AraCom IT Services AG

#2 FACH ARTIKEL

WWW.ARACOM.DE

DOCKER

MAI 18, 2020

AUTOREN: CHRISTIAN EGO & JULIA RIEGER

„Ich packe meine Docker-Container und nehme mit: Code, Laufzeitmodul, Systemwerkzeug, Systembibliotheken und dabei ist Docker-Daemon stets mein treuer Begleiter."

Docker ist eine Freie Software der Docker Inc. zur Verwaltung von Dateien durch Verwendung von Containervirtualisierung. Die Software ist eine Implementierung der Container-Technologie. Container sind die Alternativen zu virtuellen Maschinen. Damit können einzelne Services bzw. Anwendungen isoliert und sehr simpel transportiert, installiert, getestet und in Betrieb genommen werden. Docker ermöglicht es Entwicklern Softwares mit reduziertem Zeitaufwand bereitstellen zu können.

Docker setzt unterschiedliche Techniken des Kernels ein, um Anwendungen in einer losen Umgebung (Containern) zu isolieren. Ein Docker Container mit einer bestimmten Anwendung ist somit getrennt von anderen Containern und dem Host-System, wodurch das Deployment einer Anwendung sehr einfach reproduziert und die Anwendung auf andere Hosts transportiert bzw. verteilt werden kann. Aufgrund der Isolation können mehrere Container problemlos zeitgleich auf einem Host ausgeführt werden. Ein großer Vorteil ist, dass alle Container „leicht" sind, da sie keinen eigenen Kernel mitbringen, sondern auf den Kernel des Host-Rechners zurückgreifen.

INHALTSVERZEICHNIS

1. VORTEILE VON DOCKER

Die Verwendung von Docker ermöglicht es Entwickler-Teams einheitliche Dev-Umgebungen einzurichten, die sich im Idealfall sogar nahtlos auch auf Test- und Prod-Umgebungen portieren lassen. Da damit eine Applikation von Anfang an auf Prod-Umgebungen ausgerichtet ist, die Konfiguration zwischen den Umgebungen sich weitestgehend deckt und Container die minimalen Laufzeitanforderungen der Anwendung bereits enthalten, können Anwendungen mit einem deutlich verringerten Zeitaufwand bereitgestellt werden.

Dabei ist man völlig flexibel bei der Wahl des Systems und auch die Portabilität auf andere Maschinen wird ermöglicht. Diese Flexibilität entsteht durch Bündelung einer Anwendung mit allen ihrer Abhängigkeiten in einem einzigen Container. Dieser ist unabhängig von der Hostversion des Linux-Kernels, der Plattformverteilung und des Bereitstellungsmodells. So lässt sich der Container mit Docker auf einen anderen Computer simpel übertragen und auch die Ausführung ist ohne Kompatibilitätsprobleme möglich.

Weitere Vorteile sind die Wiederverwendung der Komponenten und die Versionskontrolle. Aufeinanderfolgende Versionen eines Containers lassen sich nachverfolgen, Unterschiede untersuchen und zudem kann auf vorherige Versionen zurückgegriffen werden.

Diese Stärke zeigt sich vor allem bei der Verwendung von Docker in CI-Pipelines.

VORTEILE	
Einheitliche Umgebung	**Schnelle Anwendungsbereitstellung**
Portabilität auf allen Maschinen	**Versionskontrolle und Wiederverwendung von Komponenten**
Leichtgewichtig und minimaler Aufwand bei Rollouts	**Einfachere Wartung einzelner Module**

