TEL4VN

Kubernetes networking management

THAO LUONG
09/2020

Content

- Kubernetes network model
- Network topology
- Pod networking
- Container Network Interface
- CoreDNS

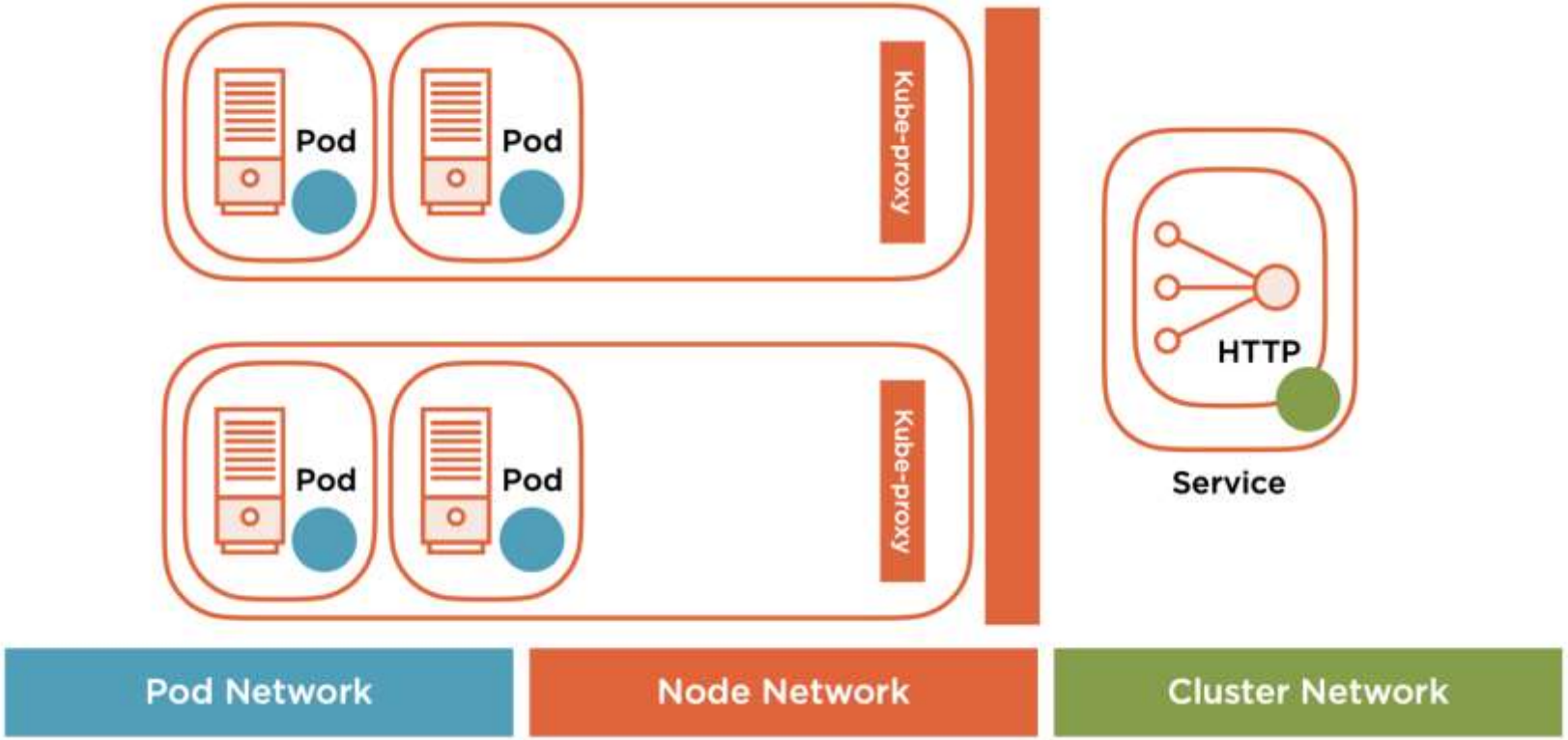
