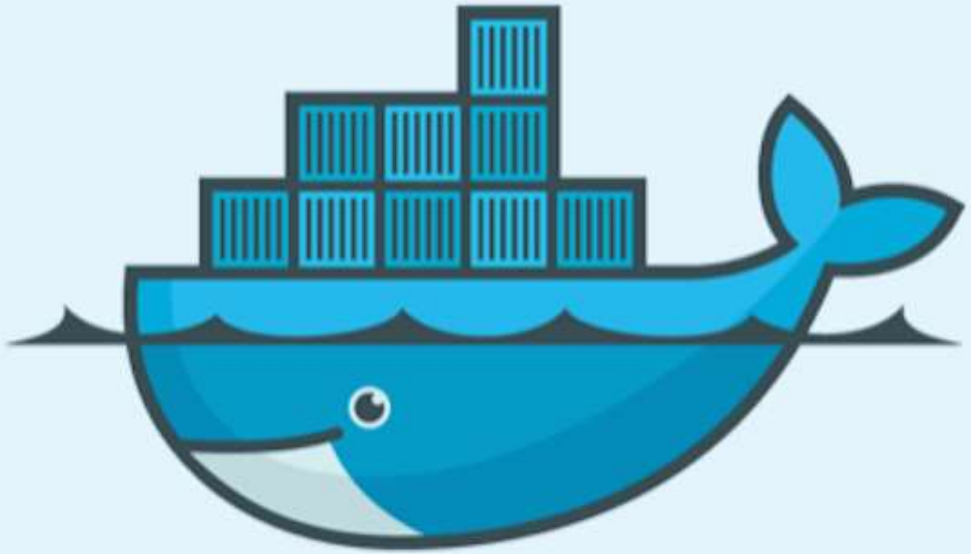
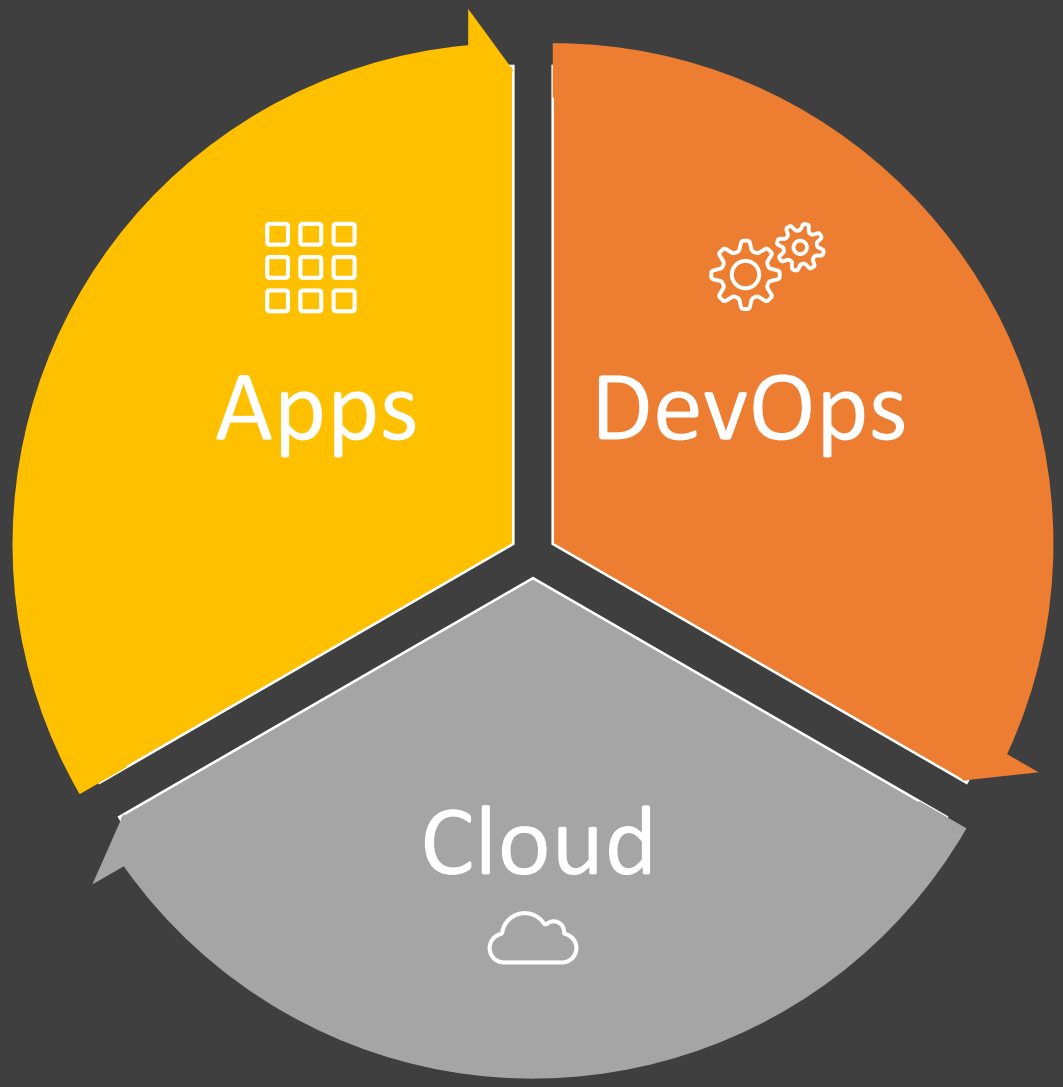
warptec
software gmbh

