ThoughtWorks®

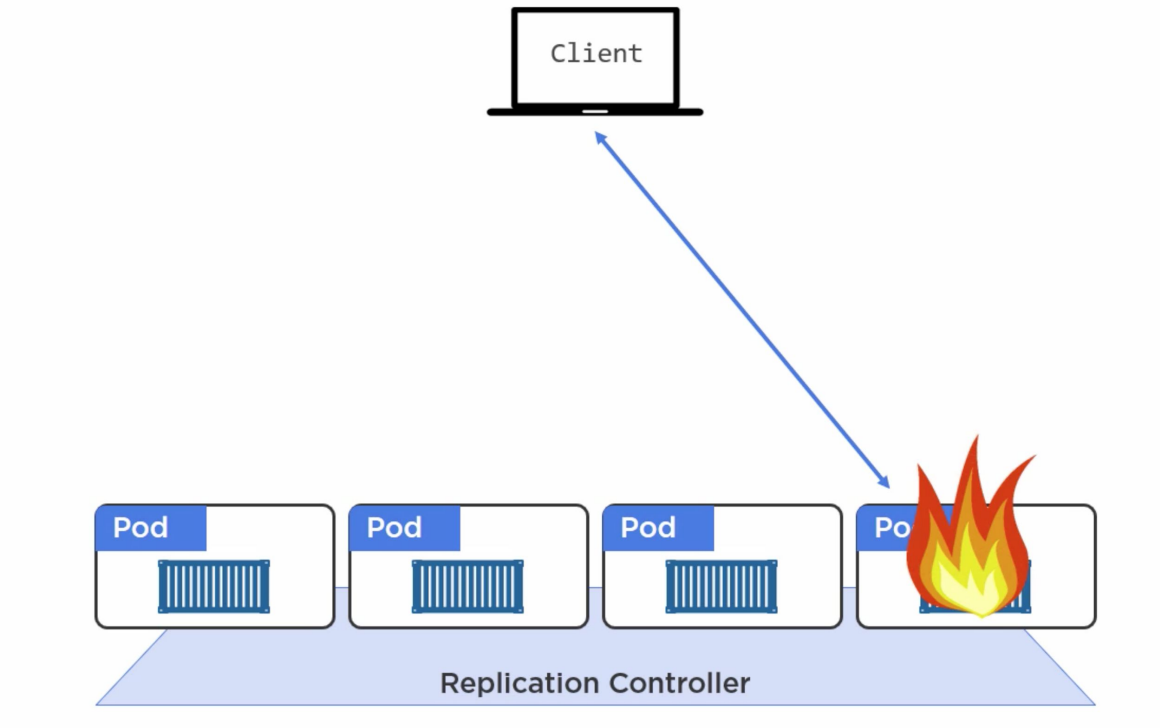
SERVICES

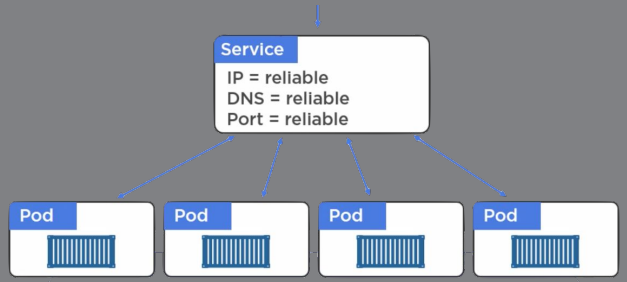
Rise of the Containers Workshop

How do we access our app ?

- From outside the cluster
- From inside the cluster

Through Pod IP ?





