ENTAGO

Dynamische Skalierung und Orchestrierung von containerbasierten Anwendungen mit OpenShift

Der vorliegende Artikel erläutert die Begriffe und Konzepte von containierbasierten Applikationen und ihrer Orchestrierung in einem dynamischen Enterprise-Umfeld und wie die damit einhergehenden Herausforderungen mit Hilfe von Orchestrierungsplattformen gemeistert werden können.

Herausforderungen von komplexen Systemen

Der Trend zu einer immer weiter gehenden Modularisierung in Form von Micro-Services, in einer Welt der verteilten Systeme auf verschiedenen Cloud-Systemen, hat für den Betrieb und das Management der Software, aber auch für IT-Verantwortliche und Entscheider, einige neue Herausforderungen und eine erhöhte Komplexität mit sich gebracht.

Container-Technologie

Das Konzept der Containerisierung hat sich als Grundbaustein etabliert, um diese Herausforderungen anzugehen und in den Griff zu bekommen. Bei der Verwendung von Containern geht es darum, dass nur die Laufzeitumgebungen, Libraries und Abhängigkeiten, die für die Lauffähigkeit eines Software-Artefakts notwendig sind, in den besagten Container «gepackt» werden und man dadurch eine sehr effiziente und ressourcenschonende Art und Weise bekommt, seinen Code zu bündeln und zum Laufen zu bringen.

Mit diesem Prinzip kann die darunterliegende Infrastruktur und Hardware relativ einfach wegabstrahiert werden. Dies bringt die folgenden essentiellen Vorteile voll zur Geltung:

- Portabilität: Ein Container ist vom darunterliegenden Host getrennt und läuft überall gleich (egal ob auf dem lokalen Entwicklergerät oder in der Cloud)
- Modularität: Container ermöglichen die Aufteilung der Software in austauschbare Komponenten oder Module, die leicht miteinander kommunizieren können. Auch *«separation of concerns»* genannt.
- Sicherheit: Container sind per Definition «immutable» und müssen bei Änderungen komplett ersetzt werden, das erleichtert Updates/Patches und Rollbacks
- Skalierbarkeit: Bei erhöhter Last kann ein Container sehr effizient skaliert werden durch das Starten von mehreren Instanzen.

Hieraus lässt sich erkennen, dass der Einsatz von Containern und deren Vorteil in Bezug auf Skalierung und Portabilität eine der Voraussetzungen dafür ist, komplexe verteilte Applikation effizient über verschiedene Infrastrukturen hinweg (Cloud oder On-Premise) zu betreiben.

Funktionsweise von Containern

Aufgrund der aufgelisteten Vorteile hat sich der Einsatz von Containern heutzutage als überlegener Ansatz für die Entwicklung, Bereitstellung und Verwaltung von Anwendungen herauskristallisiert.
Es darf jedoch nicht verschwiegen werden, dass die Arbeit mit Containern wieder eigene Herausforderungen mit sich bringt. Um diese effizient begegnen zu können, ist zu empfehlen, sich das dahinterliegende Prinzip der Containierisierung genauer anzuschauen.

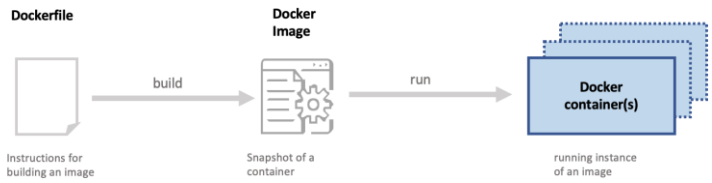

Abbildung 1: Funktionsweise Docker

Herausforderungen von Containern

Auf Abbildung 1 sieht man am Beispiel von Docker, der verbreitetsten Container-Software, dass es im Prinzip darum geht, in einem Dockerfile den Aufbau und Inhalt des Docker Images (dies entspricht einer Datei, die den Code in einem Container ausführt) zu beschreiben, aus dieser Beschreibung können nun Docker Images gebaut werden. Führt man dieses Image aus, wird eine laufende Container-Instanz erstellt. Dieses Image kann nun auf jeder Plattform identisch ausgeführt werden und je nach Bedarf mehrfach instanziiert werden.

