CCGC 5001 - Virtualization

Module 9: Managing Apps with K8s

Kubernetes in the real world

HUMBER

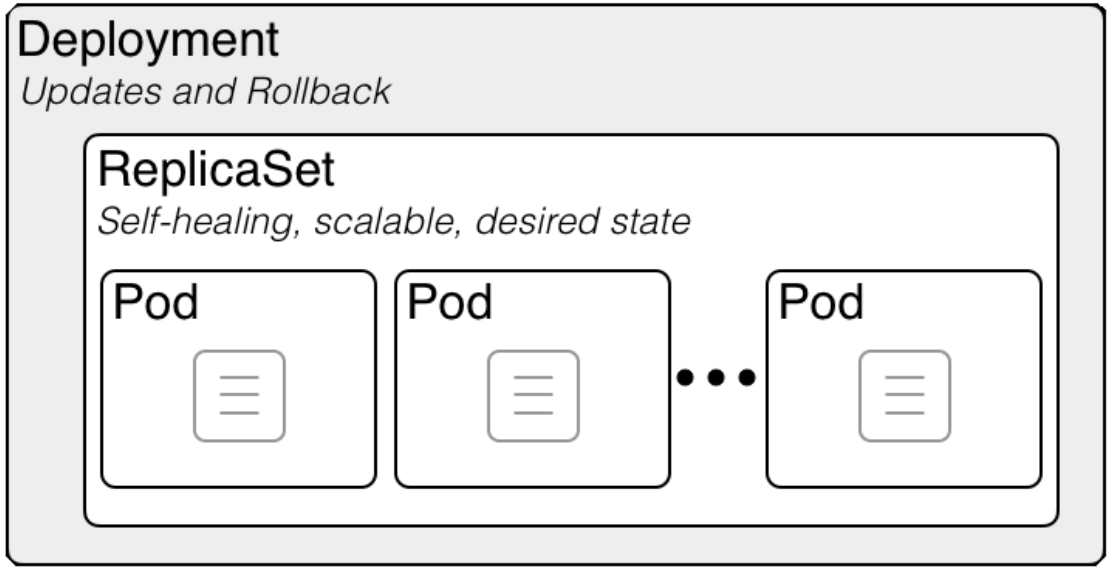
Module objectives

At the end of this module, you should be able to:

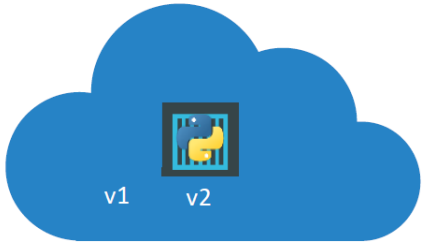
- Describe the role of deployment
- List use cases of deployment
- Apply deployment strategies
- Explain networking in K8s
- Create and deploy service
- Explain different type of service

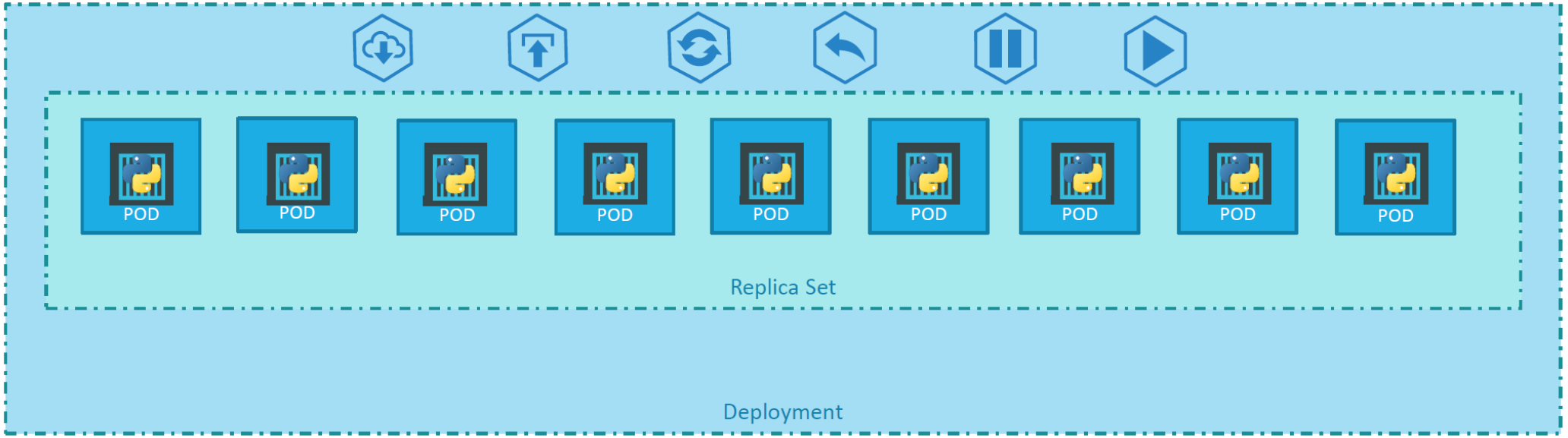
Deployment

- Deployment is really a wrapper object to a ReplicaSet
- Deployment provides declarative updates for Pods and ReplicaSets
- Deployment describes desired state
- Deployment controller changes the actual state to the desire stated
- Kubernetes deployment provides rolling updates and rollback functionality



Deployment


v1
v2


POD
POD
POD
POD
POD
POD
POD
POD
POD
Replica Set
Deployment

Deployment use case

- Create a Deployment to rollout a ReplicaSet
  - ReplicaSet creates Pods in the background
  - Check the status of the rollout to see if it succeeds or not
- Declare the new state of the Pods
  - Update the PodTemplateSpec of the Deployment
  - New ReplicaSet is created
  - Manages moving the Pods from the old ReplicaSet to the new one
- Rollback to an earlier Deployment revision
  - If the current state of the Deployment is not stable
  - Each rollback updates the revision of the Deployment
- Scale up the Deployment to facilitate more load

Deployment use case

- Pause the Deployment
  - To apply fixes to its PodTemplateSpec
- Use the status of the Deployment
  - To indicate that a rollout has stuck
- Clean up older ReplicaSets that you don't need anymore

Deployment definition

- kubectl create -f deployment.yaml
- kubectl get deployments
- kubectl get replicaset
- kubectl get pods
- kubectl get all

```yaml
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: myapp-deployment
  labels:
      app: myapp
      type: front-end
spec:
  template:
    metadata:
      name: myapp-pod
      labels:
        app: myapp
        type: front-end
    spec:
      containers:
      - name: nginx-container
        image: nginx

  replicas: 3
  selector:
    matchLabels:
      type: front-end
```

Deployment definition

- kubectl create -f deployment.yml
- kubectl get deployments
- kubectl get replicaset
- kubectl get pods
- kubectl get all
- kubectl describe deployment myapp-deployment
- kubectl rollout status deployment myapp-deployment

