DOCKER / CONTAINER WORKSHOP

En introduksjon til bruk av containerteknologi.

Bouvet backend gruppa, høsten 2023.

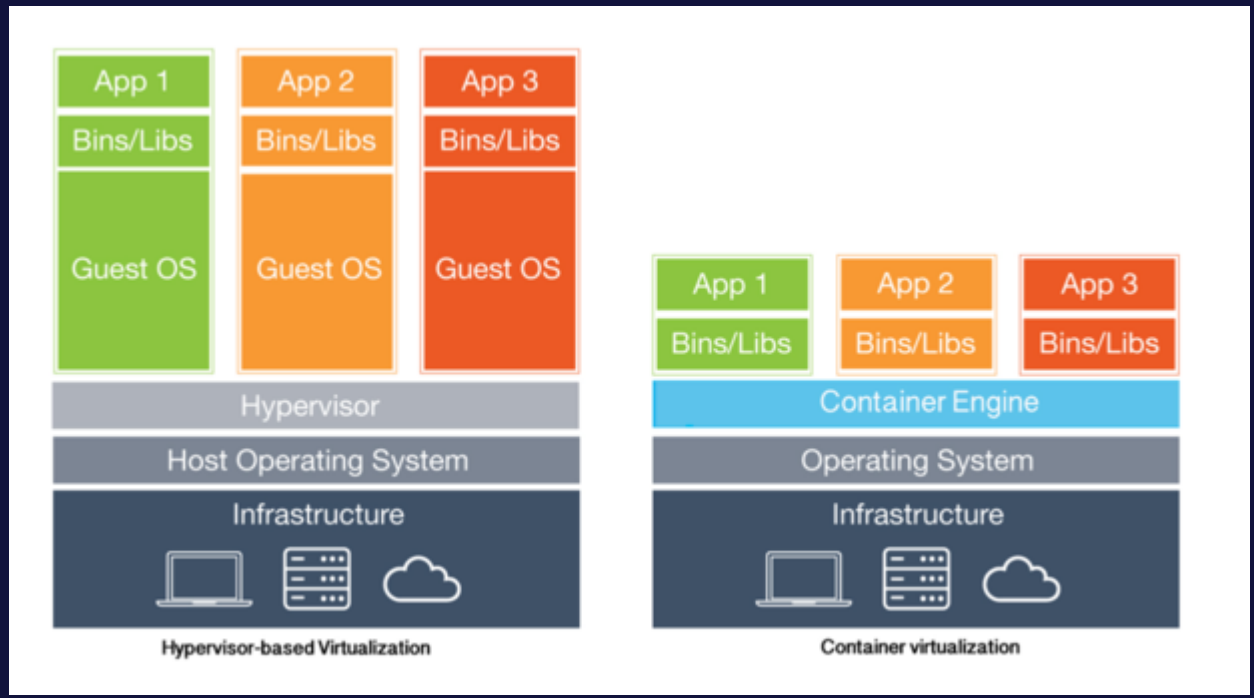
Hva er en container?

Et kjøremiljø for applikasjoner.

Containerteknologi gir oss muligheten til å pakke inn programmer i små containere som deler på maskinvare og operativ systemer.

Fordeler:

- Isolering av applikasjon
- Betydelig bedre sikkerhet
- Dele på maskinvare / nettverk etc.
- Benytte ferdige imager å bygge videre på
- Flyttbart mellom datasystemer.
- Fleksibel for legacy systemer etc.
- Automatisere utrulling av applikasjoner.
- Lagre imager i sentrale lagre.
- Lett å bygge og slette.
- Kort oppstartstid.
- Mulighet for bruk i Kubernetes etc.



