WHITE PAPER – **AUGUST 2017**

FÖRDERN DER DIGITALEN TRANSFORMATION MIT CONTAINERN UND KUBERNETES

Geschäftlicher Mehrwert durch Management von containerbasierten Anwendungen mit Kubernetes

vmware®

Inhalt

Einführung 3

Die digitale Transformation und der Wechsel zu containerbasierten Anwendungen 3

- Cloudnative Anwendungen und 12-Factor Apps 4
- Methodik zum Bereitstellen von Software as a Service 5

Überblick über Kubernetes 6

- Kubernetes-Objektmodell 7
- Aufrechterhaltung des gewünschten Zustands 7

Geschäftlicher Mehrwert von Kubernetes 8

- Kubernetes für cloudnative Anwendungen und 12-Factor-Anwendungen 8
- Ein Anwendungsbeispiel 10

Container-Technologielösungen von VMware 11

- vSphere Integrated Containers 11
- Wavefront by VMware 13
- Pivotal Container Service 13

Fazit 14

DEFINITION VON DOCKER-CONTAINERN

Mit Containern hat Docker ein Standardformat für die Paketierung und Portierung von Software definiert, vergleichbar mit ISO-Containern, die den Standard bei Frachtcontainern darstellen. Als Runtime-Instanz eines Docker-Images besteht ein Container aus drei Teilen:

- Einem Docker-Image
- Einer Umgebung, in der das Image ausgeführt wird
- Einem Satz von Anweisungen zur Ausführung des Images

– Entnommen aus dem Docker-Glossar

Einführung

Kubernetes verwaltet Container. Mit Containern werden Anwendungen und ihre Abhängigkeiten in ein verteilbares Image verpackt, das nahezu überall ausgeführt werden kann. Software kann so rationeller entwickelt und bereitstellt werden. Der Einsatz von Containern ist ein wichtiger Schritt auf dem Weg zu einem agilen digitalen Unternehmen und einer beschleunigten Bereitstellung innovativer Produkte, Services und Kundenerlebnisse. So können Unternehmen ihre Wettbewerbsfähigkeit erhöhen und sich eine Spitzenposition am Markt sichern.

Doch Container sind häufig mit Problemen beim Technologiemanagement verbunden, vor allem wenn containerbasierte Anwendungen skalierbar bereitgestellt und verwaltet werden müssen. Hier kommt Kubernetes ins Spiel. Kubernetes orchestriert containerbasierte Anwendungen und ermöglicht das Verwalten und Automatisieren von Ressourcenauslastung, Fehlerbehebung, Verfügbarkeit, Konfiguration, Skalierbarkeit und gewünschtem Zustand.

Dieses White Paper beschreibt die Lösung Kubernetes und erläutert, welchen geschäftlichen Nutzen sie hat, welche Anwendungsbereiche es gibt und wie sie die digitale Transformation Ihres Unternehmens beschleunigen kann.

Die digitale Transformation und der Wechsel zu containerbasierten Anwendungen

Gemäß der „New York Times“ beschleunigt sich die technologische Innovation immer weiter.[1] Aus diesem Grund haben viele Unternehmen sich die digitale Transformation zum Ziel gesetzt und führen zunehmend digitale Initiativen ein.[2]

Die Gründe für die in vielen Unternehmen stattfindende digitale Transformation sind klar:

- Erstellen neuer Anwendungen, die Kunden auf innovative, fesselnde Weise ansprechen
- Verbessern der Betriebsabläufe, um auf kostengünstigere, effizientere Weise noch bessere Produkte und Services bereitzustellen
- Erschließen neuer Umsatzquellen durch eine schnelle Anpassung an neue Marktbedingungen und Verbraucherpräferenzen

Laut Gartner Research „gehört die Zukunft den Unternehmen, die die effektivsten, intelligentesten und eigenständigsten Softwarelösungen erstellen können.“[3] Die Komponenten zum Erstellen effektiver, eigenständiger Anwendungen sind jedoch weniger klar als die gewünschten Ergebnisse.

Um in der heutigen Zeit effektiv zu sein, benötigen Anwendungen eine Architektur, die eine flüssige, schnelle und flexible Entwicklung und Bereitstellung ermöglicht. Gleichzeitig müssen diese Anwendungen die Sicherheit, Performance und Kosteneffizienz herkömmlicher Anwendungen bieten. Container bilden die Basis für eine neue Anwendungsarchitektur, die die digitale Transformation unterstützt und den Grundstein für Innovation legt.

1 „Digital Transformation Going Mainstream in 2016, IDC Predicts“, Steve Lohr, The New York Times, 4. November 2015

2 „New Research Finds Investment from Outside IT Is Key to Digital Transformation Success“, von Business Wire, The New York Times, 11. Mai 2017

3 „Digital Transformation Going Mainstream in 2016, IDC Predicts“, Steve Lohr, The New York Times, 4. November 2015

Unternehmen setzen zunehmend Container-Technologie ein. Eine aktuelle 451 Research-Studie offenbarte eine deutlich verstärkte Implementierung in einem wachsenden Segment.[4] Unternehmen, die Container einführen, sehen diese als schnelle Lösung zum Erstellen und Bereitstellen von cloudnativen Anwendungen sowie 12-Factor Apps.

