd-service" deleted
labsuser@master:~$ vi  network-issue-svc.yaml
labsuser@master:~$
```

3.6 Update the **network-issue-svc.yaml** file with the following code:

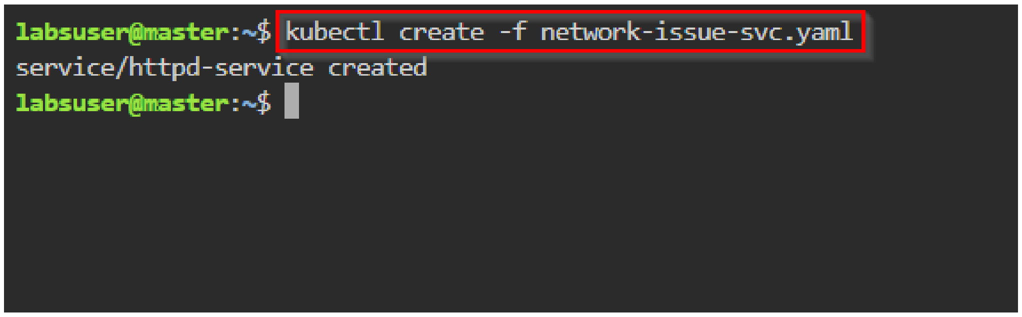
```
apiVersion: v1
kind: Service
metadata:
  name: httpd-service
spec:
  selector:
    mycka: simplilearn-network-2
  ports:
    - protocol: TCP
      port: 18080
      targetPort: 80
```



```
apiVersion: v1
kind: Service
metadata:
  name: httpd-service
spec:
  selector:
    mycka: simplilearn-network-2
  ports:
    - protocol: TCP
      port: 18080
      targetPort: 80
```

3.7 Execute the following command to create the service:

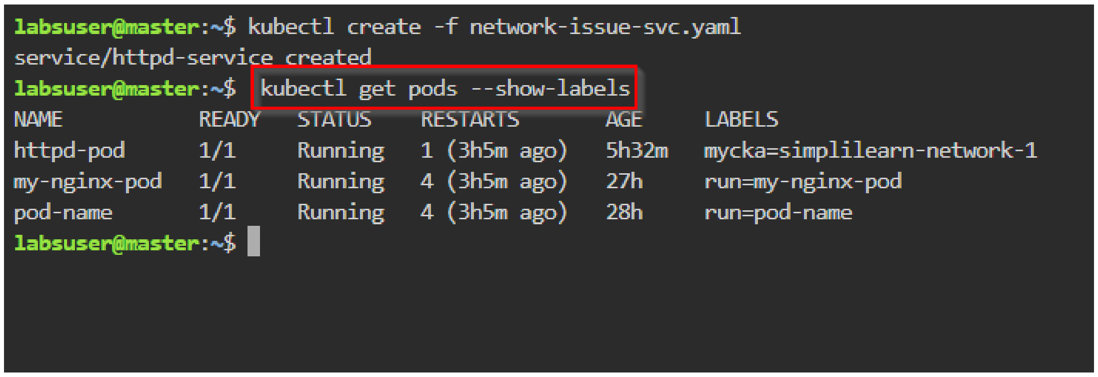
**kubectl create -f network-issue-svc.yaml**