k8s networking model

All Pods can communicate with each other on all Nodes

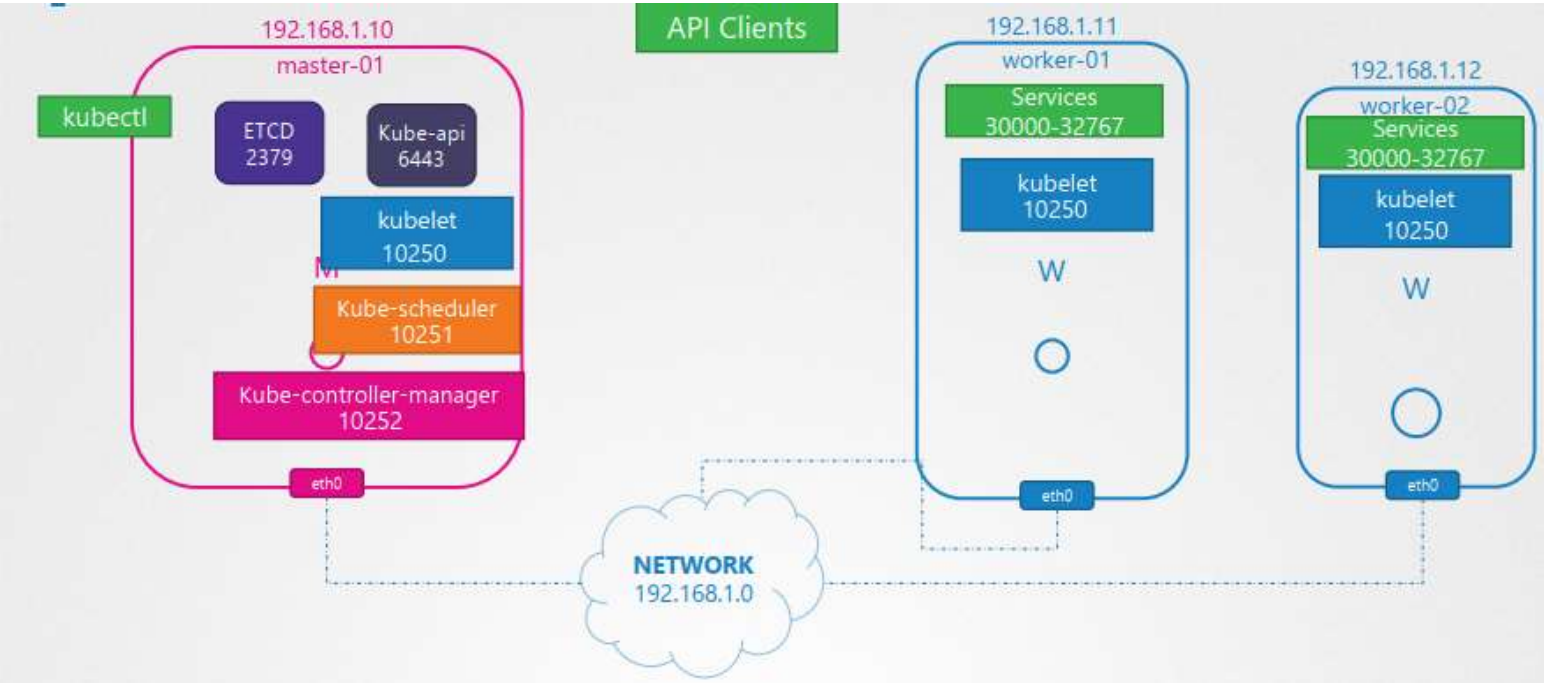
Agents on a Node can communicate with all Pods on that Node

No Network Address Translation (NAT)

