Cloud Computing

Kubernetes

Peyer Lars

Berger Adrian

Version 0.1

24. November 2020

1 Einleitung

Dieses Dokument beinhaltet eine Übersicht zur Container-Orchestrierungsplattform Kubernetes.

Damit diese Thematik erlätert werden kann, wird ein grundsätzliches Verständnis von Container vorausgesetzt. Dabei werden hauptsächlich Grundkentnisse zu Docker vorausgesetzt.

Eine kurze Übersicht über Docker kann unter dem folgenden Link gefunden werden:

https://www.redhat.com/en/topics/containers/what-is-docker

2 Übersicht

2.1 Was ist Kubernetes?

Bei Kubernetes, kurz k8s, handelt es sich um ein Open-Source Projekt, welches seinen Existenz der Idee verschiedener Google Entwickler zu verdanken hat.

Dabei war die Grundidee, ein System zu erstellen, welches die Ressourcen eines Servers automatisch idealen verwenden kann.[1]

Kubernetes ist griechisch und bedeutet Steuermann. Sowie ein Steuermann die Arbeiten auf seinem Schiff verwaltet, so verwaltet Kubernetes verschiedne Container auf einem Server. Unter Verwaltung versteht sich hierbei ebenfalls die Bereitstellung und Skalierung von Containern. [2]

Technisch spricht man dabei von einem Container-Orchestrierungs-System für Container-Applikationen.

2.2 Wieso Kubernetes?

2.2.1 Orchestrierung

Mittels Kubernetes können die Ressourcen auf dem Server je nach Auslastung automatisch skaliert und an die verschiedenen Container verteilt werden. Dadurch kann der Server bei hoher Auslastung weiterhin alle Container wie vom Entwickler vorgesehen betreiben.

Dabei beschränkt sich Kubernetes nicht auf die Verwendung von einem einzelnen Server, sondern wird im Regelfall über mehrerer Server verteilt. So können offene Anfragen stehts an den Server weitergeleitet werden, welcher noch die notwendigen Ressourcen zur Verfügung hat.

2.2.2 Self-Healing

Kubernetes kennt das Prinzip von Self-Healing (Selbstheilung). Fällt ein Server oder auch nur ein laufender Container aus, erkennt Kubernetes dies und startet automatisch auf einem anderen Server den oder die fehlenden Container neu. Dadurch kann eine hohe Verfügbarkeit von Applikationen gewöhrleistet werden, da Abstürtze im System automatisch behoben werden.

2.2.3 Verbreitung

Kubernetes ist, allem voran dank der Unterstützung von Google, aktuell eine der am besten unterstützten Container-Orchestrierungs-Systemen. So gibt es unter anderem bei Amazon Web Services (AWS), Microsoft Azure und Google Cloud einfache Konfigrationsoptionen für Kubernetes. Ebenfalls haben in den letzten Jahren auch immer mehr kleinere Hosting-Provider damit begonnen, Kuberntes als Dienstleistung anzubieten.

2.3 Alternativen

Es gibt verschiedene Alternativen zu Kubernetes. Im folgenden stellen wir zwei der beanntesten davon vor:

2.3.1 Openshift

Bei OpenShift handelt es sich um ein Projekt von Red Hat. Im Gegensatz zu Kubernetes ist OpenShift nicht Open-Source sondern wird von Red Hat als kostenpflichtigen Service angeboten. Dabei bassiert dies auf dem Open-Source Tool OKD, was wiederum auf Kubernetes aufbaut. OpenShift ist somit im wesentlichen eine Erweiterung zu Kubernetes, die das Deployment und Monitoring vereinfacht. Für Einsteiger bietet dies oft eine einfacheren Einstieg in die Orchestrierungwelt als Kubernetes. Dafür sind die Konfigurationsoptionen für Kubernetes zahlreicher. [3]

2.3.2 Docker Swarm

Docker Swarm ist verwendet als Engine direkt Docker. Dadurch ist die Einrichtung von Swarm meist einfacher als das Aufsetzen eines Kubernetes Clusters. Swarm ist generel simpler zu konfigurieren, da es weniger Funktionalitäten bietet als Kubernetes. Ebenfalls ist kein automatisches Skalieren innerhalb von Swarm möglich. [4]

3 Architektur

3.1 Master-Nodes

Kubernetes funktioniert nach dem Master-Node Prinzip. Der Master Server ist dabei der Hauptknotenpunkt von Kubernetes. Er kontrolliert und koordiniert alle Aktionen, die im Cluster erfolgen. Die Node Server warten auf Befehle vom Master und führen diese anschliessend aus.

3.2 Komponenten

3.2.1 Pod

3.2.2 Replication Controller

3.2.3 Service

3.2.4 Kubelet

3.2.5 kubectl

3.2.6 etcd

3.2.7 deploment

3.3 Schnittstellen

4 Beispiels-Applikation in einem Kubernetes Cluster

Für unser Beispiel haben wir eine simple Applikation gewählt, die aus den folgenden Containern besteht:

Vue.js Frontend Container Django Backend Container Postgres Datenbank Container

Diese Aplliaktion soll nun bei Hetzner in einem Kubernetes Cluster betrieben werden.

Literatur

[1] Red Hat. *Kubernetes (k8s) erklärt.* URL: https://www.redhat.com/de/topics/containers/what-is-kubernetes.

[2] SysEleven MetaKube. *Kubernetes einfach erklärt: Antworten vom Techie für den Noobie.* 2019. URL: https://blog.syseleven.de/kubernetes-einfach-erklaert.

[3] Tomasz Cholewa. *10 most important differences between OpenShift and Kubernetes.* 2019. URL: https://cloudowski.com/articles/10-differences-between-openshift-and-kubernetes/.

[4] Tomasz Cholewa. *10 most important differences between OpenShift and Kubernetes.* URL: https://cloudowski.com/articles/10-differences-between-openshift-and-kubernetes/.