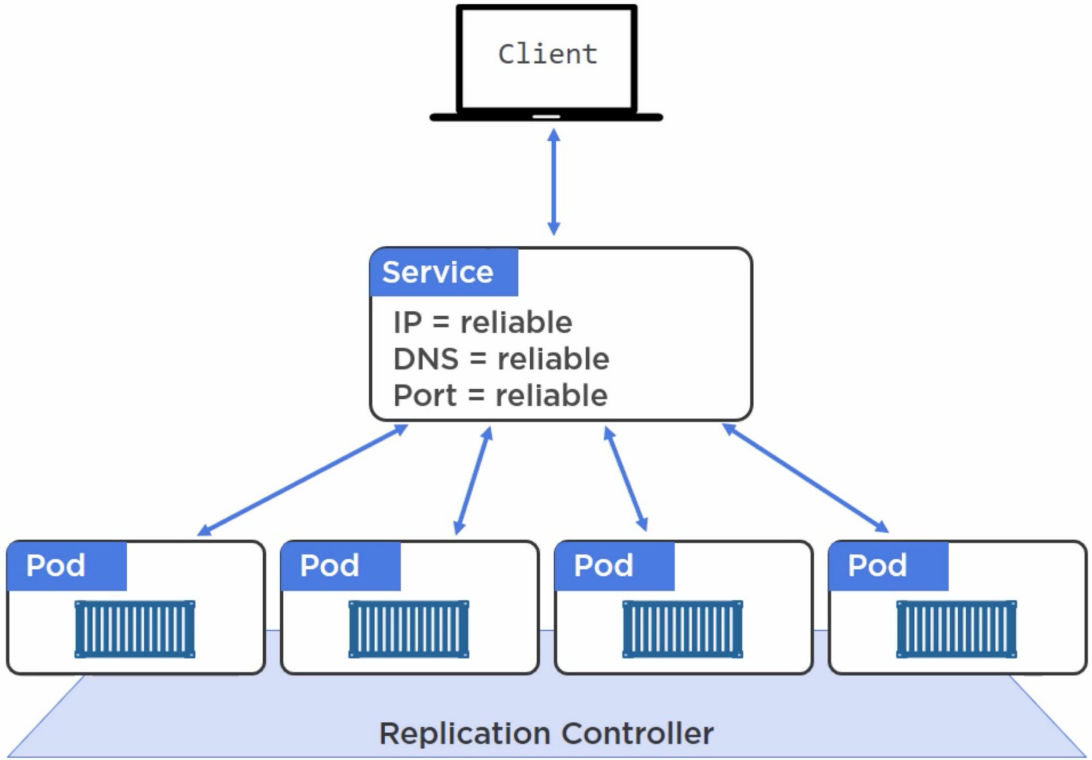
Service

A Kubernetes Service is an abstraction which defines a logical set of Pods and a policy by which to access them - sometimes called a micro-service.

