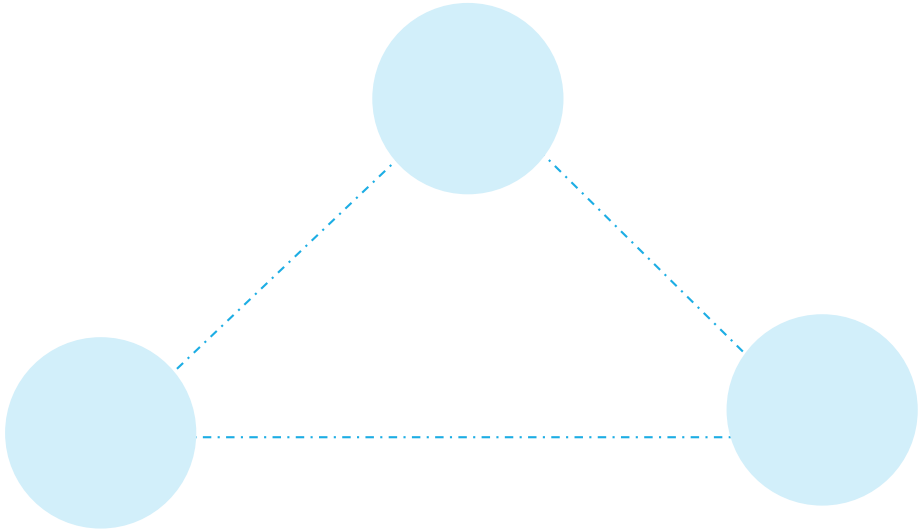
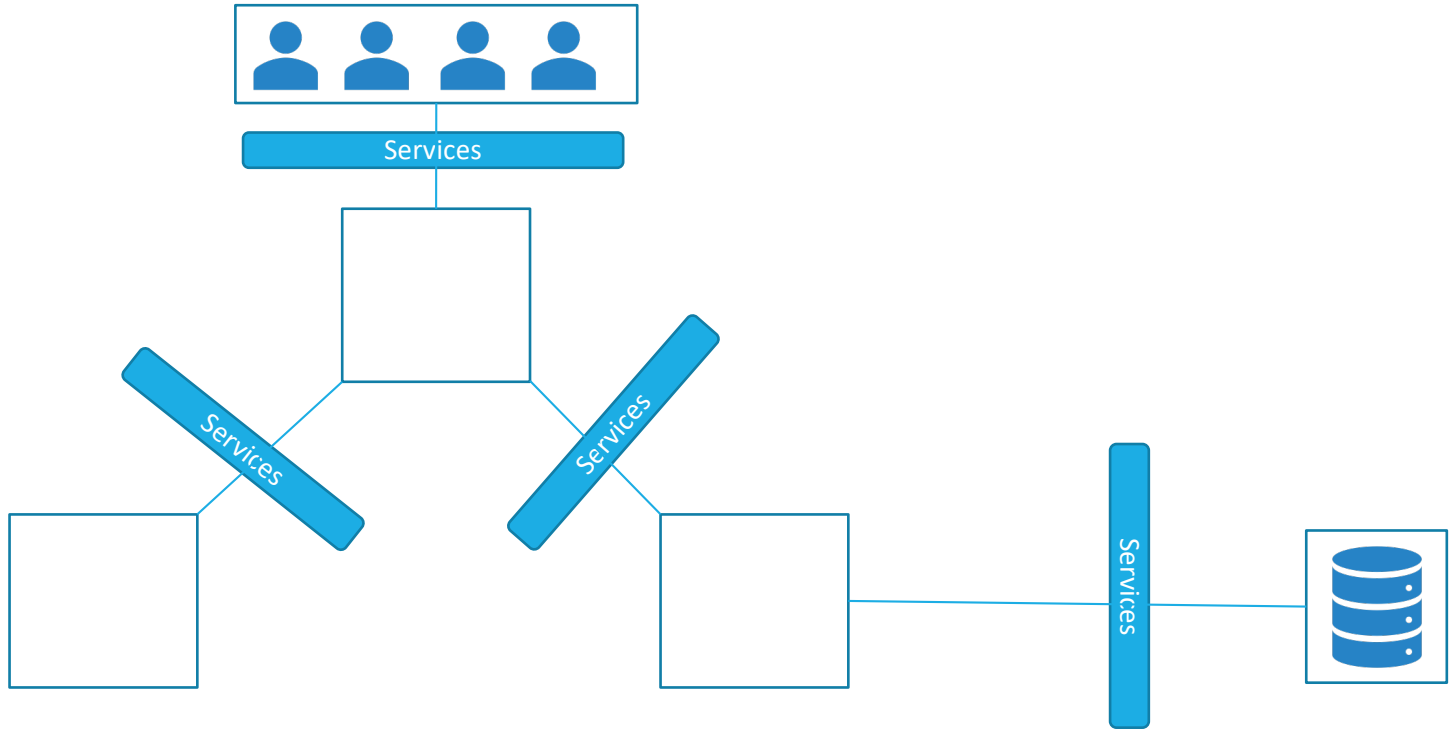
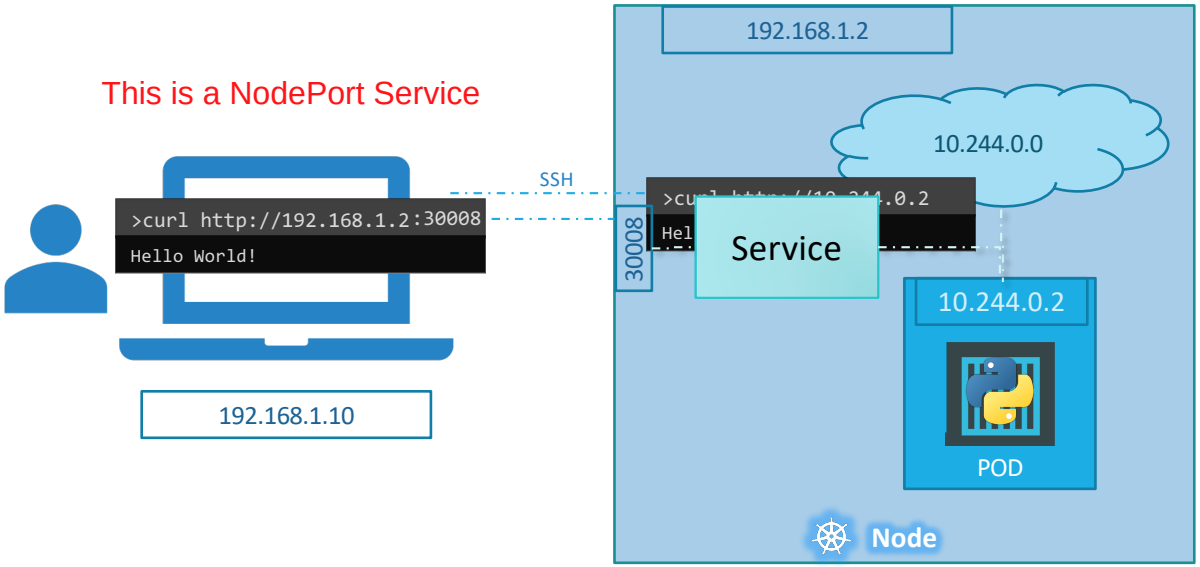
Services