In einer Microservice-Architektur können mehrere Services eine Applikation bilden und darstellen, die Services selber haben alle ihren eigenen Container und sind unabhängig voneinander.

Horizontale und vertikale Skalierung

Die Container bieten nun die Möglichkeit je nach Bedarf horizontal oder vertikal zu skalieren.

Bei einer vertikalen Skalierung könnte man dem Container mehr Ressourcen zuteilen, falls die im Container ausgeführte Applikation z.B. mehr CPU oder Arbeitsspeicher benötigt.
Bei der horizontalen Skalierung geht es darum, die Container mehrfach zu instanziieren und dadurch die Leistungsfähigkeit zu steigern.

Bei beiden Formen der Skalierungen stösst man jedoch bei grossen und komplexen Anwendungen oder Applikationslandschaften schnell an eine Grenze der Administration und Verwaltung der Container untereinander.

Container Orchestrierung

Bei einer kleinen Anzahl von Containern ist es sicherlich kein Problem diese händisch zu deployen, zu verwalten und bei Bedarf manuell neue Instanzen zu erzeugen oder mehr Ressourcen zuzuweisen. Es wird aber, wie erwähnt, schnell ein Punkt erreicht, an dem die dynamische Steuerung ohne Automatisierung nicht mehr möglich ist, insbesondere wenn es um die Anbindung an CI/CD Pipelines geht.

Hier kommt die Container Orchestrierung ins Spiel. Bei der Orchestrierung von Containern geht es darum den beschriebenen Prozess des Deployments, der Skalierung, der Netzwerkanbindung und des gesamten Lifecylemanagements des Containers zu automatisieren und über eine Orchestrierungsplattform verfügbar zu machen.

Kubernetes

Was genau eine Orchestrierungsplattform ist, lässt sich am besten am Beispiel Kubernetes beschreiben.

Kubernetes ist ein Open-Source-System zur Automatisierung der Bereitstellung, Skalierung und Verwaltung von Container-Anwendungen, das ursprünglich von Google entworfen und an die Cloud Native Computing Foundation gespendet wurde. Anders formuliert handelt es sich also um eine Orchestrierungslösung, die dem Anwender viele manuelle Schritte erspart, damit containerbasierte Anwendungen schnell ausgerollt und verwaltet werden können, um die beschriebenen Vorteile einer Nutzung von Container mit einer erhöhten Flexibilität zu nutzen.

Funktionsweise Kubernetes

Ein funktionierendes Deployment mit einem oder mehreren Images, das einem konkreten Kontext zugeordnet werden kann, bezeichnet man als Kubernetes Cluster.

Ein Kubernetes-Cluster definiert welche Anwendungen ausgeführt werden sollen, welche Images mit welchen Ressourcen verwendet werden sollen und wie dies alles konfiguriert sein soll.

Wie man in Abbildung 2 sehen kann, besteht ein Kubernetes Cluster aus zwei Teilen:

- Einem Control Plane
- Einem oder mehreren Computing Machines bzw. Nodes

Das Control Plane ist für das Management des Clusters zuständig, hier wird gesteuert welche Anwendungen wie ausgeführt werden. Im Control Plane laufen alle Prozesse für die Steuerung der Nodes (dies entspricht einer Arbeitsmaschine in Kubernetes) zusammen und alle Aufgaben für die Nodes werden hier zugewiesen.