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: myapp-deployment
  labels:
    tier: front-end
    app: nginx
spec:
  template:
    metadata:
      name: myapp-pod
      labels:
        tier: front-end
        app: myapp
    spec:
      containers:
        - name: nginx
          image: nginx
  replicas: 3
  selector:
    matchLabels:
      app: myapp
```

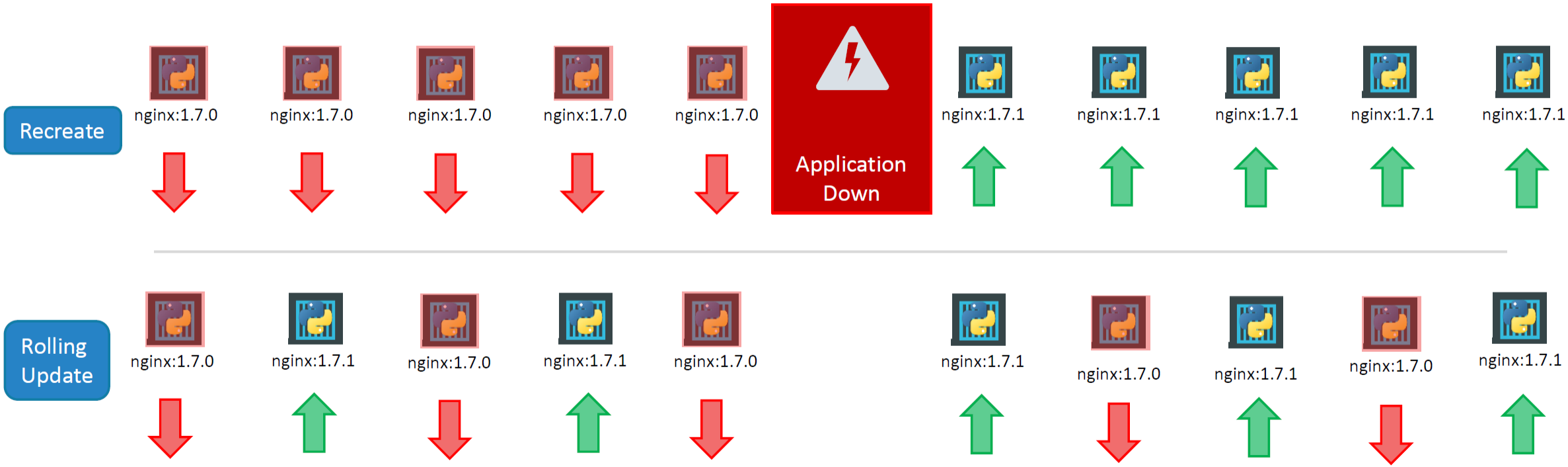
Rollout and versioning

Revision 1

Revision 2

Note: A Deployment's rollout is triggered if and only if the Deployment's Pod template (that is, .spec.template) is changed, for example if the labels or container images of the template are updated. Other updates, such as scaling the Deployment, do not trigger a rollout.

Deployment strategy

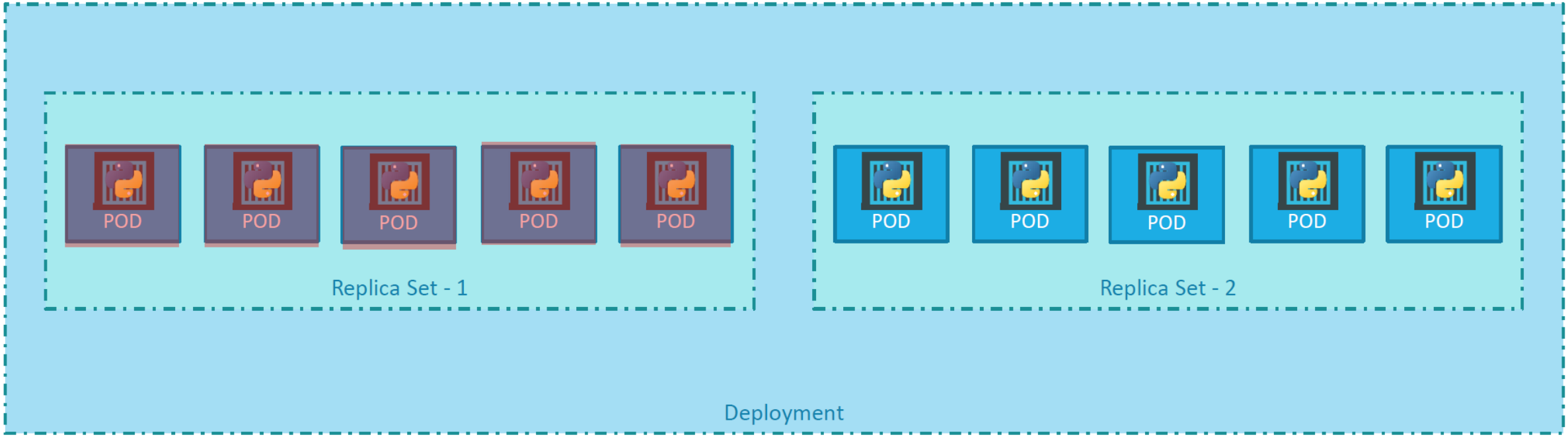


Deployment definition

- kubectl create -f deployment.yml
- kubectl rollout status deployment myapp-deployment
- kubectl rollout history deployment myapp-deployment
- kubectl delete deployment myapp-deployment
- kubectl create -f deployment.yml --record
- kubectl rollout history deployment myapp-deployment
- kubectl describe deployment myapp-deployment

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: myapp-deployment
  labels:
    tier: front-end
    app: nginx
spec:
  template:
    metadata:
      name: myapp-pod
      labels:
        tier: front-end
        app: myapp
    spec:
      containers:
        - name: nginx
          image: nginx
  replicas: 6
  selector:
    matchLabels:
      app: myapp
```

Upgrades




```
[ec2-user@ip-172-31-86-206 deployment]$ kubectl get rs
NAME                          DESIRED   CURRENT   READY   AGE
myapp-deployment-579cdfdf77   0         0         0       28m
myapp-deployment-58755fbddb   0         0         0       11m
myapp-deployment-8c5f9977c    6         6         6       18m
```

Upgrades

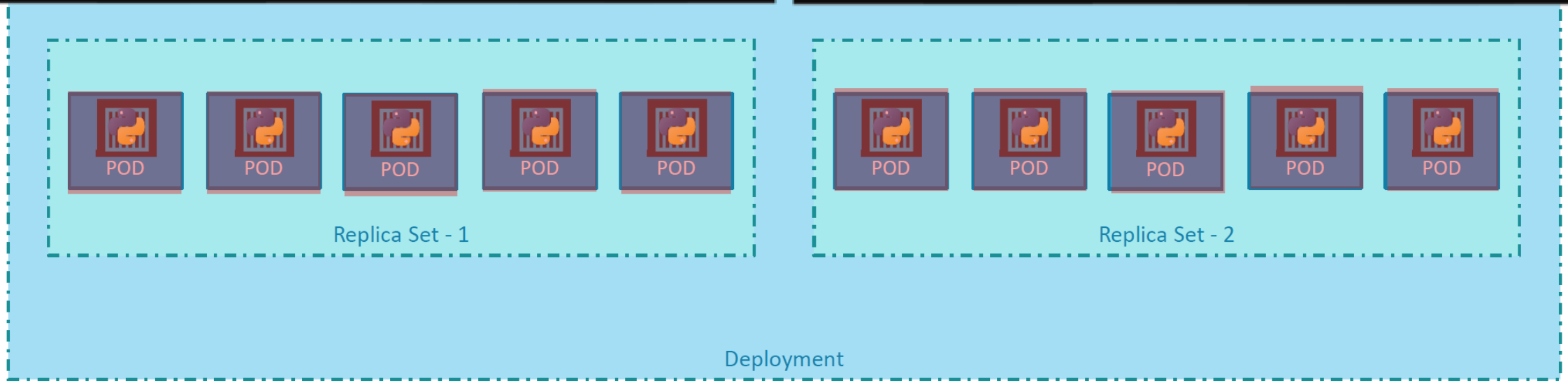
- kubectl edit deployment myapp-deployment --record
- kubectl rollout status deployment myapp-deployment
- kubectl describe deployment myapp-deployment
- kubectl rollout history deployment myapp-deployment
- kubectl set image deployment myapp-deployment nginx=nginx:1.20-perl --record
- kubectl rollout status deployment myapp-deployment
- kubectl rollout history deployment myapp-deployment

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: myapp-deployment
  labels:
    tier: front-end
    app: nginx
spec:
  template:
    metadata:
      name: myapp-pod
      labels:
        tier: front-end
        app: myapp
    spec:
      containers:
        - name: nginx
          image: nginx
  replicas: 6
  selector:
    matchLabels:
      app: myapp
```

Rollback

```
> kubectl get replicasets
NAME                          DESIRED   CURRENT   READY   AGE
myapp-deployment-67c749c58c   0         0         0       22m
myapp-deployment-7d57dbdb8d   5         5         5       20m
```

