Embedded Container mit runc

Eine Einführung

Michael Estner

21. Augsburger Linux-Infotag
April 26, 2025

Wer bin ich?

- Name: Michael Estner
- Linux Architekt @ Elektrobit
- Embedded Linux, Netzwerk, Kernel
- Aktiv in der Yocto Projekt Community
- Teilzeit Master Praktische Informatik Uni Hagen
- Freizeit: Kochen, Wandern, MMA

Embedded Container mit runc?

Agenda

- Einführung Container
- Begriffsklärung im Container Kosmos
- Einstieg runc
- Container security
- Container im Embedded Kontext
- Crun
- Zusammenfassung

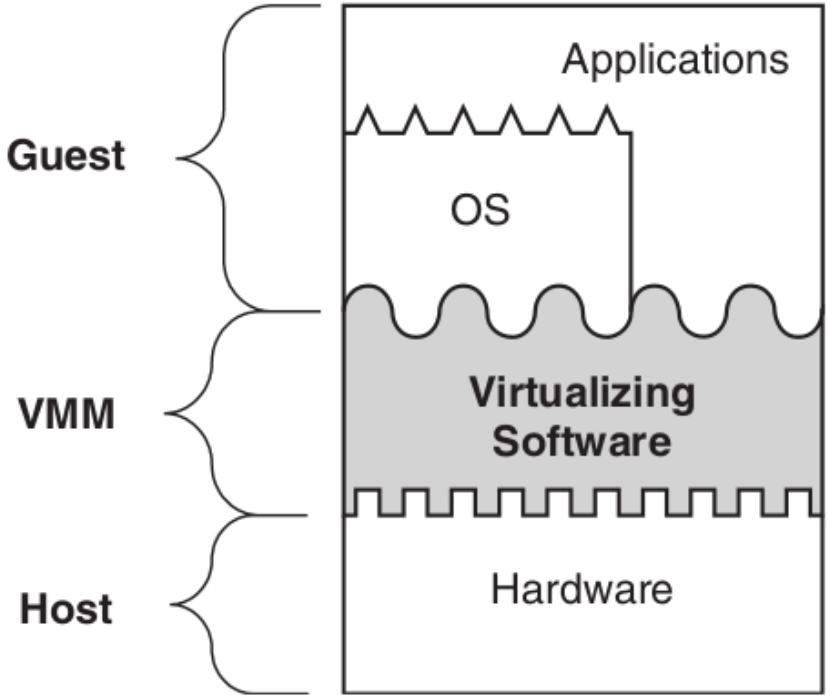
Container vs Virtualisierung

	Container	**Virtualisierung**
Ressourcen	Host	VM
Library	Container	VM
Userspace	Container	VM

Table: Vergleich

Virtuelle Maschine

System VM
Aufbau einer ”klassischen” VM

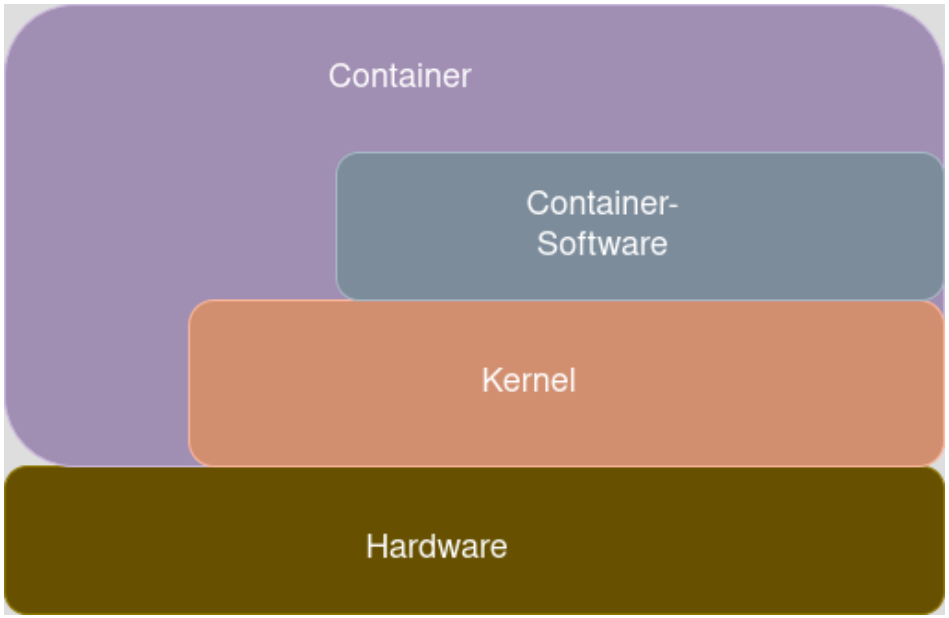


[6] S.12

Container

Container

- Nutzerprozesse direkt auf der Hardware ausgeführt
- Container greift direkt auf das Host Betriebssystem zu
- Isolation ist Unterschied zu einem Prozess



