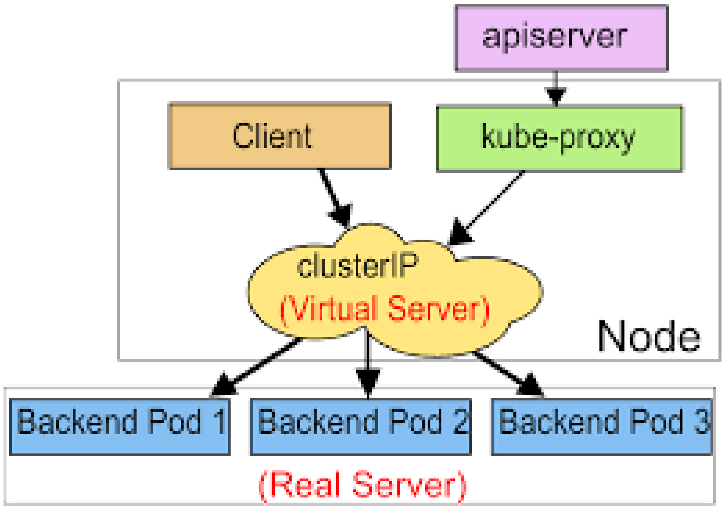
Day 5

Services



Problem Statement:

- *If load increases, we can create one more Pod? But who will create it?*
- *There are two ways: Manual or Replication Controller (it will set the desired state for you automatically).*
- *So, we cannot give all IP address of the servers to the client, => not user friendly.*
- *We are going to create an intermediate program between clients and pods. Say it has the* IP*(100) and the request is recreated and connect to respective port of backend servers.*
- *This intermediate program is frontend of the backend servers and is also known as LOAD BALANCER.*

Now challenge is, how LB will register as the new pod launches.

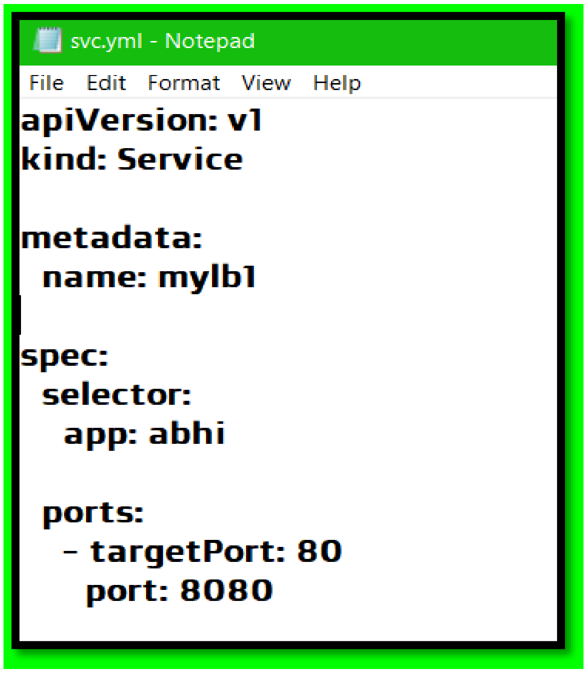
Since ip changes of system on each restart, so we have to tag the OS of label it.

LB will look for this Label and as soon as it finds it, that particular OS will get register under LB.

For this Practical,

minikube start

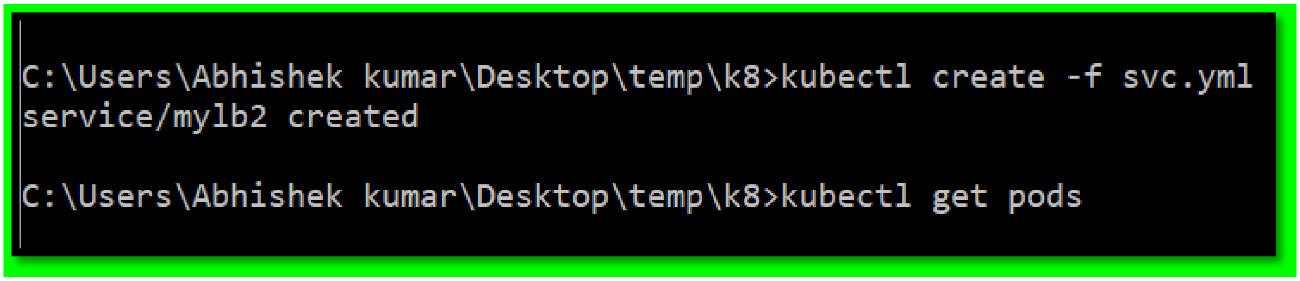
go to the directory cd /Desktop/temp/k8/notepad svc.yml



```
apiVersion: v1
kind: Service

metadata:
 name: mylb1

spec:
 selector:
   app: abhi

 ports:
  - targetPort: 80
    port: 8080
```

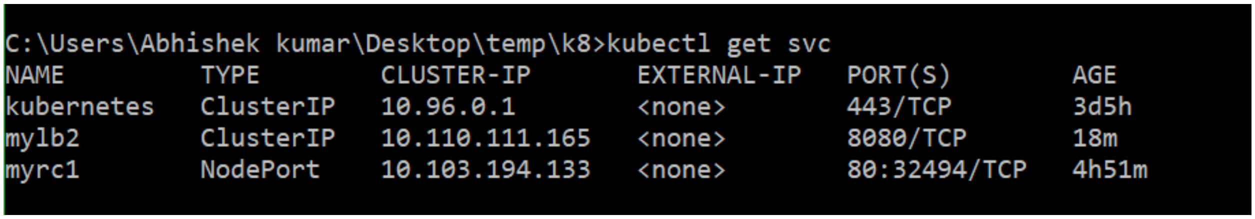
Now we have to launch this Service⇔LB bt create option.