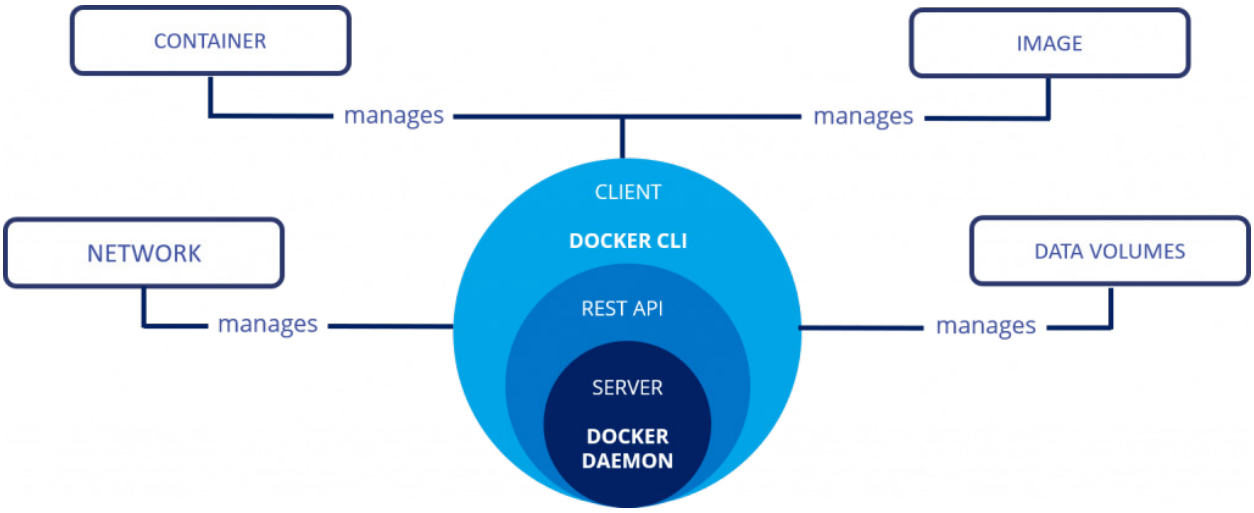
2. FUNKTIONSWEISE VON DOCKER

Um zu verstehen, wie Docker funktioniert, werden die sieben Hauptkomponenten von Docker näher erläutert:

a. Docker Daemon
b. Docker Client
c. Docker Layer
d. Docker Image
e. Docker Container
f. Docker Network
g. Dockerfile

DOCKER DAEMON

Docker Daemon ist Teil der Docker Engine und empfängt die Befehle des Docker-Clients über die CLI-Schnittstelle oder REST-API. Außerdem verwaltet der Daemon jegliche Docker-Objekte, wie Images, Container, Networks und Volumes. Ein Docker Daemon kann auch mit anderen Docker Daemons über die REST-API kommunizieren, um Dienste zu verwalten. Durch die Kommunikation untereinander, ist es möglich Docker Container über unterschiedliche Hardware zu verwalten.



DOCKER CLIENT

Der Docker-Client sendet Befehle an den Docker Daemon. Der Client kann auf dem selben Host laufen wie der Docker Daemon, aber auch auf jeglichen anderen Hosts.

DOCKER LAYER

Layer sind Bestandteile von Images und Container. Jeder Build-Befehl (RUN, COPY usw.) erzeugt einen eigenen Layer. Images bestehen aus mehreren Read-Only Layern und die Layer stellen das Dateisystem zum Zeitpunkt t dar.

Hinweis: Nur der oberste Layer eines Containers ist schreibbar!

DOCKER IMAGE

Images sind die Grundbestandteile von Docker, denn die Container-Instanzen werden aus den Docker Images aufgebaut. Betrachtet man einen Container, erkennt man das Layersystem. Es zeigt, dass jegliche Änderung eines Images als Layer über das unveränderte Image abgebildet wird.

DOCKER CONTAINER

Ein Container ist die Anwendungsumgebung von Docker und der schreibbare Layer des Images. Anwendungen lassen sich in Container verpacken und übergeben. Außerdem können daraus noch weitere Container entwickelt werden.

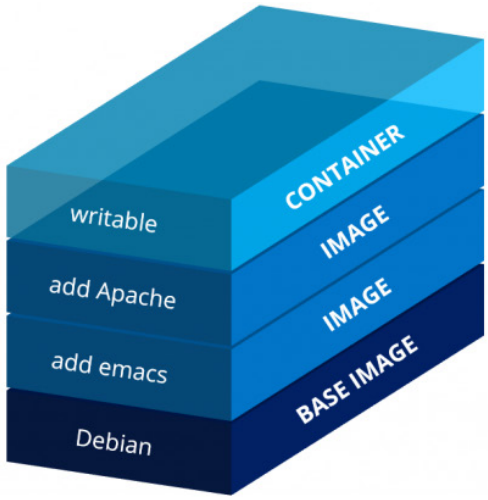
Container lassen sich: starten, stoppen, committen und beenden.