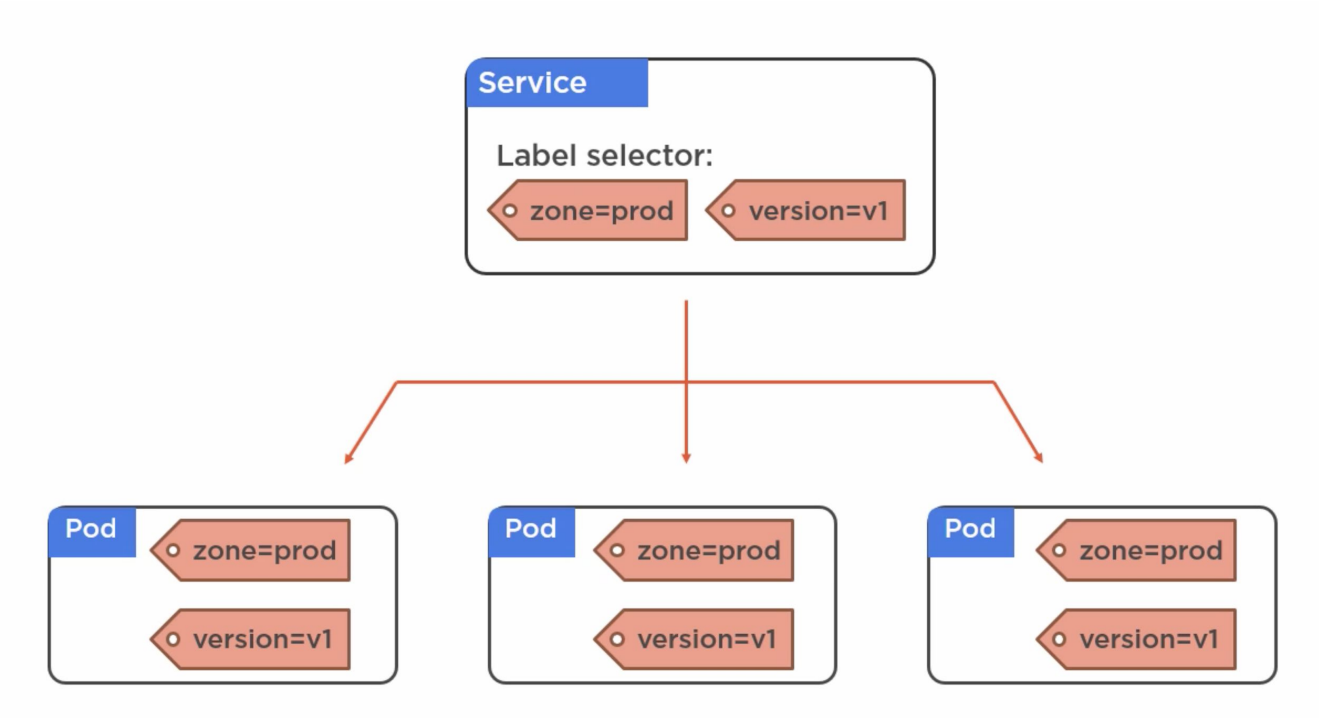
Services

- K8s objects similar to Pods and ReplicaSets
- Abstraction of a logical set of Pods
- Uses labels and selectors to match set of Pods
- Acts as intermediary for Pods to talk to each other

Through Services....



Services and Labels



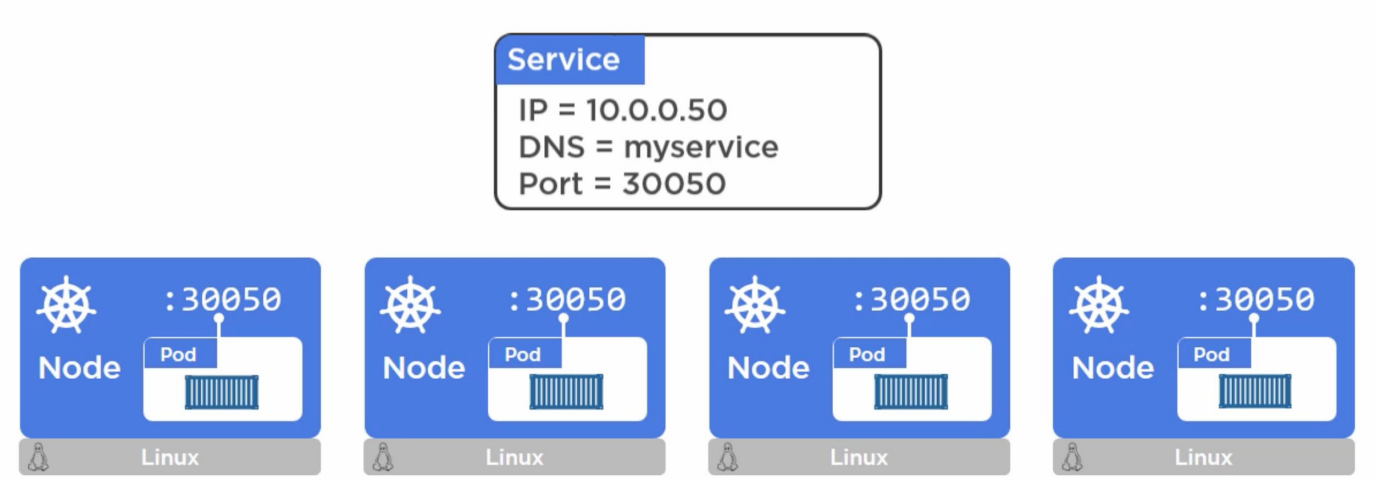
Service Demo

HANDS-ON 1

1. Create a ClusterIp service over already created pods.
2. SSH into minikube and access your app via service IP and clusterPort.

