Sisteme Tolerante la Defecte
Containere - Docker

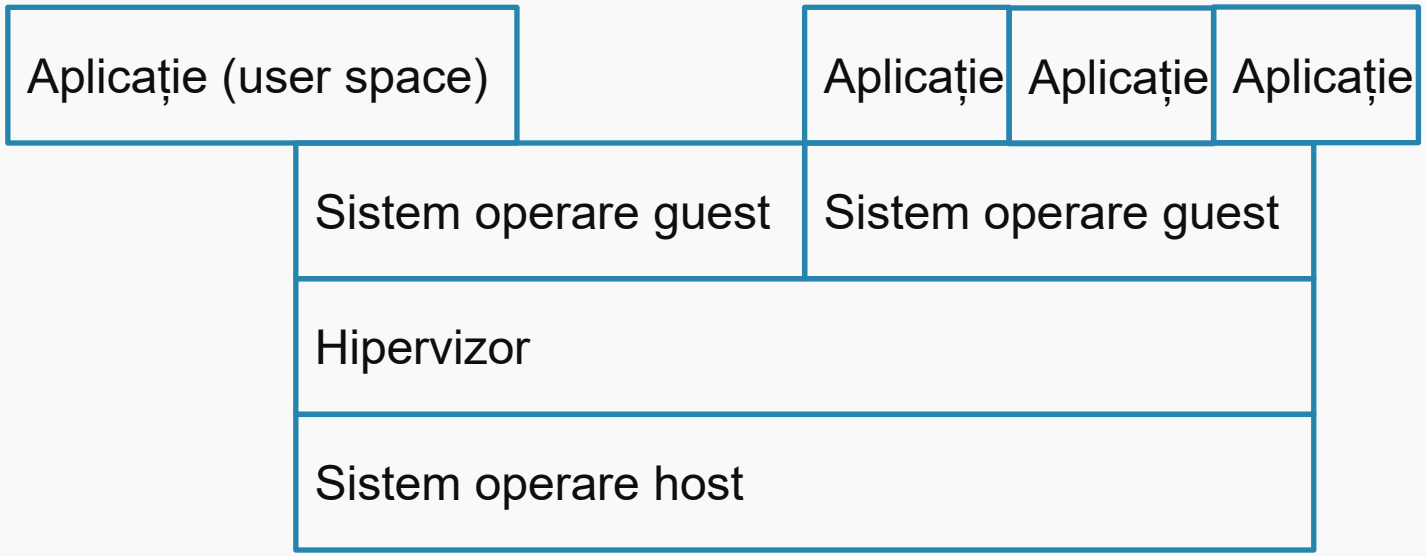
Lect. Dr. Ing. Cristian Chilipirea – cristian.chilipirea@mta.ro

