Containerization in the Linux Kernel

Kilian Calefice (796461)

Linux Internals

09.09.2024

Gliederung

- Einführung
- Relevante Features des Kernels
- Funktionsweise
- Beispiel mit Docker

Einführung

- Container sind eine Abstraktion, um Prozesse isoliert auszuführen und nutzen dabei verschiedene Kernel-Features
- Die meisten relevanten Kernel-Features wurden 2008 im Kernel eingeführt

Virtual Machines vs. Container

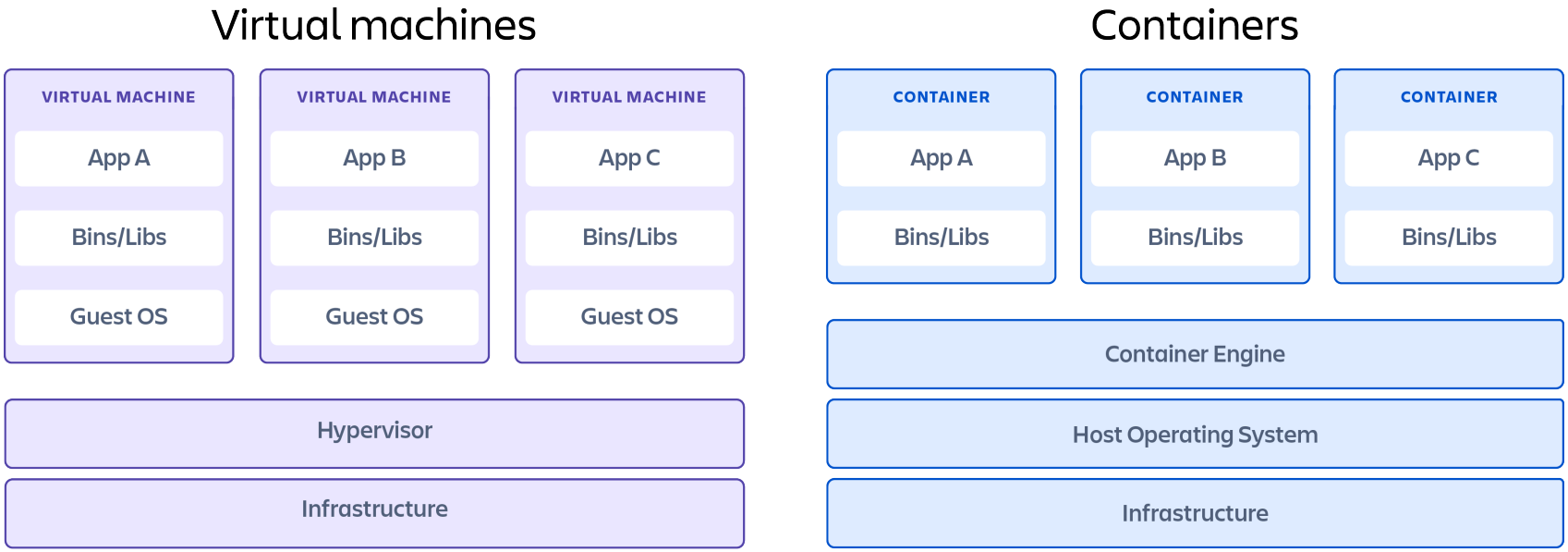

Abbildung 1

Abbildung 1: https://wac-cdn.atlassian.com/dam/jcr:92adde69-f728-4cfc-8bab-ba391c25ae58/SWTM-2060_Diagram_Containers_VirtualMachines_v03.png

Kernel-Features

1. Control Groups
2. Namespaces
3. Overlay Filesystem
4. ...

Control Groups

- Controller / Subsystems überwachen die Resourcennutzung im Kernel und sorgen dafür, dass Limits eingehalten werden
- Hierarchiche File-Struktur in `/sys/fs/cgroup`, um cgroups zu definieren
- Limits werden von den Eltern-cgroups an Kind-cgroups vererbt
- Child Prozesse treten automatisch der Cgroup des Elternprozesses bei

Namespaces

- Es gibt pid, net, mnt, ipc, uts, user und cgroup namespaces
- Sind dazu da Kernelresourcen zu isolieren
- Mit `unshare` kann man eine neue Namespace erzeugen
- Eine Namespace kann nur mit einem Prozess in ihr existieren

Overlay Filesystem

- Filesystem besteht aus directories die auch als Layers bezeichnet werden (**Lower Layer**, **Upper Layer**)
- Copy on write: Files werden nicht in tieferen Layers modifiziert sondern in eine neue Layer kopiert und dort modifiziert
- Merged view: Wenn man eine File lesen will guckt der Kernel in der obersten Layer und geht jeweils tiefer wenn er sie in der Layer nicht findet

Quellen

- Red Hat - Introduction to Linux Container
- Linux Fest - Container Pirimitives
- Init PID - Docker Container
- Docker Con - Cgroups, namepsaces and beyond