Network Topology



Networking cluster Nodes



Pod Networking

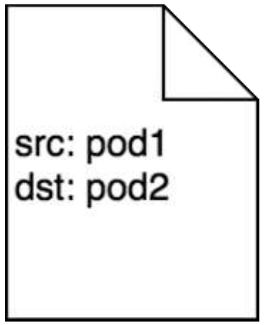
Pod share a network namespace

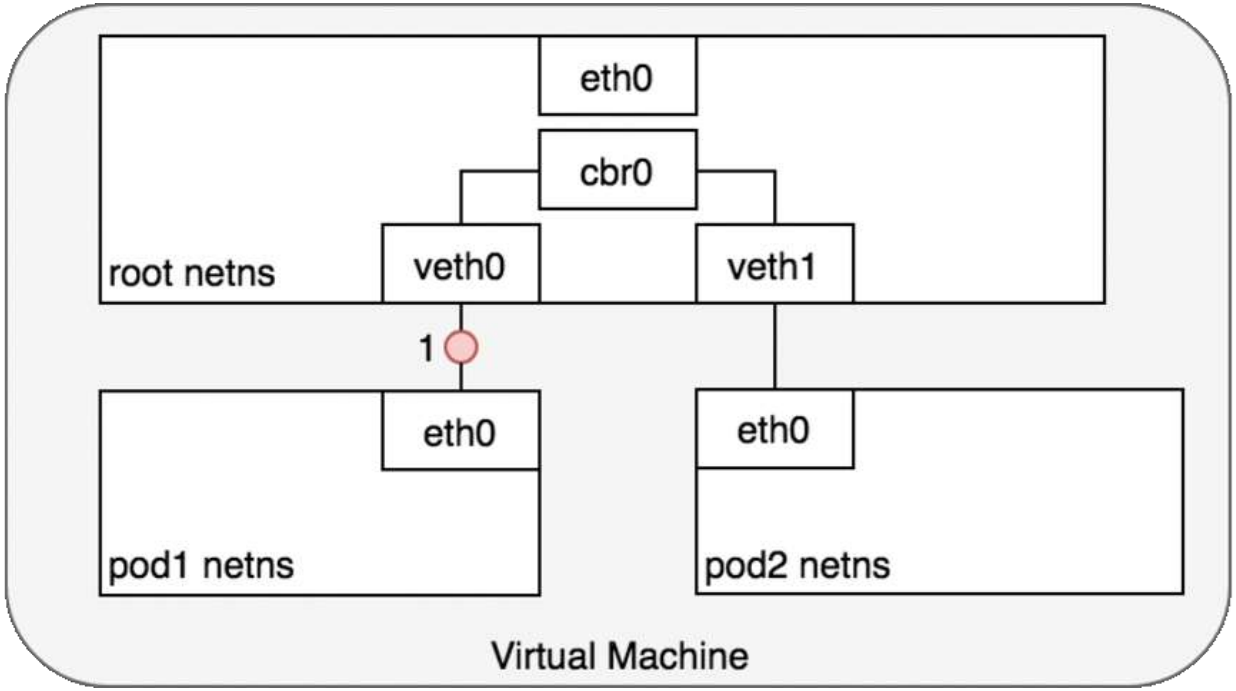
Containers in a Pod communicate over `localhost`

Pause/Infrastructure container

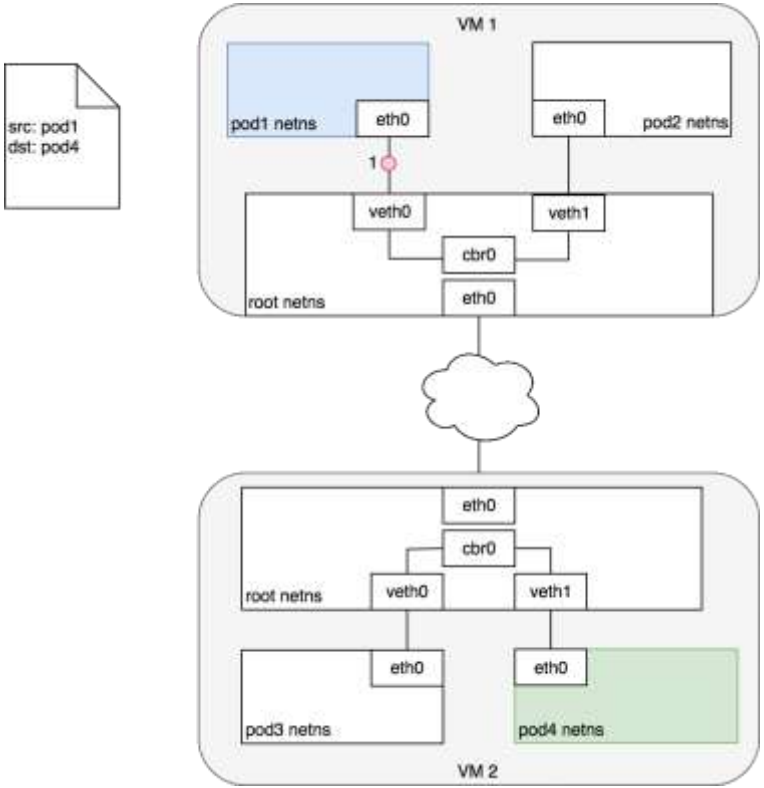
- Starts the networking namespace
- If the application container restarts the network will persist
- Lifecycle of the Pod