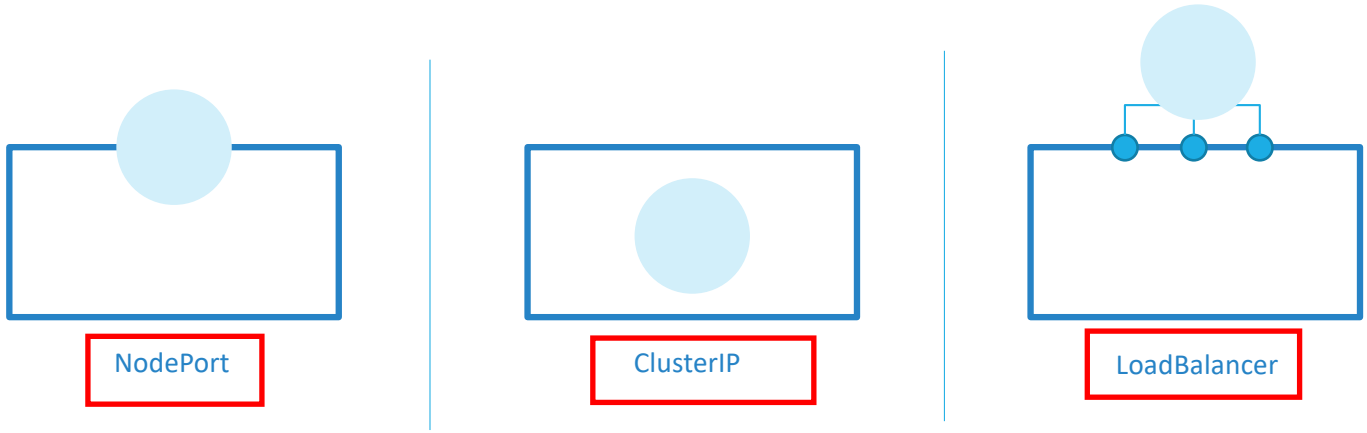
Services



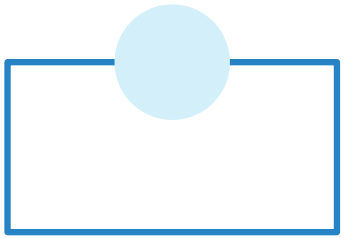
Service


This is a NodePort 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

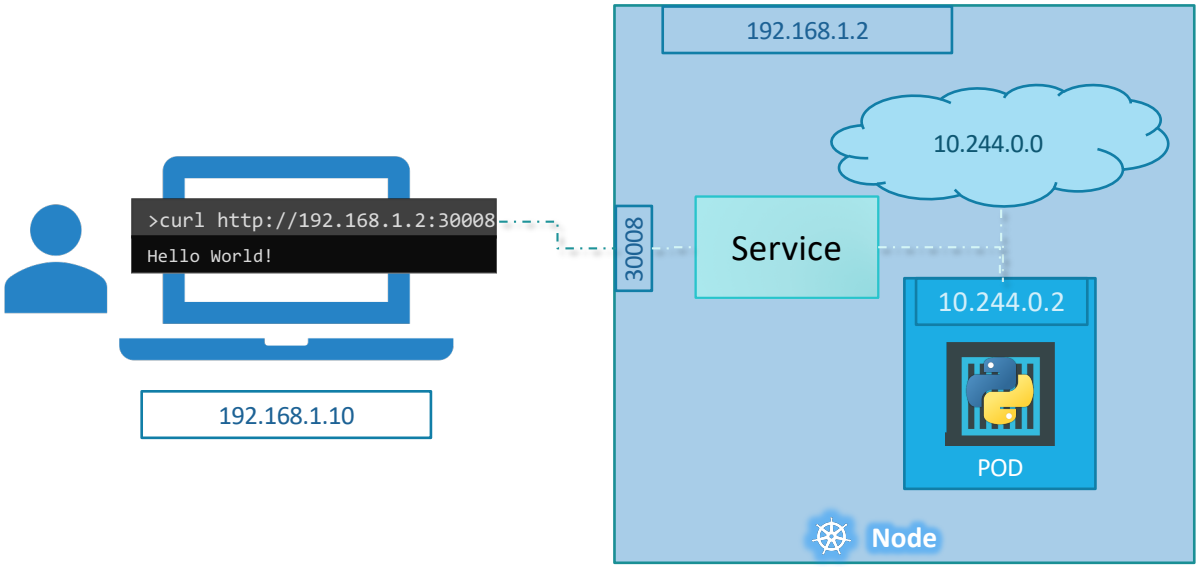
Services Types



NodePort

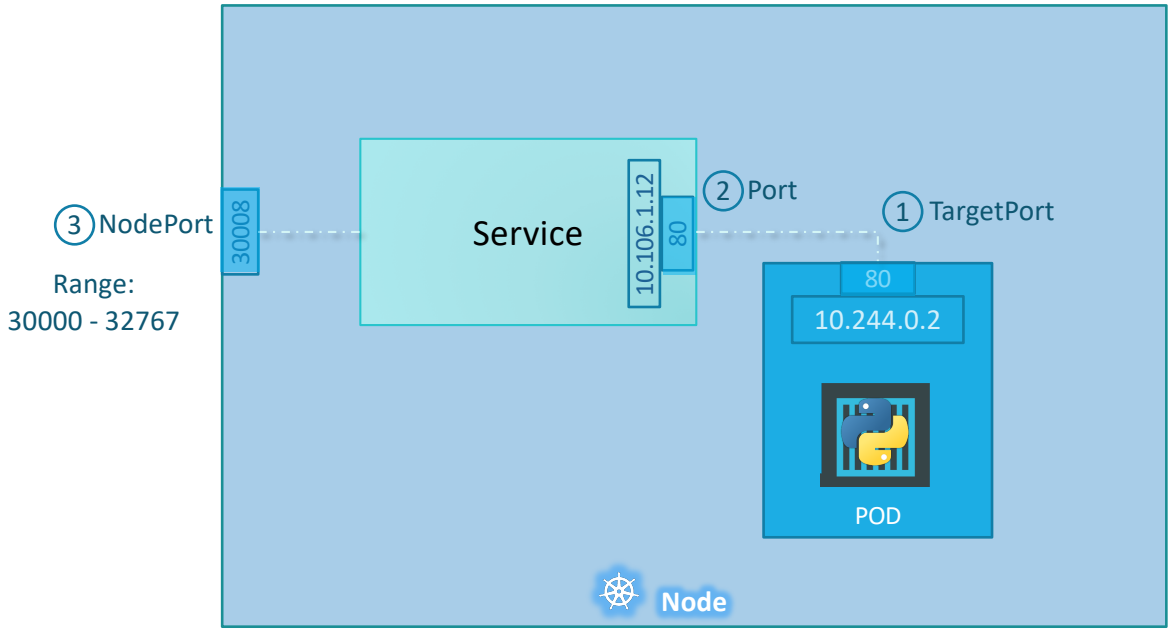


Service - NodePort



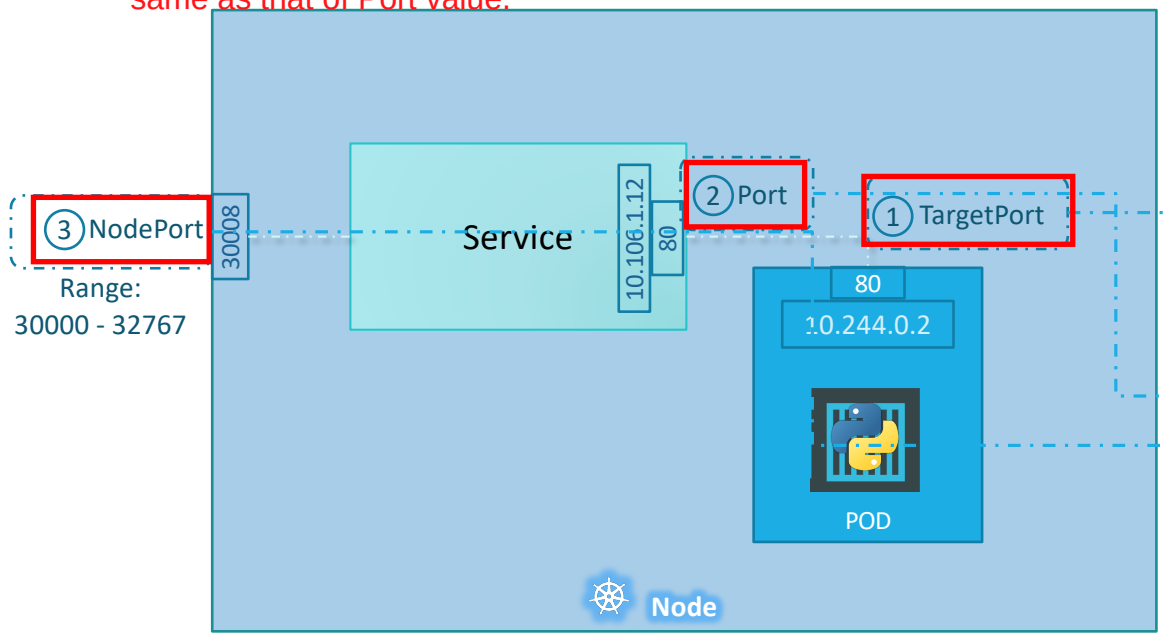
Service - NodePort