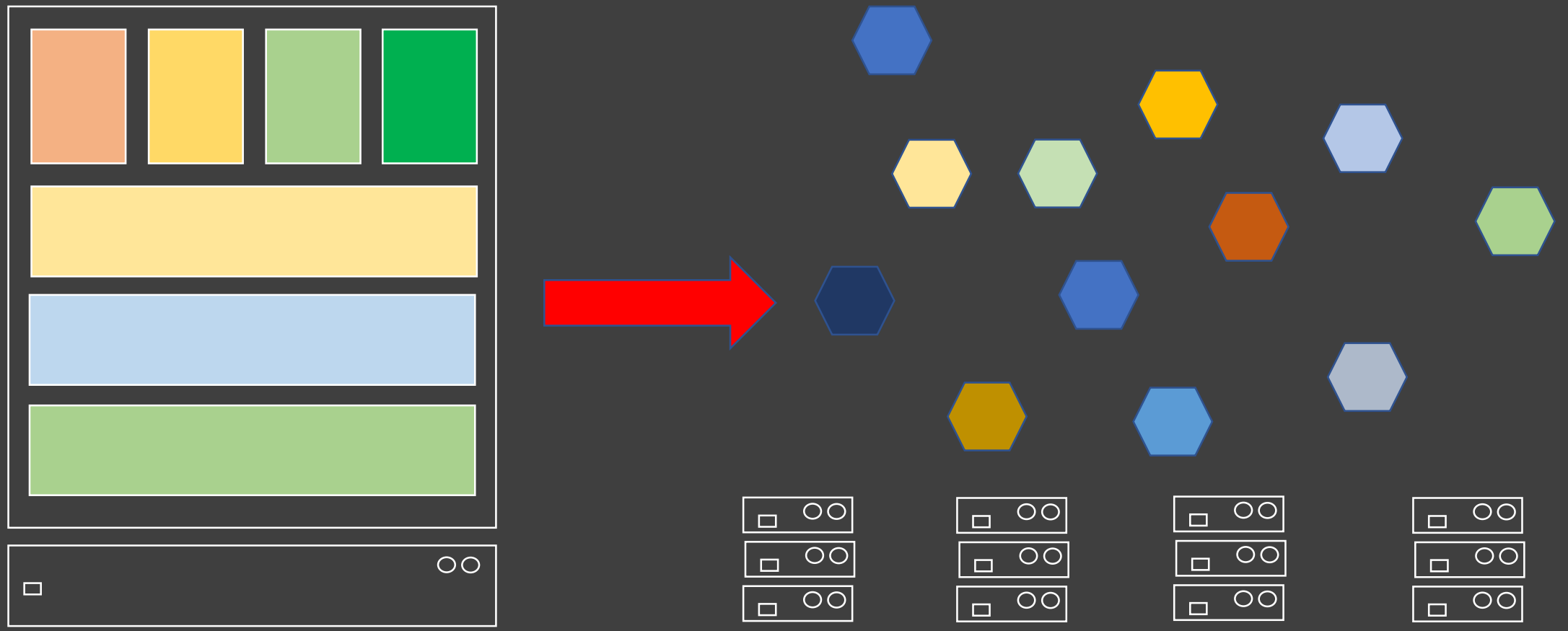


docker

Basics

Workshop
19.03.20


Apps
DevOps
Cloud

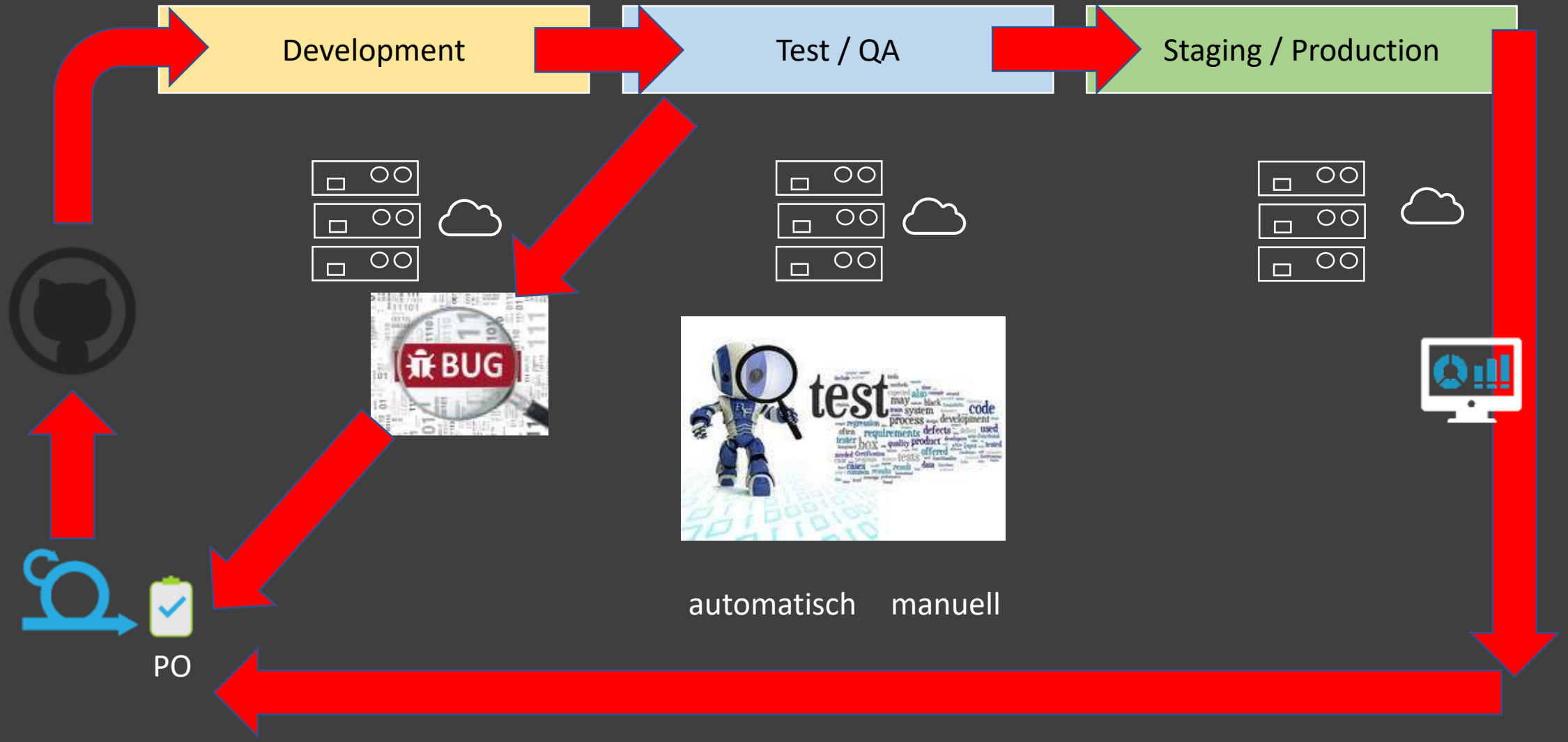


Monolith:

- Vollständige Kompilierung und Deployment
- Skalierungsschwierigkeiten