Hinweis: Wenn ein Container vor dem committen beendet wird, werden alle am Container vorgenommenen Änderungen gelöscht.

DOCKER NETWORK

Um eine Microservice-Anwendungsarchitektur abzubilden, können mehrere Container miteinander verbunden werden. Standardmäßig bringt Docker eine eigene Netzwerk-Verwaltung damit Container untereinander kommunizieren können. Die Grundeinstellungen isolieren dabei den Container vom Host System. Komplexere Anwendungsfälle lassen sich jedoch ebenfalls konfigurieren.

Goldene Regel: Ein Service pro Container!



DOCKERFILE

Ein Dockerfile beschreibt den Aufbau eines Images und dessen Layers. Dabei gilt zu beachten, dass jeder Befehl einen eigenen Layer erzeugt. Bei Ausführung von Shell-Befehlen mithilfe des RUN-Befehls sollte man daher mehrere Befehle zu einem einzigen Aufruf zusammenfassen oder ggf. in separate Scripte die ausgeführt werden auszulagern.

DOCKERFILE	BEISPIEL
ARG Build arguments Variablen, die nur während des Builds übergeben werden.	
FROM Definiert eine neue Build-Stage, die von einem vorhergehenden Basis-Image ableitet.	
COPY Kopiert Dateien aus dem Kontext (Basis-Verzeichnis).	
ENV Environment Variablen, die sowohl im Build als auch zur Laufzeit des Containers verfügbar sind.	
RUN RUN-Befehle sollten zusammengefasst werden, um weniger Layer zu generieren.	
WORKDIR Definiert das Arbeitsverzeichnis.	
ENTRYPOINT/CMD Definiert das Script oder den Befehl der beim Start ausgeführt wird.	

```
ARG PHP_VERSION=7.3.1

FROM php:${PHP_VERSION}-fpm-alpine

COPY docker-php-* install-php-extensions /usr/local/bin/
COPY config/ ${PHP_INI_DIR}/

ENV PHP_CONFIG_DIR=${PHP_INI_DIR}/conf.d

RUN set -xe; \
    rm -f /usr/local/bin/docker-php-entrypoint; \
    ln -nfs ${PHP_INI_DIR} /etc/php; \
    rm -f ${PHP_INI_DIR}/php.ini-*; \
    ln -nfs /etc/php/cli/php.ini /etc/php/php.ini; \
    install-php-extensions;

WORKDIR /srv/www

ENTRYPOINT ["docker-php-cli-entrypoint"]
CMD ["-"]
```

3. DATENVERWALTUNG MIT DOCKER

Mit Docker lassen sich unterschiedliche Docker Objekte einfach verwalten und teilen. Folgende Begriffe erklären dabei die Funktionsweise:

DOCKER REGISTRY

Bei Docker Registry handelt es sich um ein öffentliches oder privates Verzeichnis für Images. Damit können Images erstellt und mit dem Entwicklerteam ausgetauscht werden. Die Docker Inc. bietet den Docker Hub an, auf welchen man Verzeichnisse mit anderen Nutzern teilen kann. Die Funktionsweise ist dabei vergleichbar mit Git.