Pod-to-Pod Networking





Pod-to-Pod Networking Across Nodes



Route Table of Nodes

```
gke-cluster-1-default-pool-487a6374-lj8l / # route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         10.148.0.1      0.0.0.0         UG    1024   0        0 eth0
10.48.2.0       0.0.0.0         255.255.255.0   U     0      0        0 cbr0
10.148.0.1      0.0.0.0         255.255.255.255 UH    1024   0        0 eth0
169.254.123.0   0.0.0.0         255.255.255.0   U     0      0        0 docker0
```

```
gke-cluster-1-default-pool-487a6374-nz29 / # route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         10.148.0.1      0.0.0.0         UG    1024   0        0 eth0
10.48.1.0       0.0.0.0         255.255.255.0   U     0      0        0 cbr0
10.148.0.1      0.0.0.0         255.255.255.255 UH    1024   0        0 eth0
169.254.123.0   0.0.0.0         255.255.255.0   U     0      0        0 docker0
```

k8s with Calico

```
gke-cluster-2-calico-default-pool-2bb75e10-0ztm / # route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         10.148.0.1      0.0.0.0         UG    1024   0        0 eth0
10.24.1.2       0.0.0.0         255.255.255.255 UH    0      0        0 caliedfbbfb72f7
10.24.1.3       0.0.0.0         255.255.255.255 UH    0      0        0 califb09e4eac3d
10.24.1.4       0.0.0.0         255.255.255.255 UH    0      0        0 calieecd6fd5bbe
10.148.0.1      0.0.0.0         255.255.255.255 UH    1024   0        0 eth0
169.254.123.0   0.0.0.0         255.255.255.0   U     0      0        0 docker0
```

```
gke-cluster-2-calico-default-pool-2bb75e10-rk6f / # route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         10.148.0.1      0.0.0.0         UG    1024   0        0 eth0
10.24.2.4       0.0.0.0         255.255.255.255 UH    0      0        0 cali9223a8d59a0
10.24.2.5       0.0.0.0         255.255.255.255 UH    0      0        0 cali3b8b1865e83
10.24.2.6       0.0.0.0         255.255.255.255 UH    0      0        0 caliad6916ee747
10.24.2.7       0.0.0.0         255.255.255.255 UH    0      0        0 calia074902df36
10.148.0.1      0.0.0.0         255.255.255.255 UH    1024   0        0 eth0
169.254.123.0   0.0.0.0         255.255.255.0   U     0      0        0 docker0
```

CNI

CNI (Container Network Interface) is a project by CNCF that defines a specification which allows communication between containers. Kubernetes supports CNI plugins for the communication between pods

CNI

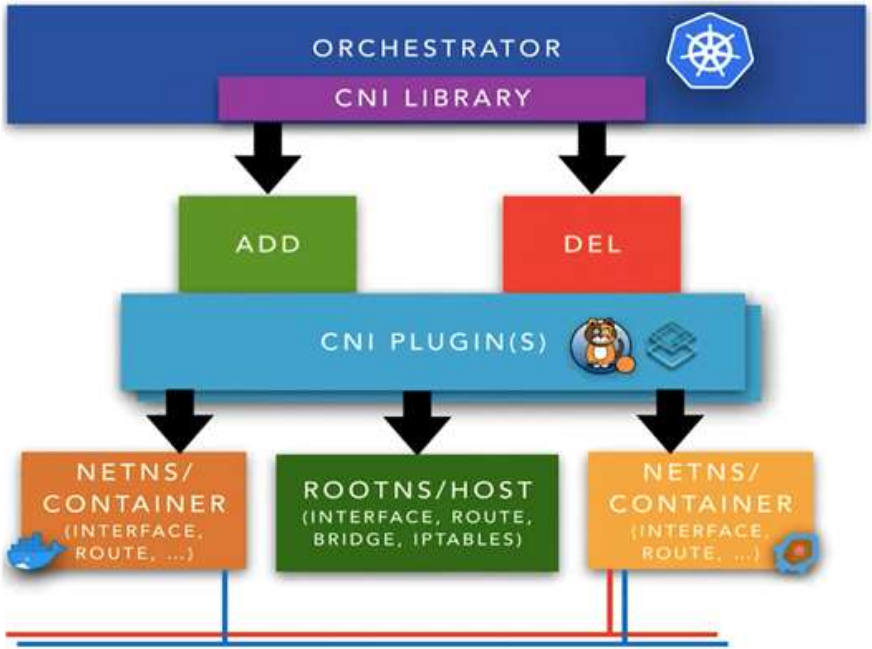


Figure 1: CNI plugin architecture

CNI

- Calico
- one of the most popular plugins
- default choice of the most of Kubernetes platforms (kubespray, docker enterprise, etc.)
- supports IP-IP encapsulation if BGP cannot be used
- supports Network Policies
- uses iptables for routing but it can be configured to use kube-proxy's IPVS mode