Kubectl create –f svc.yml



```
C:\Users\Abhishek kumar\Desktop\temp\k8>kubectl create -f svc.yml
service/mylb2 created

C:\Users\Abhishek kumar\Desktop\temp\k8>kubectl get pods
```

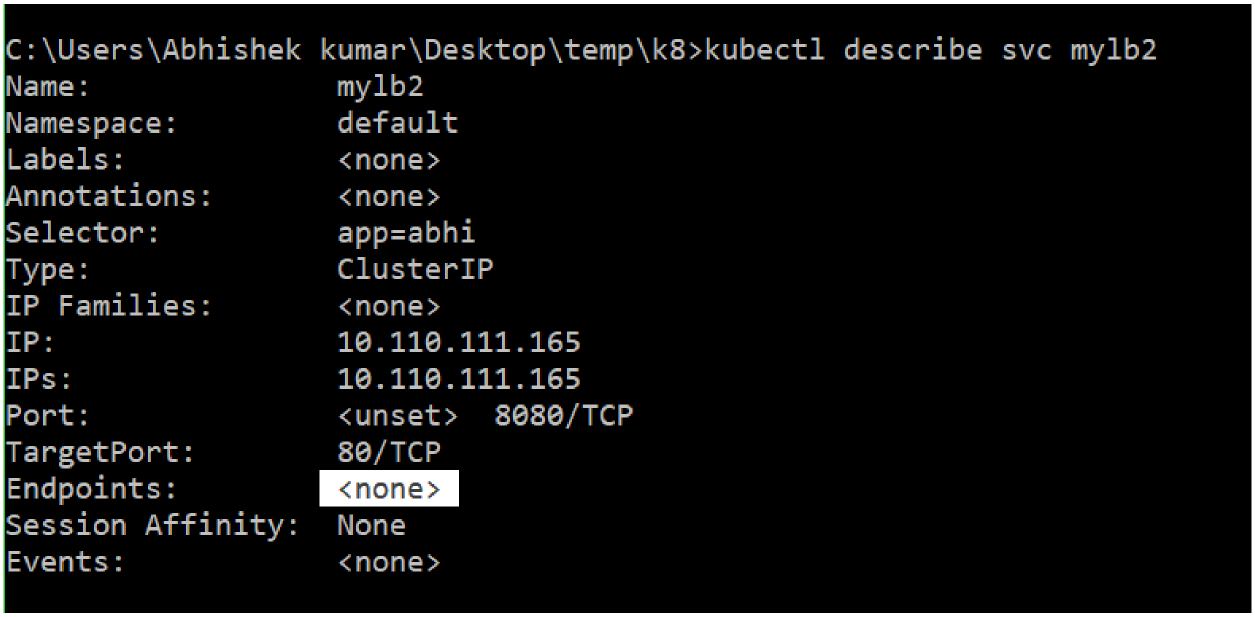
Kubectl get svc



```
C:\Users\Abhishek kumar\Desktop\temp\k8>kubectl get svc
NAME         TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
kubernetes   ClusterIP   10.96.0.1        <none>        443/TCP          3d5h
mylb2        ClusterIP   10.110.111.165   <none>        8080/TCP         18m
myrc1        NodePort    10.103.194.133   <none>        80:32494/TCP     4h51m
```

Now, in this mylb2, no webservers are attached, as only the lb is made(front end) only not (back end)

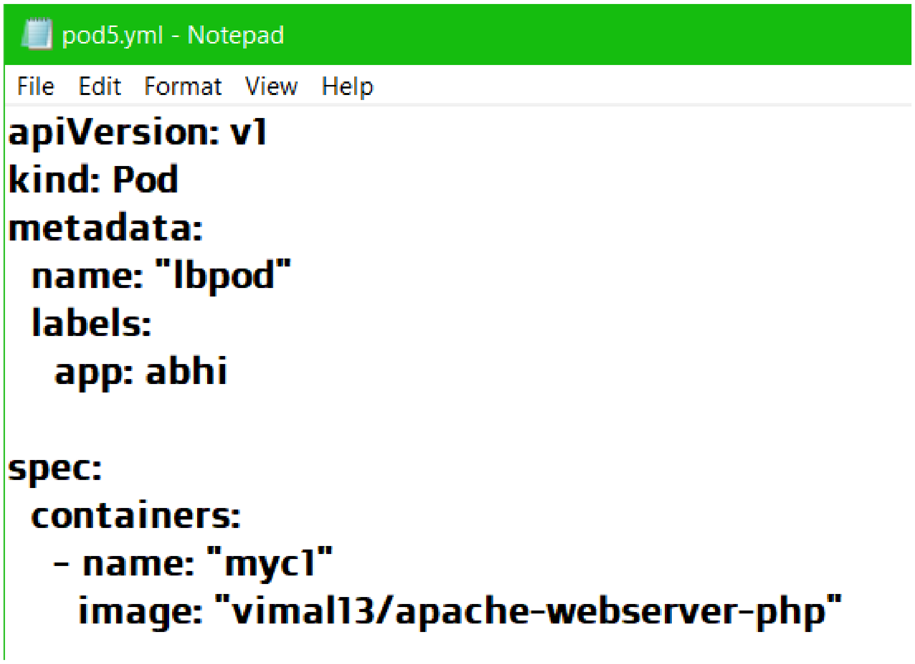
Kubectl describe svc mylb2



```
C:\Users\Abhishek kumar\Desktop\temp\k8>kubectl describe svc mylb2
Name:                mylb2
Namespace:           default
Labels:              <none>
Annotations:         <none>
Selector:            app=abhi
Type:                ClusterIP
IP Families:         <none>
IP:                  10.110.111.165
IPs:                 10.110.111.165
Port:                <unset>  8080/TCP
TargetPort:          80/TCP
Endpoints:           <none>
Session Affinity:    None
Events:              <none>
```

Now we will launch a pod with label **(app: abhi)** which is same defined as on selector of svc.yml.