```
> kubectl get replicasets
NAME                          DESIRED   CURRENT   READY   AGE
myapp-deployment-67c749c58c   5         5         5       22m
myapp-deployment-7d57dbdb8d   0         0         0       20m
```



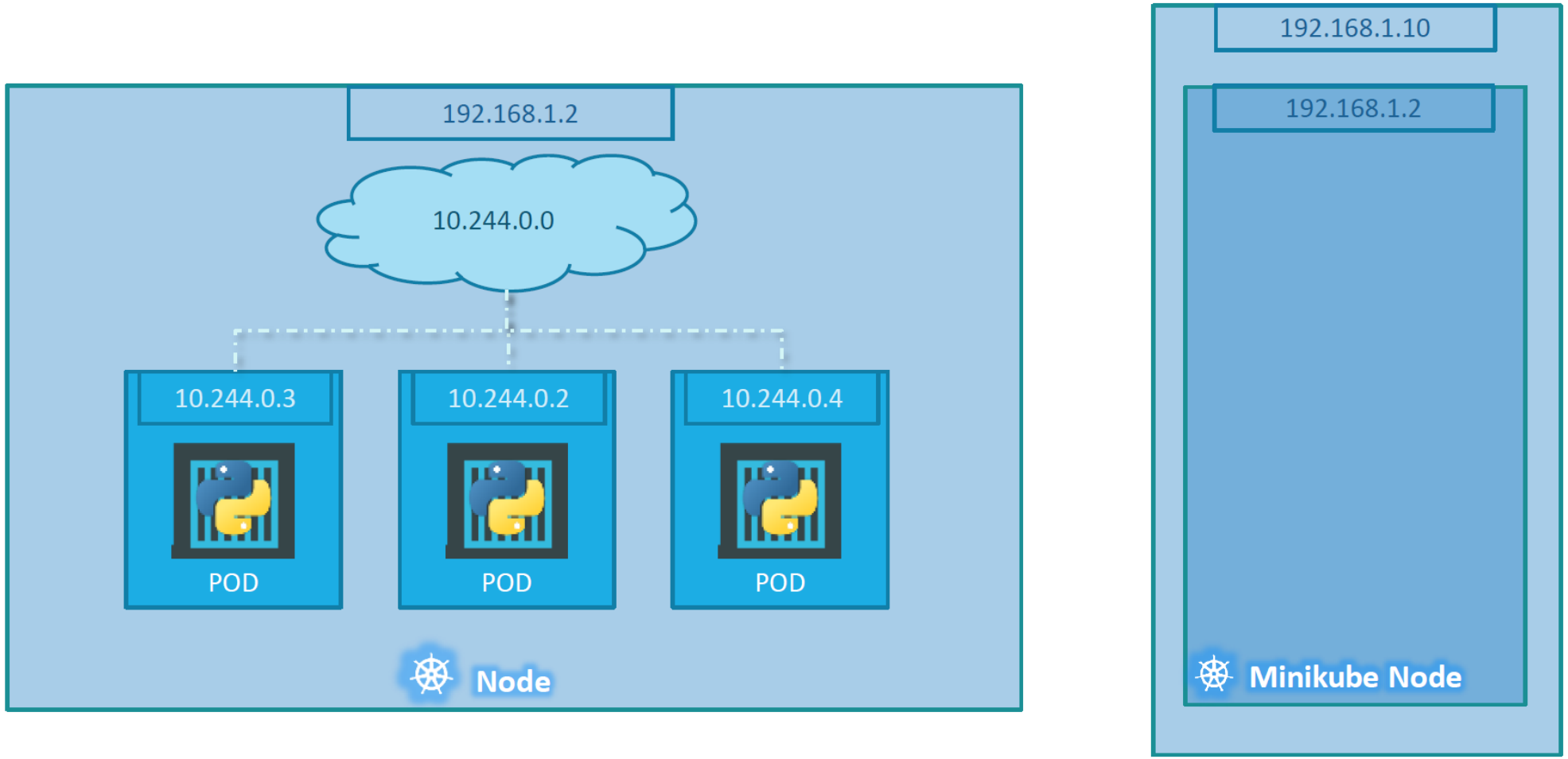

```
> kubectl rollout undo deployment/myapp-deployment
deployment “myapp-deployment” rolled back
```

Rollback

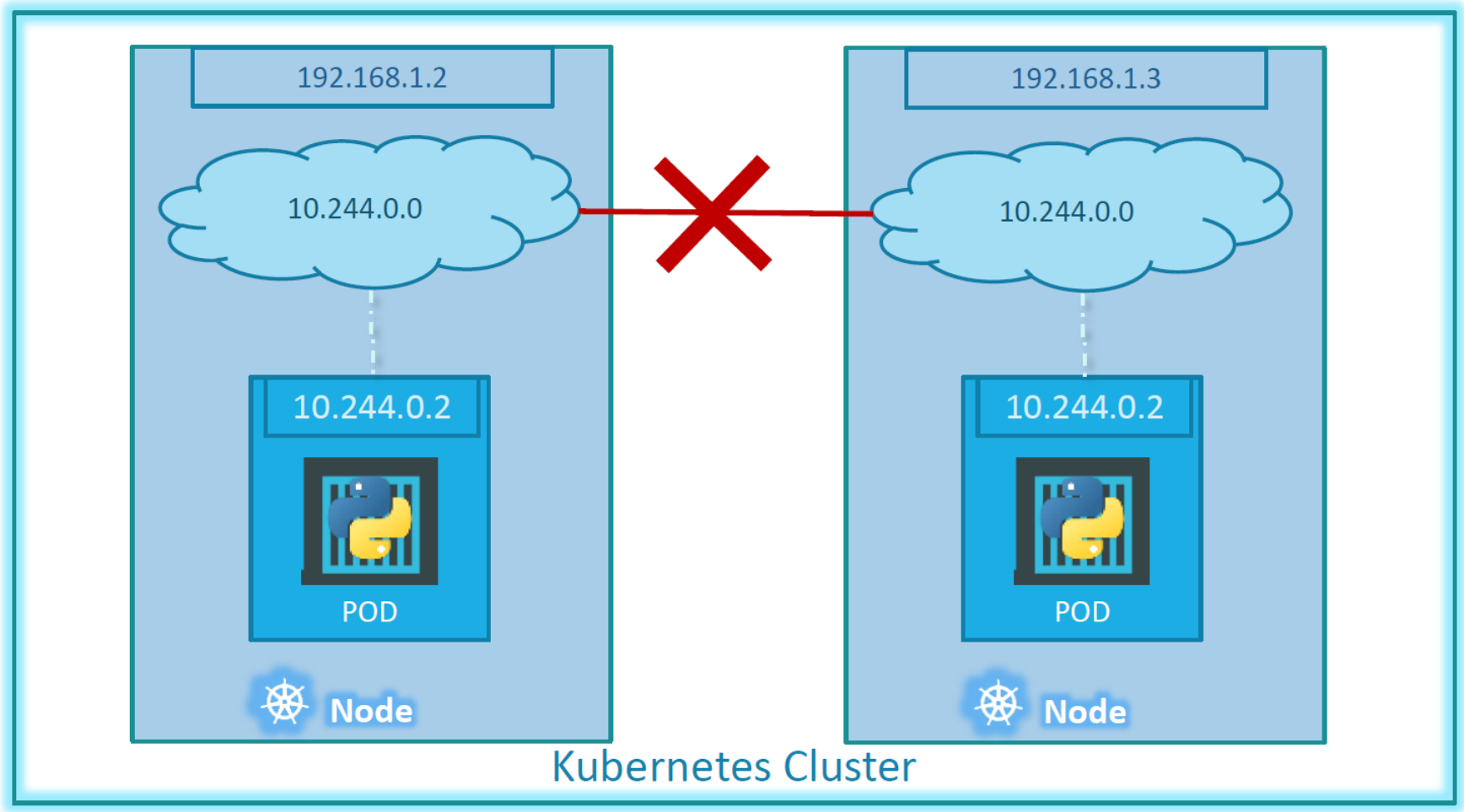
- kubectl rollout undo deployment myapp-deployment
- kubectl rollout status deployment myapp-deployment
- kubectl rollout history deployment myapp-deployment
- kubectl describe deployment myapp-deployment

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: myapp-deployment
  labels:
    tier: front-end
    app: nginx
spec:
  template:
    metadata:
      name: myapp-pod
      labels:
        tier: front-end
        app: myapp
    spec:
      containers:
        - name: nginx
          image: nginx
  replicas: 6
  selector:
    matchLabels:
      app: myapp
```