- Im Fehlerfall ist möglicherweise die gesamte Applikation betroffen

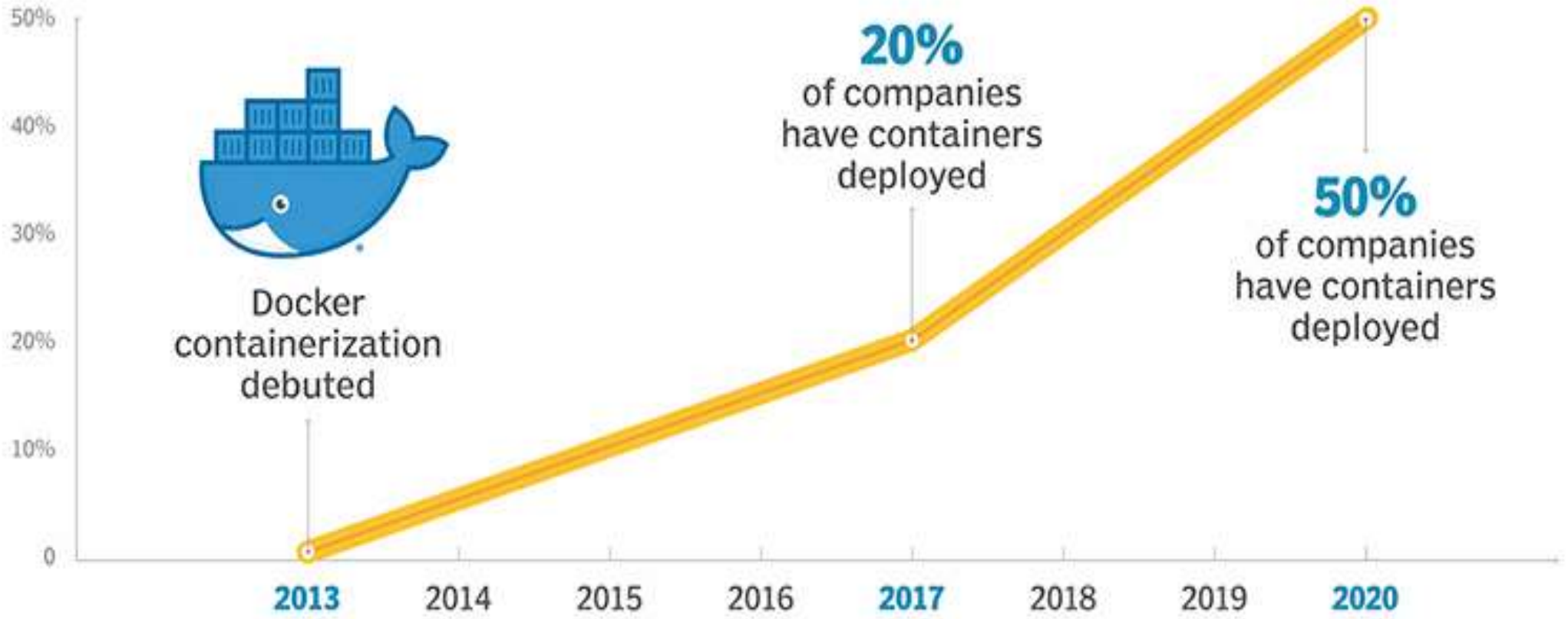
Microservice:

- Unterteilung in einzelne Dienste / Operationen
- Unabhängigkeit (technologisch…)
- Skalierbarkeit
- Fehlerisolierung