Containere

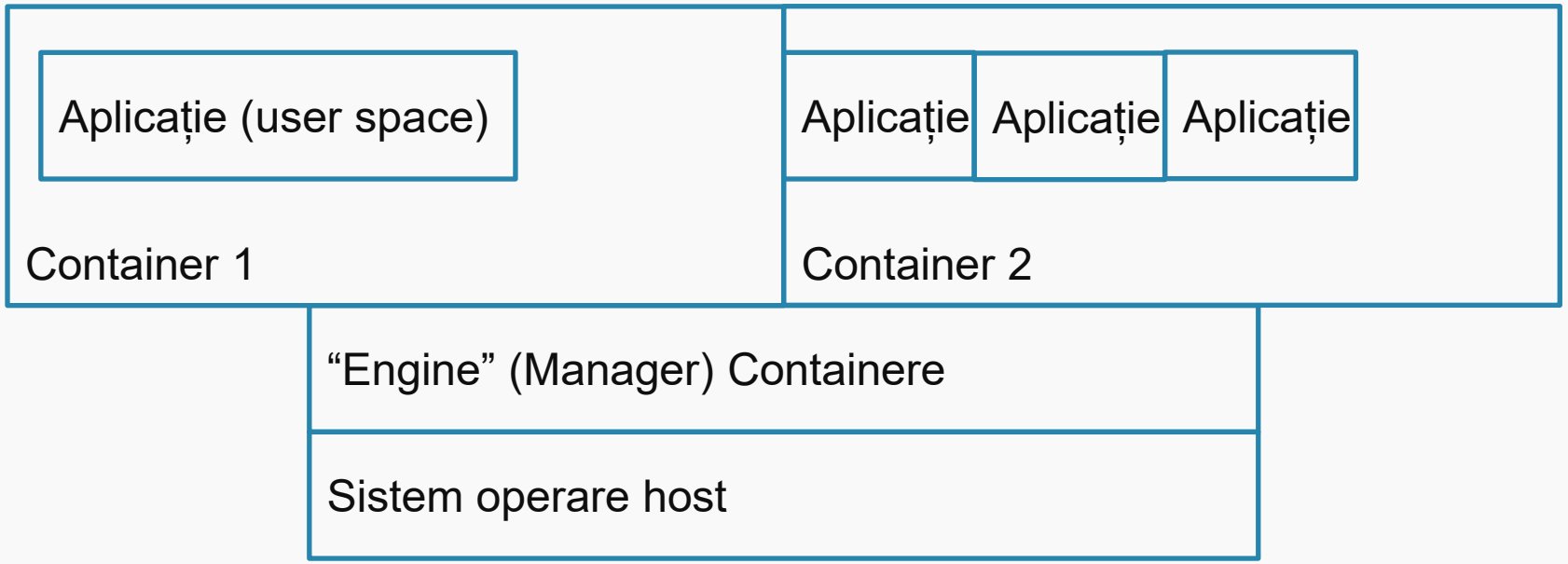
Solomon Hykes

Container vs Mașină Virtuală

Reminder Mașină Virtuală



Containere

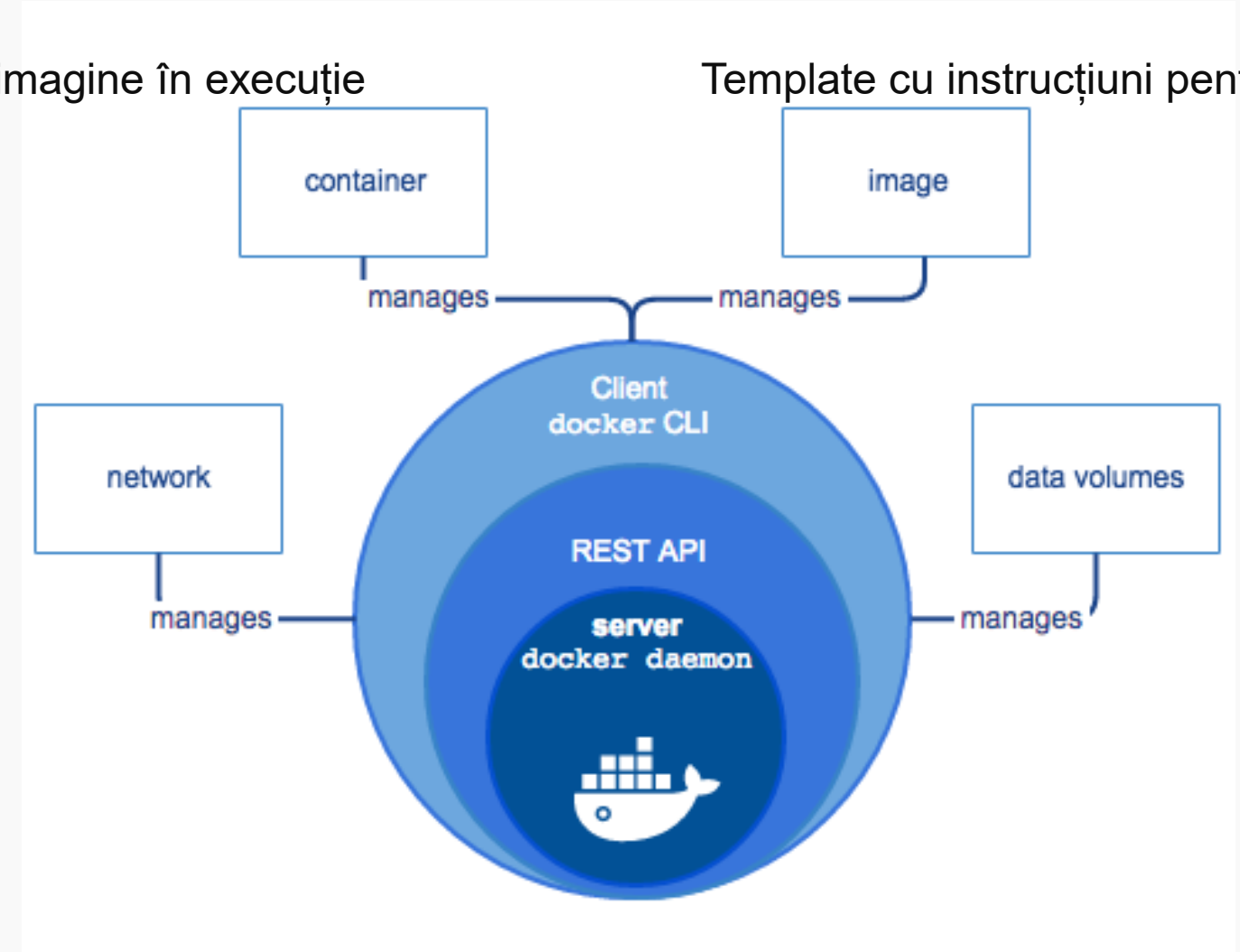

Aplicație (user space)
Container 1
Aplicație
Aplicație
Aplicație
Container 2
“Engine” (Manager) Containere
Sistem operare host


intel
Xeon

Docker Engine

O imagine în execuție

Template cu instrucțiuni pentru rulare



Funcționarea Docker

Namespaces

- **pid** - izolare procese
- **net** - izolare interfețe rețea **(+ emulare)**
- **ipc** - izolare comunicare inter procese (ex: baze de date)
- **mnt** - izolare sistem de fișiere **(+ emulare)**
  - **UnionFS** – Layered File System
- **uts** - izolare kernel, identificatori sistem (ex: hostname)

Control groups (**cgroups**) - Controlează accesul la resurse

- Memorie
- CPU
- Periferice

chroot (different /root directory)

Seccomp/AppArmor/Capabilities

Docker - comenzi

docker build

docker container run -d

docker container exec

docker container ls

docker container logs

docker container start

docker container stop

docker container rm

Dockerfile

Pe baza unui fișier Dockerfile se pot construi imagini Docker

Numele fișierului este în general chiar **Dockerfile**

Fișierul are o listă de comenzi

Pentru fiecare comandă se creează o imagine nouă

- Acest lucru permite construirea unui cache de imagini

Dockerfile – instrucțiuni

FROM baseImage

- Prima instrucțiune din orice Dockerfile
- Determină de la care altă imagine să pornească noua imagine construită (în multe cazuri un sistem de operare, gen **ubuntu**)
- Deși un container **NU** are un sistem de operare guest poate rula sisteme de operare diferite de host
  - (Ex: CentOS peste Ubuntu).
  - Folosește Kernel-ul host.
  - Chiar dacă ar fi versiuni diferite de Kernel în general sistem call-urile sunt compatibile.
  - Aplicațiile specifice OS guest sunt rualte în container.
  - Linux peste Windows (sau invers) se face folosind Mașini Virtuale.

Dockerfile – instrucțiuni

RUN cmd

- Execută comanda **în timpul creării imaginii**.

CMD cmd

- Poate exista doar o dată.
- Acesta va fi programul principal pornit în container.

LABEL name

COPY localFile containerPath

ENV variable=value

Dockerfile – instrucțiuni

EXPOSE port

- E folosit doar ca informare.