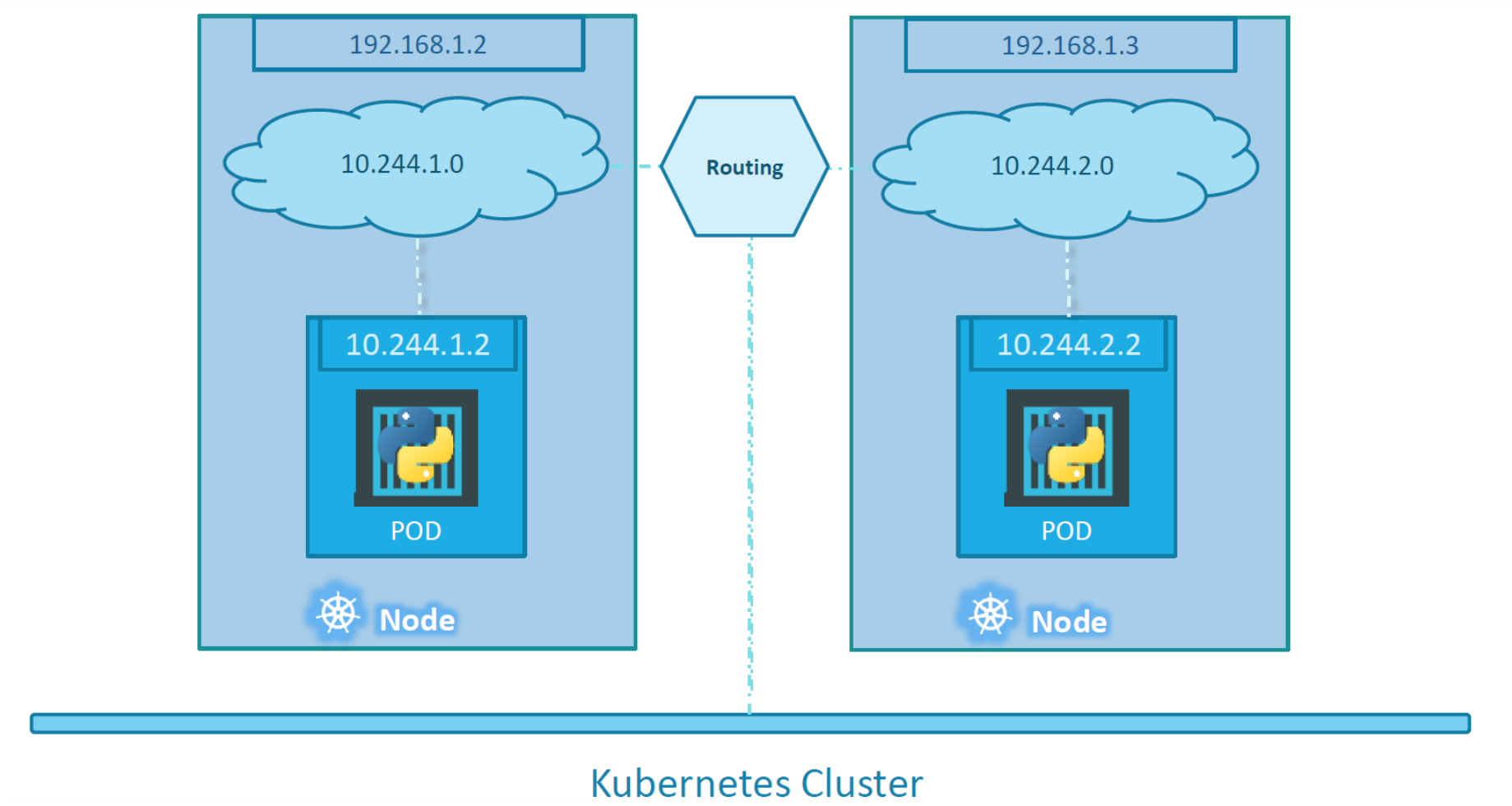
Kubernetes Networking



Cluster Networking

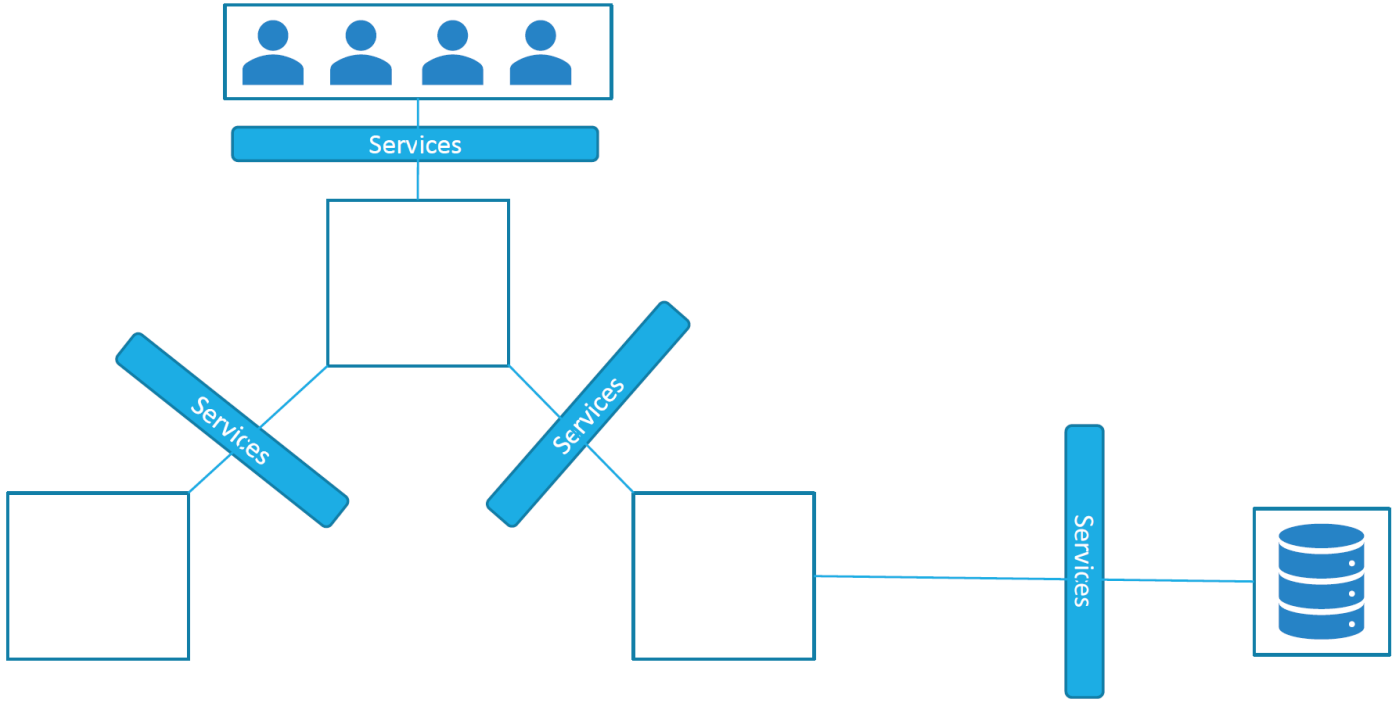


Cluster Networking



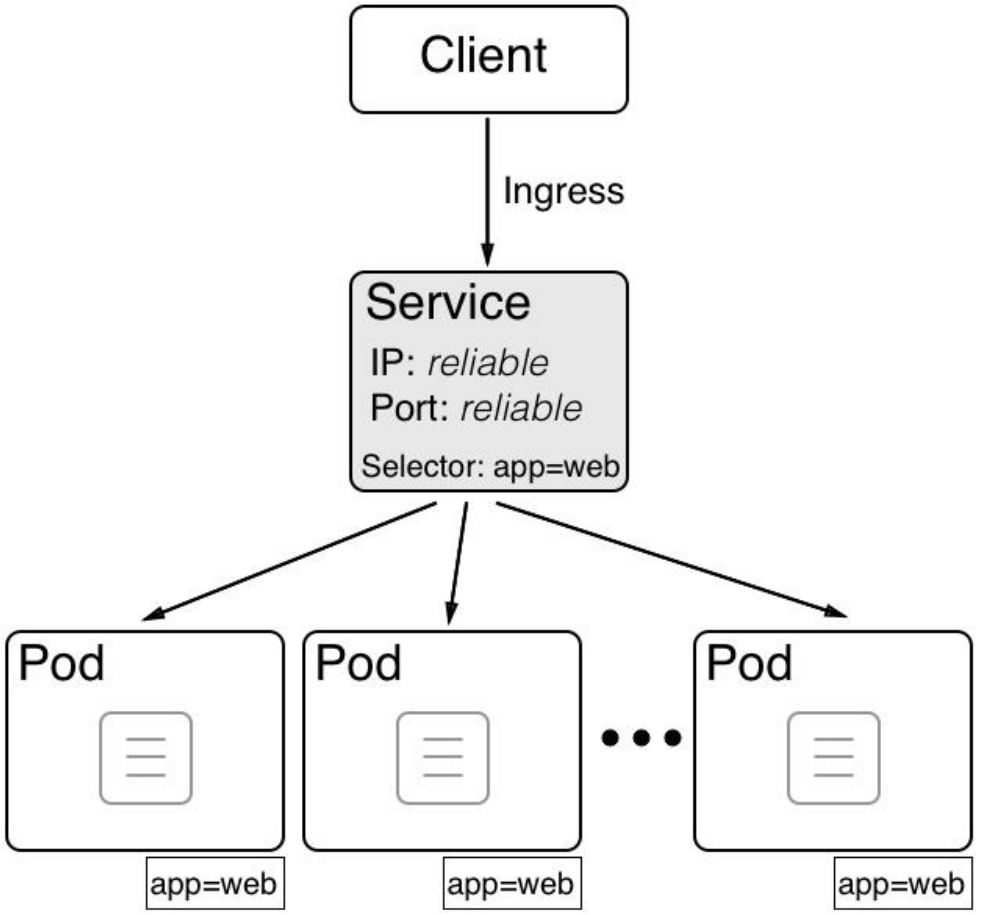
Service

How do you work with applications consisting of more than one application service?

