Kubernetes (K8s)

Background.

- **An open-source system for automating the deployment, scaling and management of containerized applications.**
  - **A container orchestration platform.**
- **Originally announced by Google in 2014.**
- **In 2015, Google donated it to the Cloud Native Computing Foundation (CNCF).**

- **Kubernetes is the Greek word for helmsman – a person that steers a ship that carries containers of goods.**

The problems it solves.

- **Problem 1: An application is comprised of multiple containers running on a cluster of nodes (on-premise server, Cloud VMs I e.g. AWS EC2). A container or an entire node crashes.**
- **K8s provides monitoring and self-healing, to ensure high availability.**

- **Problem 2: A container experiences a spike in traffic resulting in increased latency.**
- **K8s provides auto-scaling and load balancing. It replicates the container on the node and/or across the cluster of nodes and load balances the requests between them. It scales up and down based on demand.**

The problems it solves.

- **Problem 3: You regularly upgrade an image and want to apply the change to a fleet of running containers without effecting downtime.**
  - **K8s provides rolling deployments - delete and restart each container, one by one rather than all at once.**
  - **Also supports canary deployments.**

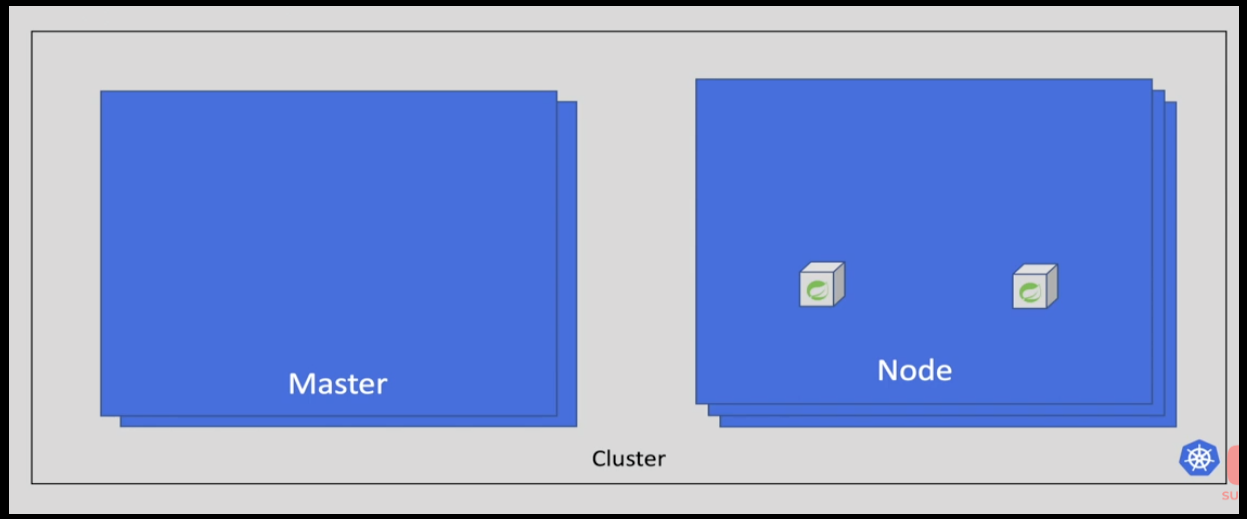
- **Problem 4: I want to upgrade my entire application from v1 to v2.**
  - **K8s provides automatic rollout and rollback.**

The problems it solves.