Gründe für Containern

- Entwicklungs- und Testumgebung
- Microservice Architekturen
- Continous Integraiton und Continous Deployment
- Cloud Anwendungen
- Effizienzsteigerung
- Plattformunabhängigkeit
- Isolierung und Sicherheit

Aufbau eines Containers

- Root Dateisystem
- Namespaces
- Cgroups
- Mountpunkte
- Security

Begriffsklärung

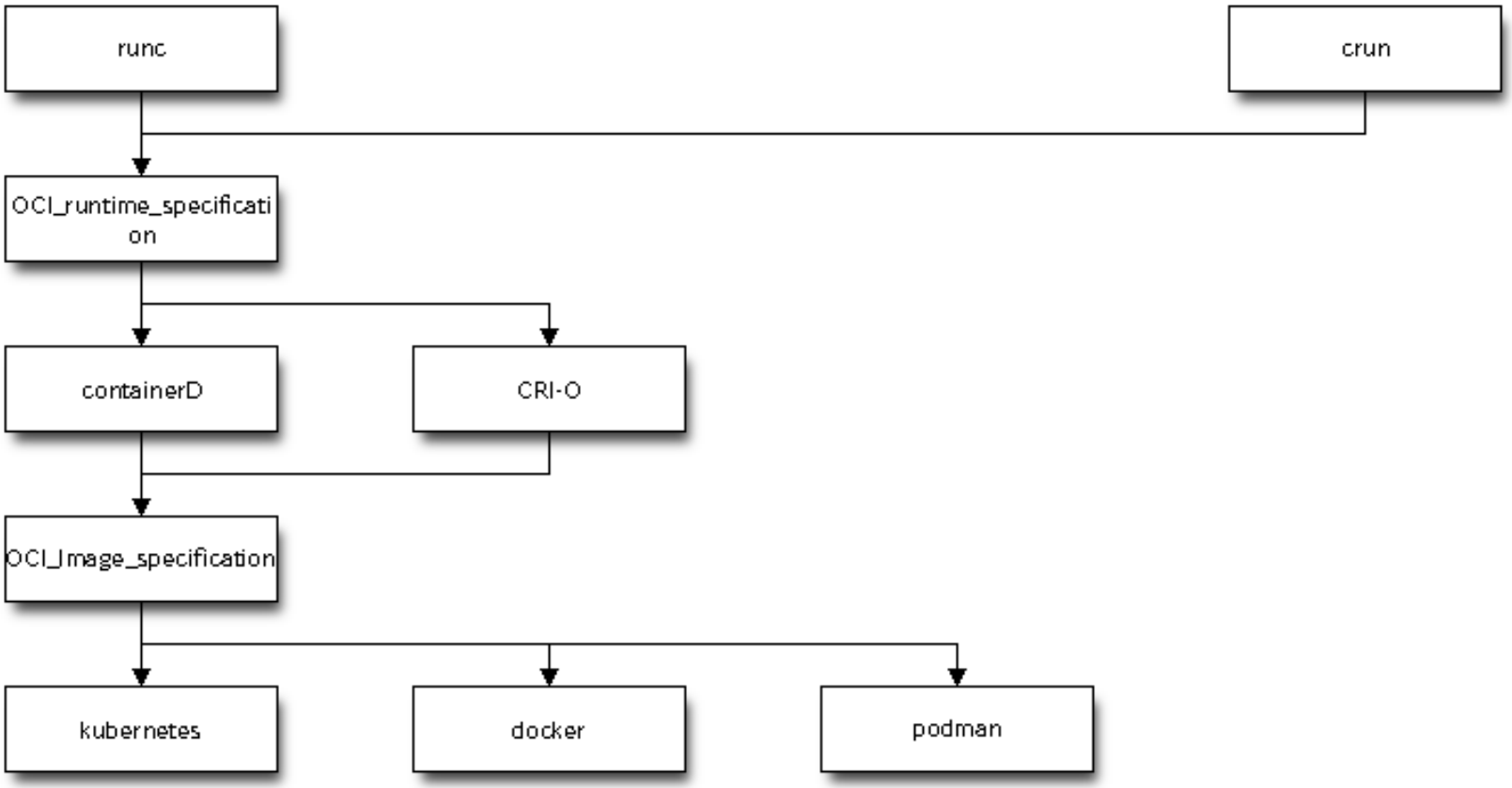


Figure: Zusammenhang Begrifflichkeiten

container management vs container runtime

container runtime

- Runc startet — stopped container
- Namespaces, cgroups, usw. wird darüber gemanaged.

