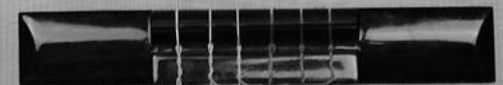


# Docker

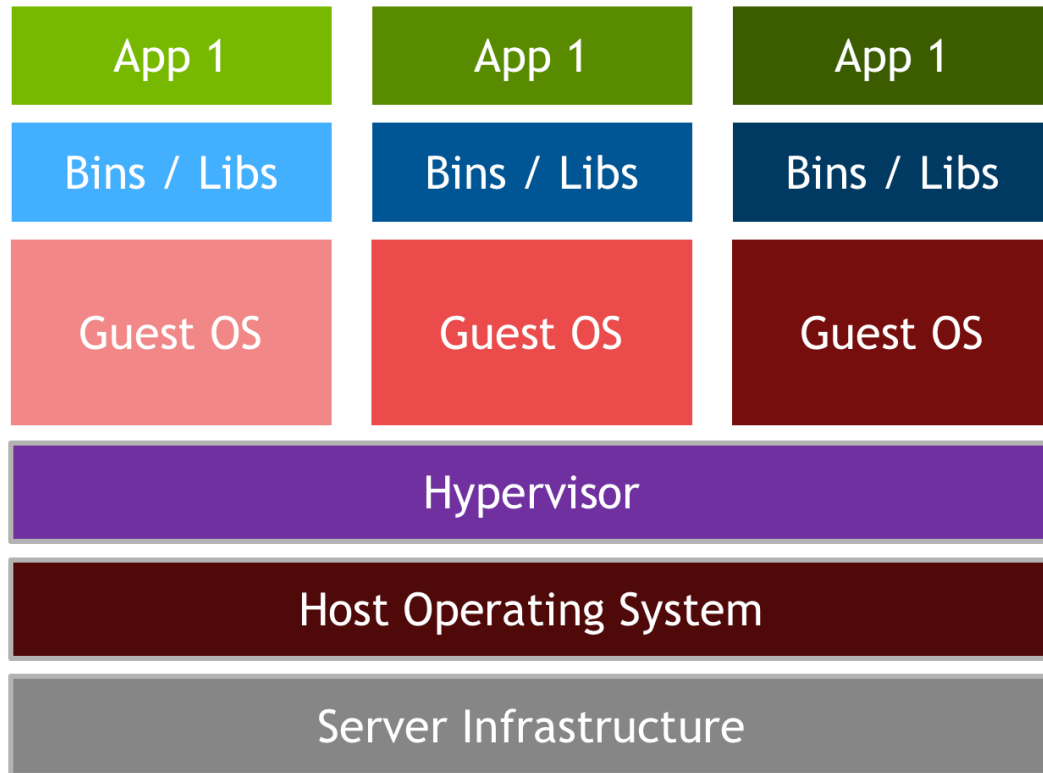


**HWZ** Hochschule für  
Wirtschaft Zürich

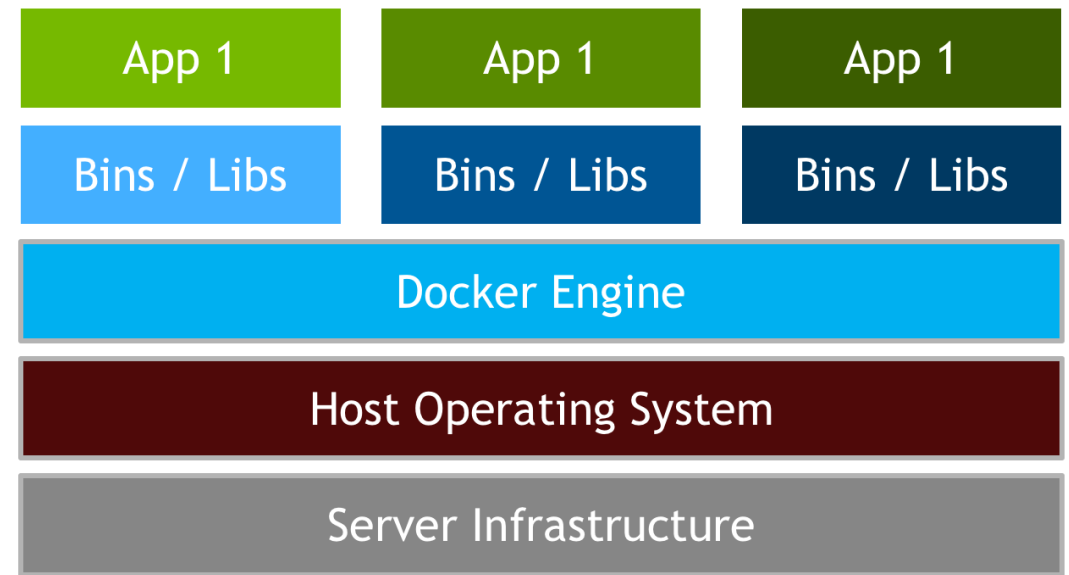


# Docker Eco System

## Virtual Machines versus