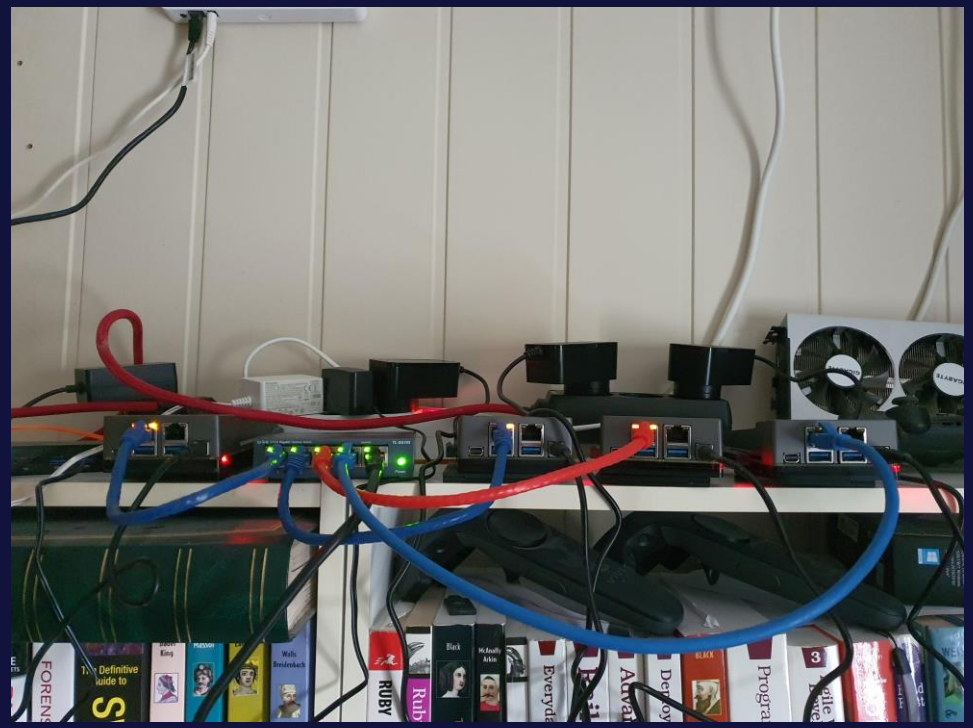
Hvorfor bruke containere?

Å pakketere programmer, tjenester, databaser etc. i små containere, gjør at man kan utnytte maskinvaren mye bedre enn ved å ha separate maskiner eller Virtuelle maskiner for hver applikasjon.

Slike containere kan dele på samme maskin eller kjøres i et cluster av maskiner som f.eks. under Kubernetes.

Da kan man flytte containere mellom noder etter behov, kjøre flere på samme maskin, eller spredt utover flere noder for last balansering.

En stor fordel er at man kan benytte seg av forhåndslagde Container image med os, databaser etc. som man selv kan Bygge videre på. Da slipper man å vedlikeholde os og annen Infrastruktur avhengigheter selv.



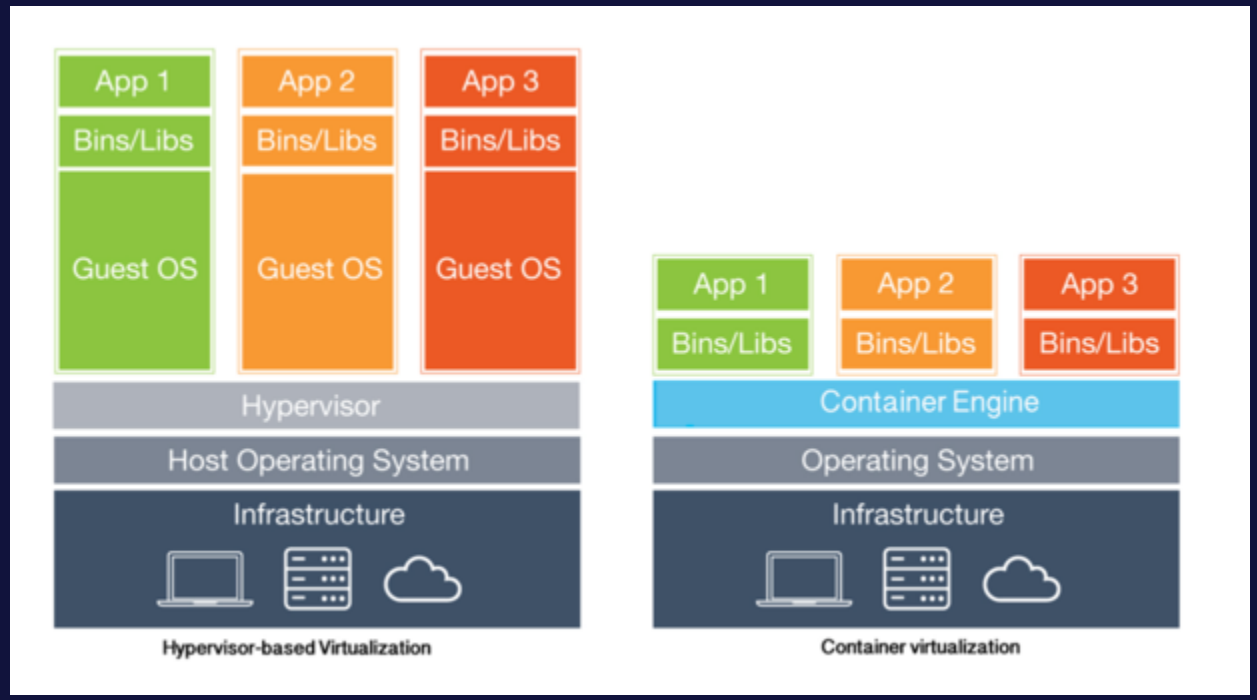
Hva er forskjellen på VM og container?

