# Service ClusterIP Headless

1. Create a headless service services/service-clusterip-headless-selectors.yaml.

Check that it doesn’t have a cluster IP:

$ kubectl apply -f services/service-clusterip-headless-selectors.yaml  
service/myservice created  
  
$ kubectl get svc  
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE  
myservice ClusterIP None <none> 80/TCP 11s

Check Endpoint and DNS:

$ kubectl get ep  
NAME ENDPOINTS AGE  
myservice <none> 86s  
  
$ kubectl run -it --image busybox:1.28 --restart=Never --rm dns-test -- nslookup myservice  
Server: 10.96.0.10  
Address 1: 10.96.0.10 kube-dns.kube-system.svc.cluster.local  
nslookup: can't resolve 'myservice'

1. Create a deployment:

$ kubectl apply -f services/dep-mygo.yaml  
deployment.apps/mygo-deployment created  
  
$ kubectl get pods -o wide  
NAME READY STATUS RESTARTS AGE IP NODE  
mygo-deployment-695974c768-lcqhc 1/1 Running 0 11s 172.17.0.3 minikube

Check DNS and see the IP:

$ kubectl run -it --image busybox:1.28 --restart=Never --rm dns-test -- nslookup myservice  
Server: 10.96.0.10  
Address 1: 10.96.0.10 kube-dns.kube-system.svc.cluster.local  
Name: myservice  
Address 1: 172.17.0.3 172-17-0-3.myservice.my.svc.cluster.local

You should see the only pods.

1. Scale the deployment up to 3 replicas.

Check IPs:

$ kubectl get pods -o wide  
NAME READY STATUS RESTARTS AGE IP NODE  
mygo-deployment-695974c768-94j4g 1/1 Running 0 32s 172.17.0.5 minikube  
mygo-deployment-695974c768-lcqhc 1/1 Running 0 94s 172.17.0.3 minikube  
mygo-deployment-695974c768-s8vfz 1/1 Running 0 32s 172.17.0.4 minikube  
  
$ kubectl run -it --image busybox:1.28 --restart=Never --rm dns-test -- nslookup myservice  
Server: 10.96.0.10  
Address 1: 10.96.0.10 kube-dns.kube-system.svc.cluster.local  
  
Name: myservice  
Address 1: 172.17.0.5 172-17-0-5.myservice.my.svc.cluster.local  
Address 2: 172.17.0.4 172-17-0-4.myservice.my.svc.cluster.local  
Address 3: 172.17.0.3 172-17-0-3.myservice.my.svc.cluster.local  
pod "dns-test" deleted

You should see all three pods.

1. Is order the same for the next DNS requests? Why it is so?
2. Clean up:

* remove the service
* remove the deployments

## Solution

1. Create a headless service services/service-clusterip-headless-selectors.yaml.

$ kubectl create -f services/service-clusterip-headless-selectors.yaml  
service/myservice created

Check that it doesn’t have a cluster IP:

$ kubectl get svc  
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE  
myservice ClusterIP None <none> 80/TCP 11s

Check Endpoint and DNS:

$ kubectl get ep  
NAME ENDPOINTS AGE  
myservice <none> 86s

Check DNS (interactive and non-interactive):

$ kubectl run dns-test --image busybox:1.28 -it --rm -- nslookup myservice  
nslookup: can't resolve 'myservice'  
  
$ kubectl run dns-test --image busybox:1.28 -it --rm  
/ # nslookup myservice  
Server: 10.96.0.10  
Address 1: 10.96.0.10 kube-dns.kube-system.svc.cluster.local  
nslookup: can't resolve 'myservice'  
/ # exit

Because there aren’t pods.

1. Create a deployment:

$ kubectl apply -f services/dep-mygo.yaml  
deployment.apps/mygo-deployment created  
  
$ kubectl get pods -o wide  
NAME READY STATUS RESTARTS AGE IP NODE NOMINATED NODE READINESS GATES  
mygo-deployment-55894d8449-wjcll 1/1 Running 0 6s 172.17.0.3 minikube <none> <none>

Check DNS and see the IP:

$ kubectl run dns-test --image busybox:1.28 -it --rm -- nslookup myservice  
Server: 10.96.0.10  
Address 1: 10.96.0.10 kube-dns.kube-system.svc.cluster.local  
  
Name: myservice  
Address 1: 172.17.0.3 172-17-0-3.myservice.my.svc.cluster.local

You should see the only pods.

1. Scale the deployment up to 3 replicas.

$ kubectl scale deployment mygo-deployment --replicas=3  
deployment.apps/mygo-deployment scaled

Check IPs:

NAME READY STATUS RESTARTS AGE IP NODE NOMINATED NODE READINESS GATES  
mygo-deployment-55894d8449-lnq4s 1/1 Running 0 14s 172.17.0.6 minikube <none> <none>  
mygo-deployment-55894d8449-nbxqd 1/1 Running 0 14s 172.17.0.7 minikube <none> <none>  
mygo-deployment-55894d8449-wjcll 1/1 Running 0 4m8s 172.17.0.3 minikube <none> <none>  
  
$ kubectl run -it --image busybox:1.28 --restart=Never --rm dns-test -- nslookup myservice  
Server: 10.96.0.10  
Address 1: 10.96.0.10 kube-dns.kube-system.svc.cluster.local  
  
Name: myservice  
Address 1: 172.17.0.7 172-17-0-7.myservice.default.svc.cluster.local  
Address 2: 172.17.0.3 172-17-0-3.myservice.default.svc.cluster.local  
Address 3: 172.17.0.6 172-17-0-6.myservice.default.svc.cluster.local  
pod "dns-test" deleted

All three pods are here. There is no single IP for service due to headless specific version.

1. Is order the same for the next DNS requests? Why it is so?

Order is different:

$ kubectl run dns-test --image busybox:1.28 -it --rm  
If you don't see a command prompt, try pressing enter.  
  
/ # nslookup myservice  
Server: 10.96.0.10  
Address 1: 10.96.0.10 kube-dns.kube-system.svc.cluster.local  
Name: myservice  
Address 1: 172.17.0.7 172-17-0-7.myservice.default.svc.cluster.local  
Address 2: 172.17.0.6 172-17-0-6.myservice.default.svc.cluster.local  
Address 3: 172.17.0.3 172-17-0-3.myservice.default.svc.cluster.local  
  
/ # nslookup myservice  
Server: 10.96.0.10  
Address 1: 10.96.0.10 kube-dns.kube-system.svc.cluster.local  
Name: myservice  
Address 1: 172.17.0.6 172-17-0-6.myservice.default.svc.cluster.local  
Address 2: 172.17.0.3 172-17-0-3.myservice.default.svc.cluster.local  
Address 3: 172.17.0.7 172-17-0-7.myservice.default.svc.cluster.local  
  
/ # nslookup myservice  
Server: 10.96.0.10  
Address 1: 10.96.0.10 kube-dns.kube-system.svc.cluster.local  
Name: myservice  
Address 1: 172.17.0.7 172-17-0-7.myservice.default.svc.cluster.local  
Address 2: 172.17.0.6 172-17-0-6.myservice.default.svc.cluster.local  
Address 3: 172.17.0.3 172-17-0-3.myservice.default.svc.cluster.local

It is so to avoid hot spot issue with single pod.

1. Clean up:

* remove the service
* remove the deployments

$ kubectl delete service myservice  
service "myservice" deleted  
  
$ kubectl delete deploy mygo-deployment  
deployment.apps "mygo-deployment" deleted