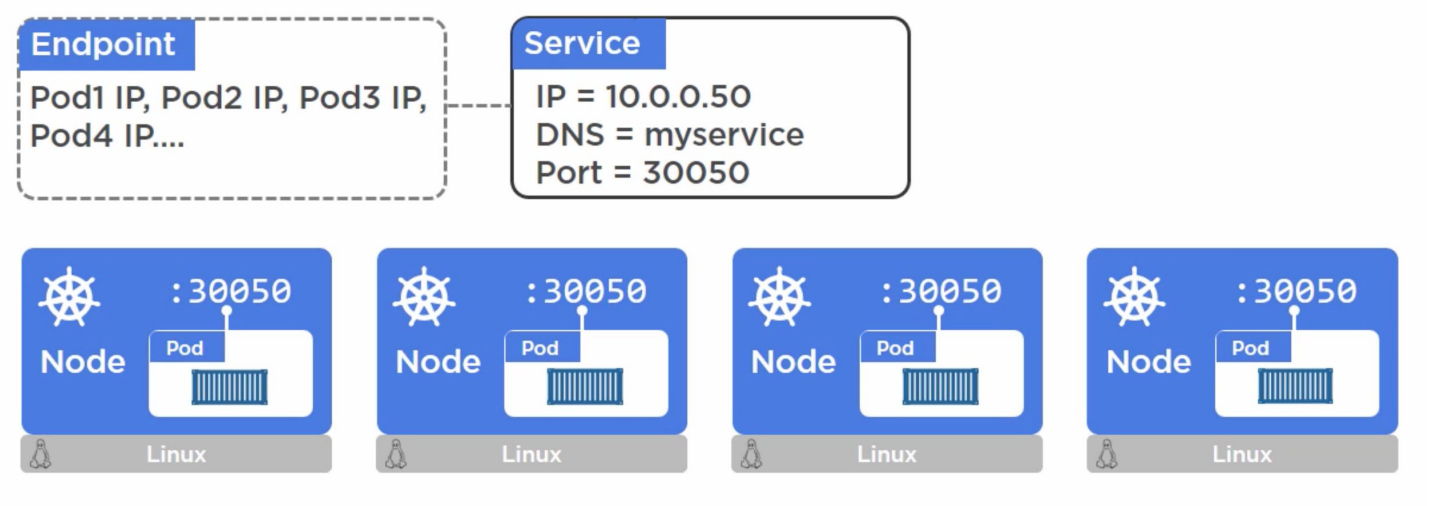
How does it work

- Every Service gets a virtual IP, port (kube-proxy) and a DNS (kube-dns) which never change
- Service load balances the request over different pods



Endpoints

- Each Service is associated with an Endpoint Object
- Contains the list of Pod IP addresses that the service is associated to.
- Keeps getting updated as the pods come and go.



Service Discovery

- Through Environment variables
- Through DNS (kube-dns)