Cloudnative Anwendungen und 12-Factor Apps

Die Cloud Native Computing Foundation, ein Projekt der Linux Foundation, definiert cloudnative Anwendungen wie folgt:[5]

1. **Containerbasiert** – Alle Elemente (Anwendungen, Prozesse usw.) werden in einen eigenen Container gepackt. Dies vereinfacht die Reproduzierbarkeit und Ressourcenisolation und verbessert die Transparenz.
2. **Dynamisch orchestriert** – Container werden aktiv geplant und verwaltet, um die Ressourcenauslastung zu optimieren.
3. **Mikroserviceorientiert** – Anwendungen werden in Mikroservices segmentiert. Durch diese Segmentierung wird die Agilität und Wartungsfreundlichkeit von Anwendungen deutlich verbessert.

Kubernetes deckt durch das Planen und Verwalten von Containern den zweiten Teil der Definition ab. Im Hinblick auf den dritten Teil unterstützen sowohl Kubernetes als auch Docker die Implementierung von Mikroservices.

Das wichtigste Element ist jedoch der Container: ein Prozess, der mit eigener Anwendung bzw. eigenem Dateisystem und Netzwerk auf einem Computer oder einer virtuellen Maschine ausgeführt wird. Ein Container verpackt eine Anwendung auf reproduzierbare Weise: Sie kann mit minimalem Aufwand verteilt und wiederverwendet werden.

Docker-Container sind die am häufigsten bereitgestellten Container. Ein Manifest, genannt Dockerfile, beschreibt, wie das Image und seine Bestandteile in einem Container auf einem Host auszuführen sind. Um die Beziehung zwischen dem Dockerfile und dem Image zu verdeutlichen, finden Sie im Folgenden ein Beispiel für ein Dockerfile, das MongoDB auf einer in einem Container ausgeführten Ubuntu-Maschine installiert. Die mit einem Rautezeichen beginnenden Zeilen sind Kommentare, die die nachfolgenden Befehle beschreiben.

```
# MongoDB Dockerfile from https://github.com/dockerfile/mongodb
# Pull base image.
FROM dockerfile/ubuntu
# Install MongoDB.
RUN \
  apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv
7F0CEB10 && \
  echo 'deb http://downloads-distro.mongodb.org/repo/ubuntu-
upstart dist 10gen' > /etc/apt/sources.list.d/mongodb.list && \
  apt-get update && \
  apt-get install -y mongodb-org && \
  rm -rf /var/lib/apt/lists/*
# Define mountable directories.
```

4 „Application containers will be a $2.7bn market by 2020, representing a small but high-growth segment of the CloudEnabling Technologies market", 451 Research, 10. Januar 2017

5 Definition entnommen aus den FAQ der Cloud Native Computing Foundation, https://www.cncf.io/about/faq/.

```
VOLUME ["/data/db"]
# Define working directory.

WORKDIR /data
# Define default command.
CMD ["mongod"]
# Expose port 27017 for the process and port 28017 for http
EXPOSE 27017
EXPOSE 28017
```

Methodik zum Bereitstellen von Software as a Service

Im Gegensatz dazu wird die 12-Factor App ebenso durch ihre Prozesse wie auch durch ihre Systemeigenschaften definiert. Bei der 12-Factor App handelt es sich um eine Methodik, mit der eine Software as a Service (SaaS)-Anwendung, d.h. eine Webanwendung, entwickelt und in der Regel auf einer Platform as a Service (PaaS) wie Pivotal Cloud Foundry bereitgestellt wird. Im Folgenden werden die 12 Faktoren kurz erläutert:[6]

1. **Beliebig häufiges Bereitstellen der Anwendung anhand einer einzigen Codebasis:** Die Codebasis wird in einem Repository gespeichert, bei einer Änderung mit einem Versionskontrollsystem wie Git verwaltet und dann viele Male als laufende Instanz der Anwendung über dieselbe Codebasis bereitgestellt. Infolgedessen wird eine Bereitstellung häufig in drei Umgebungen ausgeführt: in der lokalen Umgebung des Entwicklers, in einer Staging-Umgebung und in einer Produktionsumgebung.
2. **Deklarieren und Isolieren von Abhängigkeiten:** Die Anwendung ist nicht unbedingt auf systemweite Pakete angewiesen. Stattdessen deklariert sie die Abhängigkeiten in einem Deklarationsmanifest. Das explizite Deklarieren von Abhängigkeiten macht es für neue Entwickler einfacher, ihre Entwicklungsumgebung einzurichten.
3. **Speichern der Konfiguration in der Umgebung, nicht im Code:** Die Konfigurationsinformationen, die je nach Bereitstellung variieren, werden von der Anwendung in Umgebungsvariablen gespeichert. Die Umgebungsvariablen sind detaillierte Kontrollen, die unabhängig für jede Bereitstellung verwaltet werden. So kann die Anwendung im Lauf der Zeit mühelos in weitere Bereitstellungen skaliert werden.
4. **Verbinden mit unterstützenden Services**, z.B. einer Datenbank oder einem Storage-System, anstatt sie in den Code einzufügen. Die Anwendung behandelt solche Services als Ressourcen, die durch Änderung der Konfiguration an eine Bereitstellung angehängt oder wieder entfernt werden können.
5. **Behandeln von Build und Run als getrennte Phasen:** Die Bereitstellung der Codebasis findet in drei getrennten Phasen statt: Build, Release und Runtime. In der Build-Phase wird die Codebasis in eine ausführbare Datei (ein Build) konvertiert. In der Release-Phase wird das Build mit der Konfiguration kombiniert, um ein Release zu erzeugen, das sofort in der Runtime-Umgebung ausgeführt werden kann.
6. **Ausführen der Anwendung als zustandsfreie Prozesse:** Die Prozesse teilen keine Komponenten mit anderen Prozessen. Die Daten, die beibehalten werden müssen, werden in einer Datenbank gespeichert, die als zustandsorientierter unterstützender Service ausgeführt wird.