Containers



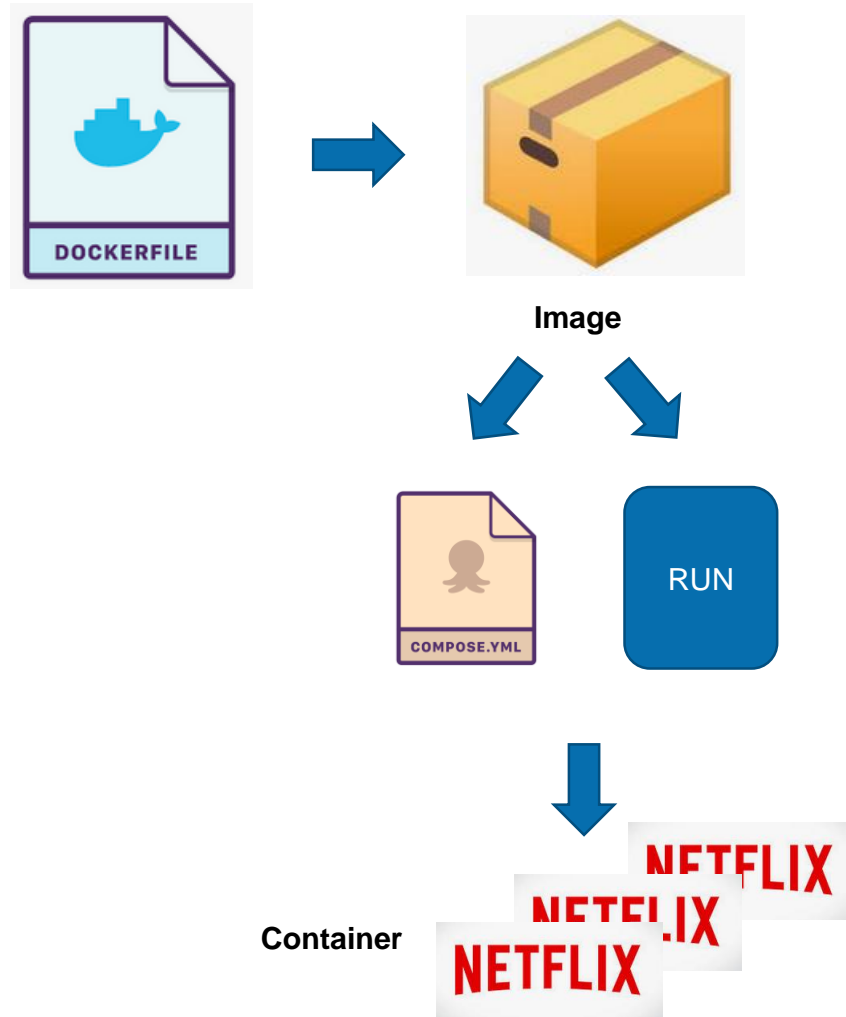
**VIRTUAL MACHINES**



**CONTAINERS**



# Dockerfile => Docker Image => Docker Container



Entwickler erstellt ein Dockerfile (oftmals Dockerfile und einige dazugehörige Dateien)