Service is like a virtual server inside the Node, which has its own unique cluster IP address.



Service - NodePort

TargetPort : The port of the container that is exposed. If not provided it is assumed to be same as that of Port value.



service-definition.yml

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

spec:
    type: NodePort
    ports:
      - targetPort: 80
        port: 80
        nodePort: 30008
```

Port : This is a mandatory , this is a port on the service object.

Nodeport : If not provided then it is automatically allocated.

Service - NodePort

service-definition.yml

```
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

```
app: myapp
```

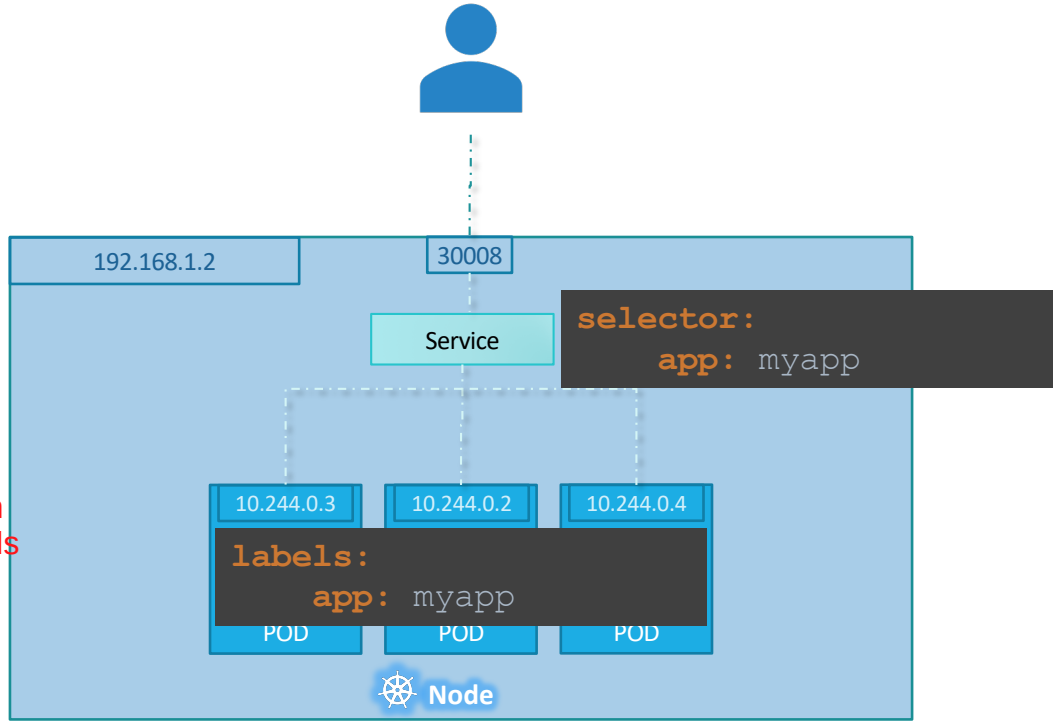
```
> kubectl create -f service-definition.yml
service "myapp-service" created
```

```
> 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>
```