6 Die Beschreibung der 12 Faktoren basiert auf den Definitionen auf der Twelve-Factor App-Website.

VERWALTEN VON CONTAINERBASIERTEN ANWENDUNGEN MIT KUBERNETES

Kubernetes orchestriert verteilte, containerbasierte Anwendungen, um

- die Auslastung von Computing-Ressourcen zu optimieren,
- Richtlinien für die Planung bereitzustellen,
- den gewünschten Zustand aufrechtzuerhalten,
- Fehler und Ausfälle durch Automatisierung zu beheben,
- Hochverfügbarkeit zu ermöglichen,
- Jobs in Echtzeit zu überwachen,
- die Konfiguration einer Anwendung zu verwalten,
- eine dynamische Skalierung zur Erfüllung wechselnder Anforderungen zu ermöglichen.

7. **Bereitstellen von Services mithilfe von Port-Bindung:** Nimmt man HTTP als Beispiel, so exportiert die Anwendung HTTP als Service, indem sie sich an einen Port bindet und diesen auf eingehende Anfragen überwacht.
8. **Horizontales Skalieren durch Hinzufügen paralleler Prozesse:** Die Anwendung verarbeitet Workloads, indem sie jede Art von Arbeit einem Prozesstyp zuweist. Ein Webprozess beispielsweise verarbeitet HTTP-Anfragen, während ein Worker-Prozess Hintergrundaufgaben verwaltet.
9. **Sicherstellen von Robustheit durch Verfügbarkeit:** Prozesse sind verfügbar – sie können schnell gestartet oder beendet werden, sodass die Anwendung auf einfache Weise geändert oder skaliert werden kann.
10. **Angleichen von Entwicklung und Produktion:** Die Anwendung ist auf eine kontinuierliche Bereitstellung hin ausgerichtet. So können Entwickler neuen Code schnell integrieren und die Anwendung selbst in einer Produktionsumgebung bereitstellen. Die Produktions- und Entwicklungsumgebung sollten möglichst ähnlich sein.
11. **Verarbeiten von Protokollen als Ereignisströme:** Die Anwendung routet den Ausgabestrom ihrer Protokolle nicht und speichert ihn auch nicht. Stattdessen schreibt sie ihn als Datenstrom in eine Standardausgabe. Hier wird er von der Ausführungsumgebung erfasst und zur Speicherung oder Analyse an ein Tool oder System wie z.B. Hadoop weitergeleitet.
12. **Ausführen einmaliger Managementskripte und -aufgaben**, wie z.B. eine Datenmigration, in einer Umgebung, die mit der Umgebung für langfristige Prozesse der Anwendung identisch ist.

Mit Containern und Kubernetes lassen sich einige Aspekte dieser Anforderungen erfüllen. Container beispielsweise spielen eine wichtige Rolle im Zusammenhang mit 12-Factor Apps, weil sie das Deklarieren und Isolieren von Abhängigkeiten ermöglichen. Container sorgen zudem durch Verfügbarkeit für eine gewisse Robustheit, da sie u.a. schnell gestartet und fehlerfrei beendet werden können. Viele der übrigen Faktoren werden von Kubernetes unterstützt.

Überblick über Kubernetes

Kubernetes wurde ursprünglich von Google entwickelt. Das Unternehmen nutzt seinen Vorgänger namens Borg, um öffentliche Anwendungen wie Gmail und Google Docs sowie interne Frameworks wie MapReduce zu initiieren, zu planen, neu zu starten und zu überwachen.[7] Kubernetes basiert auf dem ursprünglichen System von Google sowie Erweiterungen aus den Erfahrungen mit Borg. Es ist ein Open Source-Orchestrierungssystem für Container, die cloudübergreifend in Ihrem Rechenzentrum sowie in hybriden Rechenzentren eingesetzt werden können. Kubernetes kann automatisch Workloads bereitstellen, Anwendungen neu starten und bei steigendem Bedarf Ressourcen hinzufügen.

Im Folgenden erhalten Sie eine kurze Übersicht über die Funktionsweise. Ein Kubernetes-Cluster enthält einen Master-Knoten und mehrere Worker-Knoten. Wenn Sie eine Anwendung im Cluster bereitstellen, werden die Komponenten der Anwendungen auf den Worker-Knoten ausgeführt. Der Master-Knoten verwaltet die Bereitstellung.

Kubernetes umfasst folgende Komponenten:

- Die Kubernetes-API
- Die Kubernetes-Befehlszeilenschnittstelle „kubectl"
- Die Kubernetes-Steuerungsebene

7 Umfassendes Cluster-Management bei Google mit Borg, Research at Google, 2015

VORTEILE DER VERWENDUNG VON KUBERNETES

- Serverkonsolidierung und Kostensenkung durch effiziente Ressourcennutzung
- „Elegante“ Handhabung von Maschinenausfällen durch Selbstreparaturfunktionen und Hochverfügbarkeit
- Einfache und schnelle Bereitstellung, Protokollierung und Überwachung von Anwendungen
- Automatisierte Skalierbarkeit für Container und containerbasierte Anwendungen
- Entkoppeln der Anwendungen von den Maschinen für eine erhöhte Portabilität und Flexibilität
- Einfaches Ändern, Aktualisieren, Erweitern oder erneutes Bereitstellen von Anwendungen ohne Beeinträchtigung anderer Workloads

Die Steuerungsebene enthält die Prozesse, die auf dem Kubernetes-Master und den einzelnen Worker-Knoten ausgeführt werden. Auf dem Master beispielweise führt Kubernetes mehrere Prozesse aus: den API-Server, den Controller, den Scheduler und etcd. Die Worker-Knoten führen den „kubelet”-Prozess für die Kommunikation mit dem Master und den Proxy-Prozess für das Management des Netzwerks aus.

