Engineer

Podman

en containermotor, der ikke behøver en dæmon

Hvad det er

Podman er en containermotor til Linux, der bygger, henter og kører de samme OCI-images som Docker, men uden en central server imellem. Hver `podman`-kommando gør sit arbejde i sin egen proces og afslutter. En startet container kører videre under en lille overvågningsproces ved navn conmon, og den proces er barn af det, der startede den: en skal, et script eller systemd. Projektet kalder selv værktøjet dæmonløst, open source og Linux-nativt og skriver, at de fleste brugere bare kan lave et alias fra `docker` til `podman` og fortsætte som før.

Navnet står for pod manager. En pod er et begreb lånt fra Kubernetes: en gruppe containere, der deler netværksnavnerum og derfor når hinanden på localhost. Podman kan oprette pods direkte, og `podman kube play` starter containere og pods ud fra en Kubernetes-YAML-fil. Det gør værktøjet til et fornuftigt sted at afprøve et manifest, før det kommer i nærheden af en klynge.

To egenskaber skiller det ud i det daglige. Den første er rootless drift. En almindelig bruger kan køre containere, fordi kernens user namespaces gør brugeren til UID 0 inde i containeren og lægger alle andre UID'er ind i brugerens underordnede id-interval fra `/etc/subuid` og `/etc/subgid`. Netværk til rootless containere går gennem pasta, en netværksstak i brugerrummet, eftersom en uprivilegeret bruger ikke må omskrive værtens pakkefilter.

Den anden er Quadlet. En Quadlet-fil er en kort deklarativ beskrivelse af en container, en volume, et netværk, en pod eller et image-build, skrevet i samme ini-syntaks som en systemd-unit og gemt under `/etc/containers/systemd/` for hele systemet eller `~/.config/containers/systemd/` for en enkelt bruger. Ved opstart og ved hver `systemctl daemon-reload` laver en systemd-generator filen `web.container`

om til en helt almindelig `web.service`. Den ældre kommando `podman generate systemd` virker stadig, men projektet har erklæret den forældet til fordel for Quadlet.

Brug begrebet i en beslutning

Brug »Hvad er Podman?« i en konkret beslutning. Vi kan knytte begrebet til det system, I bygger eller køber. Pakket, så den opfører sig identisk på en bærbar, en testserver og i produktion, og skalerer op, når der kommer en travl uge.

Se vores ydelser Kontakt os →

Undersøg antagelserne bag ordet

Brug forklaringen og det tilknyttede arbejde til at formulere spørgsmål til jeres projekt. Vi kan undersøge antagelserne sammen, før et værktøj vælges. Har drevet 20 Docker-apps på Google Cloud og AWS — 20 containeriserede applikationer udrullet og overvåget

Har drevet 20 Docker-apps på Google Cloud og AWS Kontakt os →

Hvorfor det betyder noget

Podman tvinger et spørgsmål frem: hvem ejer procestræet? Docker styrer containerne gennem `dockerd` med egne genstartsregler, egen logning og egne sundhedstjek af containerne. På en server, der allerede kører systemd, er det en ekstra supervisor ved siden af den første, og de to taler ikke samme sprog.

Med Podman og Quadlet er systemd den eneste supervisor. En container er en tjeneste som alle andre. `systemctl status` viser den, journald gemmer dens output, `After=` og `Requires=` placerer den i forhold til den database, den har brug for, og `Restart=` og de cgroup-grænser, systemd allerede kender, gælder uændret. Den, der kan læse en unit-fil, kan også læse containerlaget.

Sikkerheden spiller også ind. Docker-dæmonen kører som root, medmindre man vælger dens rootless-tilstand, og Dockers egen sikkerhedsvejledning siger, at kun betroede brugere bør have kontrol over den, fordi en container kan få adgang til en hvilken som helst mappe på værten. Podman holder ingen privilegeret dæmon kørende mellem kommandoerne. En rootless opsætning strammer yderligere: en proces, der slipper ud af sin container, lander som en uprivilegeret bruger.

Docker vs Podman er derfor mindre en kamp om funktioner end et spørgsmål om, hvad der passer. Docker er det mere ligetil valg på en udviklers bærbare med Docker Desktop, for hold, hvis arbejdsgang er bygget op om Compose, og for værktøjer, der forventer en Docker-socket, selv om Podman også kan levere et

kompatibelt API, når man beder om det. Podman vinder typisk på servere, hvor systemd allerede er sandhedskilden, og hvor en dæmon, der omskriver firewallen, er en belastning snarere end en bekvemmelighed.

Sådan ser det ud i praksis

Podman bærer containerlaget i to nyere projekter, og før det var det fejlsøgningsværktøjet for en flåde bygget på Docker.

