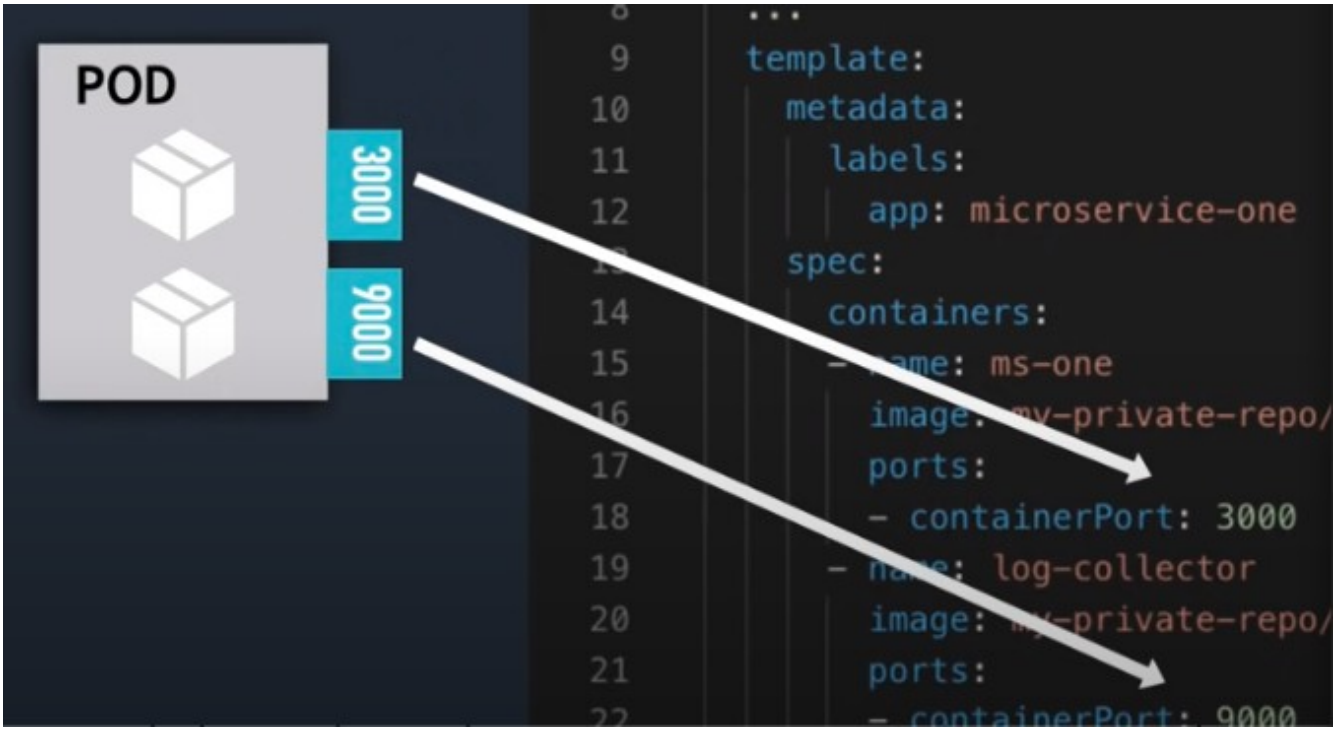
Service Types

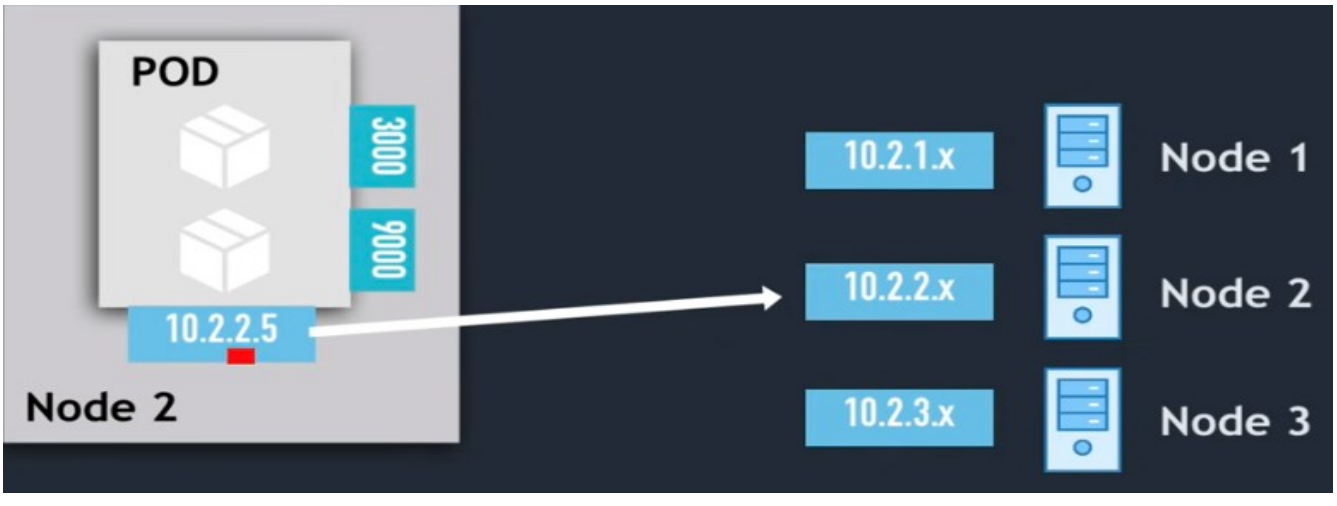
ClusterIP

ClusterIP services are needed because Pods are ephemeral and since they are destroyed frequently, we need to set a service that has a persistent IP address.

ClusterIP is a default service to enable network connectivity. Imagine you have a pod running an application on port 3000 and a logging application running on port 9000.

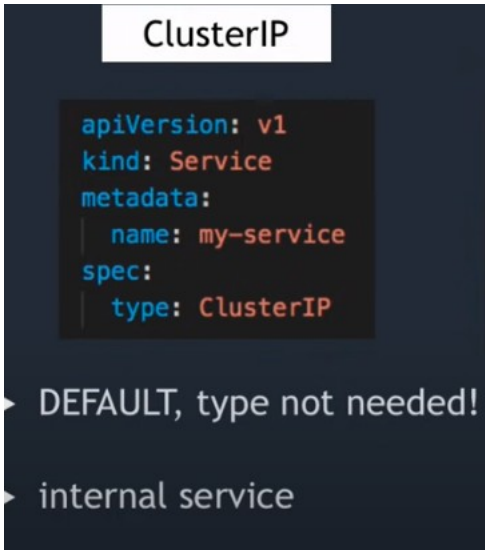


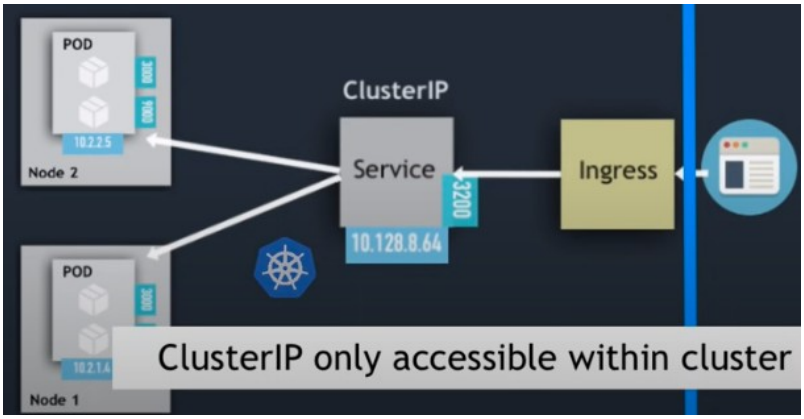
These two ports will now be open and accessible inside the pod. The pod will also get an IP from a CIDR range assigned to the node. Each node will get a range of different IP addresses.



In order to see your pod IP's we can execute the kubectl get pod -o wide command. By identifying the subnets, we can also see which node is assigned to the pods. The ClusterIP services would then give these nodes the ability to reach the internet through something called an ingress, which we will discuss later.

For now, just understand that a ClusterIP service type is an internal service that is the default type needed to enable network connectivity and is only accessible within the cluster.