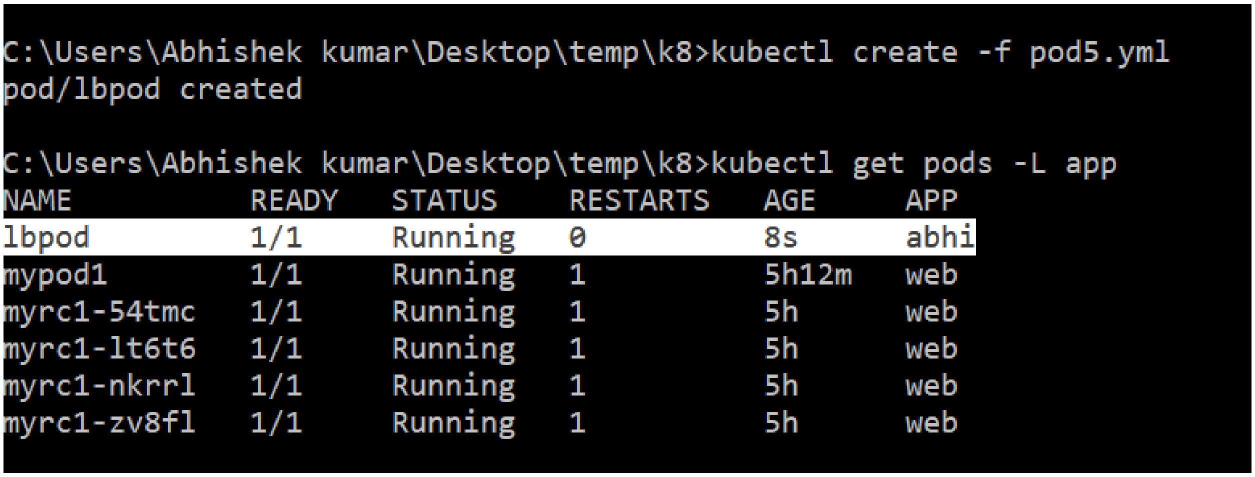
Notepad pod5.yml



```
apiVersion: v1
kind: Pod
metadata:
  name: "lbpod"
  labels:
    app: abhi

spec:
  containers:
   - name: "myc1"
     image: "vimal13/apache-webserver-php"
```

Now create this pod by using

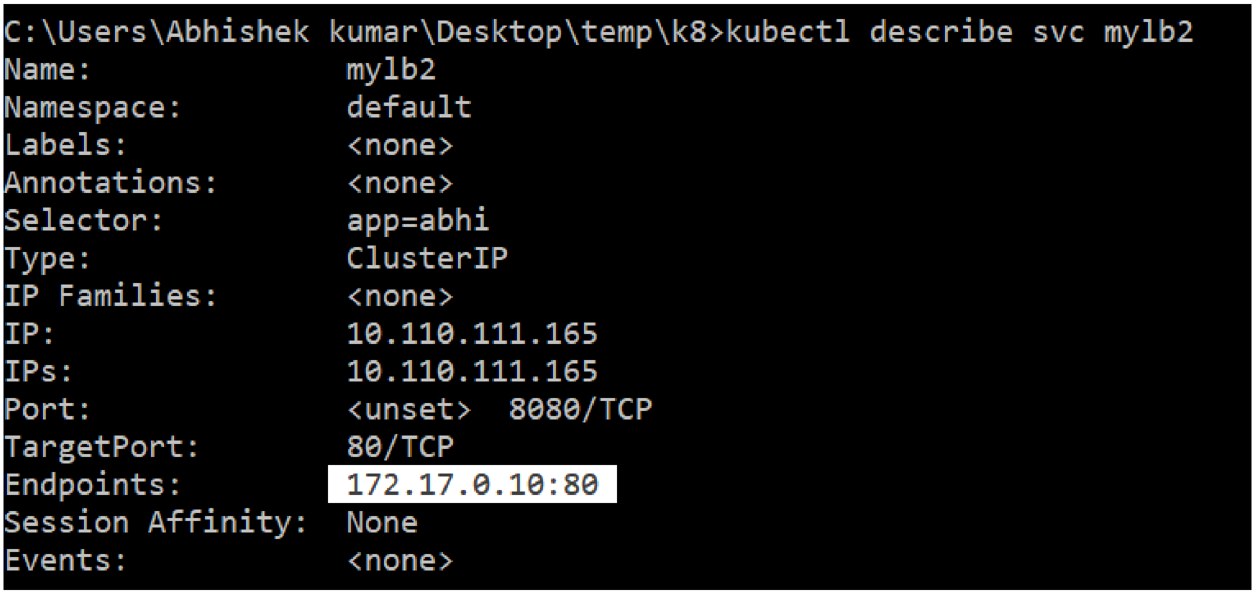
Kubectl create –f pod5.yml



```
C:\Users\Abhishek kumar\Desktop\temp\k8>kubectl create -f pod5.yml
pod/lbpod created

C:\Users\Abhishek kumar\Desktop\temp\k8>kubectl get pods -L app
NAME           READY   STATUS    RESTARTS   AGE     APP
lbpod          1/1     Running   0          8s      abhi
mypod1         1/1     Running   1          5h12m   web
myrc1-54tmc    1/1     Running   1          5h      web
myrc1-lt6t6    1/1     Running   1          5h      web
myrc1-nkrrl    1/1     Running   1          5h      web
myrc1-zv8fl    1/1     Running   1          5h      web
```

Now, in the same time let's see any endpoints are attached to mylb2 or not. It must be because we have added (app: abhi) which is defined in selectors to be used.

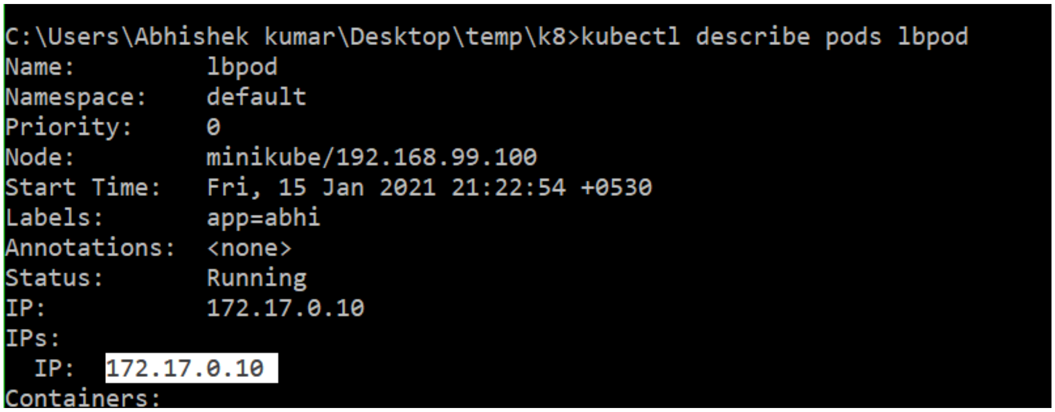
Kubectl describe svc mylb2



```
C:\Users\Abhishek kumar\Desktop\temp\k8>kubectl describe svc mylb2
Name:              mylb2
Namespace:         default
Labels:            <none>
Annotations:       <none>
Selector:          app=abhi
Type:              ClusterIP
IP Families:       <none>
IP:                10.110.111.165
IPs:               10.110.111.165
Port:              <unset>  8080/TCP
TargetPort:        80/TCP
Endpoints:         172.17.0.10:80
Session Affinity:  None
Events:            <none>
```