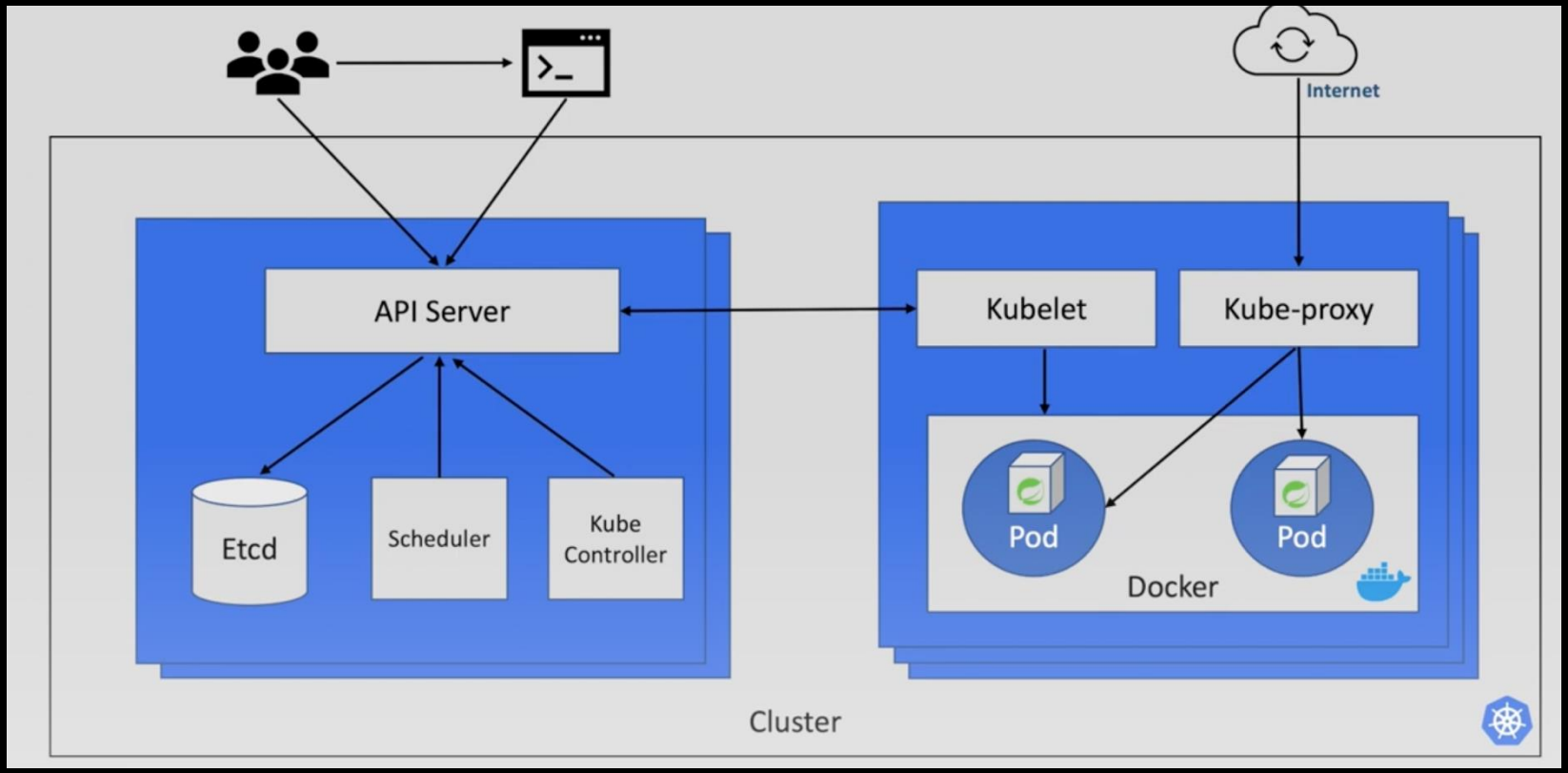
- **Problem 5: I want to reconfigure my application's without rebuilding images.**
  - **K8s provides secrets and configuration mappings to safeguard sensitive data and avoid unnecessary rebuilding of images.**

K8s Architecture

- K8s is installed over a set of nodes (VMs)
- Worker nodes host containers - the Data plane
- Master nodes (Control plane) manages the workers – More than one master for fault tolerance and high availability.
- The set of masters and workers is called a cluster