Kubernetes-Objektmodell

Ein Schlüsselelement des Kubernetes-Systems ist die Darstellung des Zustands der containerbasierten Anwendungen und Workloads, die bereitgestellt wurden. Kubernetes stellt den Zustand mithilfe von „Objekten“ wie z.B. Service, Namespace und Volume dar. Diese Objekte werden üblicherweise durch eine Objektspezifikation („spec“) festgelegt, die Sie für den Cluster erstellen.

Im Rahmen des Kubernetes-Objektmodells ist ein Pod der kleinste bereitstellbare Baustein. Ein Pod stellt eine Instanz einer Anwendung dar, die als Prozess in einem Kubernetes-Cluster ausgeführt wird. Als Runtime in einem Kubernetes-Pod wird dabei häufig Docker verwendet.

Kubernetes umfasst zudem Controller, die einen Großteil der Logik in Kubernetes umsetzen. Die Controller bieten Funktionen wie Replica Set und Stateful Set.

Aufrechterhaltung des gewünschten Zustands

Die Kubernetes-Steuerungsebene verwaltet den Zustand all dieser Objekte und stellt sicher, dass sie dem von Ihnen gewünschten Zustand entsprechen. Sie können einen gewünschten Zustand festlegen, indem Sie mit einer YAML-Datei eine Objektspezifikation für einen Service erstellen. Hier sehen Sie ein Beispiel:

```
apiVersion: v1
kind: Service
metadata:
  name: nginx-demo-service
  labels:
    app: nginx-demo
spec:
  type: NodePort
  ports:
  - port: 80
    protocol: TCP
    name: http
  selector:
    app: nginx-demo
---
apiVersion: v1
kind: ReplicationController
metadata:
  name: nginx-demo
spec:
  replicas: 3
```

```
template:
metadata:
 labels:
  app: nginx-demo
 spec:
  containers:
  - name: nginx-demo
    image: myrepo/nginx
  ports:
  - containerPort: 80
```

Wenn Sie diese Datei über die kubectl-Befehlszeilenschnittstelle an den Kubernetes-Master senden, setzt die Kubernetes-Steuerungsebene die Anweisungen in der Datei um. Dazu startet und plant sie Anwendungen, sodass der Zustand des Clusters Ihrem gewünschten Zustand entspricht. Der Kubernetes-Master und die Steuerungsebene halten dann durch Orchestrierung der Cluster-Knoten (physische Server oder virtuelle Maschinen) den gewünschten Zustand aufrecht.

Kern der Architektur ist ein API-Server, der den Zustand der Systemobjekte verwaltet. Der API-Server nutzt Kubernetes-Subkomponenten bzw. Clients, die als zusammensetzbare Mikroservices konzipiert sind, z.B. den in der YAML-Datei angegebenen Replikations-Controller. Der Replikations-Controller reguliert den gewünschten Zustand von Pod-Replikaten, wenn Fehler auftreten.

Geschäftlicher Mehrwert von Kubernetes

Kommen wir zurück zur digitalen Transformation: Kubernetes nutzt diese Architektur, um containerbasierte Anwendungen in einem verteilten Cluster zu verwalten. Die Ergebnisse tragen dazu bei, die geschäftlichen Vorteile der digitalen Transformation zu realisieren:

- Kubernetes macht das Ausführen von Anwendungen in Public, Private oder Hybrid Clouds einfacher und kostengünstiger.
- Kubernetes beschleunigt die Anwendungsentwicklung und -bereitstellung.
- Kubernetes erhöht die Agilität und Flexibilität und verbessert die Anpassungsfähigkeit.

Kubernetes für cloudnative Anwendungen und 12-Factor-Anwendungen

Durch Kubernetes wird die Ausführung von containerbasierten Anwendungen überschaubar und skalierbar. Erinnern wir uns an den zweiten Teil der Definition von cloudnativen Anwendungen: Sie werden dynamisch orchestriert, d.h., Container werden aktiv geplant und verwaltet, um die Ressourcenauslastung zu optimieren. Genau das ermöglicht Kubernetes. Kubernetes orchestriert Container und ihre Workloads, um die Auslastung der virtuellen Maschinen und physischen Server, welche die Knoten in einem Cluster bilden, zu optimieren.

Mit Blick auf die 12 Faktoren wird noch deutlicher, wie Kubernetes das Anwendungsmanagement optimiert. In der Regel kann Kubernetes 12-Factor Apps bereitstellen und ausführen.