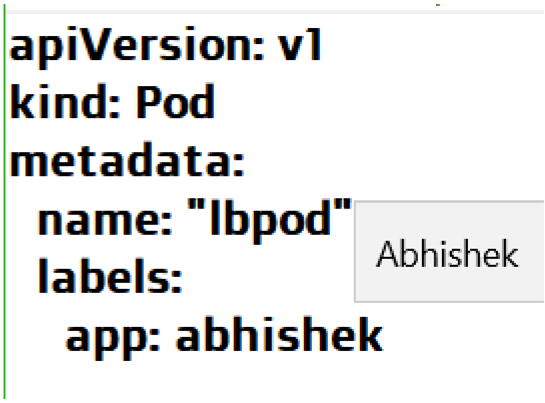
Here added one endpoints, now lets check with label not equals to abhi. Let's confirm that with checking the IP of pod.

Kubectl describe pods lbpod



```
C:\Users\Abhishek kumar\Desktop\temp\k8>kubectl describe pods lbpod
Name:         lbpod
Namespace:    default
Priority:     0
Node:         minikube/192.168.99.100
Start Time:   Fri, 15 Jan 2021 21:22:54 +0530
Labels:       app=abhi
Annotations:  <none>
Status:       Running
IP:           172.17.0.10
IPs:
  IP:  172.17.0.10
Containers:
```

Now, lets Update the pod file and check that weather it is also added to the laod balancer(mylb2). So let’s update the pod5.yml.

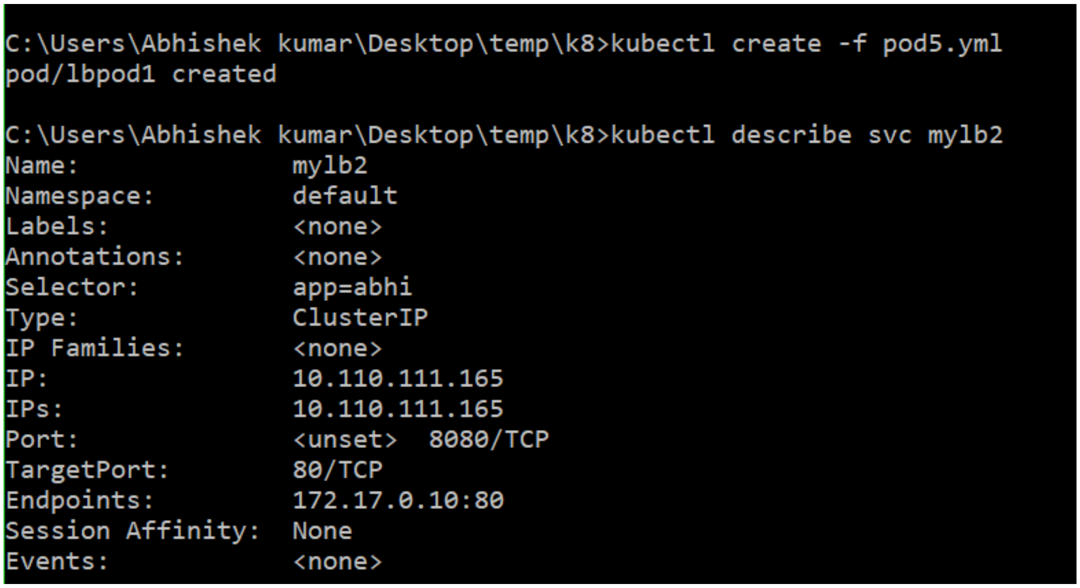


```
apiVersion: v1
kind: Pod
metadata:
  name: "lbpod"
  labels:
    app: abhishek
```

Here, pod should be created but it will not be added to the endpoints

Kubectl create –f pod5.yml

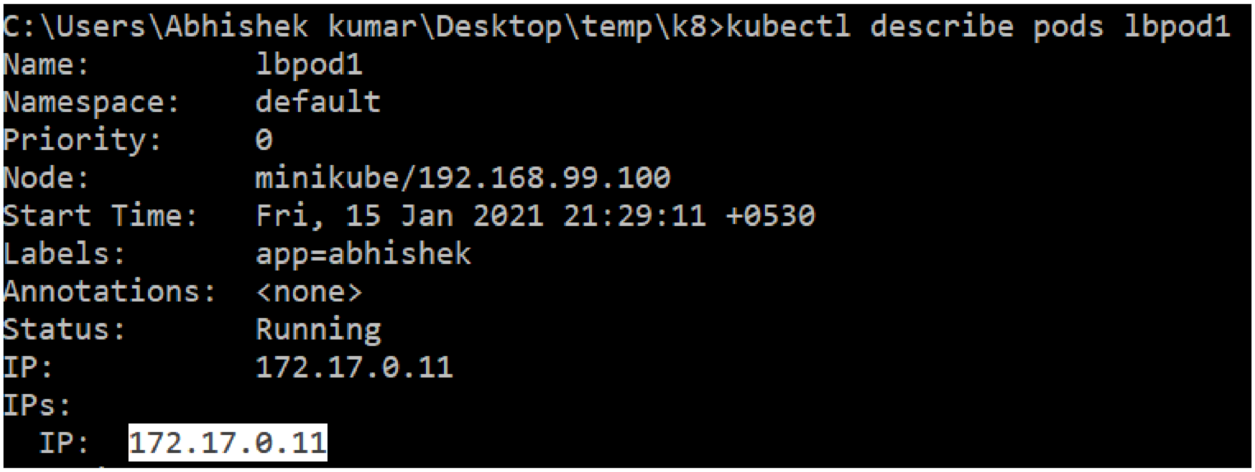
Kubectl describe svc mylb2



```
C:\Users\Abhishek kumar\Desktop\temp\k8>kubectl create -f pod5.yml
pod/lbpod1 created

C:\Users\Abhishek kumar\Desktop\temp\k8>kubectl describe svc mylb2
Name:              mylb2
Namespace:         default
Labels:            <none>
Annotations:       <none>
Selector:          app=abhi
Type:              ClusterIP
IP Families:       <none>
IP:                10.110.111.165
IPs:               10.110.111.165
Port:              <unset>  8080/TCP
TargetPort:        80/TCP
Endpoints:         172.17.0.10:80
Session Affinity:  None
Events:            <none>
```