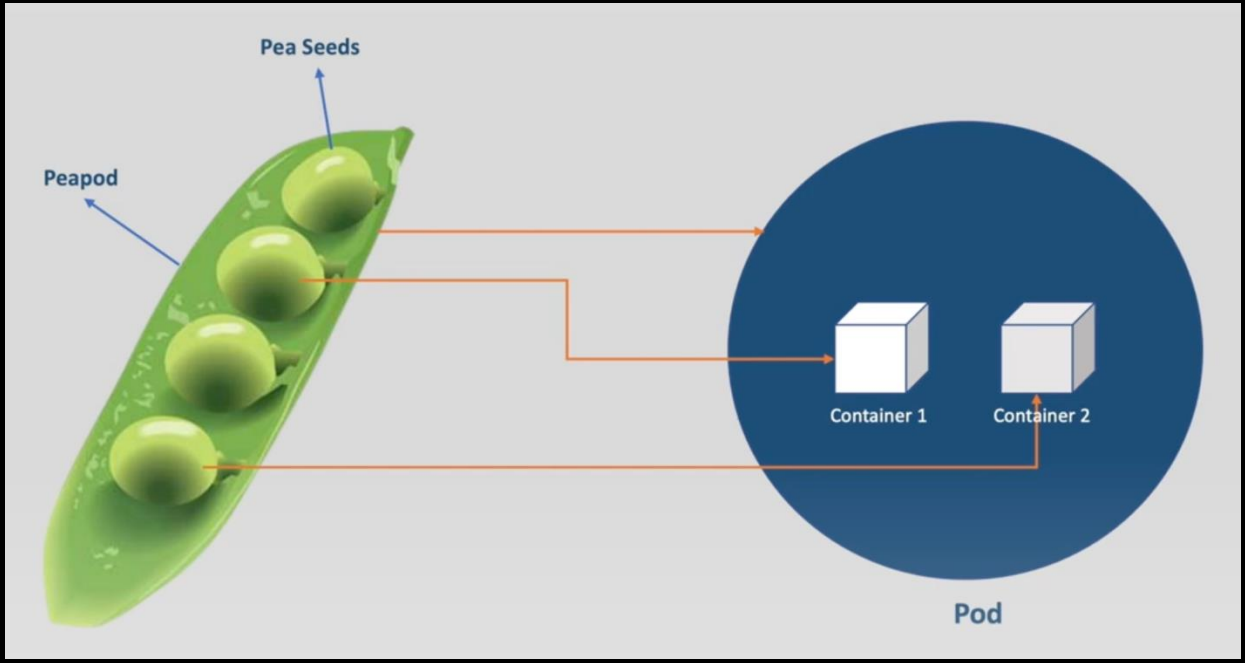


K8s Architecture



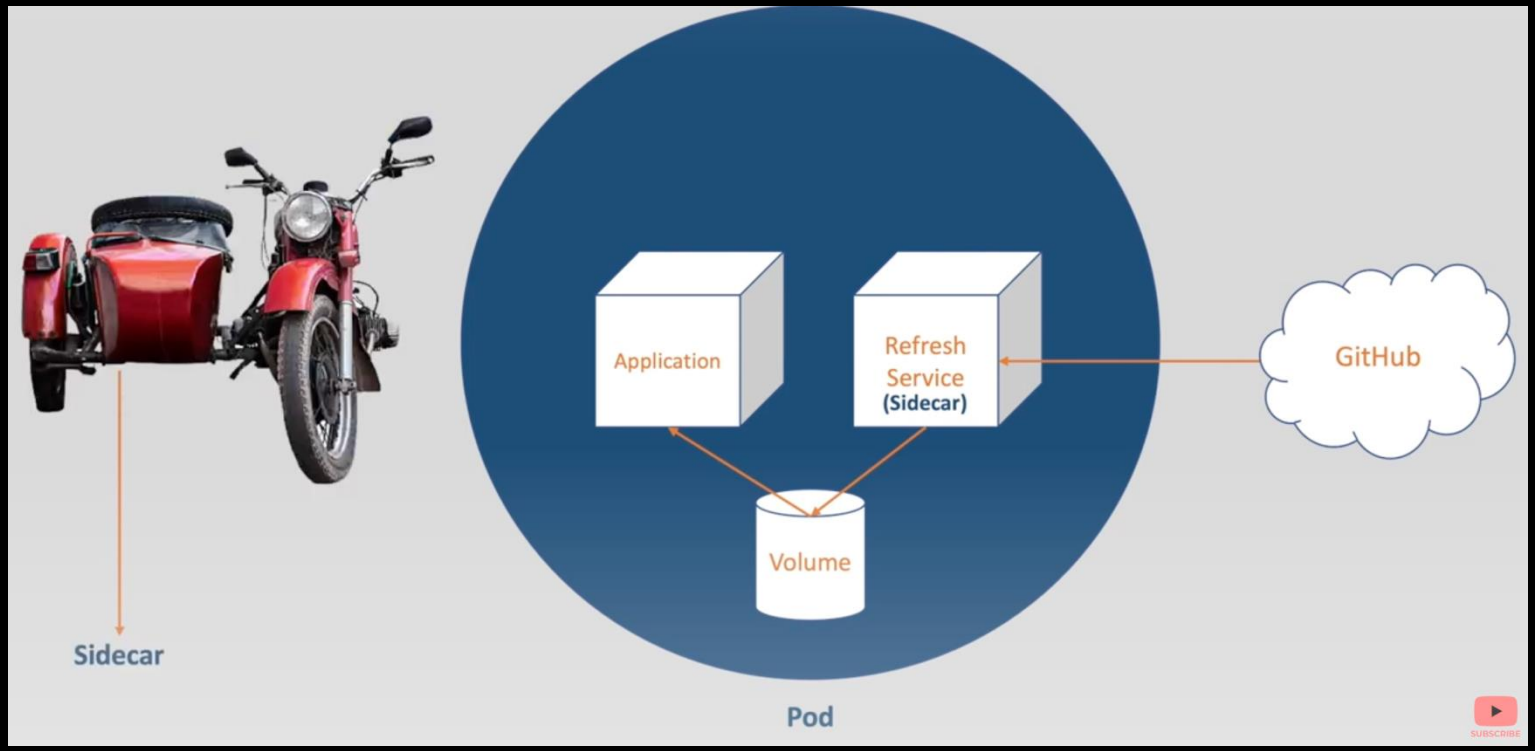
All about Pods

- The smallest unit of deployment is a pod.
- It encapsulates a group of containers, that share the same localhost network and storage.
- A pod is assigned a private IP address, but it's dynamic.



All about Pods

- When a pair of containers are tightly coupled, they must reside on the same worker node. The pod construct ensures this is satisfied.



Pod basics

- The containers in a pod share the same network (localhost), but use different ports.
- Deleting a pod will delete all its containers.
- To scale an app, we replicate its pod, where each one has its own group of containers.

- IP addressing:
  - A pod is assigned a private IP address.
  - On scale up, each pod replica is assigned its own IP.
  - Pod IPs are dynamic.

Pod basics

```
❯ oc run nginx-pod --image=nginx:latest
pod/nginx-pod created

❯ oc get pods
NAME        READY   STATUS    RESTARTS   AGE
nginx-pod   1/1     Running   0          12s
```

- Creating Kubernetes resources (e.g. pods) from the command line is cumbersome; instead, we uss declarative code files for better maintainability.
  - Files are termed manifests – yaml or json options.

Pod basics - Manifest files.

```
❯ oc apply -f 01-nginx-pod.yaml
pod/nginx-pod1 created

❯ oc get pods
NAME        READY   STATUS    RESTARTS   AGE
nginx-pod   1/1     Running   0          3d5h
nginx-pod1  1/1     Running   0          16s

❯ oc get pods -l team=integration     (Pod labels)
NAME        READY   STATUS    RESTARTS   AGE
nginx-pod1  1/1     Running   0          2m7s
```

Pod basics - Attributes

- The etcd stores lots of attributes about pods

```