Container(e) deler på maskinvare, operativsystem og selve container systemet.

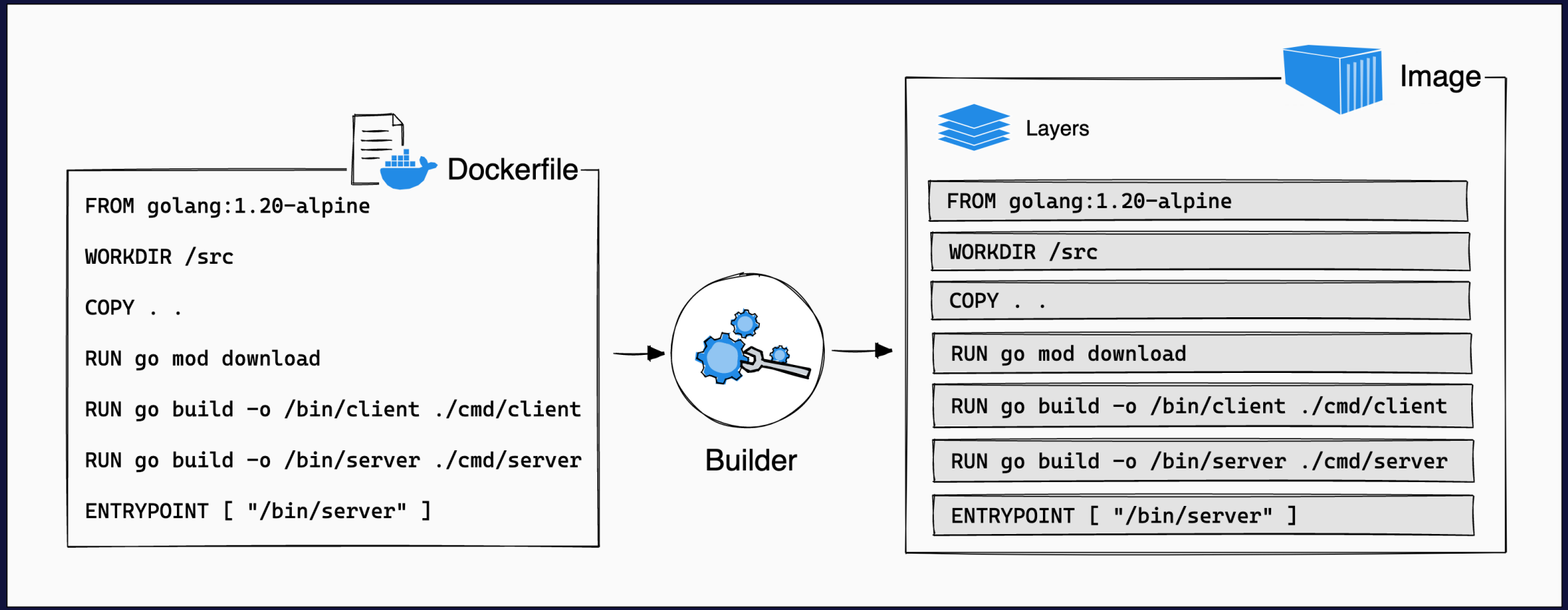
Virtuelle maskiner deler på maskinvaren, men bringer selv operativ systemer, nødvendige komponenter og applikasjonen.

Virtuelle maskiner trenger vesentlig mer maskinvare som minne og lagring. Krever mye mere administrasjon og vedlikehold og starter tregere.

Containere er ofte svært små og bygger på andre imager for nødvendige støtte systemer og deler på operativsystemet og maskinvaren I bunn.