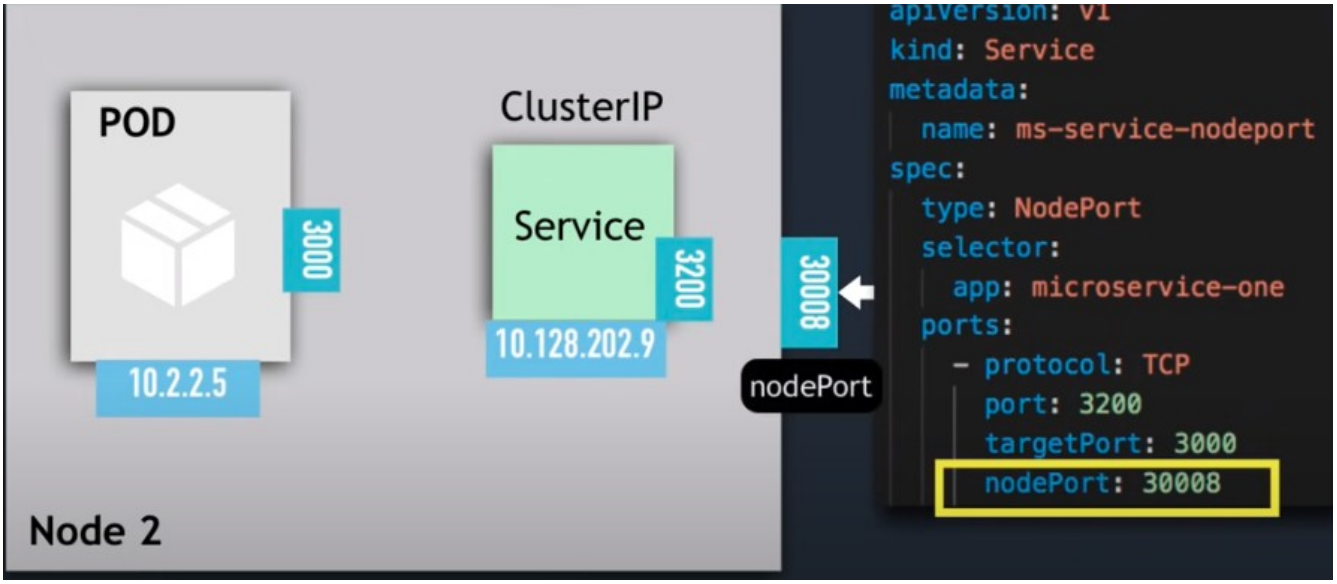




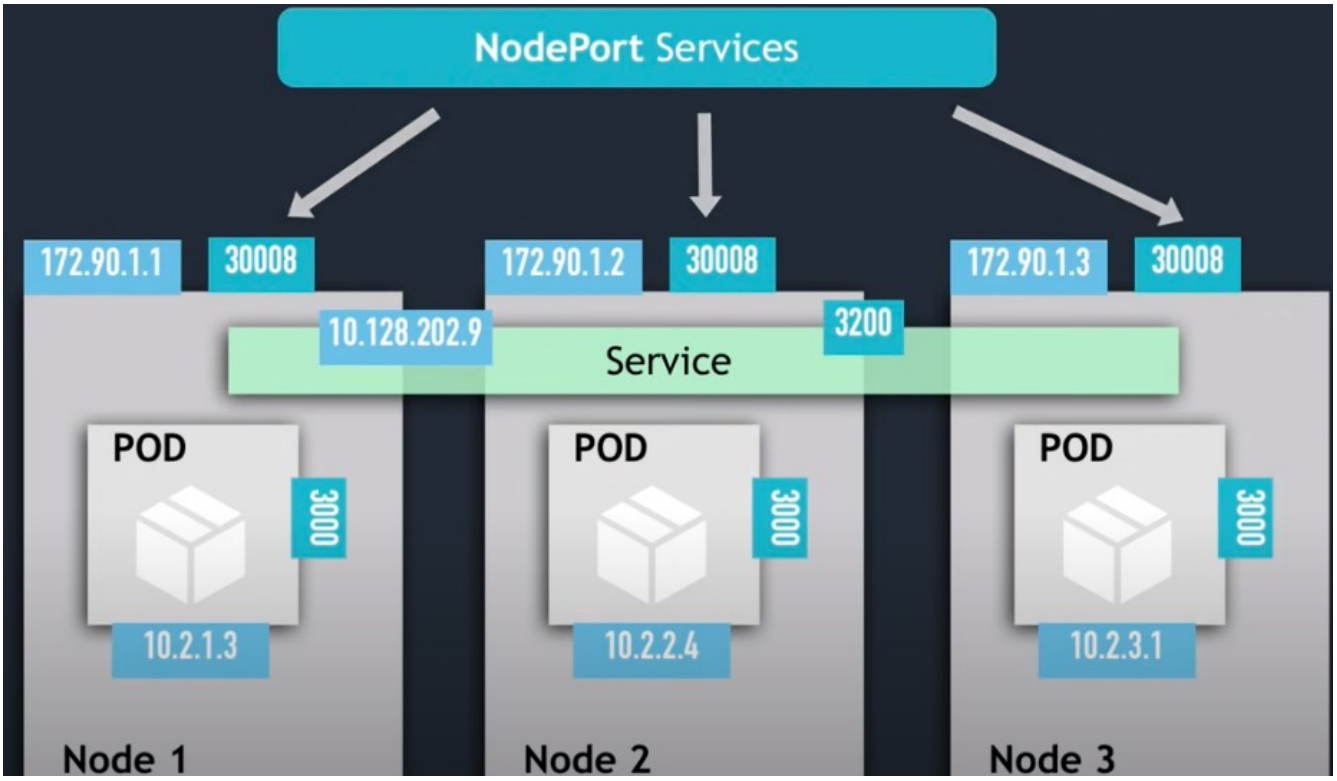
NodePort

NodePort is a service type that creates a static port that's accessible on each worker node in the cluster.

NodePort allows external traffic to come directly to the worker node on the specified port. This differs from ClusterIP as NodePort doesn't require an ingress resource to connect to the internet.