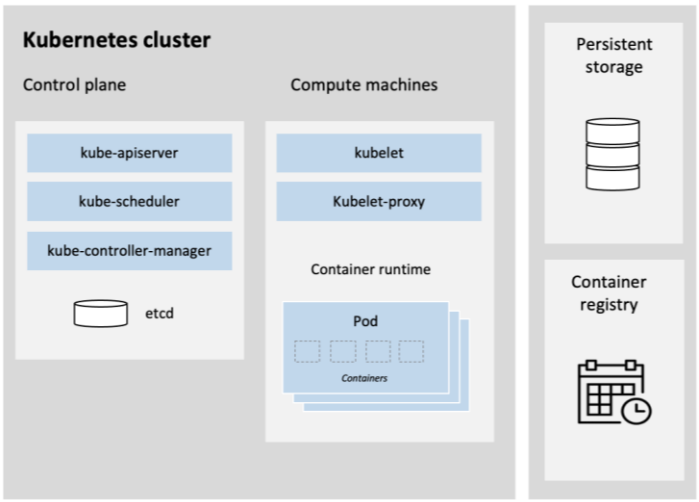

Abbildung 2: Aufbau eines Kubernetes Clusters

Das Control Plane ist ebenfalls dafür zuständig «Befehle» der Administratoren oder DevOps-Engineers entgegenzunehmen und diese Anweisungen an die Nodes weiterzuleiten. Es wird vom Control Plane automatisch entschieden, welche Nodes für die Aufgaben am geeignetsten sind. Anschliessend werden den Nodes und Pods die Ressourcen zugewiesen und mit der Erfüllung der Aufgabe betraut.

Die Nodes führen dann die Anwendungen nur noch aus.
Der Node selber besteht aus einem oder mehreren Pods, ein Pod setzt sich wiederrum aus einem oder mehreren Containern zusammen. Ein Pod ist die kleinste und grundlegendste Einheit in der Kubernetes-Welt.

Das Control Plane ist also für die Aufrechterhaltung des von uns gewünschten Zustands zuständig und die Nodes führen die Anwendungen aus.

Dieser gewünschte Zustand des Kuberntes Clusters und die Arbeit des Control Plane wird über die Konfiguration der Nodes, Pods und Container erreicht und spielt deshalb eine wichtige Rolle bei der Aufsetzung von Kubernetes.

OpenShift

Wie man bereits erkennen kann, bietet eine Kubernetes viele tolle Features für die Arbeit mit Containern an.
Gerade aber im Enterprise-Umfeld hat sich eine Lösung entwickelt die zwar auf Kubernetes aufbaut und als Basis verwendet, die ursprüngliche Idee jedoch um einige weitreichende Features und Services erweitert, um daraus eine vollumfängliche Entwicklungsplattform zu bauen.

Bei dieser Lösung handelt es sich um OpenShift, eine kommerzielle Cloud-Development Plattform der Firma Redhat, die ebenfalls das Ziel hat containerbasierte Anwendungen schnell zu bauen und dynamisch zu skalieren. Redhat hat dies jedoch um Aspekte des IT-Betriebs und der Developer-Experience ergänzt.

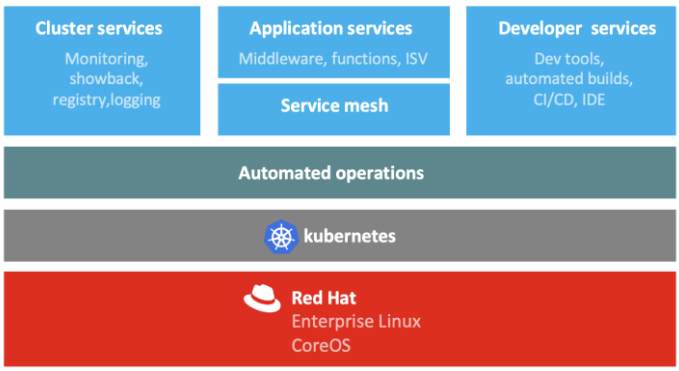

Abbildung 3: Aufbau von OpenShift

Wie man auf Abbildung 3 erkennen kann, nimmt OpenShift Kubernetes als Orchestrierungsengine für seine Plattform, hat jedoch darauf aufbauend einige Services und Erweiterungen hinzugefügt, um eine echte cloud-native Softwareentwicklung zu bieten und einen sicheren IT-Betrieb zu ermöglichen.

Die Verwaltung und die Administration der verschiedenen Cluster ist deutlich vereinfacht worden. Zum einen wäre da die sehr übersichtliche Weboberfläche, welche es erlaubt, annähernd alle administrativen Aufgaben auch ohne eine Kommandozeile ausführen zu können. Zum anderen stellt diese das gesamte System sehr übersichtlich dar. Dies ist bei Kubernetes nicht ohne weiteren Aufwand möglich.

Zusätzlich bietet OpenShift beim Aspekt der Security einige sehr spannende Features.

Während man bei Kubernetes sicherheitsrelevante Themen wie Verschlüsselung, Autorisierung, Authentifizierung u.ä. selber konfigurieren und integrieren muss, sind bei OpenShift all diese Funktionen bereits vorhanden und konfiguriert. Ebenso sind Themengebiete wie z.B. Role-Bindings und Network-Policies als zusätzliche Sicherheitsfunktionen vorhanden.