BIND-MOUNTS

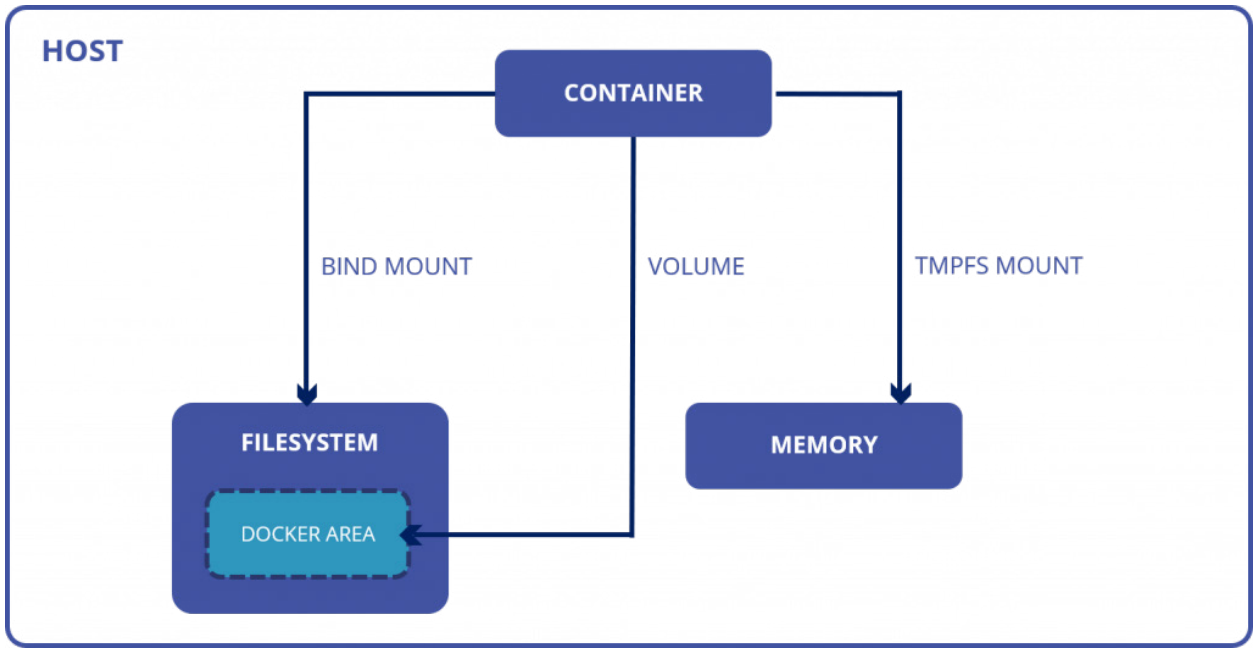
Bind-Mounts sind Daten in beliebigen Pfaden im Host-Dateisystem.

Hinweis: Andere Prozesse können Daten modifizieren.

VOLUMES

Volumes sind gespeicherte Daten im von Docker verwalteten Teil des Host-Dateisystems.

Hinweis: Volumes sind nur für Docker-Prozesse modifizierbar.



DOCKER REPOSITORY

Mit der Repository können die Versionen eines Images verwaltet werden. Die Versionierung erfolgt dabei durch die Kennzeichnung mit Tags.

4. DOCKER BEFEHLE (DOCKER CLI)

Im Folgenden finden Sie eine Auflistung der wichtigsten Docker-Befehle:

docker build: Baut Images anhand eines Dockerfiles.

docker ps-A: Listet alle laufenden und gestoppten Container.

docker images: Listet alle Images auf dem Host.

docker rmi: Löscht ein oder mehrere Images.

docker run: Führt einen Befehl in einem neuen Container aus.

docker exec: Führt einen Befehl in einem laufenden Container aus.

docker start: Starte einen gestoppten Container.

docker stop: Stoppt einen laufenden Container.

docker rm: Löscht einen oder mehrere Container.

docker pull: Abrufen eines Images aus einem Repository.

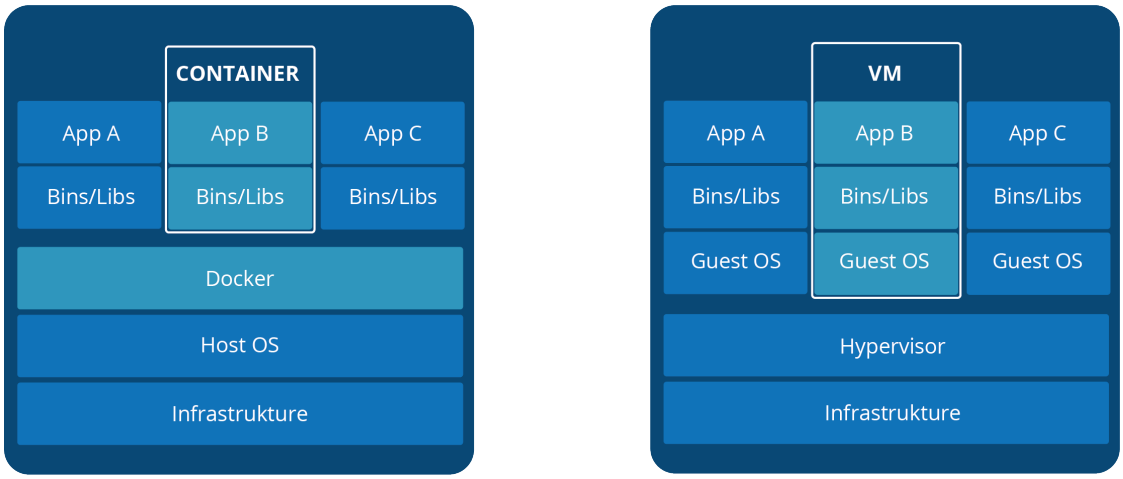
docker push: Übertragen eines Images in ein Repository.