SO OPTIMIEREN KUBERNETES UND CONTAINER DAS ANWENDUNGSMANAGEMENT	
Faktor	**Vorteil**
1. Beliebig häufiges Bereitstellen der Anwendung anhand einer einzigen Codebasis	Kubernetes kann Anwendungen mit nur einer Codebasis viele Male bereitstellen. Dazu wird einem Pod eine Spezifikation zugewiesen, die eine Container-Image-Referenz enthält.
2. Deklarieren und Isolieren von Abhängigkeiten	Container können Abhängigkeiten ausdrücken.
3. Speichern der Konfiguration in der Umgebung, nicht im Code	Sie können bestimmte Aspekte der Konfiguration einer Anwendung in der Kubernetes-Umgebung speichern. Das ConfigMaps-Konstrukt z.B. trennt Konfigurationsartefakte von den Anweisungen eines Images.
4. Verbinden mit unterstützenden Services, z.B. einer Datenbank, anstatt sie in den Code einzufügen	Kubernetes ermöglicht Ihnen das Bereitstellen von unterstützenden Services, z.B. einer Datenbank, in separaten Containern und verwaltet dann alle containerbasierten Komponenten zusammen, um Verfügbarkeit und Performance zu verbessern.
5. Behandeln von Build und Run als getrennte Phasen	Sie können die Anwendung beispielsweise mit Jenkins (einem gesonderten Pipeline-Automatisierungsserver) erstellen und dann mithilfe von Kubernetes die Docker-Images ausführen.
6. Ausführen der Anwendung als zustandsfreie Prozesse	Kubernetes vereinfacht das Ausführen von zustandsfreien Anwendungen. Kubernetes ermöglicht es beispielsweise, Zustände unabhängig voneinander in einem etcd-Datenspeicher zu verwalten, während die Anwendung ausgeführt wird. Darüber hinaus ermöglicht Kubernetes Ihnen, persistenten Storage anzuhängen. So kann z.B. für die spec-Datei, die einen Pod definiert, ein persistentes Volume erforderlich sein. Wenn der Pod ausfällt, stellt der Ersatz-Pod eine Verbindung zum selben persistenten Volume her.
7. Bereitstellen von Services mithilfe von Port-Bindung	Kubernetes umfasst Konfigurationsoptionen, um Services auf Ports verfügbar zu machen. In der oben dargestellten beispielhaften nginx-YAML-Datei wurde der nginx-Webserver an Port 80 gebunden und als Service zur Verfügung gestellt.
8. Horizontales Skalieren durch Hinzufügen paralleler Prozesse	Kubernetes skaliert eine Anwendung durch Hinzufügen weiterer Pods. Mithilfe des Replikations-Controllers kann Kubernetes z.B. mehrere Pods gleichzeitig hinzufügen.
9. Sicherstellen von Robustheit durch Verfügbarkeit	In Kubernetes ausgeführte Container werden als unveränderbar angesehen – sie müssen bedarfsorientiert oder basierend auf einem Zeitplan beendet und ersetzt werden.
10. Angleichen von Entwicklung und Produktion	In der Kubernetes-Umgebung kann Entwicklungs- und Produktionscode auf die gleiche Weise umfassend getestet werden. Sie können z.B. eine Kubernetes-Bereitstellung mit zwei Pods verwenden, einem Pod, der die Produktionsumgebung enthält, und einem anderen Pod, der die Staging-Umgebung enthält. So werden Staging und Produktion zu gleichen Umgebungen. Darüber hinaus ist die in einem Container spezifizierte Umgebung in allen Entwicklungs- und Produktionsumgebungen einheitlich.
11. Verarbeiten von Protokollen als Ereignisströme	Kubernetes ermöglicht Ihnen, auf die Standardausgabe eines Containers zuzugreifen und die Ausgabe als Datenstrom mit einem Tool Ihrer Wahl, z.B. VMware vRealize® Log Insight™, zu verarbeiten.
12. Ausführen von Managementaufgaben als einmalige Prozesse	Sie können einen Pod, der aus dem Anwendungs-Container besteht, mithilfe eines anderen Einstiegspunkts so planen, dass er einen anderen Prozess ausführt. Dies kann z.B. ein Skript zum Migrieren einer Datenbank sein.

VORTEILE FÜR ENTWICKLER

Der geschäftliche Nutzen von Containern und Kubernetes ist nicht auf das Unternehmen als Ganzes oder das Büro des CIO beschränkt. Auch Entwickler bevorzugen Container, weil sie das Entwickeln von Anwendungen einfacher, interessanter und produktiver machen.

- **Portabilität:** Mit Containern können Entwickler wählen, wie und wo Anwendungen bereitgestellt werden.
- **Geschwindigkeit:** Container beschleunigen Workflows (z.B. Tests) und ermöglichen schnellere Iterationen.
- **CI-/CD-Pipeline:** Kubernetes und Container unterstützen kontinuierliche Integration und kontinuierliche Bereitstellung.
- **Flexibilität:** Entwickler können überall und zu jeder Zeit mit den Tools ihrer Wahl Code auf ihren Laptops erstellen.
- **Der 13. Faktor:** Container und Kubernetes gelten als moderne Technologien. Deshalb sind Entwickler hoch motiviert, diese zu verwenden.

Ein Anwendungsbeispiel

Die nachfolgende kurze Fallstudie veranschaulicht ein allgemeines Anwendungsbeispiel für das Verwalten von Containern mit Kubernetes.

Ein Taxiunternehmen in einem großen Ballungsgebiet verliert Fahrgäste an Car-Sharing-Services, was den ehemals starken lokalen Marktanteil gefährdet. Das Unternehmen muss sich digitalisieren, um mit Car-Sharing-Unternehmen konkurrieren zu können. Zu diesem Zweck möchte das Unternehmen eine eigene mobile App entwickeln, diese kostengünstig im eigenen, bescheidenen Rechenzentrum ausführen und versuchen, innovative Services bereitzustellen.