container management

- Image management
- Komplexere buildtasks
- Lifecycle management

runc

runc everywhere

- runc verbirgt sich dahinter
- ideal für embedded Systeme mit begrenzten Ressourcen

[3]

runc

- runc
- runc is a CLI tool for spawning and running containers on Linux according to the OCI specification.[4]
- go geschrieben

runc

- Leichtgewichtige runtime
- Ausführen von Containern
- Json file Konfiguration
- Funktionen wie Namespaces, cgroups, capabilities und seccomp

runc - bundle

bundle

- config.json
- rootfs

- Ordner rootfs
- Runc spec - erstellt eine default config.json
- Rootfs erstellen busybox, tinybox, alpinerootfs, usw.

Root Dateisystem

- Minimalistisch
- Portabel
- Read only und schreibbare Daten

Namespaces

- mount
- network
- uts
- pid
- ipc
- uts
- user
- time

Namespaces

Figure: Einstellungen namespaces

cgroups

- Scheduler
- Ressourcen für einen Prozess begrenzen
- Gruppen: cpu, memory, pids, freezer, etc.

cgroups

Figure: Einstellungen cgroups

mountpoints

- Mountpoints im Container
- Rechte Vergabe
- Größe definieren

mountpoints

Figure: Einstellungen mounts

seccomp

- Mechanismus um syscalls zu beschränken[5]
- Best practice: So wenig wie möglich erlauben
- Möglichkeit das Edge cases nicht funktionieren, da syscall blockiert ist.
- Strace oder ebpf hook ist tracen möglich

seccomp

```
"seccomp":
{
        "defaultAction": "SCMP_ACT_KILL",
        "architectures": [
        "SCMP_ARCH_AARCH64"
        ],
        "syscalls": [
        {
        "names": [
        "exit",
        "ioctl",
        "read",
        "uname",
        "write"
        ],
        "action": "SCMP_ACT_ALLOW"
}
```

Figure: Einstellungen seccomp

Capabilities

- Es gibt unzählige Capabilities
- CAP_NET_ADMIN als Beispiel zu einem container hinzufügen welcher Änderungen beispeilsweise an der firewall vornehmen muss
- Grundsatz: So wenig Capabilities wie möglich

Capabilities

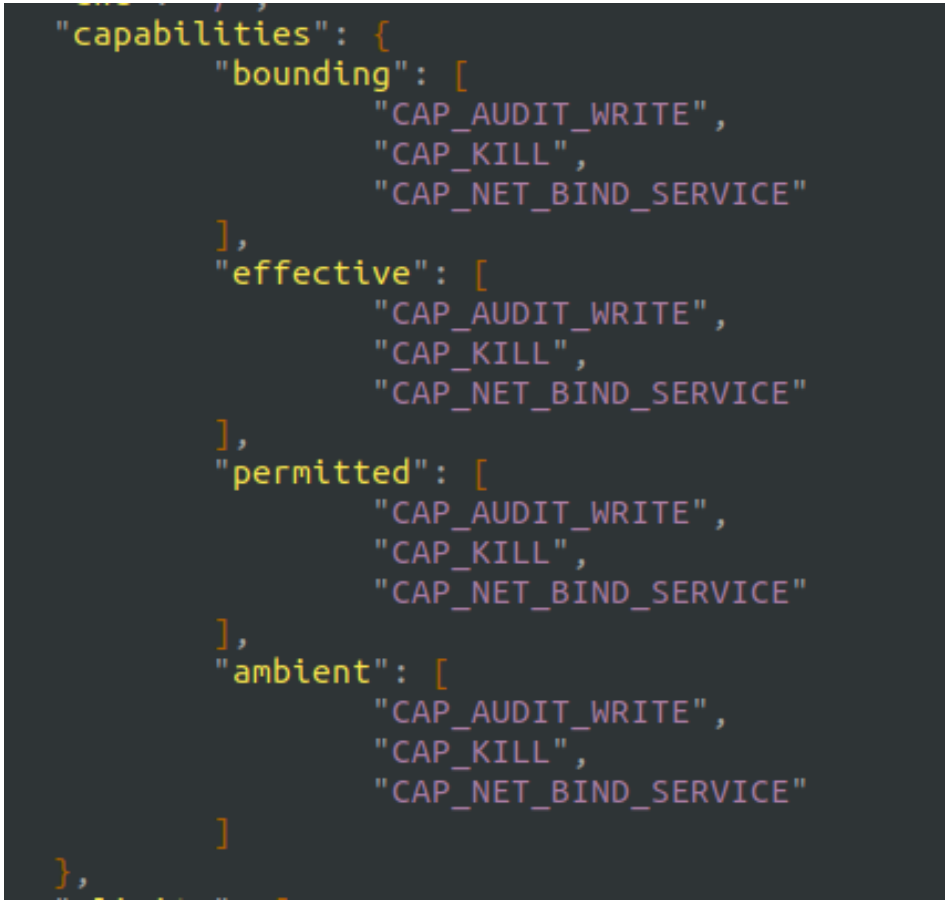


Figure: Einstellungen Capabilities

runc

Example

- runc create mycontainer
- runc run mycontainer
- runc state mycontainer
- runc delete mycontainer