5. DOCKER VS VIRTUELLE MASCHINE

Docker Container benötigen weniger Ressourcen als virtuelle Maschinen, da sie im Kontext des Host-Betriebssystems laufen.

Wohingegegen eine Virtualisierungslösung mithilfe eines Hypervisors virtuelle Maschinen betreibt, die jeweils mit eigenen Betriebssystem-Kernels laufen.

Im Gegensatz zur Virtualisierungslösung benötigt Docker keinen Hypervisor. Stattdessen verwaltet der Docker Daemon einzelne Container und deren Prozesse. Diese wiederum greifen auf den Kernel der Host Maschine zu.



6. DOCKER COMMUNITY EDITION VS ENTERPRISE EDITION

Die Docker Inc. bietet zwei unterschiedliche Haupt-Versionen von Docker an. Die Docker Community Edition (CE) ist die freie Containerisierungsplattform – es handelt sich dabei um die ursprüngliche Open-Source-Lösung der Docker Inc., welche lediglich umbenannt wurde. Die Premium-Variante ist die Docker Enterprise Edition (EE), welche einen intensiven Support der Docker Inc. bietet und zertifizierte Docker Images und Plugins enthält. Die Enterprise Edition lässt sich aber nochmals unterteilen in Basic, Standard und Advanced.

Ist die Community Edition eine abgespeckte Variante der Enterprise Edition?

Nein, denn die Kernfunktionen sind bei beiden Versionen vorhanden. Damit stellt die Community Edition keine technisch abgespeckte Variante der Enterprise Edition dar, jedoch ist mit der Enterprise Edition ein Support der Docker Inc. vorhanden.

Gibt es bei der Community Edition ebenfalls Support?

Bei der Community Edition muss leider auf den Docker Support verzichtet werden, jedoch können sich Docker-User im Docker-Forum austauschen.

Wird die Community Edition geupdatet?

Ja, auch die Community Edition wird geupdatet.

Auf welchen Systemen läuft welche Edition?

Die kostenlose Version läuft auf AWS, Azure, CentOS, Debian, Fedora, Raspbian, Ubuntu, Windows 10 und Mac.
Die Enterprise Edition läuft auf AWS, Azure, CentOS, Windows Server 2016, Windows Server 2019, Red Hat Enterprise Linux, SUSE Linux Enterprise Server, Oracle Linux und Ubuntu.

7. FAZIT ZU DOCKER

Docker bietet in der Software-Entwicklung ein Toolset, das Entwickler von Anfang an bei der Anwendungsentwicklung unterstützt. Hauptfaktoren sind dabei die Konsolidierung der Umgebungen (Dev/Test/Prod), Portierung und Verteilung von Anwendungen.

Die immer stärkere Integration in CI-Pipelines macht Docker zu einem unverzichtbaren Werkzeug im Dev-Ops Bereich. Gerade im Bereich der Microservice-Architekturen bietet Docker den entscheidenden Vorteil, dass die einzelnen Services als isolierte und lauffähige Anwendungen auf den unterschiedlichsten Systemen ohne Anpassung direkt ausgeführt werden können.

Letzteres ermöglicht es ebenfalls mithilfe von Docker skalier- und versionierbare Microservice Architekturen in der Cloud umzusetzen. Hierzu wird in der Regel Kubernetes eingesetzt, um das Ausrollen der einzelnen Versionen, die Skalierung und das Monitoring zu managen.

QUELLEN:

1. Merkel D. (2014). Docker: lightweight linux containers for consistent development and deployment.
2. Docker Documentation Commands – abgerufen am 15.05.2020
3. Why Docker? – Docker Webseite – abgerufen am 15.05.2020