- `docker build -t hackinglab/alpine-gotty-root`

Mit «docker build» wird aus einem Dockerfile ein Image

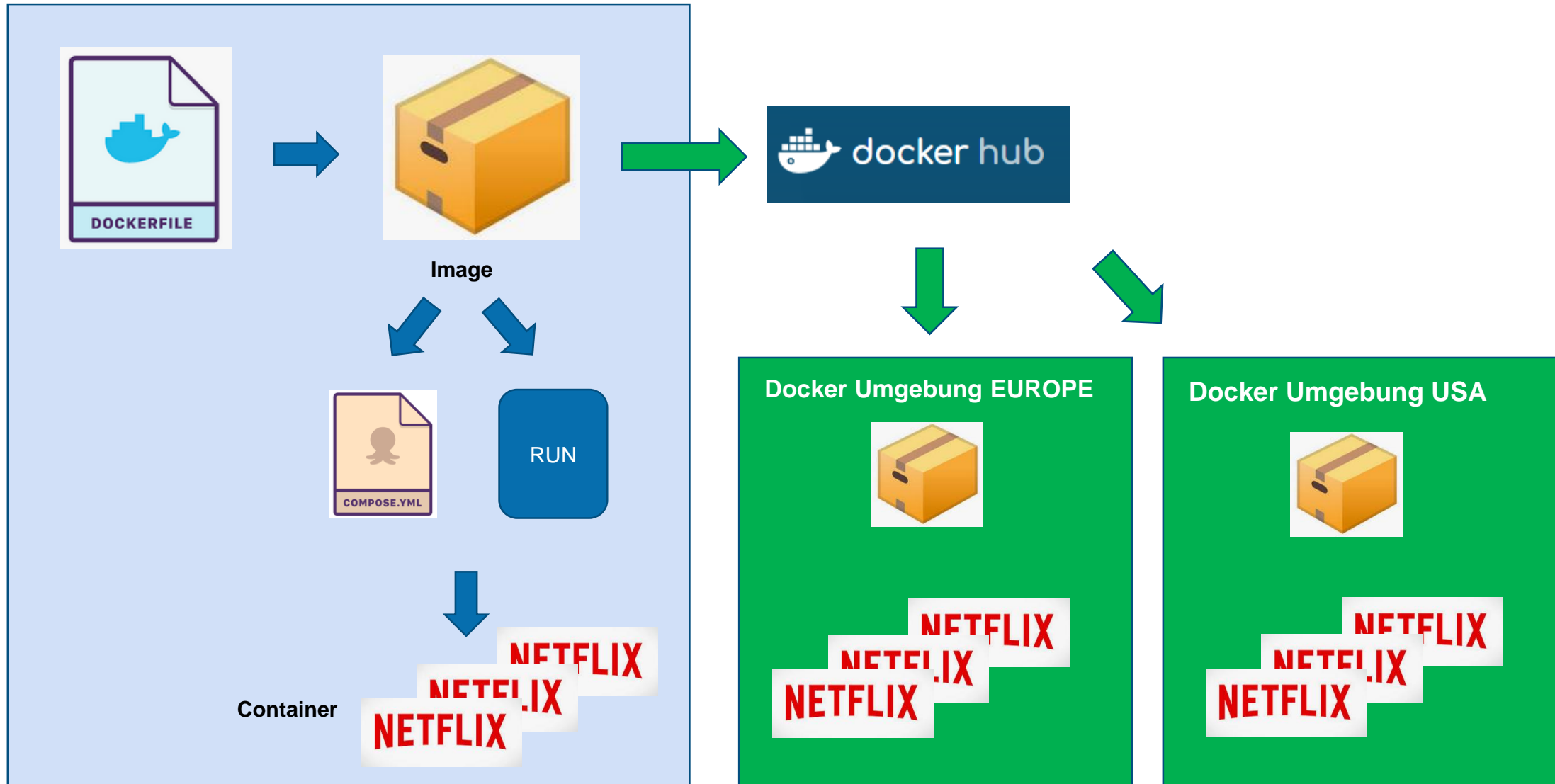
Ein Image das «gebootet» wird, nennt man Container