Types of services

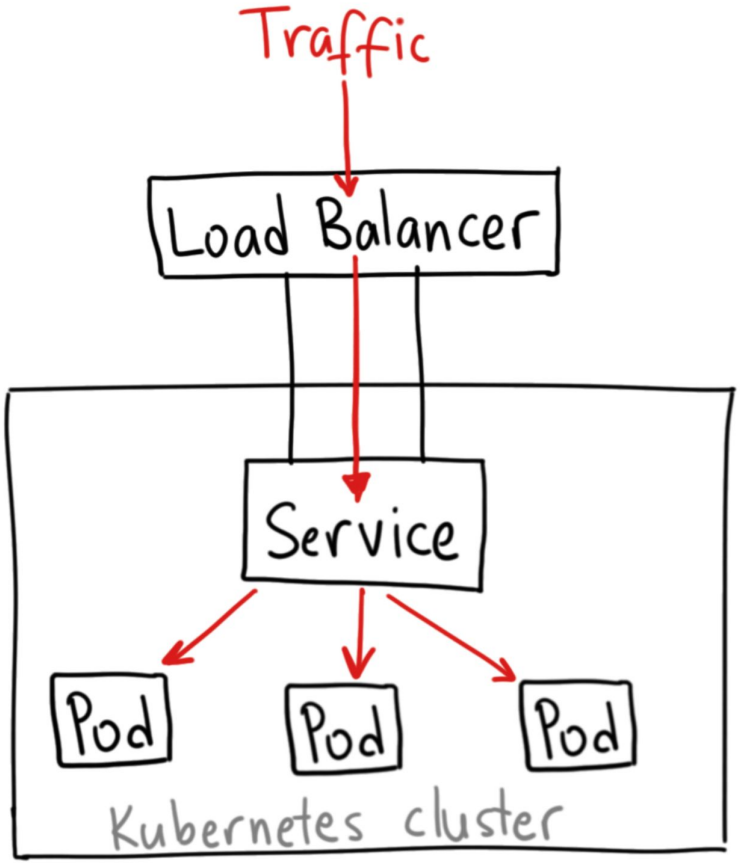
- Within the cluster
  - ClusterIP

- Outside the cluster
  - NodePort
  - LoadBalancer

    - ExternalName
    - Headless Service

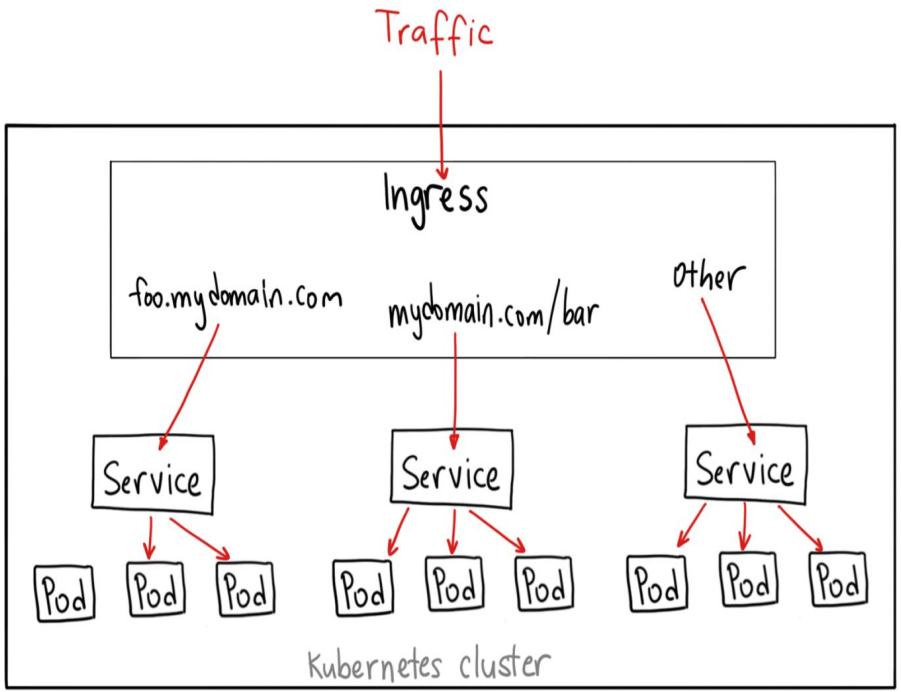
Service Demo

Load Balancer




```
apiVersion: v1
kind: Service
metadata:
  name: backend-service
spec:
  type: LoadBalancer
  ports:
  - port: 80
    targetPort: 80
    protocol: TCP
  selector:
    app: backend
```

Ingress




```
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: hello-ingress
spec:
  rules:
  - host: hello.example.com
    http:
      paths:
      - path: /a
        backend:
          serviceName: backend-a
          servicePort: 80
      - path: /b
        backend:
          serviceName: backend-b
          servicePort: 8080
```