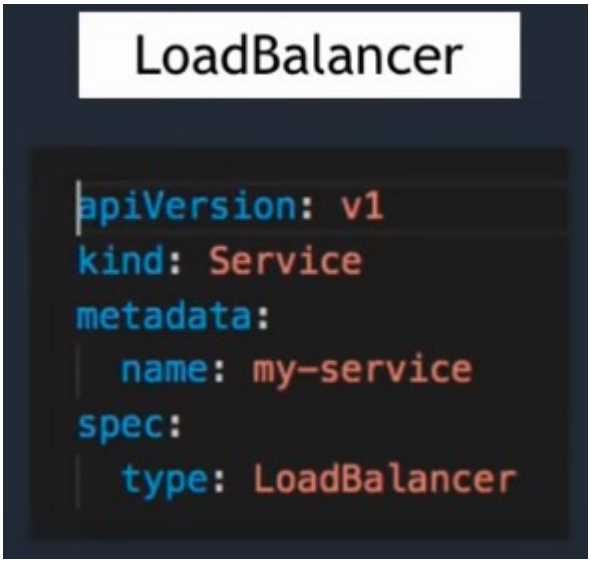


The NodePort value will use a range of ports that make browser requests possible. However, because we are opening ports to directly talk to other clients, this presents a security issue. This essentially means that we have nodes exposed to the world wide web!