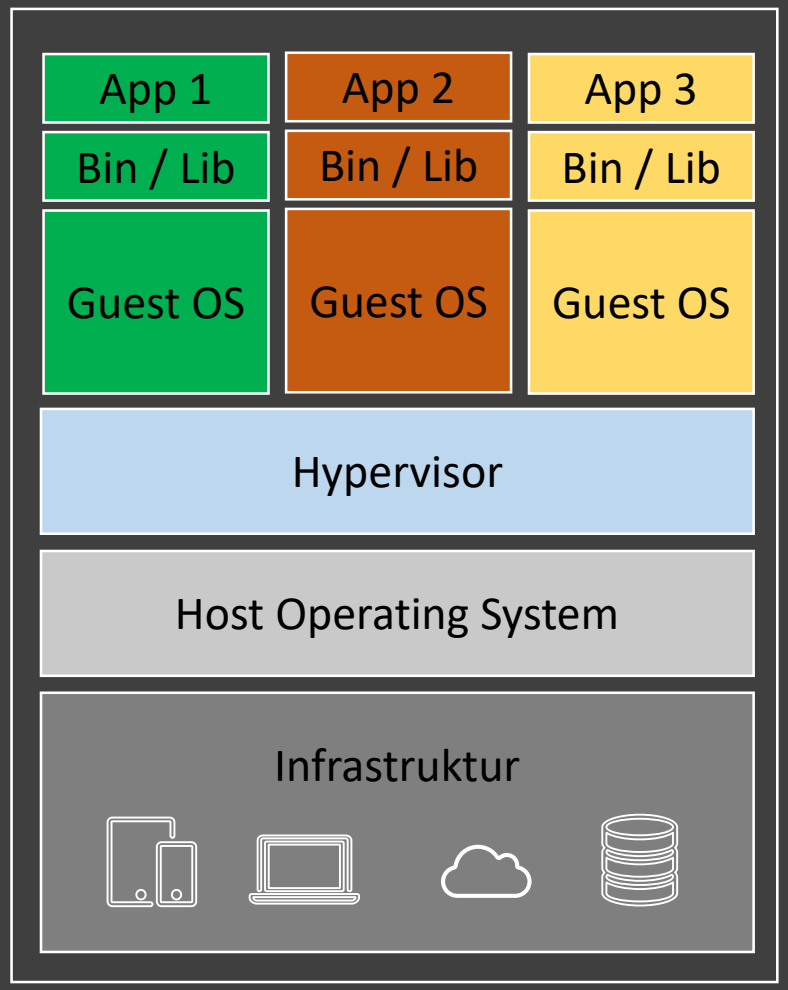

Development
Test / QA
Staging / Production
BUG
test
automatisch
manuell
PO
Entwickler
Quality Engineer
Operations

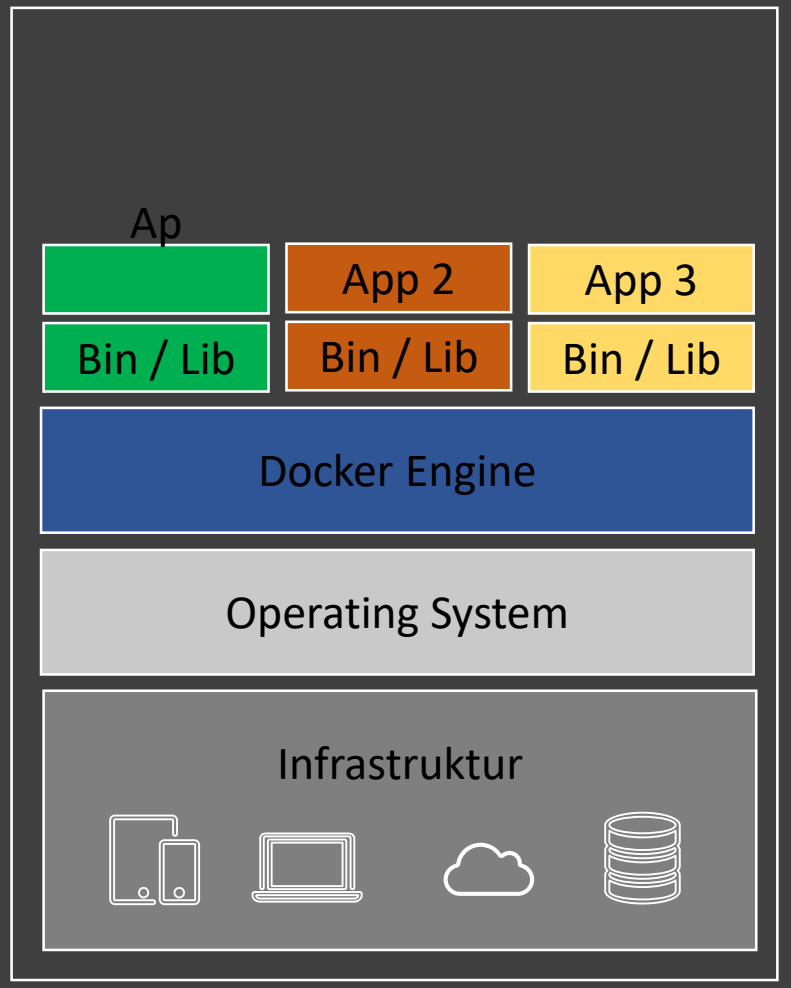

2008
Linux Container
LXC Technology
2016
Docker Enterprise
Edition
2004
Solaris Container
Zone Technology
2013
Docker wird
Open Source
Heute
sehr große
Community

Containerization timeline



SOURCE: GARTNER


App 1
App 2
App 3
Bin / Lib
Bin / Lib
Bin / Lib
Guest OS
Guest OS
Guest OS
Hypervisor
Host Operating System
Infrastruktur


Ap
App 2
App 3
Bin / Lib
Bin / Lib
Bin / Lib
Docker Engine
Operating System
Infrastruktur

