**Expose services in the cluster with node port, cluster IP, load balancer**

In kubernetes, service can be expoaed in different ways such as follow:

**1. Exposing a Service with ClusterIP:**

In the cluster IP we will create a yaml file such as follow

a) Creating a Deployment

# nginx-deployment.yaml

apiVersion: apps/v1

kind: Deployment

metadata:

name: nginx-deployment

spec:

replicas: 2

selector:­

matchLabels:

app: nginx

template:

metadata:

labels:

app: nginx

spec:

containers:

- name: nginx

image: nginx:latest

ports:

- containerPort: 80

Now we apply the deployment:

$ kubectl apply -f nginx-deployment.yaml

b) Now expose the Deployment as a ClusterIP Service

# nginx-clusterip-service.yaml

apiVersion: v1

kind: Service

metadata:

name: nginx-clusterip-service

spec:

type: ClusterIP

selector:

app: nginx

ports:

- protocol: TCP

port: 80

targetPort: 80

Again applying the service:

$ kubectl apply -f nginx-clusterip-service.yaml

c) Now verifying the Service

$ kubectl get services

This will show details about service

**2. Exposing a Service with NodePort**

NodePort exposes the service on a static port on each node’s IP.

a) Expose the Deployment as a NodePort Service

# nginx-nodeport-service.yaml

apiVersion: v1

kind: Service

metadata:

name: nginx-nodeport-service

spec:

type: NodePort

selector:

app: nginx

ports:

- protocol: TCP

port: 80

targetPort: 80

nodePort: 30007

Now applying the service:

$ kubectl apply -f nginx-nodeport-service.yaml

```

b) Verifying the Service

$ kubectl get services

**3. Exposing a Service with LoadBalancer**

a) Expose the Deployment as a LoadBalancer Service

# nginx-loadbalancer-service.yaml

apiVersion: v1

kind: Service

metadata:

name: nginx-loadbalancer-service

spec:

type: LoadBalancer

selector:

app: nginx

ports:

- protocol: TCP

port: 80

targetPort: 80

Applying the service:

$ kubectl apply -f nginx-loadbalancer-service.yaml

b) Verify the Service

$ kubectl get services

```