Es gibt viele Wege um eine Image zu booten

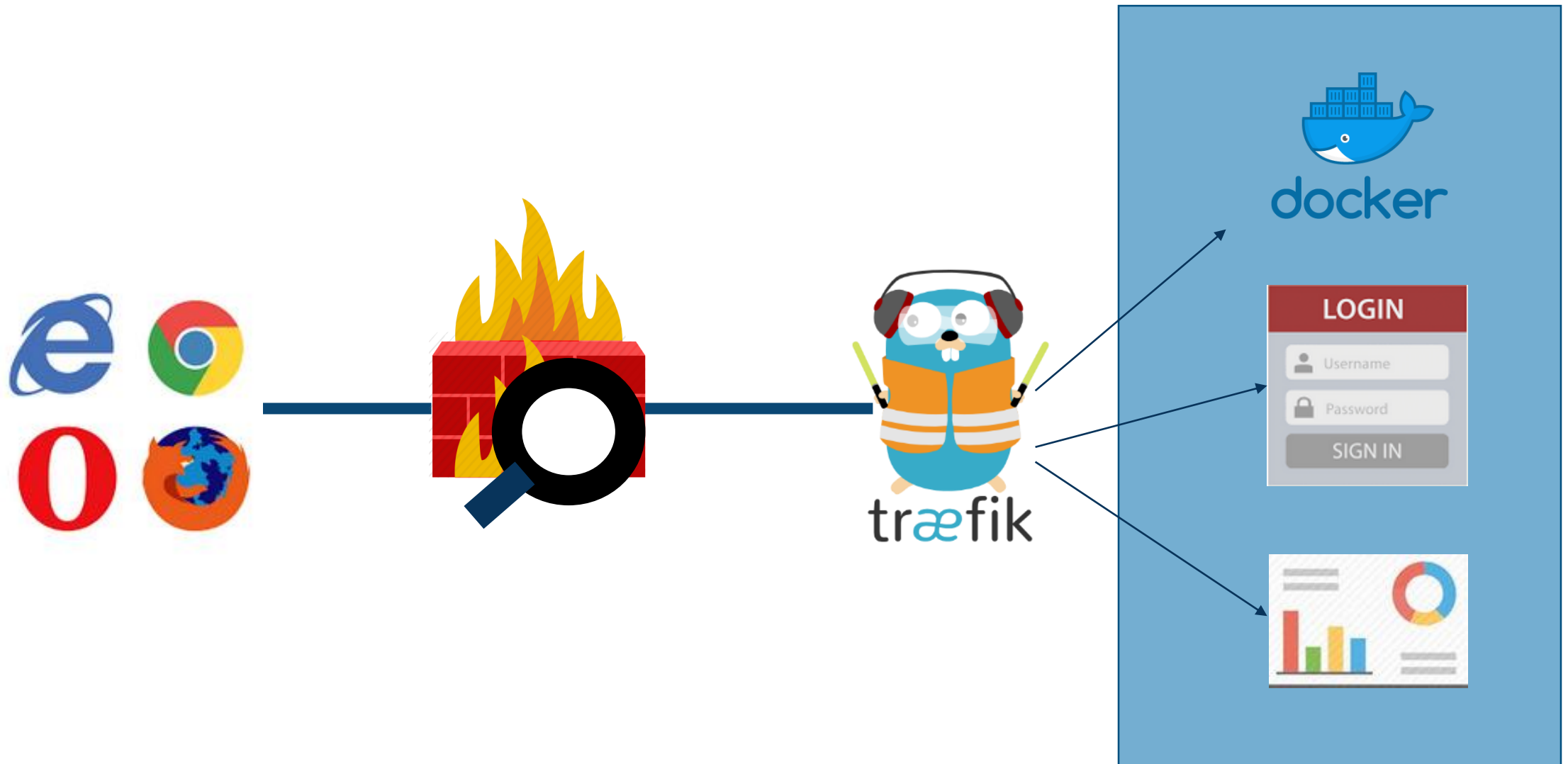
Die einfachsten 2 Varianten für das «booten»