The selector section links the Service to the Pod.

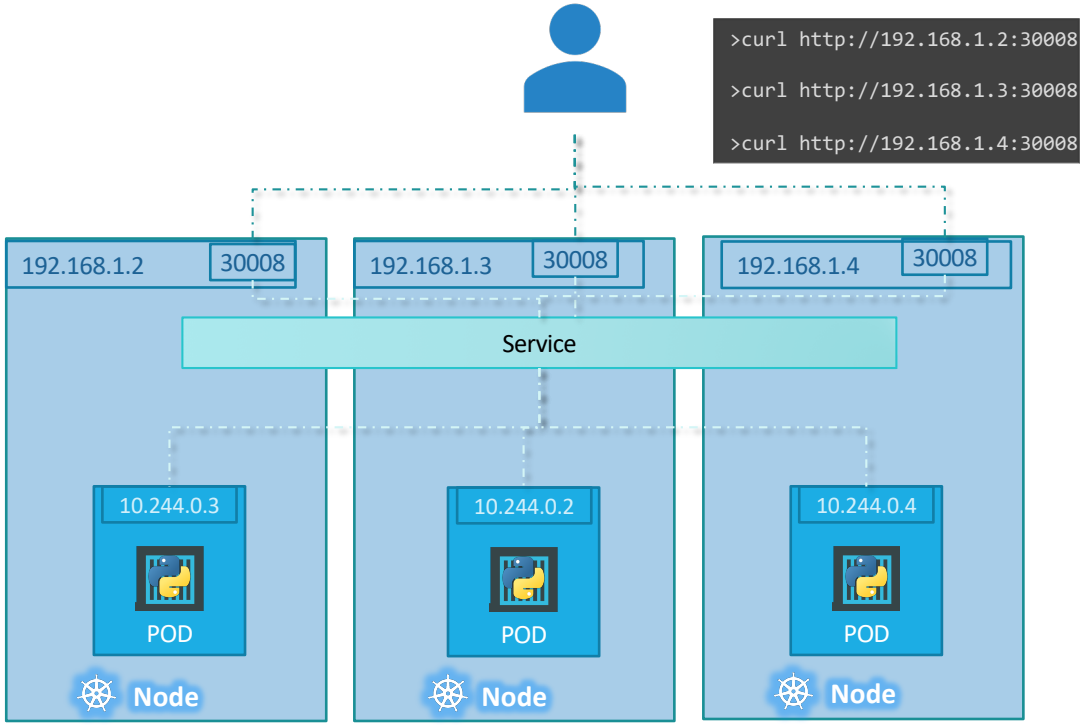
Service - NodePort



Algorithm: Random
SessionAffinity: Yes

Multiple Pods on a Single Node, then traffic is distributed across all the pods using Random selection with session Affinity.