- Weave
- provides VXLAN tunneling solution
- all of the nodes are connected as mesh which allows it to run on partially connected networks
- does not scale well because of the mesh structure
- stores configuration files on pods instead of Kubernetes CRDs or etcd
- has an encryption library
- supports Network Policies

Cluster DNS

DNS is available as a Service in a Cluster

Pods are configured to use this DNS

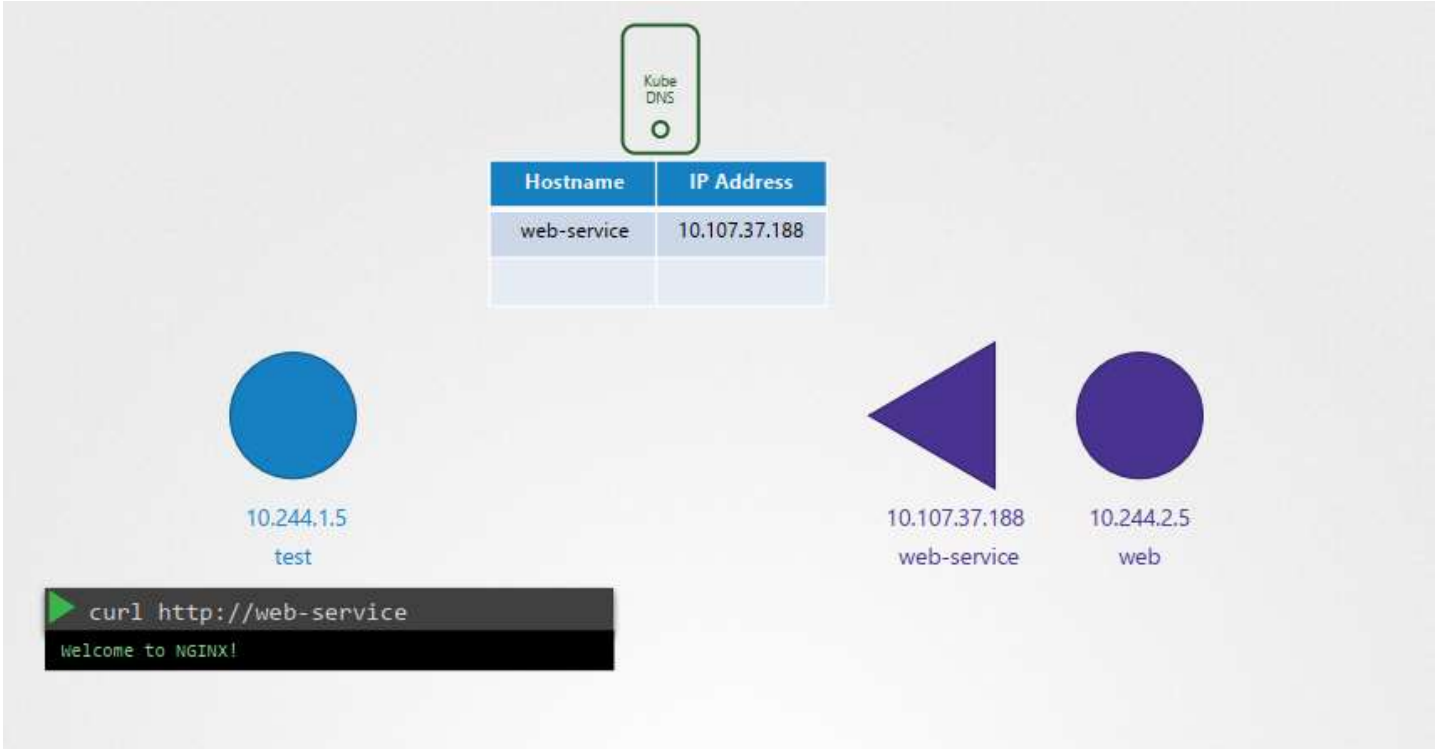
DNS records

- Services - A/AAAA records
- Namespaces - subdomains

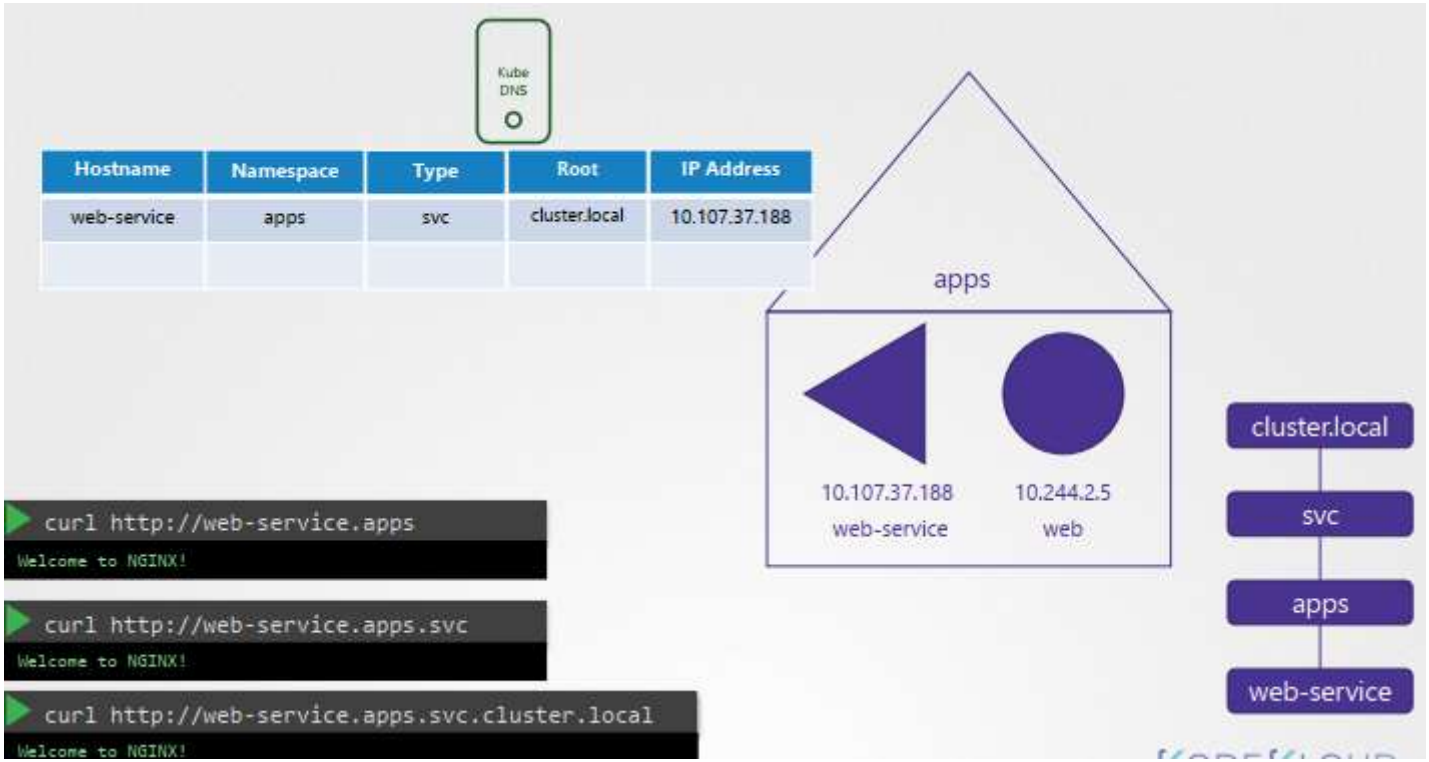
Core to Service discovery

Customize both the DNS Service and Pods configuration

Cluster DNS


Kube DNS
Hostname
IP Address
web-service
10.107.37.188
10.244.1.5
test
10.107.37.188
web-service
10.244.2.5
web
curl http://web-service
Welcome to NGINX!

Cluster DNS


Kube DNS
Hostname
Namespace
Type
Root
IP Address
web-service
apps
svc
cluster.local
10.107.37.188
apps
10.107.37.188
web-service
10.244.2.5
web
cluster.local
svc
apps
web-service
curl http://web-service.apps
Welcome to NGINX!
curl http://web-service.apps.svc
Welcome to NGINX!
curl http://web-service.apps.svc.cluster.local
Welcome to NGINX!


Q
A