Nutzen von Containerlösungen

Geschwindigkeit

- Kein Betriebssystem zu booten
- Applikationen sind in Sekunden online

Portierbarkeit

- Geringere Abhängigkeiten
- Möglichkeit Dienste innerhalb der Infrastruktur zu verschieben

Leistungsfähigkeit

- Kein Betriebssystem Over Head
- Isolation auf Prozessebene

Docker Installation

- Mac OS / Windows / Linux
  - https://hub.docker.com/

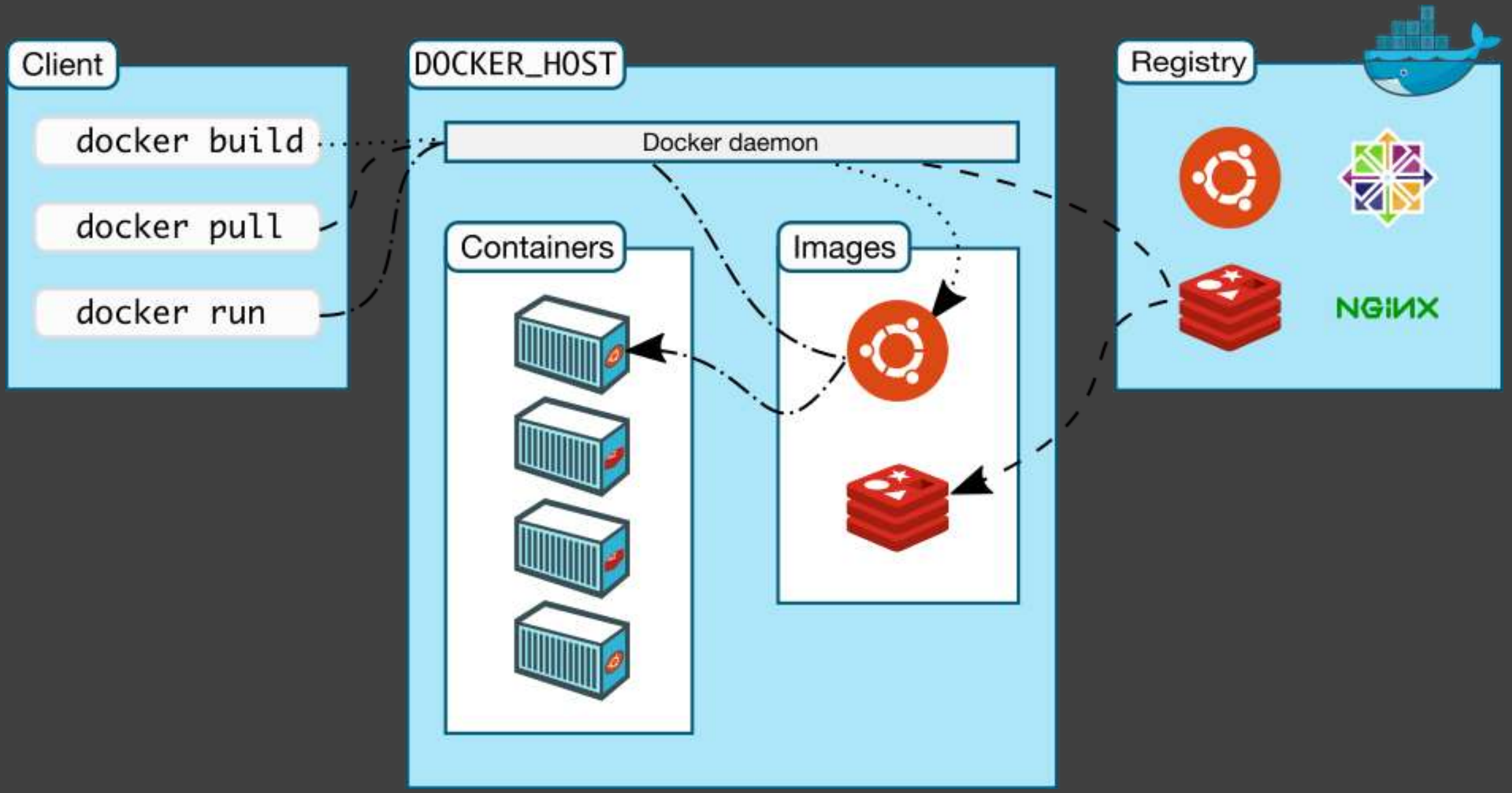
- Achtung Probleme bei parallel Installation mit anderen Virtualisierungsumgebungen
  - Z.B. VMware auf Windows

VS Code Installation

- Mac OS / Windows / Linux
  - https://code.visualstudio.com/

.net core Installation

- Mac OS / Windows / Linux
  - https://dotnet.microsoft.com/download/dotnet-core/3.1


Client
docker build
docker pull
docker run
DOCKER_HOST
Docker daemon
Containers
Images
Registry
NGINX

File

Textdokument, das alle Befehle enthält, um ein Image zu erstellen.

Image

Der Inhalt eines Containers.