Now lets check ip of the lbpod1

Kubectl describe pods lbpod1

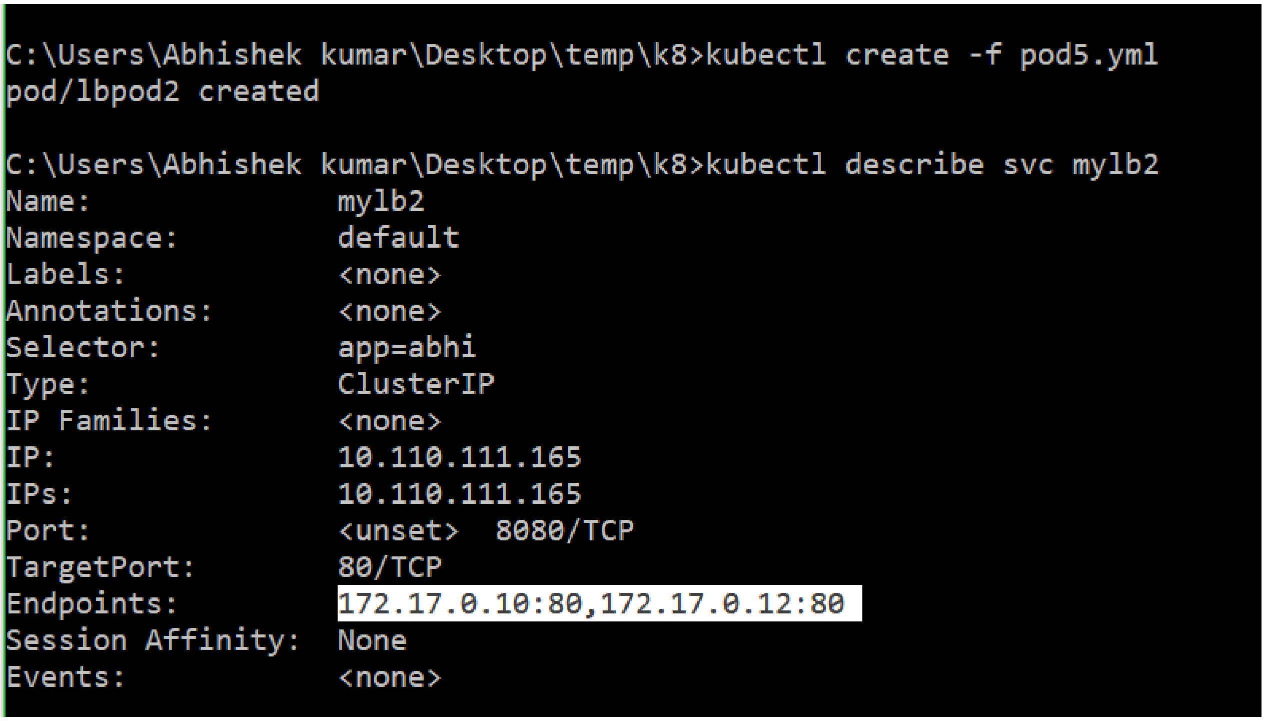


```
C:\Users\Abhishek kumar\Desktop\temp\k8>kubectl describe pods lbpod1
Name:         lbpod1
Namespace:    default
Priority:     0
Node:         minikube/192.168.99.100
Start Time:   Fri, 15 Jan 2021 21:29:11 +0530
Labels:       app=abhishek
Annotations:  <none>
Status:       Running
IP:           172.17.0.11
IPs:
  IP:  172.17.0.11
```

Again launch a New pod with (web: abhi) with same commands:

Kubectl create –f pod5.yml

Kubectl describe svc mylb2

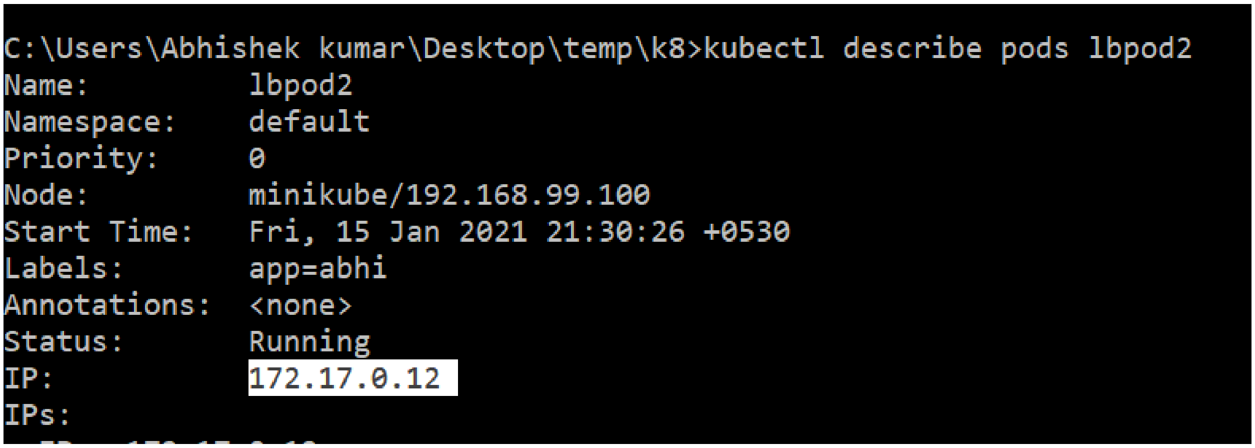


```
C:\Users\Abhishek kumar\Desktop\temp\k8>kubectl create -f pod5.yml
pod/lbpod2 created

C:\Users\Abhishek kumar\Desktop\temp\k8>kubectl describe svc mylb2
Name:              mylb2
Namespace:         default
Labels:            <none>
Annotations:       <none>
Selector:          app=abhi
Type:              ClusterIP
IP Families:       <none>
IP:                10.110.111.165
IPs:               10.110.111.165
Port:              <unset>  8080/TCP
TargetPort:        80/TCP
Endpoints:         172.17.0.10:80,172.17.0.12:80
Session Affinity:  None
Events:            <none>
```

Now, we can see 2 Ip's are added at the endpoints of the LB.