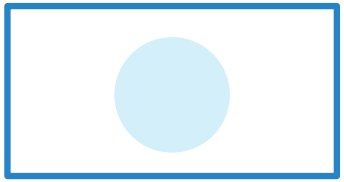
Service - NodePort



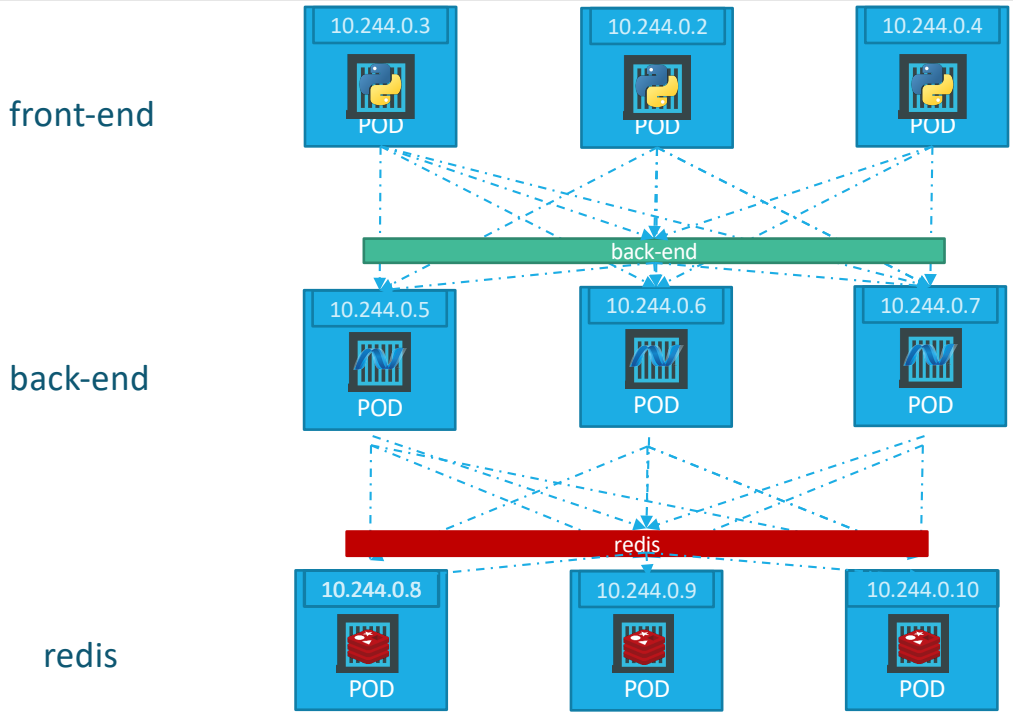
Demo

Service - NodePort

ClusterIP



ClusterIP


front-end
back-end
redis
10.244.0.3
POD
10.244.0.2
POD
10.244.0.4
POD
back-end
10.244.0.5
POD
10.244.0.6
POD
10.244.0.7
POD
redis
10.244.0.8
POD
10.244.0.9
POD
10.244.0.10
POD

ClusterIP is the default type for Service.

