OS-Level Virtualisation Teil 1

Dr. Christoph Zimmermann

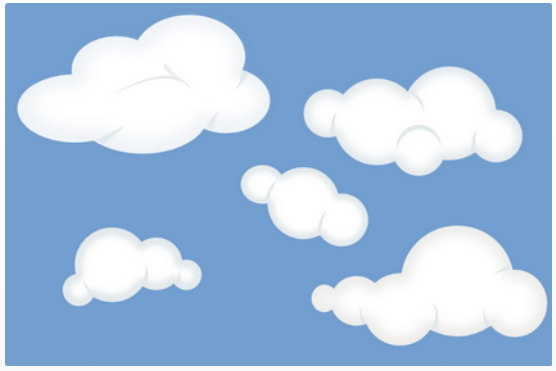
28. 2. 2023 @ FraLUG

cat /etc/motd

- Geschichtliches
- chroot
- Snaps & der ganze Rest
- Jails / Zones
- Container
- Zusammenfassung & Ausblick

type history

- Anfänge der Virtualisierung: ´70 mit IBMs VM
- Seitdem:
  - Hardware: Hypervisor, etc
  - Betriebssysteme (dieser Vortrag)
  - Software: VMs (UCSD-p System, Java, Erlang, etc.)
- Letzter Trend 😀:



chroot

- Älteste *ix “Pseudo”-Virtualiserung (1979) **V7**
- Idee: Unterverzeichnis mit separatem User-Land inkl. Befehlen, Bibliotheken, etc.
- Vorbereitung:
  - Population des Unterverzeichnis mit (minimalem) Userland
  - Je nach Funktion des Userlands:

```
mount -o rbind {/dev /proc /sys} <mnt-Pfad/...>
```

- Sicherheit: naja...

  S. u. a. https://web.archive.org/web/20160127150916/http://www.bpfh.net/simes/computing/chroot-break.html

snap & co



- Snaps, Flatpaks, AppImages
- Paketierte Software mit eigenem FS:
  - AppImage / Snap: SquashFS
  - Flatpak: libostree (git-artiges CAM -> Update-Versionierung)
- Eigene Repos:
  - snaps: snapcraft.io
  - flatpaks: flathub.org
  - AppImages: AppImage Hub (mehr Verzeichnis als Repo)

snap & co (ff)



- Sicherheit:
  - Snaps: AppArmor / seccomp
  - Flatpak: seccomp / Flatseal
  - Userland- / System-Zugriff (z. T. konfigurierbar):
    - Snaps: Connectors / Plugs
    - Flatpak: Portals
  - Snaps / Flatpaks: anscheinend nur rudimentäres Code-Vetting
  - Sandboxing:
    - Snaps: cgroups
    - Flatpaks: Bubblewrap (u. a. für Mount-Namespace)
- snapd / flatpak: Server für Admin/Ausführung (=> root-Rechte!)
- Trend: Ubuntu Core, etc.

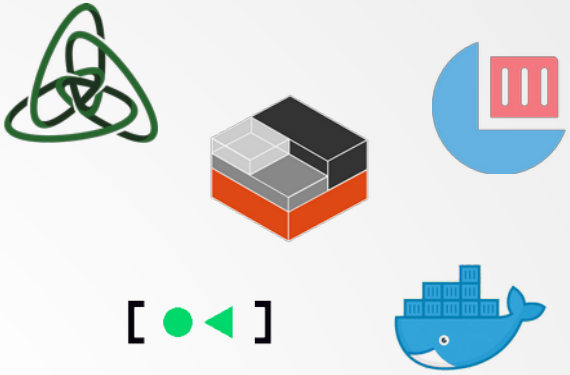
jails

- Zuerst in (Free)BSD 4.0 (2000)
- Sandkasten mit:
  - Eigenen IP-Adressen, Quotas (CPU, Speicher, etc.)
  - Normalerweise keine Raw Sockets
  - Keine IPC ausserhalb des Jails
  - Eigenem `securelevel`
  - Kein Nachladen von Kernel-Modulen
- Stark eingeschränktes `sysctl`
- Typischerweise Zugang via ssh (<=> chroot)
- ➢ Ziemlich sicher im Vergleich zu `chroot`