HANDS-ON 2

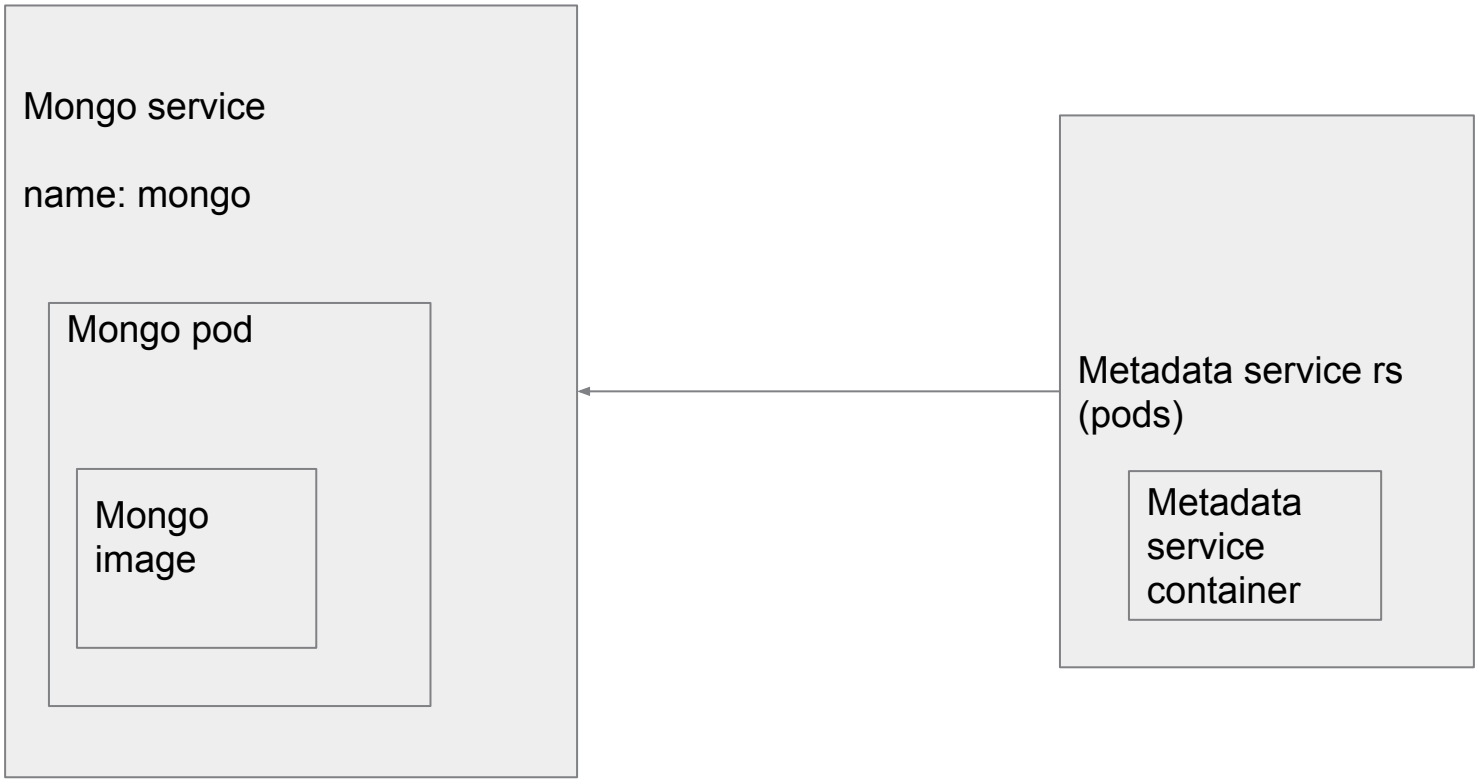
1. Create a NodePort service over the same pods.
2. Access the service via browser.

Readiness Probe

- The kubelet uses readiness probes to know when a Container is ready to start accepting traffic.
- A Pod is considered ready when all of its Containers are ready.
- One use of this signal is to control which Pods are used as backends for Services.
- When a Pod is not ready, it is removed from Service load balancers.

HANDS-ON 3

1. Create a Pod configuration(YAML) for MongoDB and create a Pod using kubectl create command.
2. Update MetadataService to connect to real MongoDB
3. Now redeploy MetadataService new version to see MongoDB connection working and Pod to Pod to communication channel.


Mongo service
name: mongo
Mongo pod
Mongo image
Metadata service rs (pods)
Metadata service container

THANK YOU

ThoughtWorks®