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
```

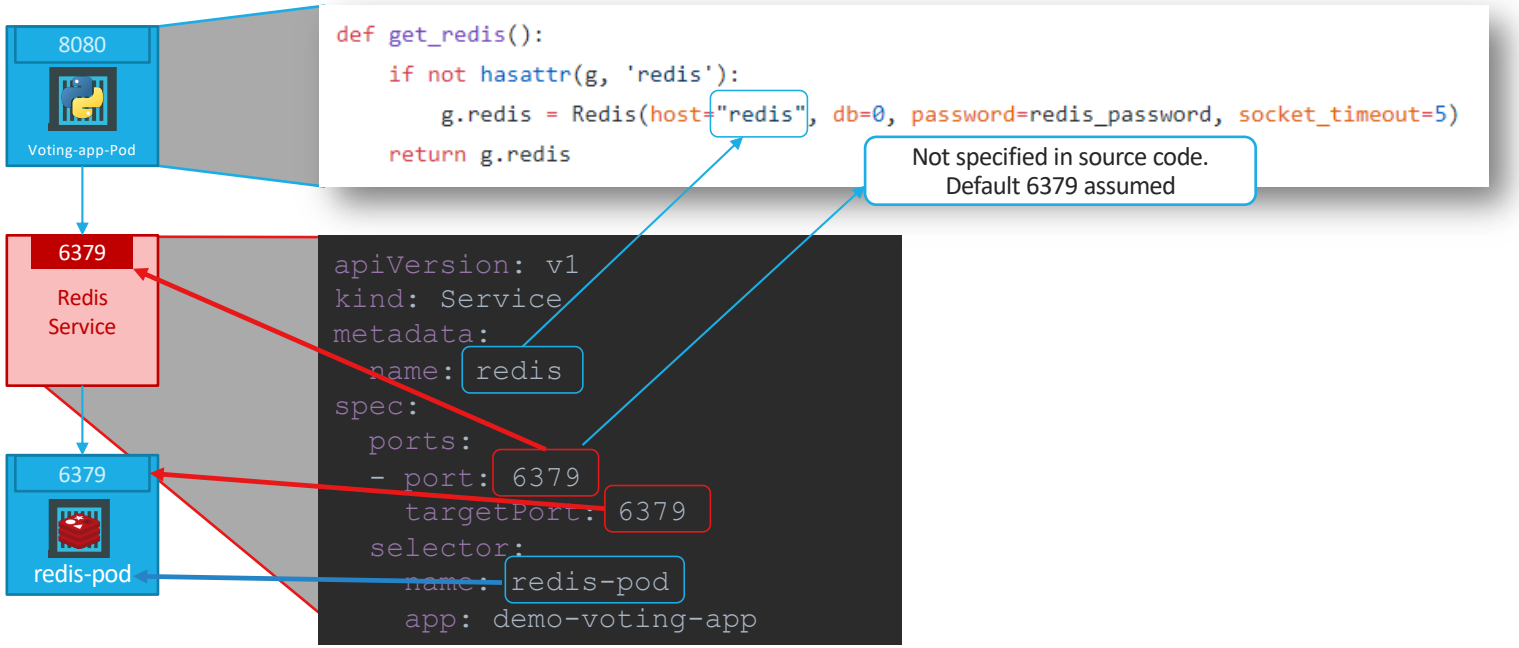
pod-definition.yml

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

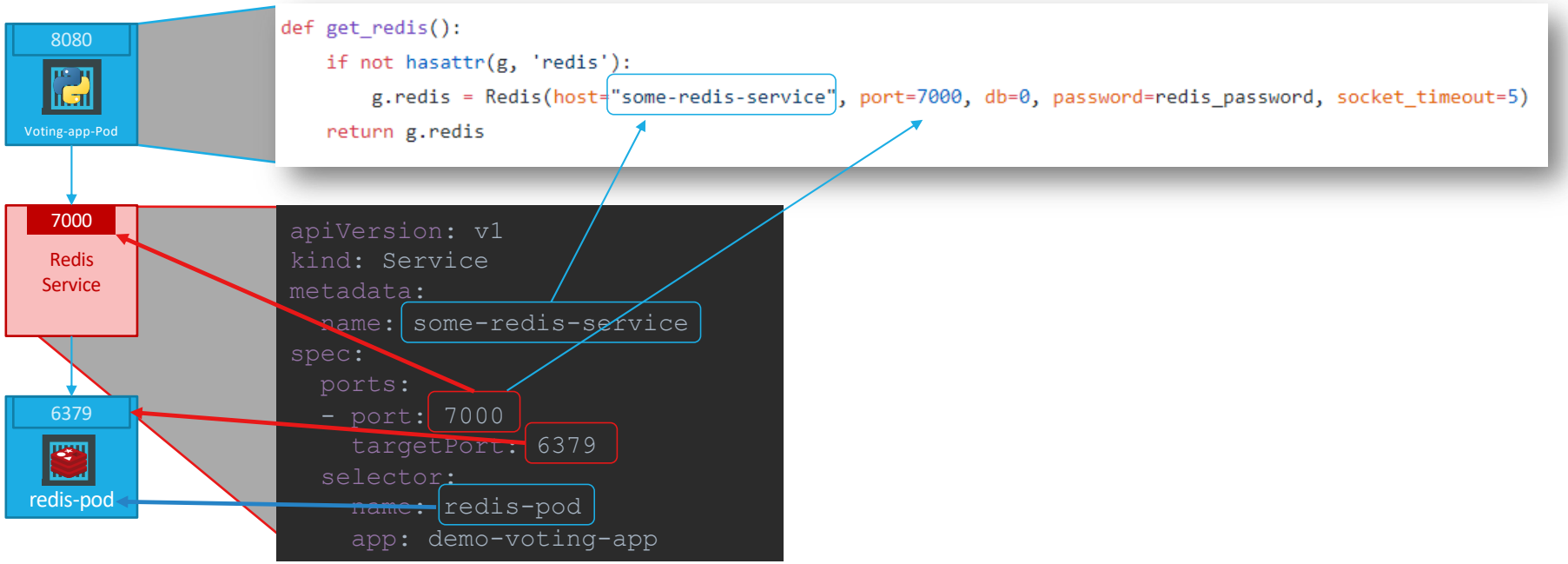
```
> kubectl create -f service-definition.yml
service "back-end" created
```

```
> 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


8080
Voting-app-Pod
6379
Redis
Service
6379
redis-pod
def get_redis():
if not hasattr(g, 'redis'):
g.redis = Redis(host="redis", db=0, password=redis_password, socket_timeout=5)
return g.redis
Not specified in source code.
Default 6379 assumed
apiVersion: v1
kind: Service
metadata:
name: redis
spec:
ports:
- port: 6379
targetPort: 6379
selector:
name: redis-pod
app: demo-voting-app

Service


8080
Voting-app-Pod
7000
Redis
Service
6379
redis-pod
def get_redis():
if not hasattr(g, 'redis'):
g.redis = Redis(host="some-redis-service", port=7000, db=0, password=redis_password, socket_timeout=5)
return g.redis
apiVersion: v1
kind: Service
metadata:
name: some-redis-service
spec:
ports:
- port: 7000
targetPort: 6379
selector:
name: redis-pod
app: demo-voting-app