Container

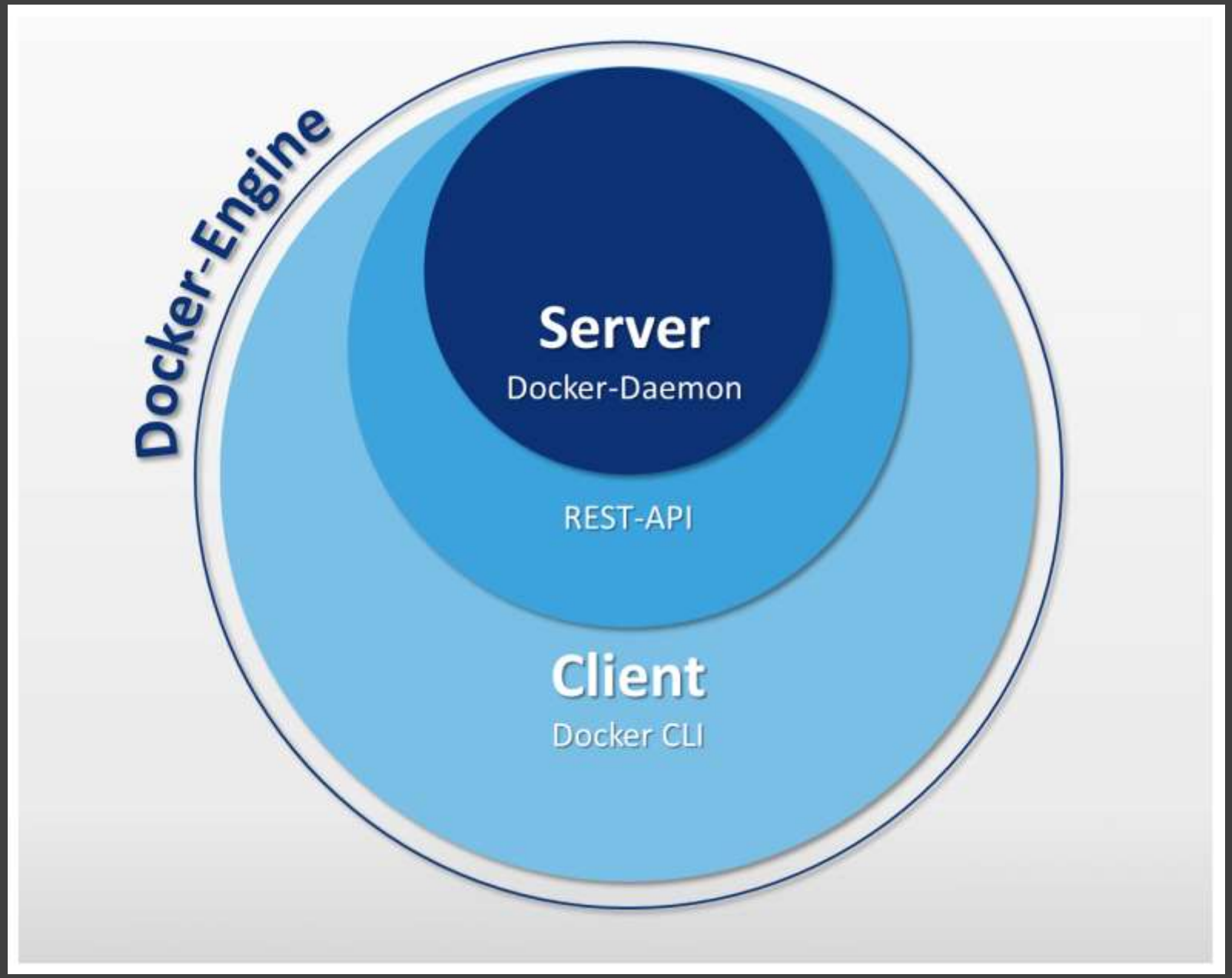
Das gestartete Image.

Engine

Führt Befehle für Container aus. Netzwerk und Speicher sind Teil der Engine.

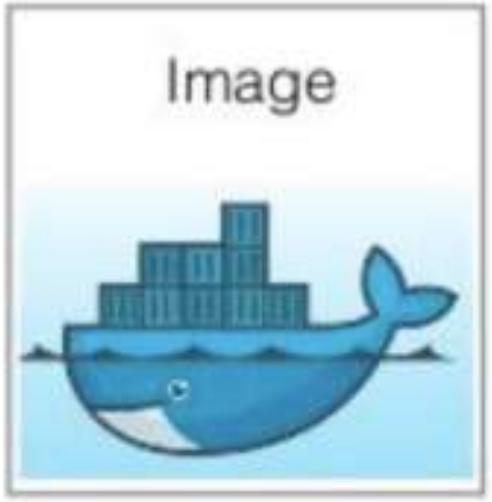
Registry

Verwalten, speichern und veröffentlichen von Images


Docker-Engine
Server
Docker-Daemon
REST-API
Client
Docker CLI

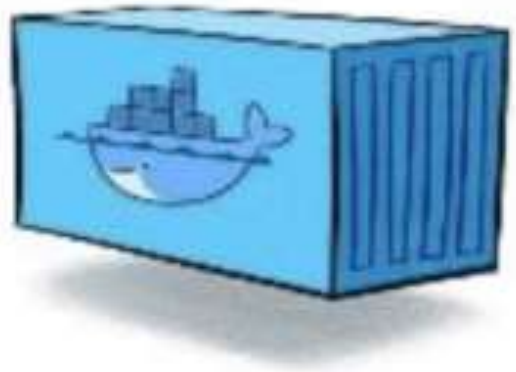
Dockerfile

build



Docker Image

run



Docker Container

Dockerfile Befehle

- FROM – Initialisierung einer neuen Build Phase und Festlegung des Basis Images. Ein gültiges Dockerfile muss mit FROM beginnen.
- RUN – führt alle Befehle in einer neuen Ebene über dem aktuellen Image aus und übertragt die Ergebnisse.
- ENV – Key Value Paar welches untergeordnet in allen weiteren Instruktionen verfügbar ist
- EXPOSE – Beschreibt auf welchem Port (TCP oder UDP) der Container Prozess veröffentlicht wird
- VOLUME – extern eingebundener Speicher (vom Host oder von anderen Containern)