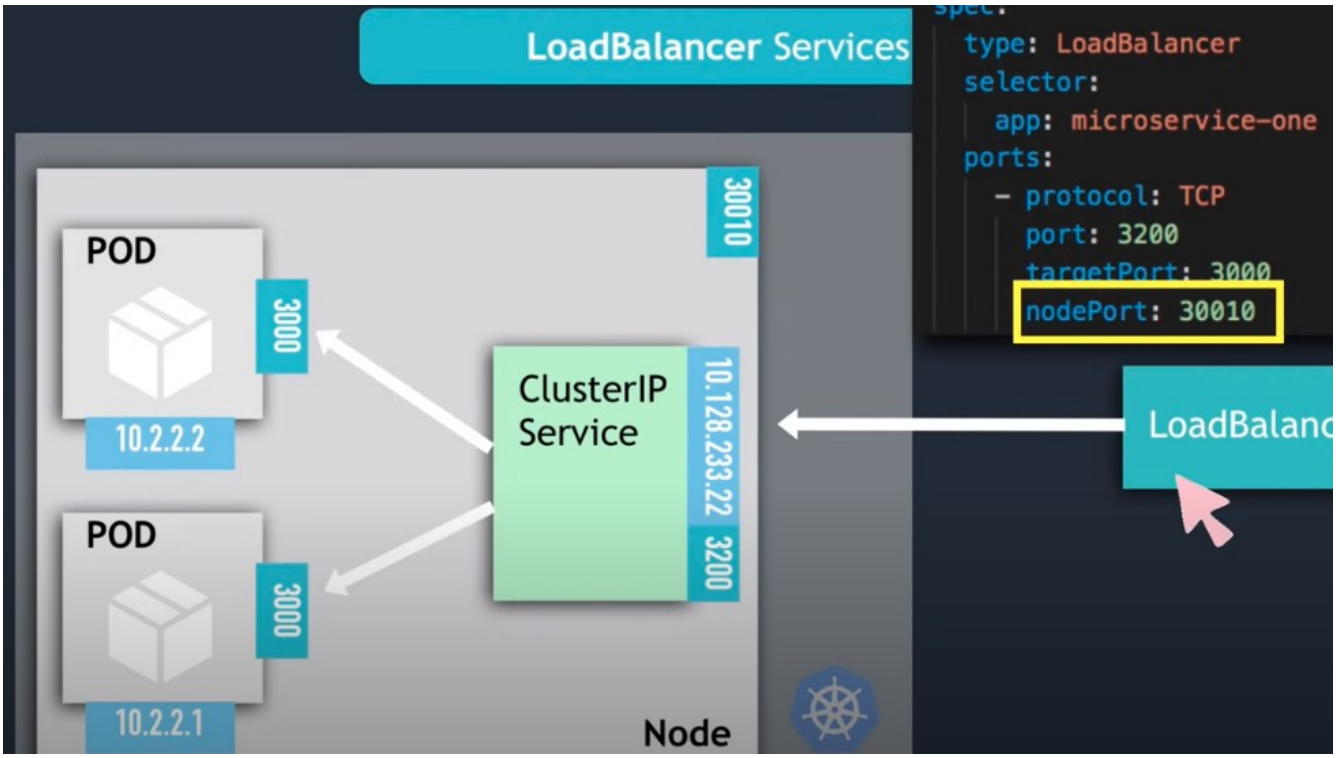


LoadBalancer

To rectify the security issues with NodePort services, we introduce a LoadBalancer service type. This is essentially the same as NodePort except we are using the LoadBalancer from our cloud provider to expose our nodes to the internet.



Each cloud provider has their own implementation of loadbalancing. Whenever we use a loadbalancing service, NodePort and ClusterIP are created automatically. Although our port 30010 is exposed, traffic from the internet will first hit the LoadBalancer, which can then direct the traffic to the destination node.



Hands On – Using the various service types

I. Create a ClusterIP service, apply it to the cluster, and verify.

```
dominickhrndz314@cloudshell:~$ cat cluster-service.yaml
kind: Service
apiVersion: v1
metadata:
  name: nginx-clusterip
spec:
  selector:
    app: nginx-clusterip
  type: ClusterIP
  ports:
  - protocol: TCP
    port: 80
    targetPort: 80
dominickhrndz314@cloudshell:~$ kubectl apply -f cluster-service.yaml
service/nginx-clusterip unchanged
dominickhrndz314@cloudshell:~$ kubectl get svc
NAME              TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)   AGE
kubernetes        ClusterIP   10.96.0.1      <none>        443/TCP   103s
nginx-clusterip   ClusterIP   10.104.96.63   <none>        80/TCP    55s
dominickhrndz314@cloudshell:~$
```

II. Create a NodePort service, apply it to the cluster, and verify.

```
dominickhrndz314@cloudshell:~$ cat nodeport-service.yaml
---
kind: Service
apiVersion: v1
metadata:
  name: nginx-nodeport
spec:
  selector:
    app: nginx-nodeport
  type: NodePort
  ports:
  - protocol: TCP
    port: 80
    targetPort: 80
dominickhrndz314@cloudshell:~$ kubectl apply -f nodeport-service.yaml
service/nginx-nodeport created
dominickhrndz314@cloudshell:~$ kubectl get svc
NAME              TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)        AGE
kubernetes        ClusterIP   10.96.0.1      <none>        443/TCP        3m7s
nginx-clusterip   ClusterIP   10.104.96.63   <none>        80/TCP         2m19s
nginx-nodeport    NodePort    10.103.186.5   <none>        80:31812/TCP   5s
dominickhrndz314@cloudshell:~$
```

III. Create a LoadBalancer service, apply it to the cluster, and verify.

```
dominickhrndz314@cloudshell:~$ cat loadbalancer-service.yaml
---
kind: Service
apiVersion: v1
metadata:
  name: nginx-loadbalancer
spec:
  selector:
    app: nginx-loadbalancer
  type: LoadBalancer
  ports:
    - port: 80
      targetPort: 80
dominickhrndz314@cloudshell:~$ kubectl apply -f loadbalancer-service.yaml
service/nginx-loadbalancer created
dominickhrndz314@cloudshell:~$ kubectl get svc
NAME                 TYPE           CLUSTER-IP     EXTERNAL-IP   PORT(S)        AGE
kubernetes           ClusterIP      10.96.0.1      <none>        443/TCP        5m39s
nginx-clusterip      ClusterIP      10.104.96.63   <none>        80/TCP         4m51s
nginx-loadbalancer   LoadBalancer   10.101.143.3   <pending>     80:31886/TCP   7s
nginx-nodeport       NodePort       10.103.186.5   <none>        80:31812/TCP   2m37s
dominickhrndz314@cloudshell:~$
```