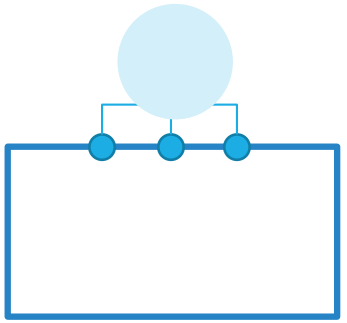
Demo

Service - NodePort

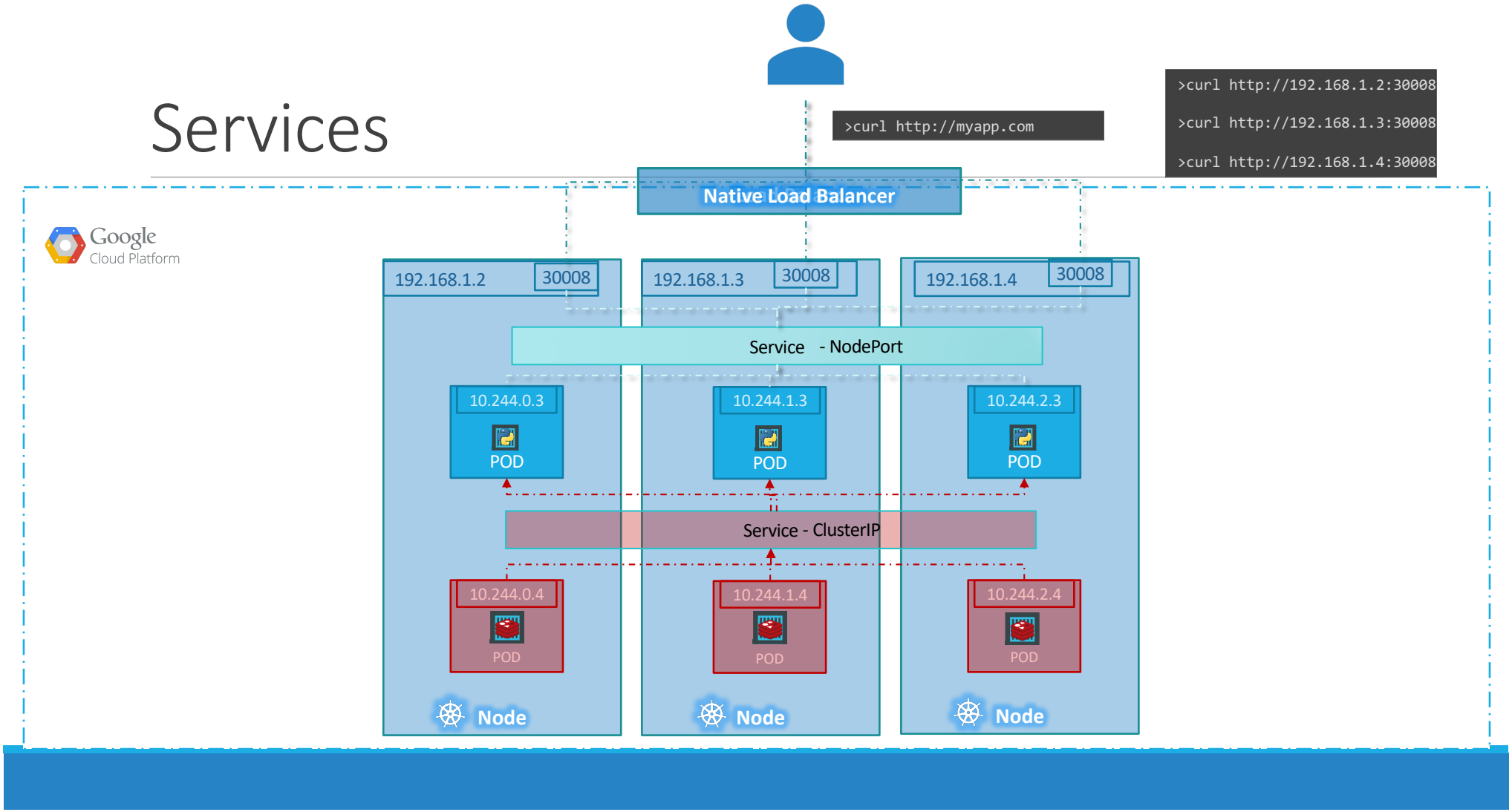
References

https://kubernetes.io/docs/concepts/services-networking/dns-pod-service/

Service - LoadBalancer



Services


>curl http://myapp.com
>curl http://192.168.1.2:30008
>curl http://192.168.1.3:30008
>curl http://192.168.1.4:30008
Native Load Balancer
Google Cloud Platform
192.168.1.2
30008
192.168.1.3
30008
192.168.1.4
30008
Service - NodePort
10.244.0.3
POD
10.244.1.3
POD
10.244.2.3
POD
Service - ClusterIP
10.244.0.4
POD
10.244.1.4
POD
10.244.2.4
POD
Node
Node
Node

service-definition.yml

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

spec:
    type: NodeBalancer
    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
```