Allgemeine Befehle

- docker version – zeigt detaillierte Infos über die Docker Client und Server Versionen
- docker login – Login in eine Docker Registry (z.B. Docker Hub)
- docker system prune -a – löscht alle unbenutzten container, unbenutzte Netzwerke und alle unbenutzten images
- docker info – Dieser Befehl zeigt systemweite Informationen zur Docker Installation an, Kernel-Version, die Anzahl der Container und Images.
- docker exec – Ausführen eines Befehls in einem laufenden Container (z.B. docker exec -it <container id> /bin/bash)

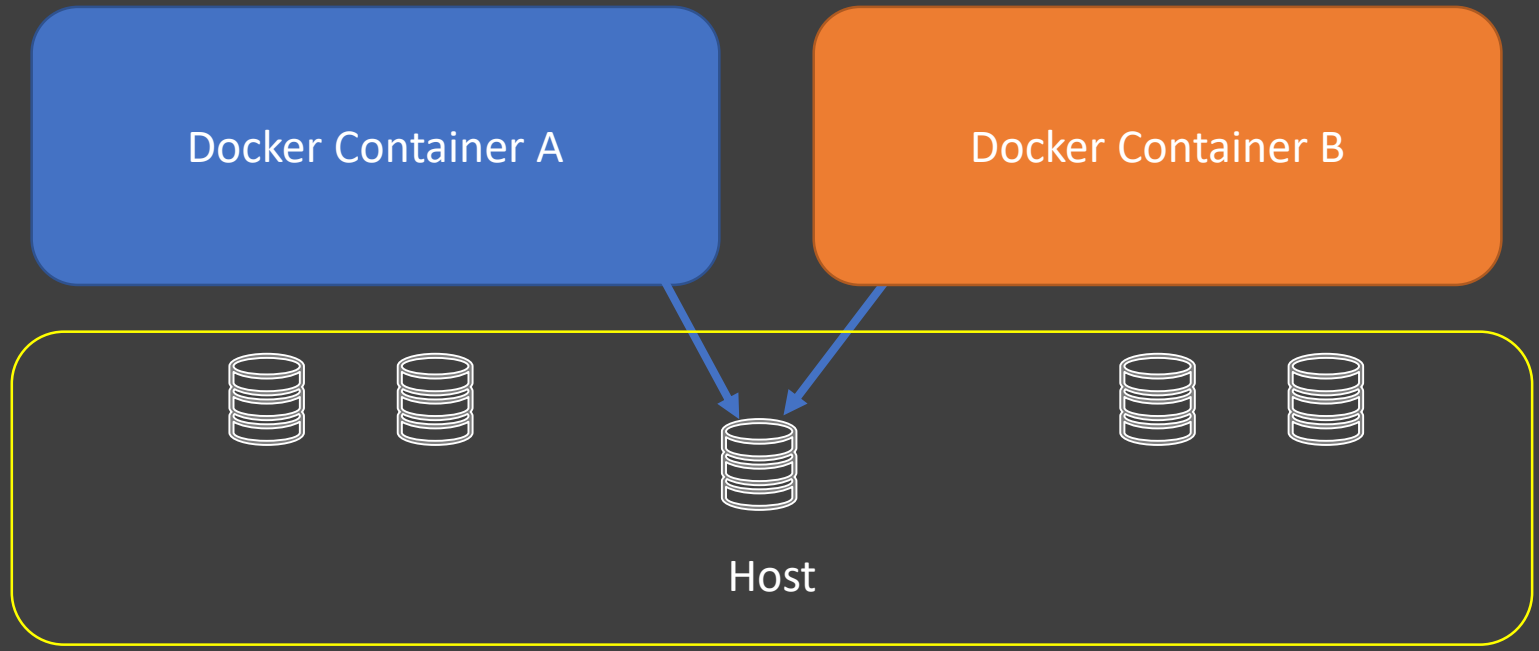
Image Befehle

- docker image ls– zeigt alle Images
- docker image build – erstellt ein Image aus einem Dockerfile
- docker image push – veröffentlicht ein Image in eine entfernte Registrierung
- docker image history – detaillierte Informationen zur image history
- docker image inspect – detaillierte Informationen zum image
- docker image rm – Delete an image.

Container Befehle

- docker container create– erstellt einen Container aus einem Image
- docker container start – startet einen existierenden Container
- docker container run – erstellt einen neuen Container und startet diesen
- docker container ls – zeigt eine Liste aller laufenden Container
- docker container inspect – detaillierte Informationen zum Container
- docker container logs – zeigt logs des Container Prozesses auf der Konsole
- docker container stop – Herunterfahren des Containers
- docker container kill – Hauptprozess im Container abrupt stoppen.
- docker container rm – Löschen eines Containers.