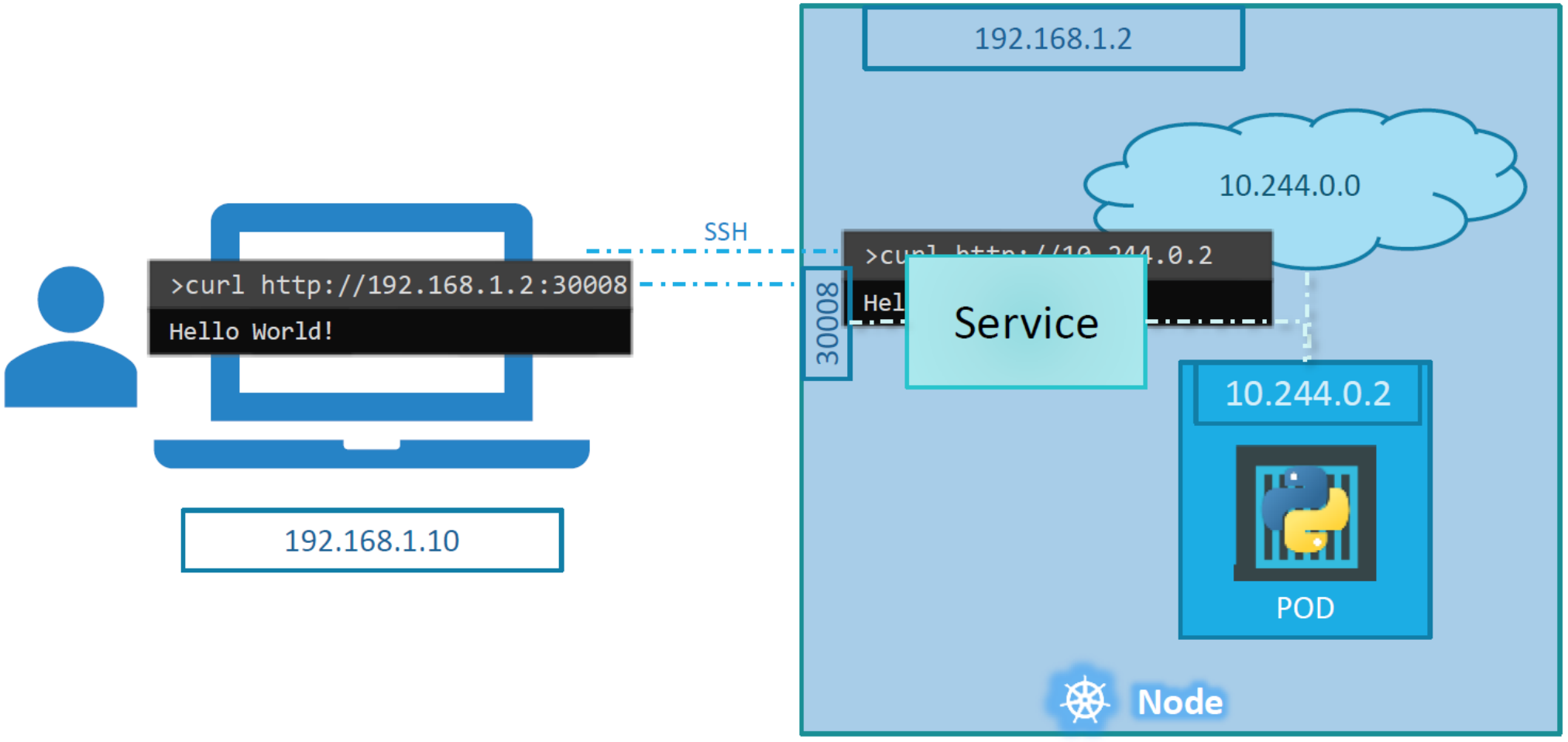


Service

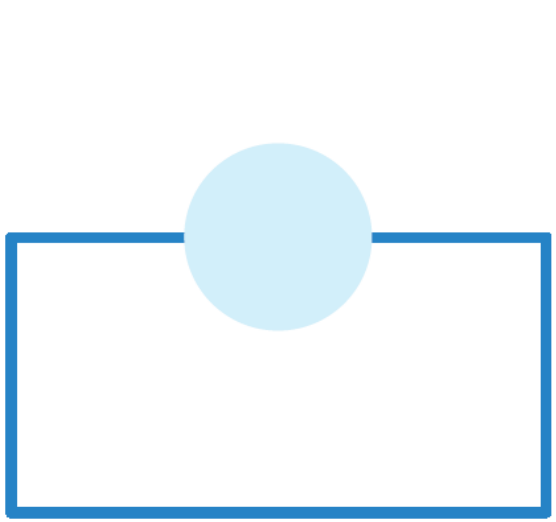
- Kubernetes service provide stable endpoints to ReplicaSets or Deployments
- Provides reliable cluster-wide IP address, also called a virtual IP (VIP), as well as reliable port
- Pods are determined by the Selector defined in the service specifications
- Selectors are based on labels
- Kubernetes objects can have zero to many labels



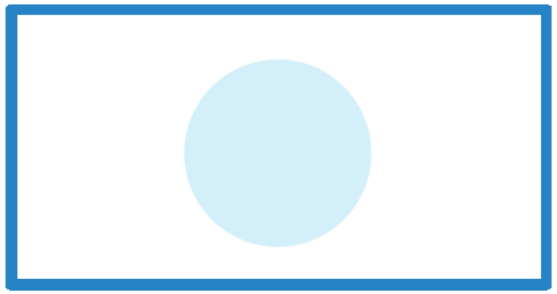
Service


>curl http://192.168.1.2:30008
Hello World!
192.168.1.10
SSH
192.168.1.2
10.244.0.0
30008
Service
10.244.0.2
POD
Node