❯ oc get pod nginx-pod1 -o wide
NAME        READY   STATUS    RESTARTS   AGE   IP             NODE      NOMINATED NODE
READINESS GATES
nginx-pod1  1/1     Running   0          13m   10.130.1.156   worker1   <none>
<none>
```

- Use describe command to get all etcd info:

```
❯ oc describe pod nginx-pod1
```

  - Response also shows the ‘events’ that occurred for a pod.

Pod basics - Debugging

- Three main options:

1. Port-forwarding is possible with cluster access:

```
❯ oc port-forward nginx-pod1  3000:80
Forwarding from 127.0.0.1:3000 -> 80
Forwarding from [::1]:3000 -> 80
```

2. Check container's logs:

```
❯ oc logs nginx-pod1
```

3. Open a terminal shell inside the container:

```
❯ oc exec -it nginx-pod1 -- /bin/bash
root@nginx-pod1:/#
```

ReplicaSets

- For high availability of an app (pod), K8s can create multiple copies in the cluster, called replica sets.

```
❯ oc apply -f 02-nginx-replica.yaml
replicaset.apps/nginx-replica created
❯ oc get pods
NAME                READY   STATUS              RESTARTS   AGE
nginx-replica-dmfvn   0/1     ContainerCreating   0          7s
nginx-replica-wc7j9   0/1     ContainerCreating   0          7s
❯ oc get rs       (rs - replicaset)
NAME            DESIRED   CURRENT   READY   AGE
nginx-replica   2         2         2       14s
```

Self-healing

❯ oc delete pod nginx-replica-dmfvn
pod "nginx-replica-dmfvn" deleted

❯ oc get pods

NAME	READY	STATUS	RESTARTS	AGE
nginx-replica-kkhs2	0/1	ContainerCreating	0	3s
nginx-replica-wc7j9	1/1	Running	0	7m36s

- The new pod's IP address is different to the one it replaced.
- A replica set's spec.selector property determines the pods it controls.
  - A pod's labels match the replica set's selector

To be continued ….