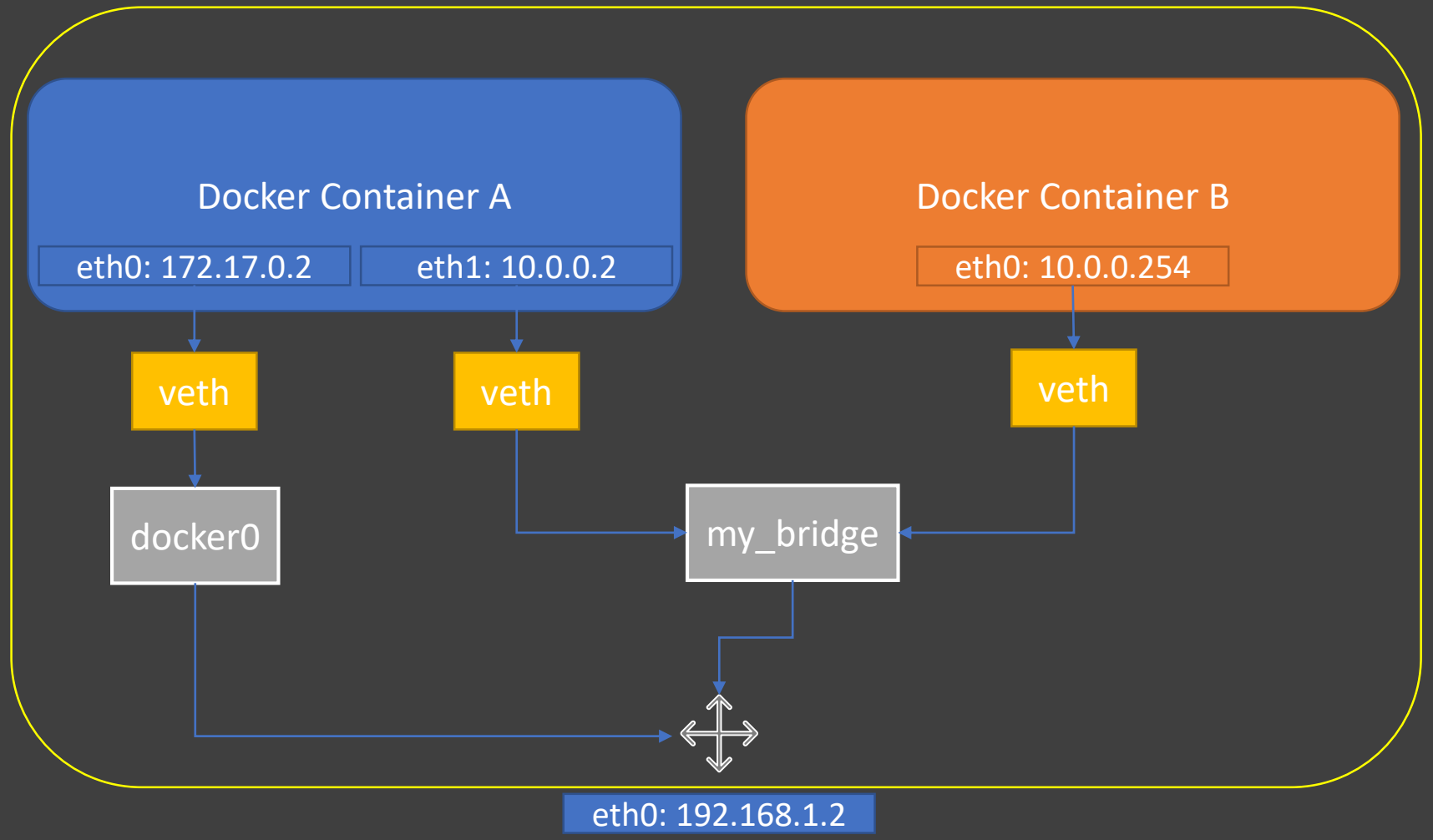
Volumes



Volumes Befehle

- docker volume ls – zeigt eine Liste aller Volumes
- docker volume inspect – detaillierte Informationen zum Volume
- docker volume create – erstellt ein neues Volume
- docker volume rm – entfernt ein Volume

Network

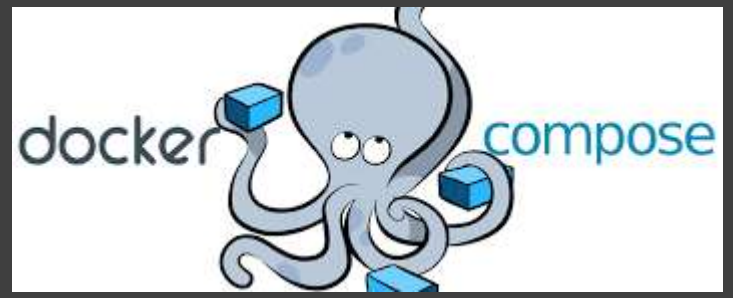

Docker Container A
eth0: 172.17.0.2
eth1: 10.0.0.2
Docker Container B
eth0: 10.0.0.254
veth
veth
veth
docker0
my_bridge
eth0: 192.168.1.2

Network Befehle

- docker network ls – zeigt eine Liste aller Netzwerke
- docker network inspect – detaillierte Informationen zum Netzwerk
- docker network create – erstellt ein neues Netzwerk
- docker network rm – entfernt ein Netzwerk

Befehls Referenz

- https://docs.docker.com/engine/reference/commandline/docker/



- Mit Docker Compose lassen sich innerhalb einer Datei mehrere Container erstellen, Beziehungen definieren und verwalten.
- Diese Container können mit einem einzigen Befehl gestartet werden.
- Die Beschreibung erfolgt in einer docker-compose.yml Datei

Compose Befehle

- docker-compose up – erzeugt und startet alle container
- docker-compose build – erzeugt alle Container und tagged diese, somit sind sie beim nächsten Start schneller verfügbar.
- docker-compose ps – listet die aktiven Container auf
- docker-compose start/stop – starten oder stoppen einzelner Container aus dem Compose File
- docker-compose rm –entfernen eines einzelnen Containers

Vielen Dank für die Aufmerksamkeit