Was dem Taxiunternehmen zugute kommt, ist seine etablierte lokale Marke; zudem ist es bekannt für pünktliche, höfliche und sichere Fahrer.

Während neu eingestellte Entwickler an der mobilen App arbeiten, modernisiert das Taxiunternehmen sein Rechenzentrum mit Standardhardware und Virtualisierung. Zur Maximierung der Ressourcenauslastung in seinem kleinen Rechenzentrum und zur Kostenminimierung plant das Unternehmen, seine neue Anwendung in Docker-Containern auf virtuellen Maschinen auszuführen. Mit Kubernetes soll die containerbasierte Anwendung orchestriert werden.

Nach ihrer Einführung und ihrer Bewerbung in und auf den Taxis wird die Anwendung zu einem vollen Erfolg. Das Unternehmen nutzt Kubernetes, um Schwankungen in der Nutzung der Anwendung aufzufangen und die Anzahl der Container, in denen die Anwendung ausgeführt wird, dynamisch zu skalieren. Wenn die Kennzahlen für die Anwendung beispielsweise den vordefinierten Schwellenwert für hohe Auslastung erreichen, was üblicherweise in der Rushhour der Fall ist, maximiert das DevOps-Team des Unternehmens mithilfe der Horizontal Pod Autoscaling-Funktion von Kubernetes automatisch die Anzahl der Container, sodass das System den Bedarf erfüllen kann. Um 4:00 Uhr nachts dagegen wird die Anzahl der Container reduziert, um die geringe Nachfrage zu diesem Zeitpunkt elastisch zu erfüllen und Ressourcen zu schonen.

Die mobile App korreliert die Fahrtwünsche mit dem jeweiligen Standort. Durch Analyse der Daten und Kombination mit seinen detaillierten Kenntnissen der Struktur der Stadt ist das Unternehmen in der Lage, an den besten Plätzen Taxis für winkende Kunden zu positionieren und so Fahrzeuganfragen an die Konkurrenz zuvorzukommen. Und weil das Unternehmen die Protokolle der Anwendung als Ereignisströme verarbeitet, können die Taxis darüber hinaus zu jeder Tag- und Nachtzeit an den jeweiligen Hotspots stationiert werden.

Da das Unternehmen die Anwendung mithilfe von Containern implementiert hat, können Entwickler täglich neue Änderungen umsetzen. Die von der Anwendung erfassten Daten helfen dem Unternehmen dabei, neue Funktionen zu ermitteln und schnell Innovationen durchführen. So kann das Unternehmen sich auf seine Stärken konzentrieren und z.B. Stammkunden identifizieren und Bonusprogramme einführen, um diese Kunden zu halten.

Die geschäftlichen Vorteile durch die technische Agilität, die containerbasierte Anwendung und die Kubernetes-Orchestrierung bescheren dem Unternehmen einen klaren Wettbewerbsvorteil:

- Die Planungsrichtlinien in Kubernetes bieten dem Unternehmen die nötige Elastizität, um den Bedarf dynamisch und kostengünstig zu erfüllen, und zwar mit seinem kleinen, aber jetzt modernisierten Rechenzentrum.
- Kubernetes kümmert sich automatisch um Fehler und Ausfälle, sodass der Fehlerbehebungsaufwand für das kleine DevOps-Team reduziert wird.
- Durch die Möglichkeit zur nahtlosen Änderung der Anwendung und ihrer Funktionen kann das Unternehmen sich gegen größere, weniger lokale Konkurrenten durchsetzen, da es agiler ist und seine Kenntnisse der örtlichen Gegebenheiten besser anwenden kann.

- Durch Container und Kubernetes wird die Ausführung der Anwendung einfacher und kostengünstiger.
- Das DevOps-Team kann Container mühelos aus der Test- in die Produktionsumgebung portieren, wodurch die Entwicklung und Bereitstellung neuer Funktionen beschleunigt wird.

Container-Technologielösungen von VMware

In einem kürzlich Report mit dem Titel „Closing the Digital Transformation Confidence Gap in 2017“ untersuchte The Hackett Group die Aussagen von Führungskräften aus mehr als 180 Großunternehmen. Der Report ließ eine große Lücke „zwischen den hohen Erwartungen an den geschäftlichen Nutzen der digitalen Transformation und dem geringen Vertrauen in die Durchführbarkeit einer solcher Transformation im Unternehmen” erkennen. Der Hackett Group zufolge verdeutlichen die Ergebnisse die Notwendigkeit für die IT, in die erforderlichen Tools zu investieren und Techniken zur schnellen Anwendungsentwicklung einzuführen, z.B. agile Prozesse.[8]

Cloudnative Lösungen von VMware helfen Ihnen dabei, Container schnell und kostengünstig in die Produktion zu überführen und damit Ihre Fähigkeit zur Umsetzung einer digitalen Transformation zu verbessern.

Durch das Ausführen von Containern auf VMs wird auch die Sicherheit für containerbasierte Anwendungen erhöht, insbesondere im Kontext des dritten Grundsatzes cloudnativer Anwendungen – Mikroservices. Laut einem White Paper von Docker zum Thema Sicherheit „ermöglicht das Bereitstellen von Docker-Containern in Kombination mit VMs, eine gesamte Gruppe von Services voneinander zu isolieren und dann innerhalb eines VM-Hosts zu gruppieren”.[9]

Durch das Bereitstellen von Containern mit VMs wird eine Anwendung von zwei Isolationsschichten eingeschlossen. Dieser Ansatz eignet sich hervorragend für Cloud-Umgebungen mit Mandantenfähigkeit und mehreren Workloads. “Docker-Container lassen sich gut zusammen mit Virtualisierungstechnologien einsetzen, denn sie schützen die virtuelle Maschine selbst und bieten darüber hinaus einen wirksamen Schutzmechanismus für den Host”, heißt es im Sicherheits-White Paper von Docker.

vSphere Integrated Containers

VMware vSphere Integrated Containers ist eine umfassende Container-Lösung, die auf VMware vSphere basiert. Sie führt moderne und herkömmliche Workloads gleichzeitig Ihrem Software-Defined Datacenter von VMware aus und bietet Netzwerke, Storage, Sicherheit, Performance und Transparenz der Enterprise-Klasse.