WORKDIR path

VOLUME path

ARG var=default

STOPSIGNAL signal

HEALTHCHECK opt **CMD** cmd

Dockerfile - example

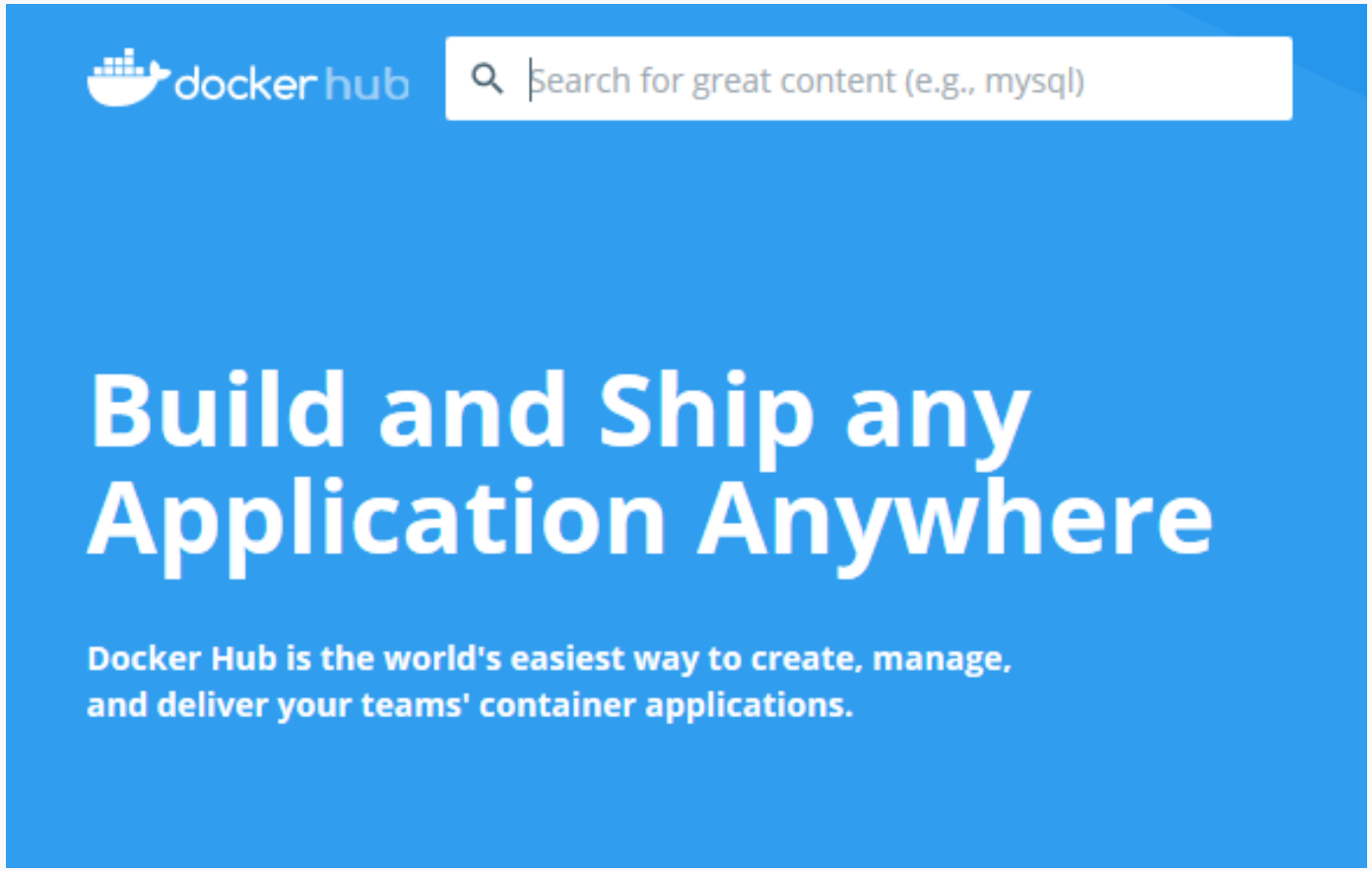
```
# Firefox over VNC
#
# VERSION               0.3

FROM ubuntu

# Install vnc, xvfb in order to create a 'fake' display and firefox
RUN apt-get update && apt-get install -y x11vnc xvfb firefox
RUN mkdir ~/.vnc
# Setup a password
RUN x11vnc -storepasswd 1234 ~/.vnc/passwd
# Autostart firefox (might not be the best way, but it does the trick)
RUN bash -c 'echo "firefox" >> /.bashrc'

EXPOSE 5900
CMD    ["x11vnc", "-forever", "-usepw", "-create"]
```

Dockerfile - example

```
FROM debian

# ARCH is only set to avoid repetition in Dockerfile since the binary download only supports amd64
ARG ARCH=amd64

RUN apt-get update && \
  DEBIAN_FRONTEND=noninteractive apt-get install -y \
    curl \
    unzip \
    jq \
    && apt-get clean \
    && rm -rf /var/lib/apt/lists/*

EXPOSE 19132/udp

VOLUME ["/data"]

WORKDIR /data
```

https://github.com/itzg/docker-minecraft-bedrock-server/blob/master/Dockerfile

Docker Hub



https://hub.docker.com/

Play With Docker

https://labs.play-with-docker.com

Necesită un cont Docker (free)

Se poate să necesite să opriți add-blocker

https://training.play-with-docker.com/

https://www.katacoda.com