Let's check the Ip of the lbpod2.

Kubectl describe pods lbpod2

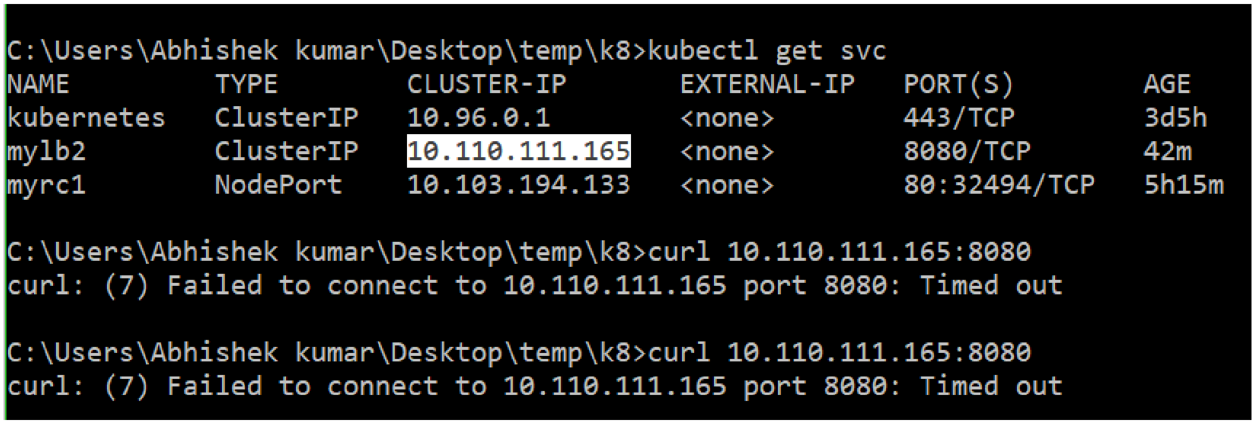


```
C:\Users\Abhishek kumar\Desktop\temp\k8>kubectl describe pods lbpod2
Name:         lbpod2
Namespace:    default
Priority:     0
Node:         minikube/192.168.99.100
Start Time:   Fri, 15 Jan 2021 21:30:26 +0530
Labels:       app=abhi
Annotations:  <none>
Status:       Running
IP:           172.17.0.12
IPs:
```

This selection of IP/POD to set as Endpoints in LB is called Auto Discovering.

Now, if we want to check our service status by curl to the cluster IP with port given.

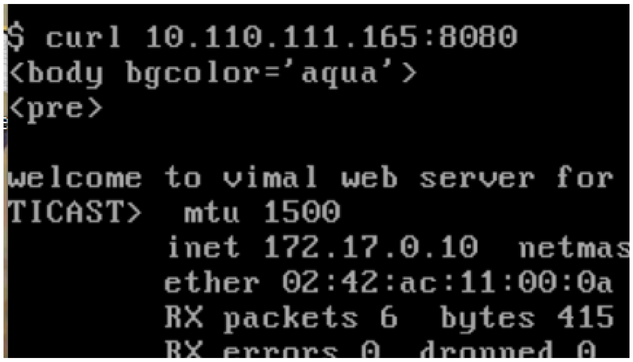
Here, **curl 10.110.111.165:8080**



```
C:\Users\Abhishek kumar\Desktop\temp\k8>kubectl get svc
NAME         TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)        AGE
kubernetes   ClusterIP   10.96.0.1        <none>        443/TCP        3d5h
mylb2        ClusterIP   10.110.111.165   <none>        8080/TCP       42m
myrc1        NodePort    10.103.194.133   <none>        80:32494/TCP   5h15m

C:\Users\Abhishek kumar\Desktop\temp\k8>curl 10.110.111.165:8080
curl: (7) Failed to connect to 10.110.111.165 port 8080: Timed out

C:\Users\Abhishek kumar\Desktop\temp\k8>curl 10.110.111.165:8080
curl: (7) Failed to connect to 10.110.111.165 port 8080: Timed out
```

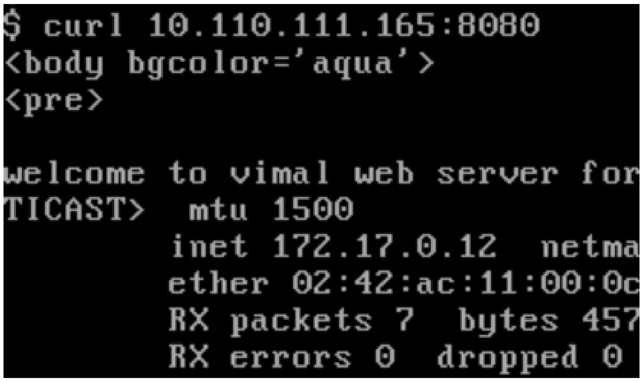
Can’t connect as this windows cli, is outside world for this Node. But, if we go to the **VM of minikube login with docker and password :tcuser** and curl here

curl 10.110.111.165:8080



```
$ curl 10.110.111.165:8080
<body bgcolor='aqua'>
<pre>

welcome to vimal web server for
TICAST>  mtu 1500
        inet 172.17.0.10  netmas
        ether 02:42:ac:11:00:0a
        RX packets 6  bytes 415
        RX errors 0  dropped 0
```