Ein wichtiger Vorteil für die Developer Experience und die Geschwindigkeit wie Software bereitgestellt werden kann, ist sicherlich die CI/CD Integration, die einen zentralen Baustein von OpenShift darstellt. Es werden beispielsweise fertige CI/CD Pipelines mit Tekton bereitgestellt, durch Jenkins-Images wird eine CI/CD Integration stark vereinfacht oder es wird ein ServiceMesh Layer unterstützt.

Die vielen genannten Features, Standards und Vorinstallationen, welche die gesamte Plattform bilden (siehe Abbildung 3), bieten zusammen mit der webbasierten Benutzeroberfläche einen leichten Einstieg in die Welt für das dynamische Entwickeln und Betreiben von komplexen und verteilten, cloud-nativen Systemen und Applikationen.

Wie kann OpenShift eingesetzt werden?

RedHat bietet sowohl Self-Hosted Systeme zur eigenen Installation auf beliebigen Linux Systemen an, als auch eine cloudbasierte Managed Solution, welche in der RedHat Cloud läuft. Weiter bieten AWS, Azure und IBM eigene Managed Installationen an. Als «OpenShift Dedicated» kann es sogar in der Google Cloud ausgeführt werden.

Die Installation eines OpenShift-Clusters ist dank des Open-Shift Installationstools nicht besonders aufwändig. Es bietet verschiedene Voreinstellungen, um ein OpenShift-Cluster in verschiedenen Cloud Umgebungen (AWS, Azure usw.) zu installieren. Der Bedarf an virtueller Hardware sollte jedoch nicht unterschätzt werden, da ein OpenShift-Cluster mit Hilfe des Installers nicht mit weniger als drei Master und drei Worker-Nodes gestartet werden kann.

Kubernetes	OpenShift
Kubernetes ist ein Open-Source Framework	OpenShift ist ein Produkt
Steuerung über Kommandozeile	Einfache Benutzeroberfläche
Keine Jenkins-Integration	Erleichtertes Einbinden von CI/CD Pipelines dank Jenkins-Integration
Sicherheit abhängig von Nutzer und Konfiguration	Hohe Sicherheitsstandards
Hohe Flexibilität	Viele Default-Einstellungen die den Einstieg erleichtern, die aber auf Kosten der Flexibilität gehen

Abbildung 4: Unterschiede von Kubernetes und OpenShift

Fazit

Im Bereich der Container-Orchestrierung gibt es inzwischen verschiedene Orchestrierungs-Plattformen, die für die unterschiedliche Bedürfnisse von Anwendungen und Systemlandschaften geschaffen worden sind. Neben Kubernetes und OpenShift ist hier noch Docker Swarm oder Apache Mesos zu nennen.

Kubernetes bietet als Open-Source Framwork eine gute Basis für die Orchestrierung von Containern. Gerade für kleinere Projekte, die keine komplexe Konfiguration erfordern, kann es durchaus ausreichen, ein Kubernetes Cluster zu betreiben. Durch die Verfügbarkeit bei allen grossen Providern kann hier auch mit einem kostengünstigen Cluster begonnen werden. Sollte es nötig sein, kann das Cluster auch vergleichsweise einfach zu anderen Providern migriert werden.

Für den Betrieb von grossen Microservice-Systemen im Enterprise-Umfeld hat sich Open-Shift für eine dynamische Container-Orchestrierung und als vollumfängliche Entwicklungsplattform etabliert.

Aufgrund der mitgelieferten Features wie der integrierten CI-/CD Pipeline, der wesentlich übersichtlicheren UI und der besseren Sicherheitsfunktionen kann der Einsatz von OpenShift guten Gewissens empfohlen werden.

Gerne unterstützen wir Sie bei der Transformation in die Welt der containerbasierten Anwendungen und stehen als Gesprächspartner bei allen Fragen zum Thema Orchestrierung und OpenShift für Sie zur Verfügung.

Josip Cosic
Senior Software Engineer

Entago AG
Buckhauserstrasse 34
CH-8048 Zurich

info@entago.ch

www.entago.ch