```
labsuser@master:~$ kubectl create -f network-issue-svc.yaml
service/httpd-service created
labsuser@master:~$
```

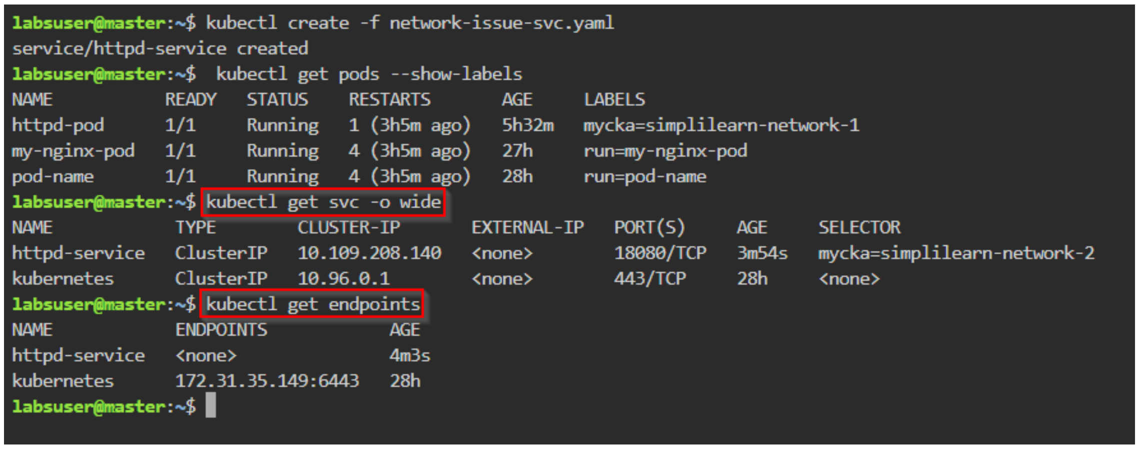
3.8 To list the pods with labels, execute the following command:

**kubectl get pods --show-labels**



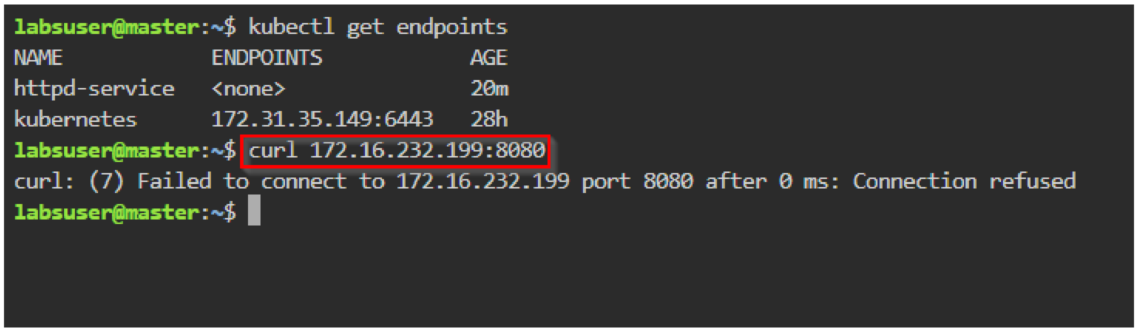
```
labsuser@master:~$ kubectl create -f network-issue-svc.yaml
service/httpd-service created
labsuser@master:~$ kubectl get pods --show-labels
NAME           READY   STATUS    RESTARTS      AGE     LABELS
httpd-pod      1/1     Running   1 (3h5m ago)  5h32m   mycka=simplilearn-network-1
my-nginx-pod   1/1     Running   4 (3h5m ago)  27h     run=my-nginx-pod
pod-name       1/1     Running   4 (3h5m ago)  28h     run=pod-name
labsuser@master:~$
```

3.9 Execute the commands below to retrieve the cluster IP and endpoints along with their respective ports:
**kubectl get svc -o wide**
**kubectl get endpoints**



```
labsuser@master:~$ kubectl create -f network-issue-svc.yaml
service/httpd-service created
labsuser@master:~$  kubectl get pods --show-labels
NAME          READY   STATUS    RESTARTS       AGE     LABELS
httpd-pod     1/1     Running   1 (3h5m ago)   5h32m   mycka=simplilearn-network-1
my-nginx-pod  1/1     Running   4 (3h5m ago)   27h     run=my-nginx-pod
pod-name      1/1     Running   4 (3h5m ago)   28h     run=pod-name
labsuser@master:~$ kubectl get svc -o wide
NAME           TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)     AGE     SELECTOR
httpd-service  ClusterIP   10.109.208.140  <none>        18080/TCP   3m54s   mycka=simplilearn-network-2
kubernetes     ClusterIP   10.96.0.1       <none>        443/TCP     28h     <none>
labsuser@master:~$ kubectl get endpoints
NAME           ENDPOINTS            AGE
httpd-service  <none>               4m3s
kubernetes     172.31.35.149:6443   28h
labsuser@master:~$
```

3.10 Reaccess the service using the **curl** command followed by the cluster IP of the service along with the correct port number
Example: **curl 172.16.232.199:8080**



```
labsuser@master:~$ kubectl get endpoints
NAME           ENDPOINTS            AGE
httpd-service  <none>               20m
kubernetes     172.31.35.149:6443   28h
labsuser@master:~$ curl 172.16.232.199:8080
curl: (7) Failed to connect to 172.16.232.199 port 8080 after 0 ms: Connection refused
labsuser@master:~$
```

By following these steps, you have successfully troubleshot networking issues, set up an httpd-pod and its associated service, and validated their operation within a Kubernetes environment.