Container status

Example

- runc list

```
ID          PID         STATUS      BUNDLE                                                    CREATE
D                       OWNER
apache      30718       created     /[redacted]/work/privat/luga_LIT_2025/container/apache    2025-0
4-24T07:10:28.606860242Z    root
hello       35373       running     /[redacted]/work/privat/luga_LIT_2025/container/hello     2025-0
4-24T07:17:58.163199287Z    root
```

Figure: Status der laufenden Container

runc execute process in container

Example

- sudo runc exec test echo ”Hello”
- sudo runc ps test

runc

Hello world container

- mkdir hello
- cd hello
- wget http://dl-cdn.alpinelinux.org/alpine/v3.10/releases/x86_64/alpine-minirootfs-3.10.1-x86_64.tar.gz
- mkdir rootfs && tar -xzf alpine-minirootfs-3.10.1-x86_64.tar.gz -C rootfs
- runc spec
- sed -i 's;"sh";"sh -c echo Hallo Augsburg;'
- sudo runc run hello[4]

Container hello world!

Figure: Hallo Welt, Container

runc container with docker

- mkdir nginx
- cd nginx
- mkdir rootfs
- docker export $(docker create nginx:alpine) — tar -C rootfs -xvf -
- runc spec

Container to host connection

setup network to container

- sudo brctl addbr runc0
- sudo ip link set runc0 up
- sudo ip addr add 192.168.10.1/24 dev runc0
- sudo ip link add name veth-host type veth peer name veth-guest
- sudo ip link set veth-host up
- sudo brctl addif runc0 veth-host
- sudo ip netns add runc
- sudo ip link set veth-guest netns runc
- sudo ip netns exec runc ip link set veth-guest name wlan0
- sudo ip netns exec runc ip addr add 192.168.10.101/24 dev wlan0
- sudo ip netns exec runc ip link set wlan0 up

runc rootless container

Example

- runc spec –rootless

Figure: UID/GID Mapping

runc checkpoint container

Example

- runc checkpoint –image-path $(pwd)/image-checkpoint test

- Sichert den aktuellen Stand des containers
- Restore ist möglich

runc hooks

Example

- createRuntime;
- createContainer;
- startContainer;
- poststart hook;
- poststop hook;

[2]

runc and systemd

Systemd service file

Ein einfacher systemd service um einen container zu starten und zu stoppen[4]

```
[Unit]
Description=Start My Container

[Service]
Type=forking
ExecStart=/usr/local/sbin/runc run -d --pid-file /run/mycontainerid.pid mycontainerid
ExecStopPost=/usr/local/sbin/runc delete mycontainerid
WorkingDirectory=/mycontainer
PIDFile=/run/mycontainerid.pid

[Install]
WantedBy=multi-user.target
```

Embedded container - why?

- Why not?
- Isolation
- Weit verbreitet
- Ressourcen Begrenzung

crun

- C basierte runc Alternative
- Kein go overhead
- OCI konform[1]

Comparising of embedded runtime's

- Comparing embedded container runtimes
- Performance Vergleich der verschiedenen runtimes
- Crun wird hier präveriert.[7]

Zusammenfassung

- Container Grundlagen
- Runc Grundlagen
- Beispiele
- Warum sinnvoll im Embedded Kontext
- Ausblick

Embedded Container mit runc

Fragen?

Kontakt

- Linkedin
- michaelestner@web.de


crun.
containers/crun, April 2025.
original-date: 2017-09-13T20:20:58Z.


Mihail Kirov.
mihailkirov/runc-article, June 2024.
original-date: 2022-04-18T21:15:08Z.


mkdev.
runc & OCI Deep Dive: Running Containers Explained | mkdev.


Opencontainer.
opencontainers/runc: CLI tool for spawning and running containers according to the OCI specification.


seccomp.
seccomp(2) - Linux manual page.


James Smith.
Virtual Machines.
Elsevier, 2005.


Simone Weiß and Thomas Irgang.
A Comparative Analysis of Embedded Container Runtimes, 0200.