Durch Unterstützung für Docker-Container ermöglicht vSphere Integrated Containers Ihnen die sofortige Nutzung von Container-Technologie, um die Produktivität von Entwicklern zu erhöhen und die geschäftliche Agilität zu verbessern. Die Lösung hilft Ihnen dabei, Ihre Organisation in ein digitales Unternehmen zu transformieren und Ihr Rechenzentrum durch Bereitstellen von containerbasierten Anwendungen zu modernisieren.

8 Despite High Expectations for Digital Transformation Led by Cloud, Analytics, Robotic Process Automation, Cognitive & Mobile, IT & Other Business Services Areas See Low Capability to Execute, The Hackett Group, 16. März 2017 Nach Registrierung können Sie die Studie hier herunterladen: http://www.thehackettgroup.com/research/2017/social-media/key17it/.

9 Introduction to Container Security, Docker-White Paper, Docker.com

Übersicht über die Architektur

Die folgende Abbildung veranschaulicht die allgemeine Architektur von vSphere Integrated Containers.

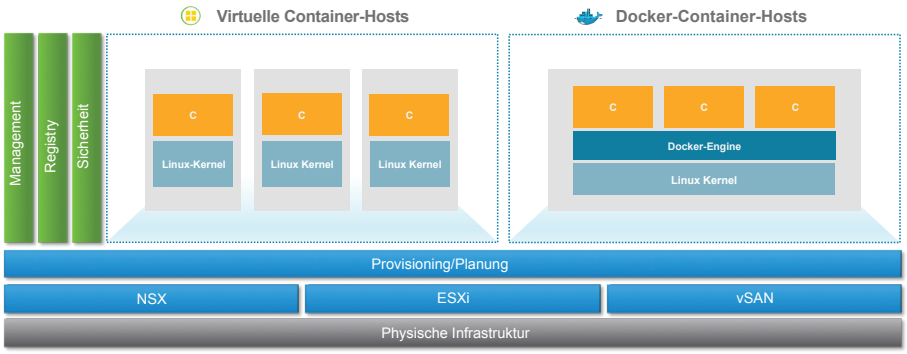

Abbildung 1: Allgemeine Architektur von vSphere Integrated Containers

Diese Architektur bietet Ihnen zwei Container-Bereitstellungsmodelle:

- **Virtuelle Container-Hosts:** vSphere Integrated Containers nutzt die nativen Konstrukte von vSphere für das Provisioning von Containern. Indem jedes Container-Image als virtuelle Maschine bereitgestellt wird, erweitert vSphere Integrated Containers die Verfügbarkeits- und Performance-Funktionen von vSphere auf containerbasierte Workloads, einschließlich VMware HA, vMotion und Distributed Resource Scheduler. Darüber hinaus können Entwickler eine Docker-API nutzen.
- **Docker-Container-Hosts:** Entwickler können native Docker-Container-Hosts bedarfsorientiert selbst bereitstellen und sie auf vSphere ausführen. In der ticketlosen Umgebung, die vSphere Integrated Containers bietet, können Entwickler Docker-Tools nutzen, während Governance und Kontrolle der Infrastruktur in den Händen von IT-Teams bleiben.

Komponenten

Alle Komponenten von vSphere Integrated Containers sind Open Source-Projekte:

- **vSphere Integrated Containers Engine:** Die Engine bietet eine Container-Runtime für vSphere und ermöglicht Softwareingenieuren, Anwendungen in Containern zu entwickeln und die containerbasierten Anwendungen zusammen mit herkömmlichen VM-basierten Workloads auf vSphere-Clustern bereitzustellen.
- **Harbor:** Harbor ist eine private Container-Registry der Enterprise-Klasse, in der Container-Images gespeichert und verteilt werden. Harbor erweitert die Open Source-Distribution von Docker mit Enterprise-Funktionen wie Identitätsmanagement, rollenbasierte Zugriffssteuerung und Auditing.
- **Admiral:** Admiral ist ein Portal für das Container-Management und stellt eine Benutzerschnittstelle bereit, über die DevOps-Teams und andere Anwender Container bereitstellen und verwalten können. Admiral kann beispielsweise Kennzahlen zu Container-Instanzen anzeigen. Cloud-Administratoren können Container-Hosts verwalten und Governance-Regeln auf die Nutzung dieser Hosts anwenden, einschließlich Kapazitätsquoten.

VORTEILE VON MIKROSERVICES

In Kombination mit Containern sind Mikroservices immer häufiger bevorzugtes Architekturmuster für das Entwickeln neuer Anwendungen. Die Architektur zerlegt die Funktionen einer Anwendung in eine Reihe kleiner, eigenständiger, dezentralisierter, zielorientierter Prozesse. Jeder dieser Prozesse kann unabhängig entwickelt, getestet, bereitgestellt, ersetzt und skaliert werden.