På én vært kører containerne som Quadlet-units under systemd med Podman i stedet for Docker. Grunden er firewallen. Dockers standard publicerer containerporte over de regler, hærdningsrollen installerer, mens disse containere bruger værtens netværk helt uden publicerede porte. Units kører på systemniveau og ikke rootless, og siden noterer det som en afvejning: enklere automatisering mod den strengere isolation, rootless ville give, med en note om, hvilken vej man skal gå, hvis containerudbrud en dag vejer tungere. Udrulning er skilt fra provisionering. Root forbereder udrulningsstien én gang. Bagefter bygger en uprivilegeret bruger containerens image og genstarter tjenesten med ét fast script, som en afgrænset regel tillader, og intet andet.

Den samme opbygning fungerer på meget lidt hardware. En vært med én kerne og 512 MB hukommelse kører virksomhedens offentligt tilgængelige TLS-endepunkter, git-forge og onion-tjeneste, mens automatisering af sikkerhedskopiering er implementeret separat. Dens containerlag er Quadlet-units under den samme supervisor som alt andet i stedet for en dæmon med sin egen. Over et registreret døgns stikprøver var den laveste målte tilgængelige hukommelse omkring 310 MB af de 464 MB, der kan bruges.

Tidligere, på tværs af tyve Docker-pakkede applikationer fordelt på Google Cloud og AWS, gik fejlsøgningen gennem Podman, mens Kubernetes tog sig af orkestreringen. Imagene var de samme, værktøjet omkring dem var et valg.

Hvor det går galt

Det meste besvær med Podman opstår, når man forventer, at det opfører sig som Docker eller som en almindelig systemd-unit, for det er lidt af begge.

En rettet Quadlet-fil gør tilsyneladende intet. Generatoren kører ved opstart og ved `systemctl daemon-reload`, så en ændring på disken er usynlig, indtil en af delene sker. Dernæst afviser `systemctl enable` den genererede tjeneste, fordi systemd betragter den som midlertidig. Start ved opstart hører hjemme i filens egen `[Install]`-sektion:

```
# /etc/containers/systemd/web.container
[Unit]
Description=Web front end

[Container]
Image=docker.io/library/nginx:1.27
Network=host

[Service]
Restart=always

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

Når filen er gemt, kører man `systemctl daemon-reload` og `systemctl start web.service`.

En rootless tjeneste stopper, når ejeren logger ud. Bruger-units lever i brugerens egen systemd-manager, som lukker sammen med den sidste session. `loginctl enable-linger` for den bruger holder manageren i live fra opstart, og det kræver enhver rootless tjeneste, der skal overleve en genstart.

En rootless webserver beder om port 80 og bliver afvist. Porte under 1024 kræver root eller kapabiliteten `CAP_NET_BIND_SERVICE`, og den regel gælder også den bruger, Podman kører som. Publicér en høj port og sæt en proxy foran, eller sænk kernens `ip_unprivileged_port_start` med vilje, vel vidende at det gælder alle brugere på værten.

Filer i en bind mount ejes af en fremmed. I rootless-tilstand bliver brugerens eget UID til root inde i containeren, og alle andre UID'er lægges ind i det underordnede interval. En proces, der kører som UID 999 inde i containeren, skriver filer, som værten ser som ejet af UID 100998 eller deromkring. `podman unshare` åbner en skal i den samme mapping, hvor `chown` virker, som containeren forventer, og indstillingen `U` på en mount retter ejerskabet automatisk, så det passer til containeren.

Læs mere

- What is Podman? — projektets egen dokumentation, der begynder med dets definition af værktøjet.
- podman-systemd.unit — opslagsværket for Quadlet: filtyper, søgestier og hver eneste nøgle.
- Basic Setup and Use of Podman in a Rootless environment — underordnede id'er, user namespaces og netværk med pasta.
- podman-unshare — sådan træder man ind i det rootless navnerum for at se eller rette ejerskab.
- podman-generate-systemd — den ældre måde at lave units på og dens udfasning til fordel for Quadlet.

Ordliste

Vælg ud fra jeres faktiske vilkår

Medbring udrulningsforløbet, driftsrisiciene og dem, der har vagten. Vi kan afgrænse en ændring med kontroller, en vej tilbage og tydeligt ansvar.

Se, hvad det koster Kontakt os →

Gør begreberne praktiske

Spørg, hvordan »Hvad er Podman?« gælder i jeres organisation. Et konkret eksempel fra jeres nuværende system er nok til at begynde samtalen.

Se vores ydelser

welcome@engineer.company +45 50 10 61 84 København, Danmark

Tag denne side med dig

Visitkort

Lavet med kærlighed af D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Copyright © 2025 – nu · Engineer ApS · Alle rettigheder forbeholdes.

Udskrevet fra https://engineer.company/da/glossary/podman/