Intetnal Service Discovery with ClusterIP

**Internal Service Discovery**

* Visit the **vote** app from browser
* Attemp to vote by clicking on one of the options

observe what happens. Does it go through?

Debugging,

1. kubectl get pod
2. kubectl exec vote-xxxx ping redis

[replace xxxx with the actual pod id of one of the vote pods ]

keep the above command on a watch. You should create a new terminal to run the watch command.

e.g.

1. watch kubectl exec vote-kvc7j ping redis

**where, vote-kvc7j is one of the vote pods that I am running. Replace this with the actual pod id.**

Now create **redis** service

1. kubectl apply -f redis-svc.yaml
3. kubectl get svc
5. kubectl describe svc redis

Watch the ping and observe if it’s able to resolve **redis** by hostname and its pointing to an IP address.

e.g.

1. PING redis (10.102.77.6): 56 data bytes

**where 10.102.77.6**is the ClusterIP assigned to the service.

What happened here?

* Service **redis** was created with a ClusterIP e.g. 10.102.77.6
* A DNS entry was created for this service. The fqdn of the service is **redis.instavote.svc.cluster.local** and it takes the form of my-svc.my-namespace.svc.cluster.local
* Each pod points to internal DNS server running in the cluster. You could see the details of this by running the following commands

1. kubectl exec vote-xxxx cat /etc/resolv.conf

**[replace vote-xxxx with actual pod id]**

[sample output]

1. nameserver 10.96.0.10
2. search instavote.svc.cluster.local svc.cluster.local cluster.local
3. options ndots:5

where **10.96.0.10** is the ClusterIP assigned to the DNS service. You could co relate that with,

1. kubectl get svc -n kube-system

4. NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
5. kube-dns ClusterIP 10.96.0.10 <none> 53/UDP,53/TCP h
6. kubernetes-dashboard NodePort 10.104.42.73 <none> 80:31000/TCP 3m

**where, 10.96.0.10**is the ClusterIP assigned to **kube-dns** and matches the configuration in **/etc/resolv.conf** above.

**Creating Endpoints for Redis**

Service is been created, but you still need to launch the actual pods running **redis**application.

Create the endpoints now,

1. kubectl apply -f redis-deploy.yaml
2. kubectl describe svc redis

[sample output]

1. Name: redis
2. Namespace: instavote
3. Labels: role=redis
4. tier=back
5. Annotations: kubectl.kubernetes.io/last-applied-configuration={"apiVersion":"v1","kind":"Service","metadata":{"annotations":{},"labels":{"role":"redis","tier":"back"},"name":"redis","namespace":"instavote"},"spec"...
6. Selector: app=redis
7. Type: ClusterIP
8. IP: 10.102.77.6
9. Port: <unset> 6379/TCP
10. TargetPort: 6379/TCP
11. Endpoints: 10.32.0.6:6379,10.46.0.6:6379
12. Session Affinity: None
13. Events: <none>

Again, visit the vote app from browser, attempt to register your vote and observe what happens now.