K8’s Service

An abstract way to expose an application running on a set of Pods as a network service.

Kubernetes gives Pods their own IP addresses and a single DNS name for a set of Pods, and can load-balance across them.

In Kubernetes, a Service is an abstraction which defines a logical set of Pods and a policy by which to access them

Defining a Service

A Service in Kubernetes is a REST object, similar to a Pod. Like all of the REST objects, you can POST a Service definition to the API server to create a new instance. The name of a Service object must be a valid [DNS label name](https://kubernetes.io/docs/concepts/overview/working-with-objects/names#dns-label-names).

For example, suppose you have a set of Pods that each listen on TCP port 9376 and carry a label app=MyApp:

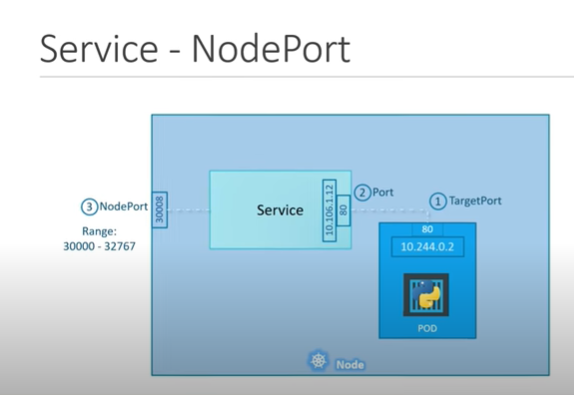
* ClusterIP: Exposes the Service on a cluster-internal IP. Choosing this value makes the Service only reachable from within the cluster. This is the default ServiceType.
* [NodePort](https://kubernetes.io/docs/concepts/services-networking/service/#nodeport): Exposes the Service on each Node’s IP at a static port (the NodePort). A ClusterIP Service, to which the NodePort Service routes, is automatically created. You’ll be able to contact the NodePort Service, from outside the cluster, by requesting <NodeIP>:<NodePort>.
* [LoadBalancer](https://kubernetes.io/docs/concepts/services-networking/service/#loadbalancer): Exposes the Service externally using a cloud provider’s load balancer. NodePort and ClusterIP Services, to which the external load balancer routes, are automatically created.
* [ExternalName](https://kubernetes.io/docs/concepts/services-networking/service/#externalname): Maps the Service to the contents of the externalName field (e.g. foo.bar.example.com), by returning a CNAME record

with its value. No proxying of any kind is set up.

# Node Port service

Nodeport Exposes the Service on each Node’s IP at a static port or A NodePort is an open port on every node of your cluster. Any traffic that is sent to this port is forwarded to the service.

Kubernetes controller allocates a port from a range specified by (typically 30000–32767).



We can access the application or service from outside the cluster, by requesting **<any NodeIP>:<NodePort>**

kubectl run mypod --generator=run-pod/v1 --image=nginx --port=80 --labels="myapp=mynginxapp"

Deploy Service:

kubectl create –f <https://raw.githubusercontent.com/t2run/learning/master/K8s/services/service3.yaml>

apiVersion: v1

kind: Service

metadata:

name: mynginxsvc

spec:

type: NodePort

ports:

- port: 80

nodePort: 30180

name: http

- port: 443

nodePort: 31443

name: https

selector:

myapp: mynginxapp

kubectl get svc

kubectl get pods -o wide

kubectl get node –o wide

**NodeIP>:<NodePort>**

# ClusterIP

kubectl apply -f https://raw.githubusercontent.com/openshift-evangelists/kbe/master/specs/services/rc.yaml

kubectl apply -f https://raw.githubusercontent.com/openshift-evangelists/kbe/master/specs/services/svc.yaml

kubectl get pods -l app=sise

curl 172.17.0.3:9876/info

kubectl get svc

curl 172.30.228.255:80/info