zones

- Zuerst in Solaris 10 (2004/5)
- x86/SPARC
- Ziemlich Container-ähnlich (deswegen auch „Solaris Containers“):
  - Abstrahierte H/W
  - Quotas
  - Vollständige Administration (analog zu Jails)
  - BrandZ: Zones mit zusätzlichen Merkmalen, u.a.:
    - Unterschiedliche Solaris-Versionen
    - Linux-Emulation (RH)
    - Cluster-Funktionalität, etc.
  - → Z. T. eigene Repos

container(d)



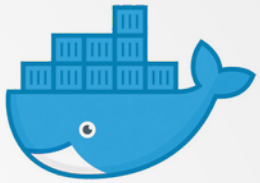
- Geschichtliches:
  - OpenVZ: 2005
  - LXC: 2008
  - Docker: 2013
- Zentrale Funktion:
  - S/W-Packaging + Resourcen-Kontrolle (!)
  - Leichtgewichtige „VM“
  - Primäres Userland: Linux
- Zentrale Kernel-Mechanismen:

cgroups / namespaces

- Control Groups:
  - V1: Kernel 2.6.24 (2008) <= Google
  - V2: Kernel 4.5 (2015)
  - Resourcen-Kontrolle: Speicher, CPU, I/O-Bandbreite
  - Priorisierung
  - Accounting
  - Administration: Checkpointing, Freezing
- Namespaces (2.4.19, 2002): Isolation von Kernel-Primitiven
  - pid, mnt, net, ipc, user (verschachtelt), cgroups, UTS/time

container(d) (ff)

- Fallstudie: Docker
- Container:
  - Image
  - Layered Filesystem
  - Runtime (containerd / runc)
- Ökosystem:
  - Klient / Server
  - Docker Hub (Images)
  - Dockerfile / docker compose
  - Infrastruktur: Volumes, Ports, etc.
- Ausblick: OCI / CNCF



cat Dockerfile

```
FROM ubuntu:20.04

RUN apt-get update && \
    DEBIAN_FRONTEND=noninteractive TZ=Etc/UTC \
    apt-get install -y \
        sudo time git-core subversion build-essential gcc-multilib-arm-linux-gnueabihf \
        libncurses5-dev zlib1g-dev gawk flex gettext wget unzip \
        grep rsync python3 python3-distutils && \
    apt-get clean

RUN useradd -m openwrt && \
    echo 'openwrt ALL=NOPASSWD: ALL' > /etc/sudoers.d/openwrt

USER openwrt
WORKDIR /home/openwrt

RUN git clone git://git.openwrt.org/openwrt/openwrt.git -b openwrt-22.03 && \
    openwrt/scripts/feeds update -a
```

sha256 /etc/motd

- chroot -> container: Funktionalität vs. Komplexität
- Container:
  - Skalierbarkeit
  - Container vs. VMs
  - Cloud
- Herausforderungen:
  - CPU-Architekturen jenseits von x86
  - Sicherheit (runc: CVE-2019-5736) => Privilege Escalation @ Host
  - Orchestrierung / Clustering (Teaser :-)

apropos

- Snaps: https://snapcraft.io/docs
- Snap Store: https://snapcraft.io
- flatpak: https://docs.flatpak.org/en/latest
- flathub: flathub.org
- AppImage Hub: https://www.appimagehub.com
- Jails: http://phk.freebsd.dk/sagas/jails/
- Zones: https://docs.oracle.com/cd/E26502_01/html/E29024/toc.html
- cgroups / Namespaces: https://www.kernel.org/doc/Documentation/cgroup-v2.txt
- Docker: docker.io

Fragen?

Vielen Dank!

Dr. Christoph Zimmermann

monochromec at <ignore>space</ignore>gmail<dot></dot>com