After Refresh

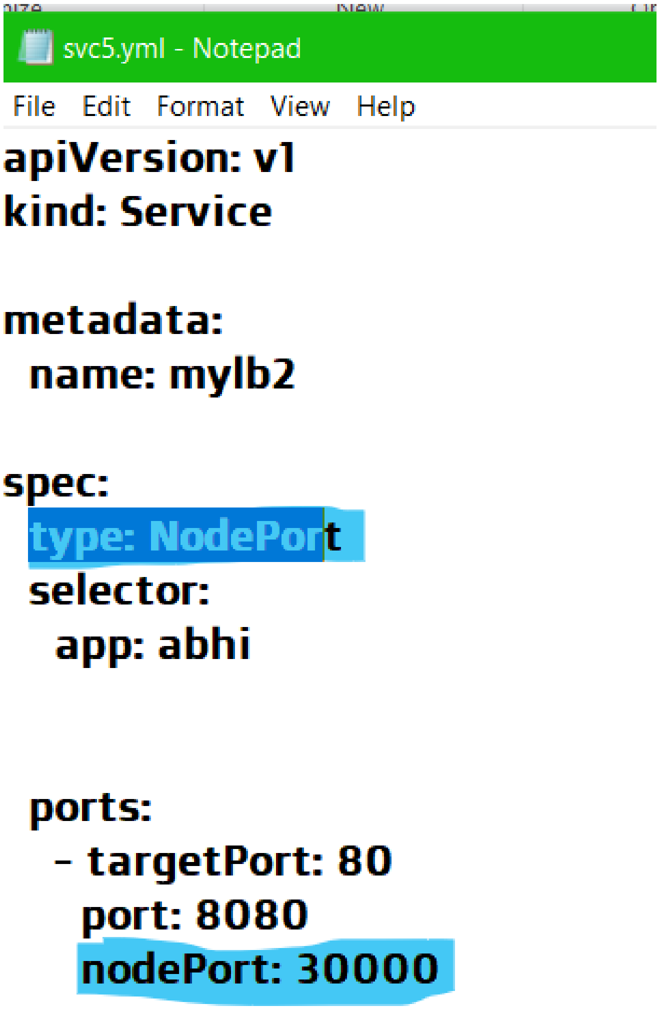


```
$ curl 10.110.111.165:8080
<body bgcolor='aqua'>
<pre>

welcome to vimal web server for
TICAST>  mtu 1500
        inet 172.17.0.12  netma
        ether 02:42:ac:11:00:0c
        RX packets 7  bytes 457
        RX errors 0  dropped 0
```

But, we need to expose that

So we need NodePort which act as Second reverse Proxy and will expose to the world. So that any one can connect with this ip.

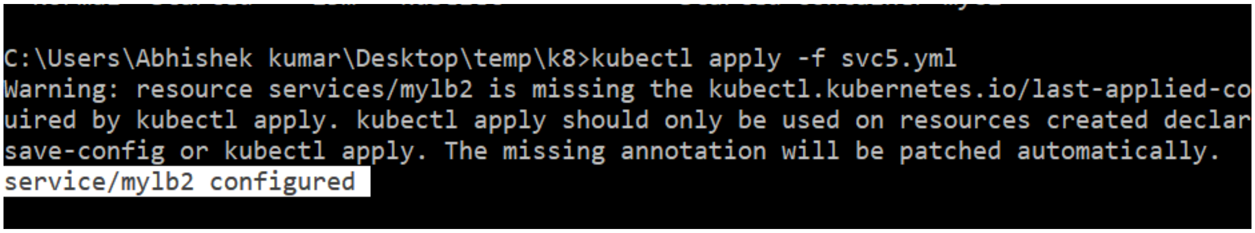
We need to change in service file (svc5.yml)



```
apiVersion: v1
kind: Service

metadata:
  name: mylb2

spec:
  type: NodePort
  selector:
    app: abhi


  ports:
    - targetPort: 80
      port: 8080
      nodePort: 30000
```

Now, after that run that file with Apply here as we already runned this file.

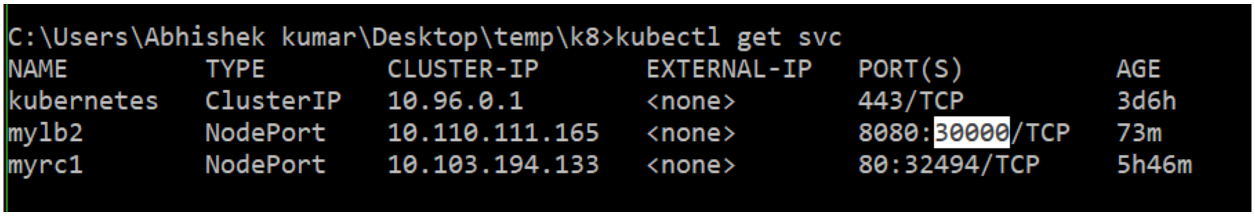
kubectl apply –f svc5.yml



```
C:\Users\Abhishek kumar\Desktop\temp\k8>kubectl apply -f svc5.yml
Warning: resource services/mylb2 is missing the kubectl.kubernetes.io/last-applied-co
uired by kubectl apply. kubectl apply should only be used on resources created declar
save-config or kubectl apply. The missing annotation will be patched automatically.
service/mylb2 configured
```

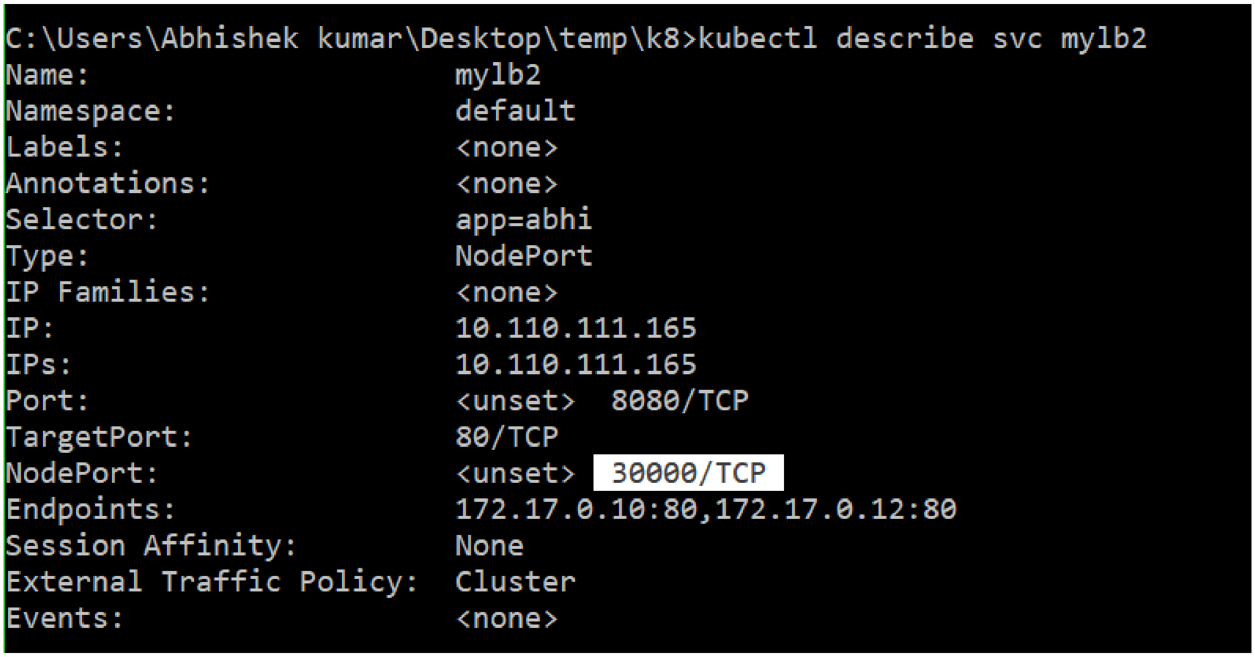
Now, confirm for the port and it is nodePort