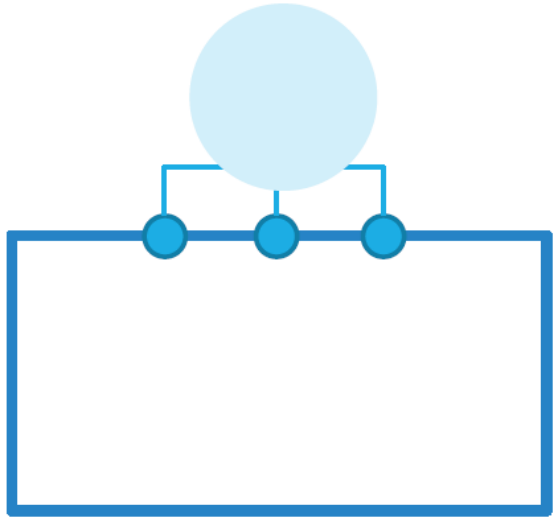
Service types



NodePort

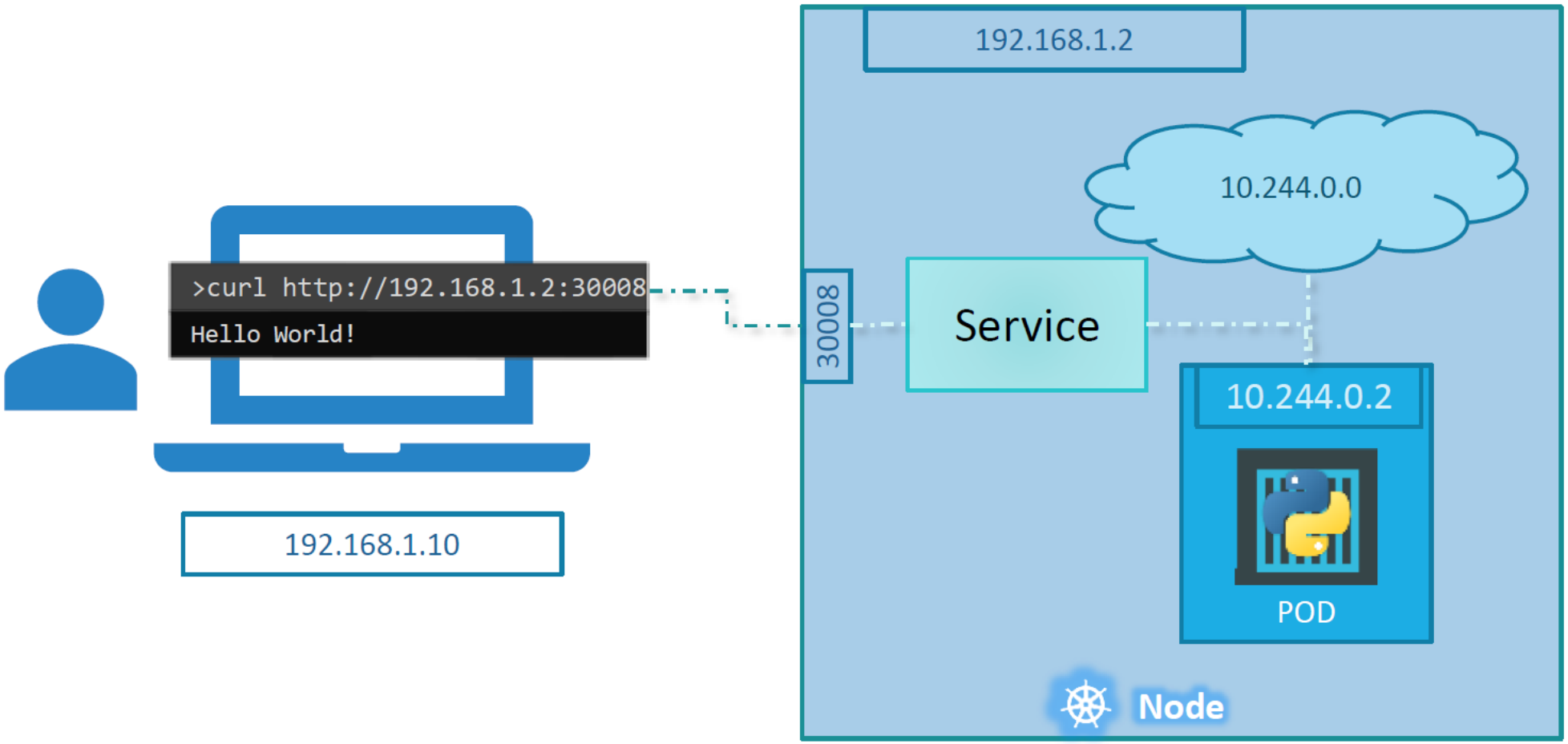


ClusterIP

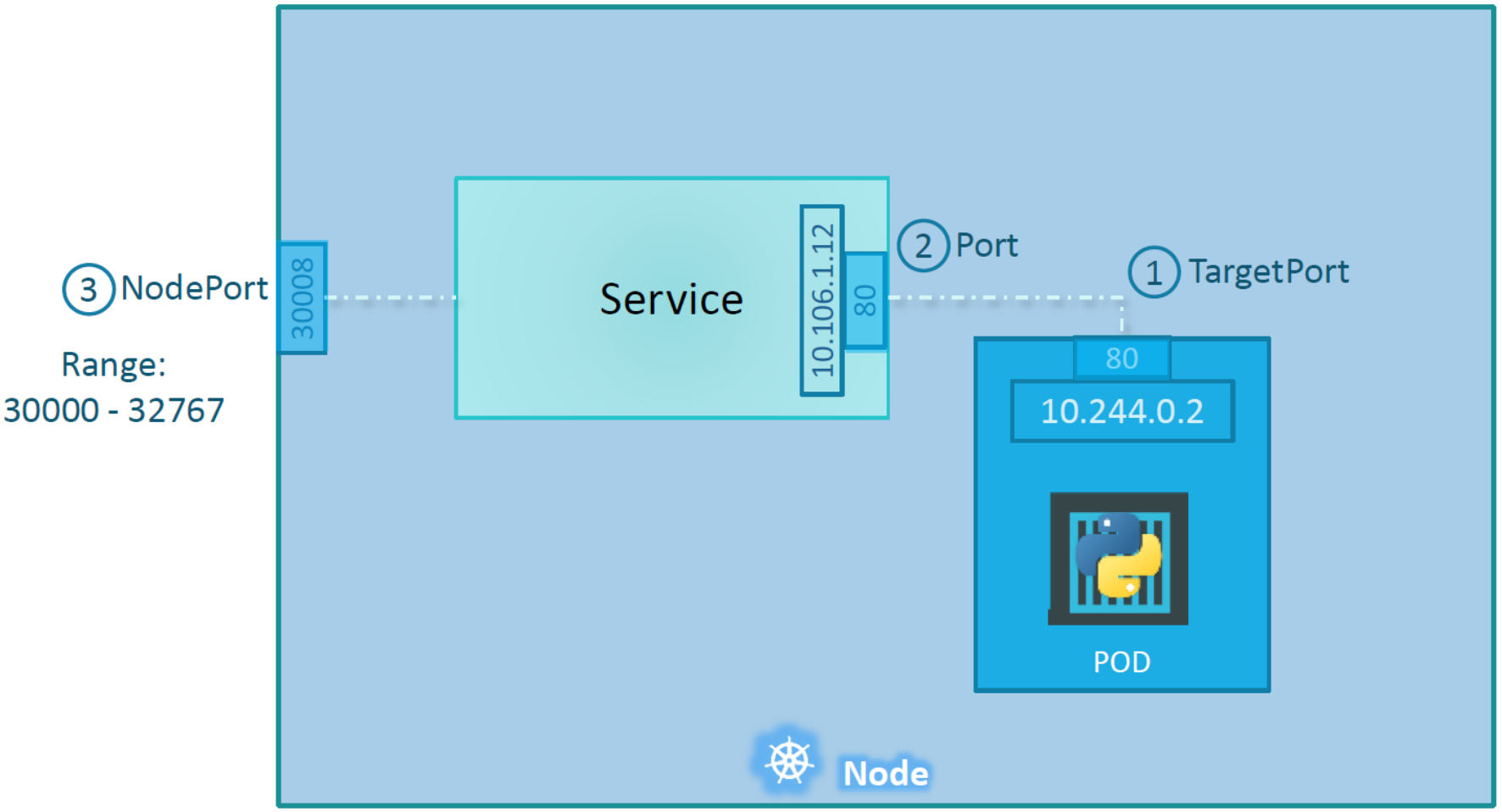


LoadBalancer

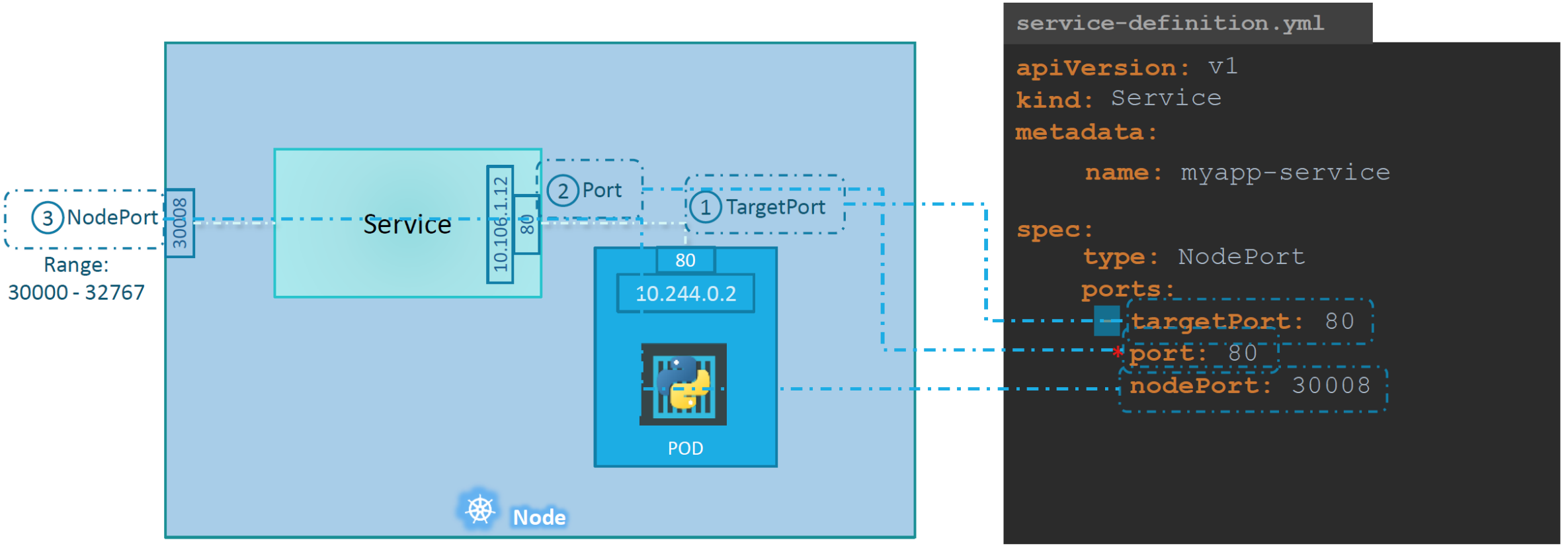
Service - NodePort



Service - NodePort



Service - NodePort



Service - NodePort

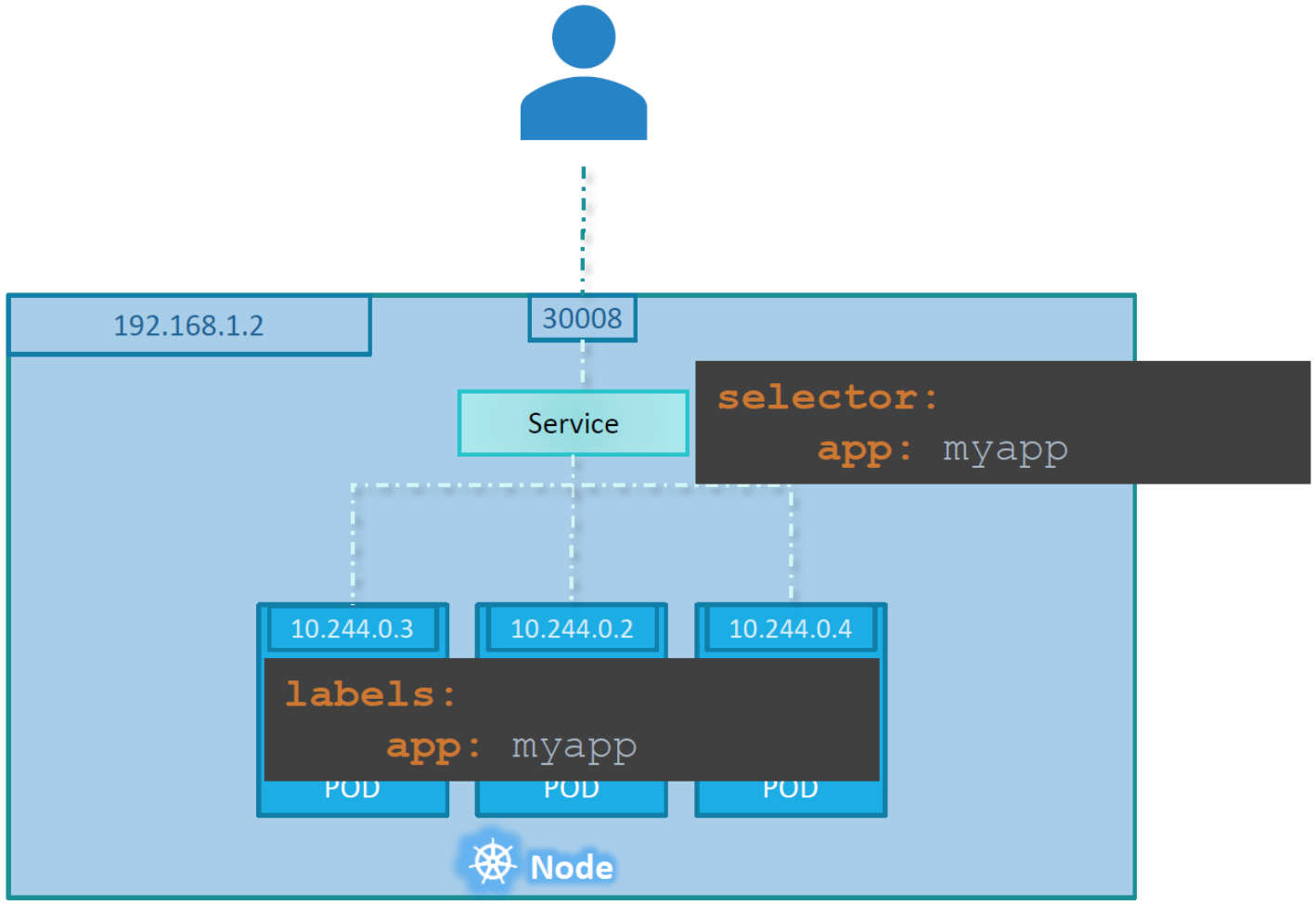
service-definition.yml

```yaml
apiVersion: v1
kind: Service
metadata:
    name: myapp-service

spec:
    type: NodePort
    ports:
     - targetPort: 80
       port: 80
       nodePort: 30008
    selector:
       app: myapp