- Erhöhte Modularität
- Einfacheres Entwickeln und Testen von Anwendungen
- Parallele Entwicklung: Ein Team kann einen Service unabhängig von anderen Teams, die an anderen Services arbeiten, entwickeln und bereitstellen.
- Unterstützung von kontinuierlichem Code-Refactoring, um die Vorteile von Mikroservices im Lauf der Zeit noch weiter zu steigern
- Förderung eines Modells der kontinuierlichen Integration und kontinuierlichen Bereitstellung
- Verbesserte Skalierbarkeit
- Vereinfachte von Komponenten-Upgrades

Merkmale und Funktionen

vSphere Integrated Containers beinhaltet ein einheitliches Managementportal, das zum sicheren Bereitstellen von Containern in das Identitätsmanagement integriert ist. Entwickler und DevOps können ihre jeweiligen Anforderungen erfüllen, indem sie bedarfsorientiert Docker-Container-Hosts erstellen.

Das Ergebnis ermöglicht Anwendungsentwicklungsteams, containerbasierte Anwendungen zu erstellen, zu testen und bereitzustellen. Die Lösung unterstützt agile Entwicklungspraktiken und DevOps-Methoden wie kontinuierliche Integration und kontinuierliche Bereitstellung (CI/CD).

Wavefront by VMware

Wavefront® by VMware ermöglicht effizientes, skalierbares Überwachen von Containern. Die Wavefront-Plattform beinhaltet Dashboards, die DevOps-Teams Echtzeittransparenz für die Abläufe und Performance von containerbasierten Anwendungen und Kubernetes-Clustern bieten.

Mit dem Wavefront-Service lassen sich Daten in verschiedenen Containern und Kubernetes-Clustern messen, korrelieren und analysieren. Das Dashboard zeigt Daten zur Performance von Mikroservices und zur Ressourcenauslastung an, sodass Sie Probleme erkennen und Anwendungen optimieren können. Mithilfe dieser Daten können Sie z.B. Entscheidungen darüber treffen, wie und wann eine Container-Umgebung skaliert werden sollte. Weitere Informationen finden Sie unter VMware und Wavefront.

Pivotal Container Service

VMware® Pivotal Container Service bietet eine Kubernetes-Plattform auf Produktionsniveau, die Unternehmen das zuverlässige Bereitstellen, Ausführen und Operationalisieren moderner und herkömmlicher Anwendungen über Private und Public Clouds hinweg ermöglicht. Pivotal Container Service basiert auf dem Open Source-Projekt Kubo und bietet Hochverfügbarkeit, erweiterte Sicherheit und betriebliche Effizienz. Dadurch können Unternehmen die Time-to-Market verkürzen, die Produktivität von Entwicklern erhöhen und Betriebskosten senken.

Um Mikroservices und containerbasierte Workloads schnell in die Produktionsumgebung zu intergieren, errichtet Pivotal Container Service eine einheitliche Virtualisierungs- und Container-Infrastruktur auf VMware vSphere oder in einem Software-Defined Datacenter von VMware.

Komponenten

Pivotal Container Service umfasst die folgenden Komponenten:

- **Kubernetes auf Produktionsniveau**
- **BOSH:** Hierbei handelt es sich um ein Open Source-System, das Release-Entwicklung, Bereitstellung und Lebenszyklusmanagement für kleinere und größere Cloud-Softwarelösungen vereinheitlicht. Das System eignet sich optimal für große verteilte Systeme und führt Überwachungsfunktionen, Recovery bei Ausfällen und Software-Updates ohne bzw. mit minimalen Ausfallzeiten aus. BOSH unterstützt mehrere „Infrastructure as a Service“-Anbieter, einschließlich VMware vSphere, Google Cloud Platform, Amazon Elastic Compute Cloud (EC2) und OpenStack.

WEITERE INFORMATIONEN ...

Informationen dazu, wie die Lösungen von VMware Sie beim Erstellen, Ausführen und Verwalten von cloudnativen Anwendungen unterstützen, finden Sie unter: cloud.vmware.com/cloud-native-apps

- **VMware ESXi™:** ESXi ist der branchenführende zweckgerichtete Bare-Metal-Hypervisor und wird direkt auf dem physischen Server installiert. Dieser kann dann in virtuelle Maschinen partitioniert werden.
- **VMware NSX®:** Die Netzwerkvirtualisierungstechnologie für moderne Anwendungsarchitekturen bietet wichtige Netzwerkfunktionen für Kubernetes-Cluster.

Fazit

Das Entwickeln und Bereitstellen von containerbasierten Anwendungen auf VMware-Infrastruktur ermöglicht die Schaffung eines Mehrwerts durch digitale Transformation. VMware-Lösungen verbessern die Produktivität von Entwicklern, die geschäftliche Agilität, die IT-Flexibilität und die Skalierbarkeit von Anwendungen. Die daraus resultierenden Ergebnisse helfen Ihnen dabei, sich an Marktveränderungen anzupassen und die Markteinführungszeit für Anwendungen zu verkürzen.

VMware, Inc. 3401 Hillview Avenue Palo Alto CA 94304 USA Tel 877-486-9273 Fax 650-427-5001 www.vmware.com Zweigniederlassung Deutschland Willy-Brandt-Platz 2, 81829 München Telefon: +49 89 370 617 000 Fax: +49 89 370 617 333 www.vmware.com/de

Artikelnr.: VMW_17Q3_WP_Driving-Digital-Transformation-with Kubernetes_FINAL2_081617
08/17