Kubectl get svc



```
C:\Users\Abhishek kumar\Desktop\temp\k8>kubectl get svc
NAME         TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
kubernetes   ClusterIP   10.96.0.1        <none>        443/TCP          3d6h
mylb2        NodePort    10.110.111.165   <none>        8080:30000/TCP   73m
myrc1        NodePort    10.103.194.133   <none>        80:32494/TCP     5h46m
```

Kubectl describe svc mylb2



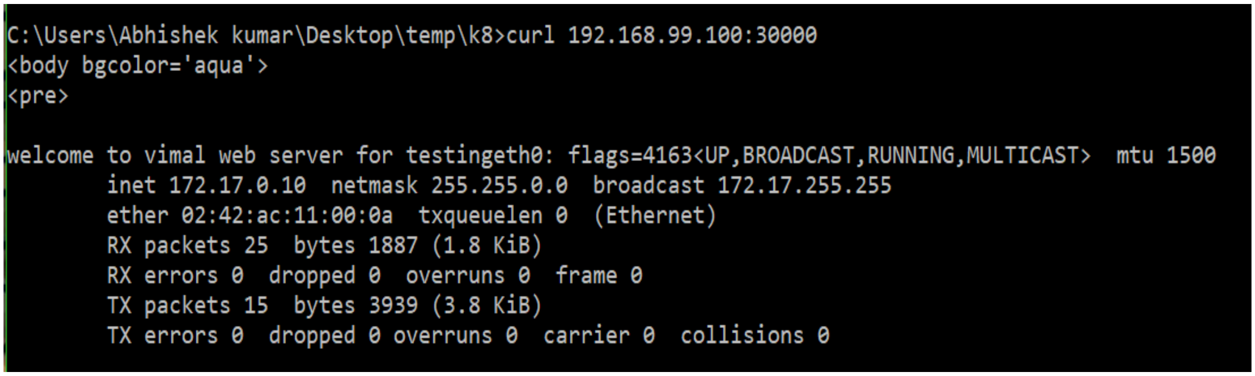
```
C:\Users\Abhishek kumar\Desktop\temp\k8>kubectl describe svc mylb2
Name:                     mylb2
Namespace:                default
Labels:                   <none>
Annotations:              <none>
Selector:                 app=abhi
Type:                     NodePort
IP Families:              <none>
IP:                       10.110.111.165
IPs:                      10.110.111.165
Port:                     <unset>  8080/TCP
TargetPort:               80/TCP
NodePort:                 <unset>  30000/TCP
Endpoints:                172.17.0.10:80,172.17.0.12:80
Session Affinity:         None
External Traffic Policy:  Cluster
Events:                   <none>
```

Now, if we use curl command then we can see it’s running from windows command prompt.

With IP of the minikube (**ifconfig | less**) and then use

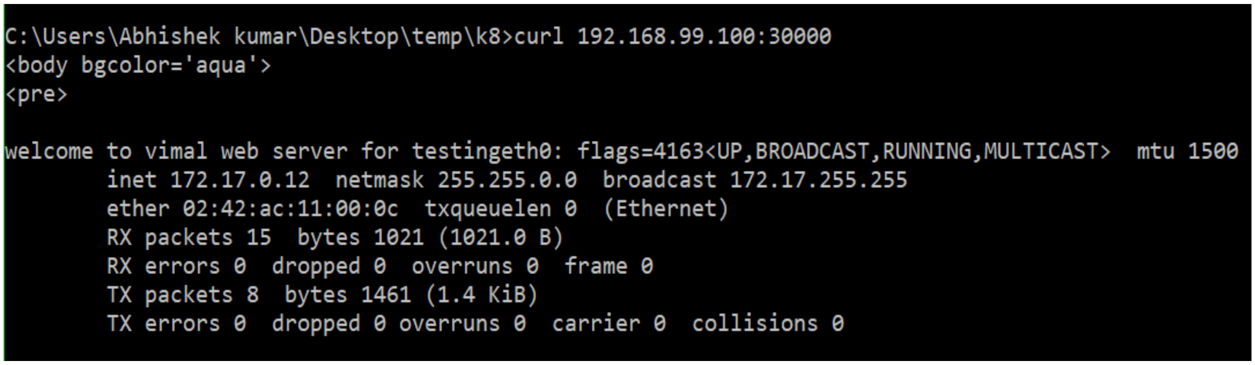
Curl minikubeIP:port

Curl 192.168.99.100:30000



```
C:\Users\Abhishek kumar\Desktop\temp\k8>curl 192.168.99.100:30000
<body bgcolor='aqua'>
<pre>

welcome to vimal web server for testingeth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.17.0.10  netmask 255.255.0.0  broadcast 172.17.255.255
        ether 02:42:ac:11:00:0a  txqueuelen 0  (Ethernet)
        RX packets 25  bytes 1887 (1.8 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 15  bytes 3939 (3.8 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Again Refresh-



```
C:\Users\Abhishek kumar\Desktop\temp\k8>curl 192.168.99.100:30000
<body bgcolor='aqua'>
<pre>

welcome to vimal web server for testingeth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.17.0.12  netmask 255.255.0.0  broadcast 172.17.255.255
        ether 02:42:ac:11:00:0c  txqueuelen 0  (Ethernet)
        RX packets 15  bytes 1021 (1021.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 8  bytes 1461 (1.4 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Done!!!