       type: front-end
```

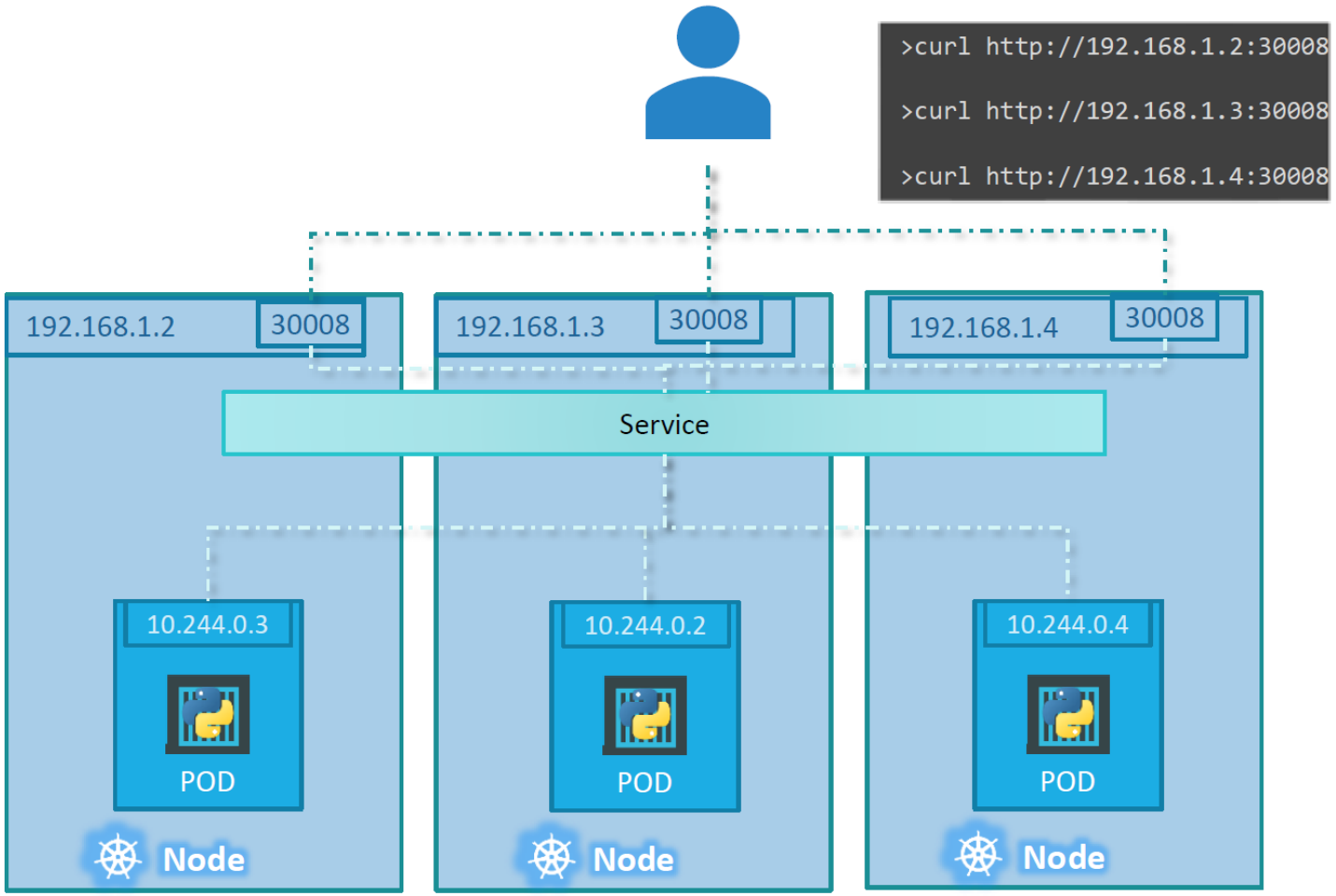
pod-definition.yml

```yaml
apiVersion: v1
kind: Pod

metadata:
 name: myapp-pod
 labels:
    app: myapp
    type: front-end
spec:
  containers:
  - name: nginx-container
    image: nginx
```

Service - NodePort



Service - NodePort



Service - NodePort

service-definition.yml

```yaml
apiVersion: v1
kind: Service
metadata:
    name: myapp-service

spec:
    type: NodePort
    ports:
     - targetPort: 80
       port: 80
       nodePort: 30008
    selector:
       app: myapp
       type: front-end
```

