Kubernetes

Pronounced kubernets

# Container Orchestration

So, you have containerized your app now what? With an evil mastermind mindset, you should probably think about world domination. That is, scaling your app. How? More containers of course! How do you set up a network of these containers? Well, you could do it manually, or you could use a container orchestration tool.

A container orchestration tool helps in deployment, scaling, networking, and monitoring of your containers. It provides a framework for managing containers and the microservices they contain at scale.

## Deployment

* Deploying containers on a server.

## Scaling

* Having multiple containers running the same image

## Network

* Load balancing, service discovery

## Monitoring

* Service health checks, automatic spinning up of a service instance if one instance goes down, logging, monitoring ops per second

# K8s

Kubernetes is a portable, extensible, open-source platform for managing containerized workloads and services, that facilitates both declarative configuration and automation. It has a large, rapidly growing ecosystem. Kubernetes services, support, and tools are widely available.

Kubernetes is a container orchestration tool. It provides tools that help deploy and manage containers that run over a network. It lets us build application services that span multiple containers, schedule containers across a cluster, scale those containers, and manage their health over time. Similar tools are Docker Swarm and Apache Mesos

Note: K8s allows you to take advantage of existing docker workloads

## K8s Architecture

### Cluster

* A set of nodes that run containerized apps (i.e what you call a group of nodes)
  + Specifically a master node and a couple of worker nodes
* Allow containers to run across multiple machines and environments
* Your MSA app would run across a cluster

### Node

* May be a virtual or physical machine (basically a machine) depending on the cluster
* Managed by the control pane and contains the necessary services to run pods
  + The control plane is the container orchestration layer that exposes the API and interfaces to define, deploy, and manage the lc of containers

#### Master Node

* Controls the state of the cluster
* Origin of all task assignments
* Coordinates processes such as:
  + Scheduling and scaling apps
  + Maintaining cluster state
  + Implementing updates

#### Worker Nodes

* Contains the pods that run your microservices
* Perform tasks assigned by the master node
* Composed of: kubelet, container runtime, kube-proxy

##### Kubelet

* Primary node agent
* Registers the node with the api server (i.e the master node)
* Part of the node that is used in monitoring

##### Container Runtime

* Software responsible for running containers
* For example: Docker

##### Kube-proxy

* Network proxy
* Maintains network rules on nodes, these network rules allow network communication to your pods from network sessions inside or outside of your cluster

### Pod

* Smallest execution unit in K8s
* Made up of one or more containers (usually just one)
* Ephemeral in nature, if a pod (or the node it executes on) fails, K8s can automatically create a new replica of that pod to continue ops
* Represent the processes running on a cluster
* By limiting pods to a single process, K8s can report on the health of each process running in the cluster
* Components of a pod: unique IP address, persistent storage volumes, config info that determines how a container should run

### Container

* A container is a container is a container is a container
* A container is a process that you want to run on a virtual file system (an image)

## K8s Tools

### Kubectl

* K8s command line tool
* Allows you to interact with the API in the master node to deploy and manage your cluster

### API

* Lets you query and manipulate the state of objects in K8s
* The kubectl uses the API

The core of Kubernetes' [control plane](https://kubernetes.io/docs/reference/glossary/?all=true#term-control-plane) is the [API server](https://kubernetes.io/docs/concepts/overview/components/#kube-apiserver). The API server exposes an HTTP API that lets end users, different parts of your cluster, and external components communicate with one another.

### Minikube

* Local K8s

### Docker

* Container runtime used in your nodes

### AKS

* Azure Kubernetes service

## Deployment with K8s

### Object config file

* used to create objects in your cluster
* Written in yaml

### Pods

* You can manually deploy a process to run on a pod, you’ll get a specific ip address
* You run into an issue if you want to deploy a certain number of containers, thus you use deployments

### Deployment

* An abstraction over pods, instead of manually deploying a process to a pod you can actually specify how many pods you want deployed
* Deployments schedule and automatically deploy pods based on specification
* So if a pod goes down, you try to recreate an instance of that pod to run in a node

### Services

* Allow you to be able to leverage built in load balancing and the service registry of k8s
* Allows you to access your pods as the service they represent rather than individual copies of the service
* Gives you a singular access point to the multiple pods that run a singular process

Resources:

[Vid on container orchestration](https://www.youtube.com/watch?v=kBF6Bvth0zw)

[Kubernetes Explained](https://www.youtube.com/watch?v=aSrqRSk43lY)

[Kubernetes and Docker](https://www.youtube.com/watch?v=2vMEQ5zs1ko)

[K8s Cluster](https://www.vmware.com/topics/glossary/content/kubernetes-cluster)

[K8s Pods](https://www.vmware.com/topics/glossary/content/kubernetes-pods)

[Master Node Components](https://medium.com/@AADota/kubernetes-architecture-master-node-components-summary-b44ceae59e4)

[K8s Docs](https://kubernetes.io/docs/concepts/overview/)