Kubernetes Primer

Kubernetes Background

Kubernetes is an application orchestrator.

It orchestrates containerized cloud-native microservices apps.

Orchestrator

An orchestrator is a system that deploys and manages applications. It can deploy your applications and dynamically respond to changes.

Containerized App

A containerized application is an app that runs in a container.

Before containers, apps ran on phtsical servers or virtual machines. Containers are the next iteration of package and ru napps.

- Faster, lightweight, suited to business reqs than servers and VMs
- Noew apps run in containers in could-native ecosystems

Cloud Native

A cloud-native app is designed to meet cloud-like demands of

- auto-scaling
- self-healing
- rolling updates
- rollbacks

Cloud-native apps are not just fot public cloud. The y can also run anywhere that have Kubernetes. Cloud native is about the way applications behave and react to events

Microservices Apps

A microservices app is built from lots of small, specialised, independient parts that work together to form a meanigful app.

Each individual services is called a microservice. Web frontend, catalog, shopping cart, authentication, logging, persistent store, etc. Each microservice runs as one or more containers.

Kubernetes

Kubernetes deploys and manages (orchestrates) applications that are packaged and run as a containers (containerized) and that are built in ways (cloud-native microservices) that allow them to scale, self-heal and be updated in-line with modern cloud-like requeriments.

History

Google created a new platform called Kubernets that it donated it to the newly formed Cloud Native Computing Fundation CNCF in 2014 as an open-source project.

Kubernetes enables abstracts underlying infrastructure and it makes it easy to move applications on and off clouds. Kubernetes in 2014 has become the most important cloud-native technology on the planet.

- Written in Go
- Built in GitHub
- Discussed on Twitter @kubernetsio and slack.k8s.io

Kubernetes and Docker

Docker builds applications into containeri mages and can run them as containers.

Kubernets can't do either of those. Instead, it sits a higher level and orchestrates things.

Kubernetes make high-level orchestration decisions such as which nodes should ru nthe containers. The Docker runtime is bloated and overkill for what Kubernetes needs. The Kubernetes project began work to make the container runtime layer pluggable so that users could choose the best container runtime for their needs.

In 2016 Kubernetes introduced the container runtime interface (CRI) that made this cntaoiner runtime layer pluggable. In 2020 Kubernetes deprecated the Docker runtime.

Containerd has replaced Docker as default container runtime in most. Containerd is a stripped-down version of Docker optimized for Kubernetes. All container images created by Docker will work on Kubernetes.

Develop teams uses Docker to build images and operations teams uses Kubernetes to run them.

Kubernetes as OS of the Cloud

In many ways, Kubernetes is like an OS for the cloud.

- You install traditional Linux on a server, and it abstracts server resources and schedules application processes.
- You install Kubernetes on a cloud, and it abstracts cloud resources and schedules application microservices.

In the same way Linux abstracts the hardware differences between server plataforms, Kubernetes abstracts the differences between different private and public clouds.

Kubernetes is a major step towards a true hybrid cloud. allowing to seamlessly move and balance workloads across multiple different public and private cloud infrastructures.

You can also migrate to and from differente clouds, meaning you can choose a cloud today and not have to stick with that decision for the rest of your life.

Application Scheduling

As a OS abstracts a specific computer resources (CPU, memory, storage, networking) Kubernetes does a similar thing with cloud and datacenter resources.