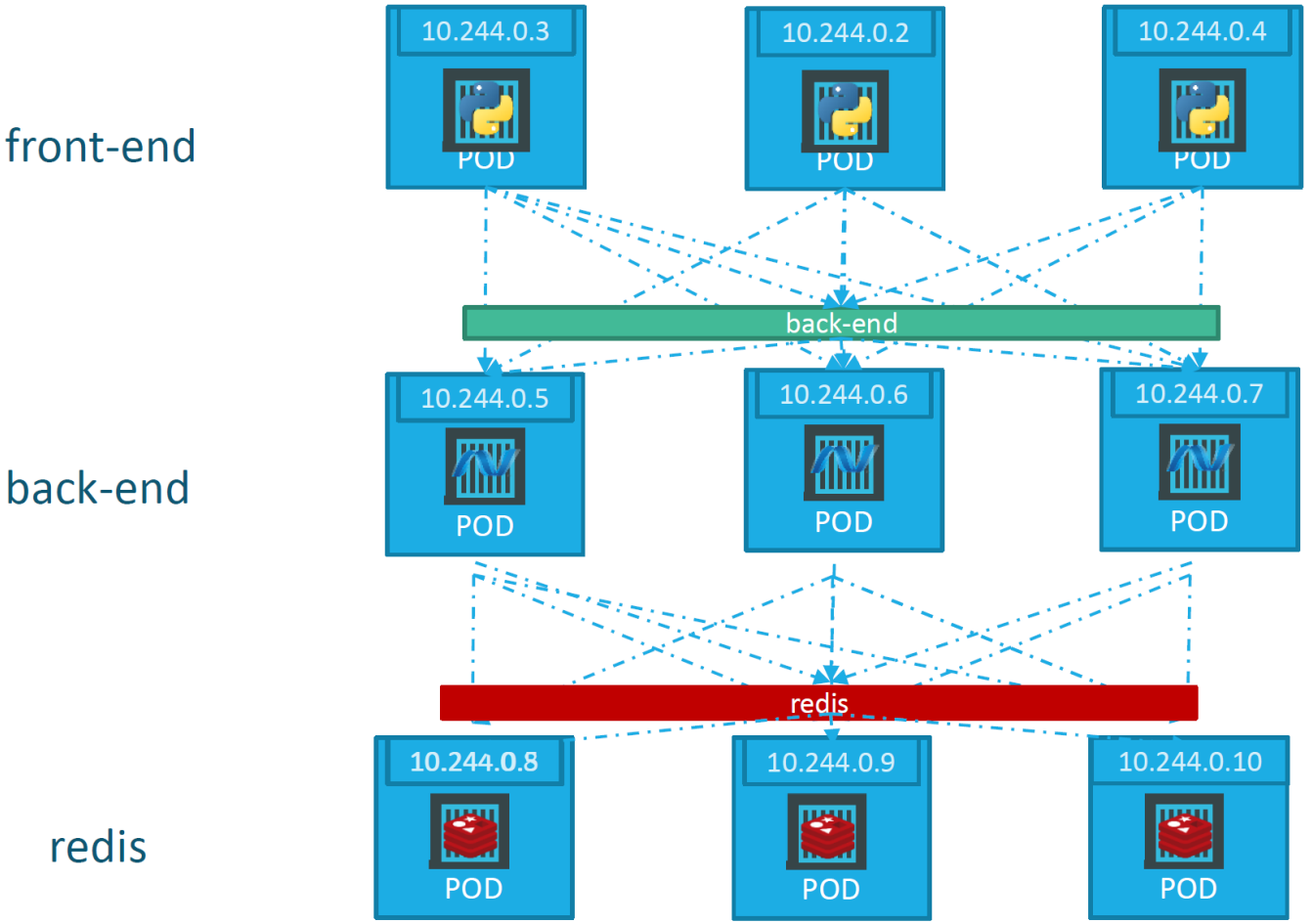
```
> kubectl create -f service-definition.yml
service "myapp-service" created
```

```
> kubectl get services
NAME            TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
kubernetes      ClusterIP   10.96.0.1        <none>        443/TCP          16d
myapp-service   NodePort    10.106.127.123   <none>        80:30008/TCP     5m
```

```
> curl http://192.168.1.2:30008
<html>
<head>
<title>Welcome to nginx!</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
```

Note: To get minikube node IP address:
minikube service myapp-service --url

Service - ClusterIP



Service - ClusterIP

service-definition.yml

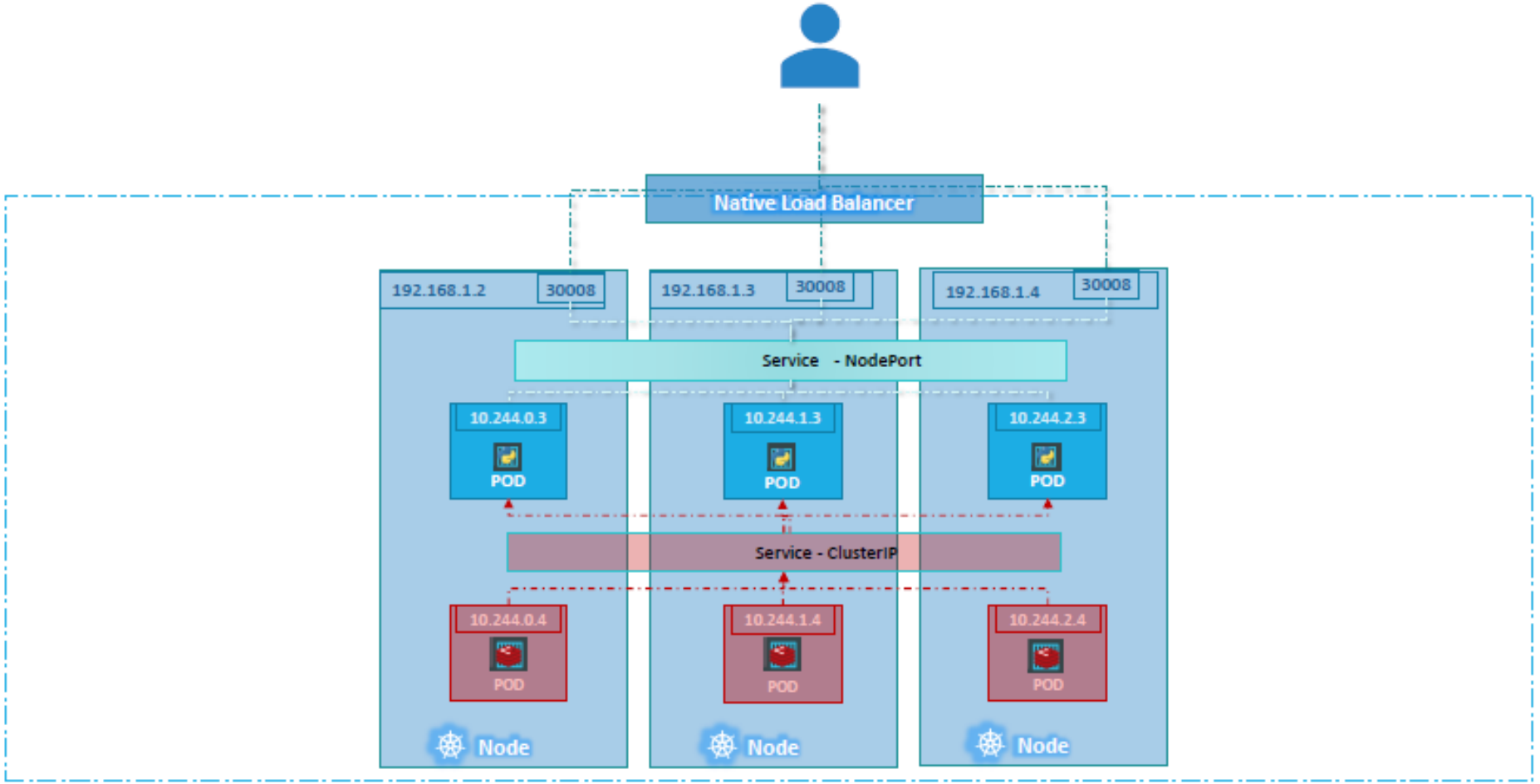
```yaml
apiVersion: v1
kind: Service
metadata:
    name: back-end

spec:
    type: ClusterIP
    ports:
     - targetPort: 80
       port: 80

    selector:
       app: myapp
       type: back-end
```

pod-definition.yml

```yaml
       app: myapp
       type: back-end
spec:
  containers:
  - name: nginx-container
    image: nginx
```

```
> kubectl create -f service-definition.yml
service “back-end" created

> kubectl get services
NAME          TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)    AGE
kubernetes    ClusterIP   10.96.0.1        <none>        443/TCP    16d
back-end      ClusterIP   10.106.127.123   <none>        80/TCP     2m
```

Service - Load Balancer



Service - Load Balancer

service-definition.yml

```yaml
apiVersion: v1
kind: Service
metadata:
    name: front-end

spec:
    type: LoadBalancer
    ports:
     - targetPort: 80
       port: 80

    selector:
        app: myapp
        type: front-end
```

```
> kubectl create -f service-definition.yml
service "front-end" created
```

```
> kubectl get services
NAME          TYPE          CLUSTER-IP       EXTERNAL-IP   PORT(S)    AGE
kubernetes    ClusterIP     10.96.0.1        <none>        443/TCP    16d
front-end     LoaBalancer   10.106.127.123   <Pending>     80/TCP     2m
```

Module summary

In summary, in this module, you learned:

- Deployments and deployment strategies
- How to build and deploy K8s deployments
- Networking in K8s
- Create and deploy different type of service

Thank you