Chemnitzer Linux Tage

podman

als Alternative zu

docker

Agenda

- Motivation zum Thema
- Begriffsdefinition
- Architekturvergleich
- Besonderheiten podman
- Umgang mit compose

Motivation

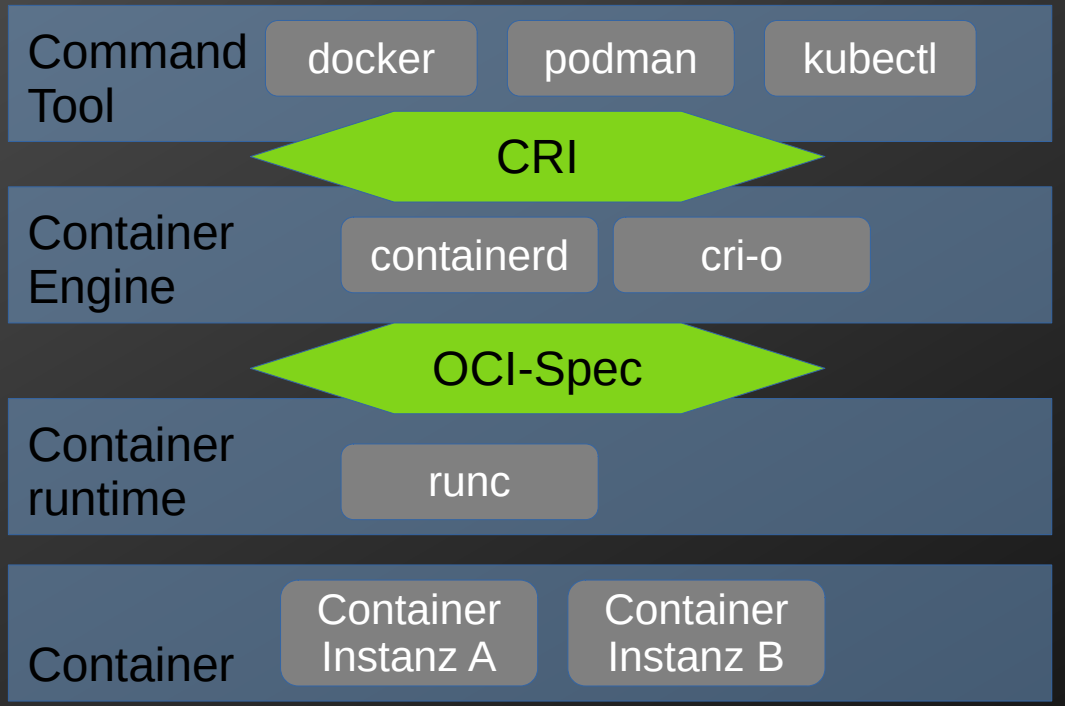
- Am Anfang war Docker
- Heute eine große Vielfalt, Kubernetes, K3s, OpenShift und auch podman
- Grundlage Pods, ein Modell das Docker nicht kann
- Außerdem docker != Container, es gibt inzwischen OCI 
- Bessere Integration in Linux, später mehr
- Aufzeigen, dass es eine Wahlmöglichkeit gibt

podman

- Open Source container Virtualisierer:
  - Zitat github: „A tool for managing OCI containers and pods.“
- Befehlszeile nahezu identisch zu docker
- Unterstützt durch Red Hat
- Ursprünge liegen im CRI-O Projekt
- Eigenständiges Repository seit ca. 2018

Begriffsdefinition

- Command-Tool
- Container Runtime Interface (CRI)
- Container Engine
- Open Container Initative Specification (OCI Spec)
- Container runtime



Beispiele

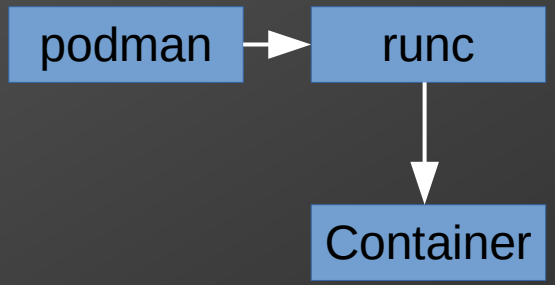
- OCI
  - runc
  - crun
  - kata-runtime
  - gvisor
- CRI
  - cri-o
  - Containerd (libcontainer)

Vergleich podman docker kubernetes

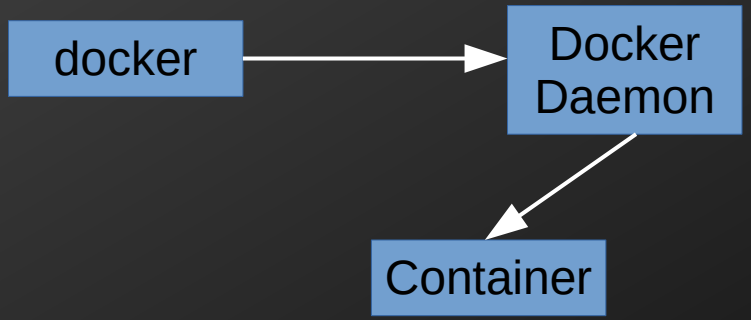
	podman	docker	Kubernetes
Lizenz	Apache 2.0	Proprietär	Apache 2.0
Architekturmuster	Forking	Client Server	Verteiltes System
Unterstützt pods	ja	nein	ja
Cluster	nein	Mittels Swarm ❗	ja

Architektur

- Podman
- forking, analog dem Standardprozessmodell von Linux



- Docker
- Client-Server
- Zentraler daemon startet alle container



podman ersetzt docker

- Ganz bequem:
  - alias docker=‘/usr/bin/podman‘
- „docker“-Socket:
  - systemctl --now enable podman.socket
- Oder im User Context:
  - systemctl --user --now enable podman.socket
  - Export DOCKER_HOST=unix:///run/user/$UID/podman/podman.sock

podman wie docker?

- mehr Befehle
- Container erzeugen ohne ihn gleich zu starten:
  - podman create alpine:latest ash

Nach Diskussion im Vortrag inzwischen auch in docker vorhanden

- Konzept der pods
- Zusammenspiel mehrere Container genauer steuern
- Gemeinsame Volumes
- Lokale Test für Kubernetes

Rootless Container

- Jeder User kann Container ausführen
- Bleibt in seinem Kontext (Sicherheitszugewinn)
- Mehrere User auf einem Server vorstellbar und machbar
- ~/.local/share/containers hier liegen die Images (besser Layers)

Integration und Interaktion

- podman generate <TYPE> <pod|container>

systemd

- Erzeugt SystemD Service File
- Container als Service automatisch starten

kube

- Kubernetes YAML erzeugen
- Übertragbar auf Kubernetes

Fragen?

Und was ist mit compose

- 2 Varianten, simpel:
  - Wie bereits gezeigt, den Socket aktivieren
  - Docker compose ausführen
- Oder:
  - podman-compose (python package)
  - Innerhalb des Verzeichnis mit dem compose File
    - podman-compose up

Fragen?