- `docker run hackinglab/alpine-gotty-root`
- `docker-compose up -d`

# Dockerfile => Docker Image => Docker Container

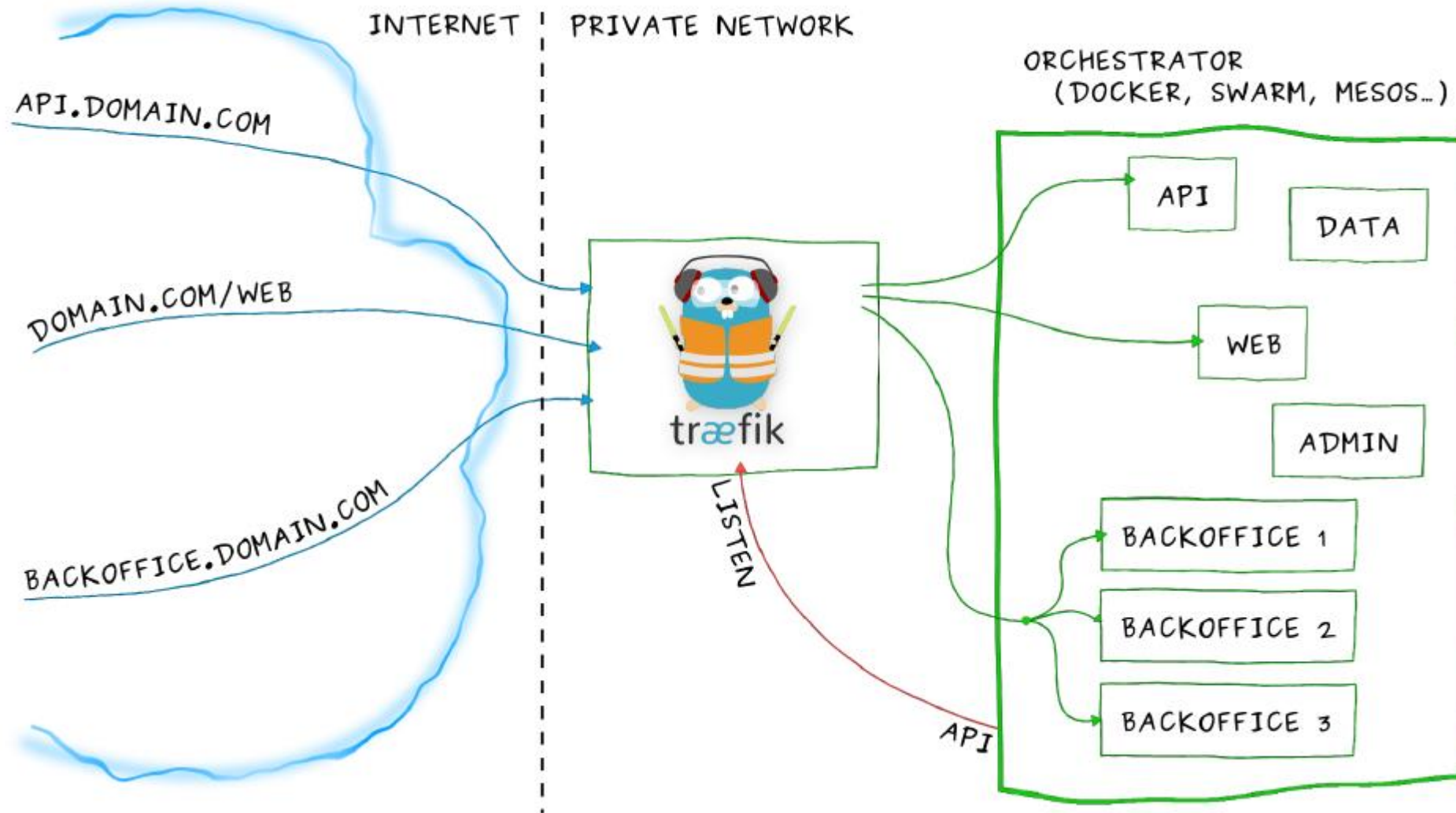


# OnDemand Docker Services



# Traefik Load Balancer

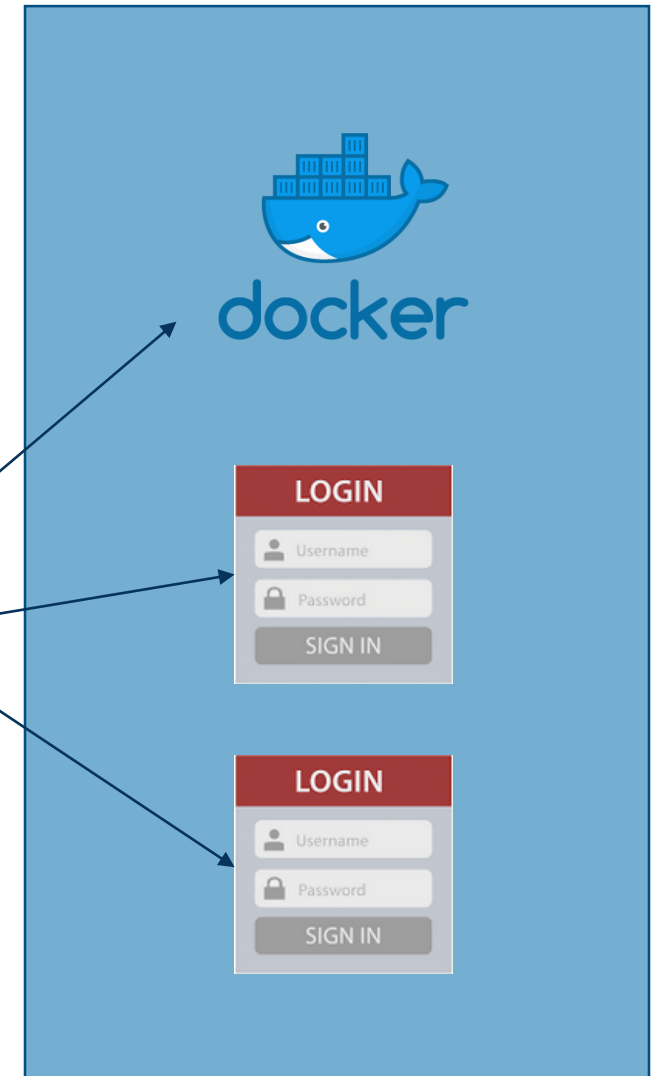
# Microservice Architecture





# Hacking-Lab Test Suite

Wildcard SSL Certificate \*.idocker.hacking-lab.com  
New Routes without restarting the service



# Traefik Status Page - DOCKER

The screenshot displays the Traefik dashboard in a web browser. The browser's address bar shows the URL `idocker.hacking-lab.com:8080/dashboard/`. The dashboard header includes the Traefik logo, navigation links for **PROVIDERS** and **HEALTH**, and version information **V1.7.4 / MAROILLES** with a link to **DOCUMENTATION**.

A search bar at the top of the dashboard content area is labeled "Filter by name or id ...". Below it, two tabs are visible: **docker** (selected) and **file**.

The dashboard is divided into two main sections: **11 FRONTENDS** and **11 BACKENDS**.

**FRONTENDS:**

- frontend-Host-auth-idocker-hacking-lab-com-7**:
  - Route Rule: `Host:auth.idocker.hacking-lab.com`
  - Entry Points: `http` and `https`
  - Backend: `backend-keycloak-authidockerhackinglabcom`
- frontend-Host-devgit-idocker-hacking-lab-com-8**:
  - Route Rule: `Host:devgit.idocker.hacking-lab.com`
  - Entry Points: `http` and `https`
  - Backend: `backend-devgit-devgitidockerhackinglabcom`

**BACKENDS:**

- backend-devgit-devgitidockerhackinglabcom**:
  - Server: `http://172.26.0.5:80`
  - Weight: 1
- backend-globalgit-globalgitidockerhackinglabcom**:
  - Server: `http://172.29.0.2:80`
  - Weight: 1
- backend-keycloak-authidockerhackinglabcom**:
  - Server: (empty)
  - Weight: (empty)

# Traefik Status Page - FILE

The screenshot shows the Traefik status page in a web browser. The browser's address bar displays `idocker.hacking-lab.com:8080/dashboard/`. The page header includes the Traefik logo, navigation links for **PROVIDERS** and **HEALTH**, and version information **V1.7.4 / MAROILLES** with a **DOCUMENTATION** link. A search bar at the top allows filtering by name or ID.

The main content area is divided into two sections: **3 FRONTENDS** and **2 BACKENDS**.