IV. Create 3 nginx pods for each service and verify all 9 deploy.

```
dominickhrndz314@cloudshell:~$ cat clusterip-pods.yaml
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-clusterip
  labels:
    app: nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx-clusterip
  template:
    metadata:
      labels:
        app: nginx-clusterip
    spec:
      containers:
      - name: nginx
        image: nginx
        ports:
        - containerPort: 80

dominickhrndz314@cloudshell:~$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-clusterip-5699d6c8c9-519b8   1/1     Running   0          27s
nginx-clusterip-5699d6c8c9-v9xx8   1/1     Running   0          27s
nginx-clusterip-5699d6c8c9-wnp2j   1/1     Running   0          27s
dominickhrndz314@cloudshell:~$
```

```
dominickhrndz314@cloudshell:~$ cat nodeport-pods.yaml
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-nodeport
  labels:
    app: nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx-nodeport
  template:
    metadata:
      labels:
        app: nginx-nodeport
    spec:
      containers:
      - name: nginx
        image: nginx
        ports:
        - containerPort: 80
dominickhrndz314@cloudshell:~$ kubectl apply -f nodeport-pods.yaml
deployment.apps/nginx-nodeport created
dominickhrndz314@cloudshell:~$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-clusterip-5699d6c8c9-5l9b8    1/1     Running   0          2m15s
nginx-clusterip-5699d6c8c9-v9xx8    1/1     Running   0          2m15s
nginx-clusterip-5699d6c8c9-wnp2j    1/1     Running   0          2m15s
nginx-nodeport-84d8bb796f-6fv2x     1/1     Running   0          6s
nginx-nodeport-84d8bb796f-mmqcq     1/1     Running   0          6s
nginx-nodeport-84d8bb796f-t46sj     1/1     Running   0          6s
```

```
dominickhrndz314@cloudshell:~$ cat loadbalancer-pods.yaml
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-loadbalancer
  labels:
    app: nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx-loadbalancer
  template:
    metadata:
      labels:
        app: nginx-loadbalancer
    spec:
      containers:
      - name: nginx
        image: nginx
        ports:
        - containerPort: 80
dominickhrndz314@cloudshell:~$ kubectl apply -f loadbalancer-pods.yaml
deployment.apps/nginx-loadbalancer created
dominickhrndz314@cloudshell:~$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-clusterip-5699d6c8c9-5l9b8    1/1     Running   0          3m56s
nginx-clusterip-5699d6c8c9-v9xx8    1/1     Running   0          3m56s
nginx-clusterip-5699d6c8c9-wnp2j    1/1     Running   0          3m56s
nginx-loadbalancer-5c7c865f9d-bhfgv 1/1     Running   0          5s
nginx-loadbalancer-5c7c865f9d-hcfb8 1/1     Running   0          5s
nginx-loadbalancer-5c7c865f9d-p7196 1/1     Running   0          5s
nginx-nodeport-84d8bb796f-6fv2x     1/1     Running   0          107s
nginx-nodeport-84d8bb796f-mmqcq     1/1     Running   0          107s
nginx-nodeport-84d8bb796f-t46sj     1/1     Running   0          107s
dominickhrndz314@cloudshell:~$
```

IV. Verify all services and discuss why LoadBalancer is stuck in pending.

```
dominickhrndz314@cloudshell:~$ kubectl get svc
NAME                 TYPE           CLUSTER-IP     EXTERNAL-IP   PORT(S)        AGE
kubernetes           ClusterIP      10.96.0.1      <none>        443/TCP        15m
nginx-clusterip      ClusterIP      10.104.96.63   <none>        80/TCP         14m
nginx-loadbalancer   LoadBalancer   10.101.143.3   <pending>     80:31886/TCP   10m
nginx-nodeport       NodePort       10.103.186.5   <none>        80:31812/TCP   12m
dominickhrndz314@cloudshell:~$
```

The LoadBalancer service type remains in pending status unless your cluster has integration with a cloud provider that provisions one for you through GCP or AWS.

If you can see all 9 of your pods and your 3 services list, you have completed this training. Please remember to delete all of you deployments using `kubectl delete pods --all` and `kubectl delete svc --all`.