Anatomien til en container



Enkel image script

Et enkelt Python basert eksempel

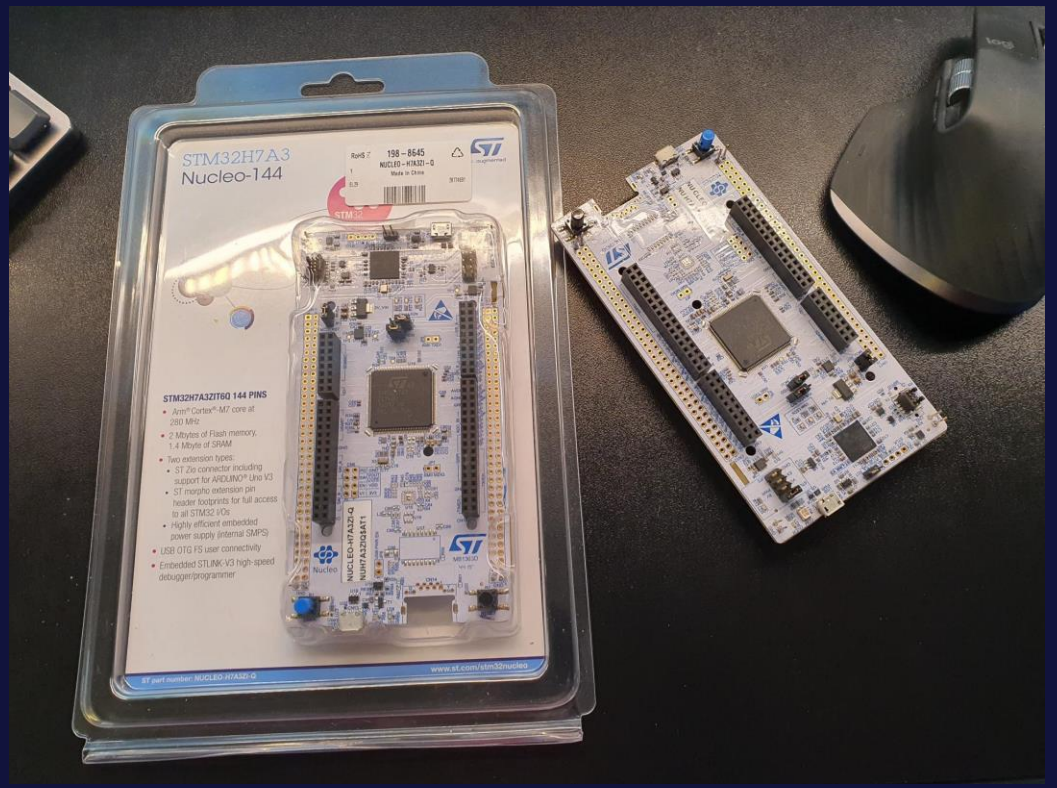
```
from flask import Flask
app = Flask(__name__)

@app.route("/")
def hello():
    return "Hello World!"
```

```
FROM python:3-alpine
RUN python -m pip install --upgrade pip
RUN pip install flask==2.1.*

# install app
COPY hello.py /

# final configuration
ENV FLASK_APP=hello
EXPOSE 8000
CMD flask run --host 0.0.0.0 --port 8000
```

Utviklermiljø i en container

Utvikle for mikro kontrollere fra din egen datamaskin.

- Hvordan utvikle for STM32
  - Egen PC med kryss kompilering.
  - Raspberry PI e.l. med kompilering
  - Skreddersydd container?



bouvet

Utviklermiljø i en container

Bygge og kjøre vår krysskompilering container

- For å bygge en container:
  - podman build -t stenbror/gcc-arm .

- For å kjøre en container i interaktiv modus:
  - podman run --rm -it --privileged -v "$(PWD):/home/app" stenbror/gcc-arm:latest bash

bouvet

Utviklermiljø i en container

Følgende skjer i containeren og til slutt utenfor i Power Shell.

- Start med å gå inn i bygge mappen:
  - cd /home/app/dev
- Sjekk om du har test.c og build.sh filene
  - ls –l
- Kjør bygge skriptet:
  - ./build.sh
- Se om du har en kompilert fil både inni containeren og utenfor.
  - ls –l (container) / dir (Windows maskinen i powershell / Command Line)

Mere informasjon på nettet:

- https://docs.podman.io/en/latest
- https://docs.docker.com
- https://kubernetes.io/docs/home/
- https://linuxcontainers.org
- https://www.proxmox.com/en/