**FRONTENDS:**

- frontend1:** Features a **Route Rule** with `Host: api.idocker.hacking-lab.com` and `PathPrefix: /api`. It has **Entry Points** for `http` and `https`, and is configured to use **Backend** `backend1`.
- frontend2:** Features a **Route Rule** with `Host: idocker.hacking-lab.com` and `PathPrefix: /`. It has **Entry Points** for `http` and `https`.

**BACKENDS:**

- backend1:** A table showing the **Server** `http://localhost:81` with a **Weight** of `10`.
- backend2:** A table showing the **Server** `http://localhost:82` with a **Weight** of `10`.

# Conclusion

Paradigm shift into the direction of “Microservice Architectures”

Small apps together shape the big application

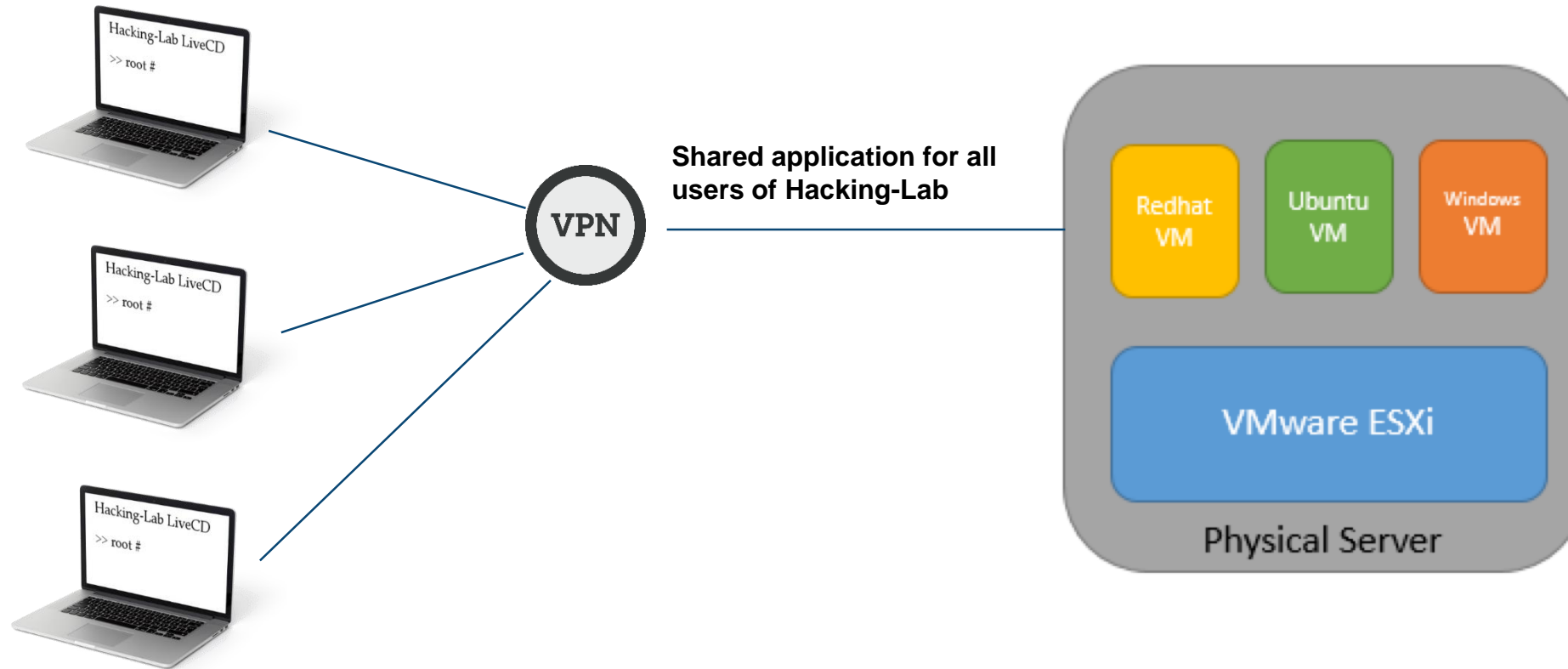
We will face more and more RESTful service architectures

# **Docker & Hacking-Lab 1.0**

## Appendix



# How we deployed vulnerable services in the past (and still do...)



# How we deploy vulnerable services today

