

Container orchestration

More containers!!!



AP HOGESCHOOL
ANTWERPEN

AP.BE

Overzicht

- De probleemstelling
- Container orchestration: the concepts
- Container orchestration: the tools
 - Scripting
 - Docker compose
 - Kubernetes

De probleemstelling

- Veel containers zijn onoverzichtelijk
- Versiebeheer?
- Updates?
- Redundantie?
- Container networking?
- Code / configuratie?

Container orchestration

The concepts

Container orchestration

- Een tool om alles te beheren met:
 - Versiebeheer
 - Networking
 - Redundantie
 - Opslag

Container orchestration: Versiebeheer

- Configuratie in een tekst bestand
- Eenvoudige manier om nieuwe versie te pushen

Infrastructure as Code

- DevOps mantra: alles automatiseren + in Git bijhouden
- Dus ook deployments
- 2 flavors: declarative vs imperative
 - Declarative: uitkomst vastleggen (desired state)
 - Imperative: commando's vastleggen
- Gebruik declarative waar mogelijk
- Nadeel: learning curve

Netwerken bouwen

- Containers zijn geïsoleerd van het host systeem = ze kunnen er niet mee communiceren
- Enkel omgekeerd door port binding
- Wat als je een databank connectie wil maken vanuit je container?
- Oplossing: 2 containers in hetzelfde (Docker) netwerk

Container orchestration: Networking

- Netwerken tussen containers zijn gedefinieerd
- DNS
- Isolatie
- Open te stellen poorten

Container orchestration: Redundantie

- Containers deployen op meerdere hosts
- Containers automatisch herstarten
- Meerdere identieke containers draaien

Container orchestration: Opslag

- Volumes voorzien om data in op te slaan
- Volumes delen tussen containers

Container orchestration

The tools

Container orchestration

- Gewenste parameters instellen (declarative)
- Orchestration middleware zorgt voor de rest
- Platformen
 - Docker Compose = same host
 - Docker swarm = orchestration on multiple hosts
 - Kubernetes (K8s) = orchestration on multiple hosts

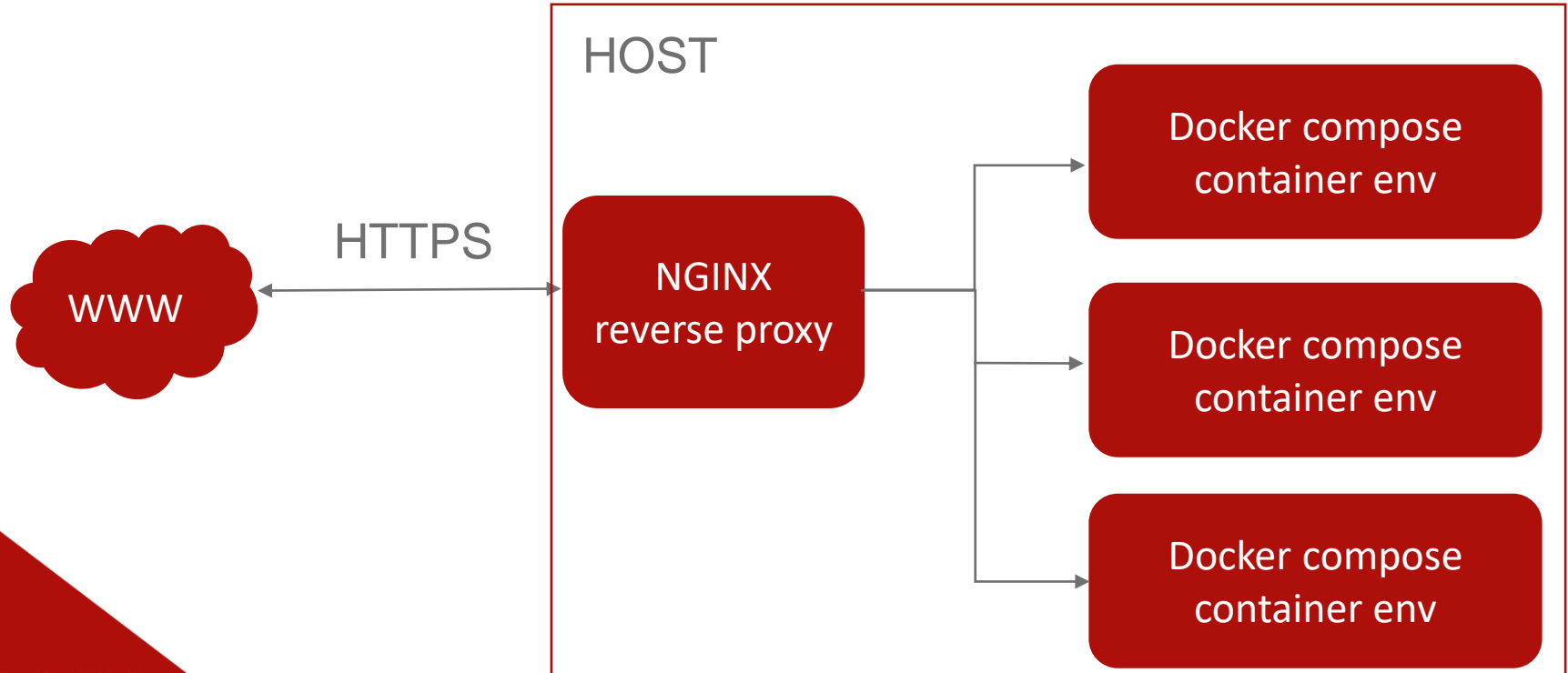
Docker compose

- Single host container orchestration
- Geen remote api
- Support voor docker volumes
- Container networking/DNS
- Geïsoleerde omgeving
- Container dependencies

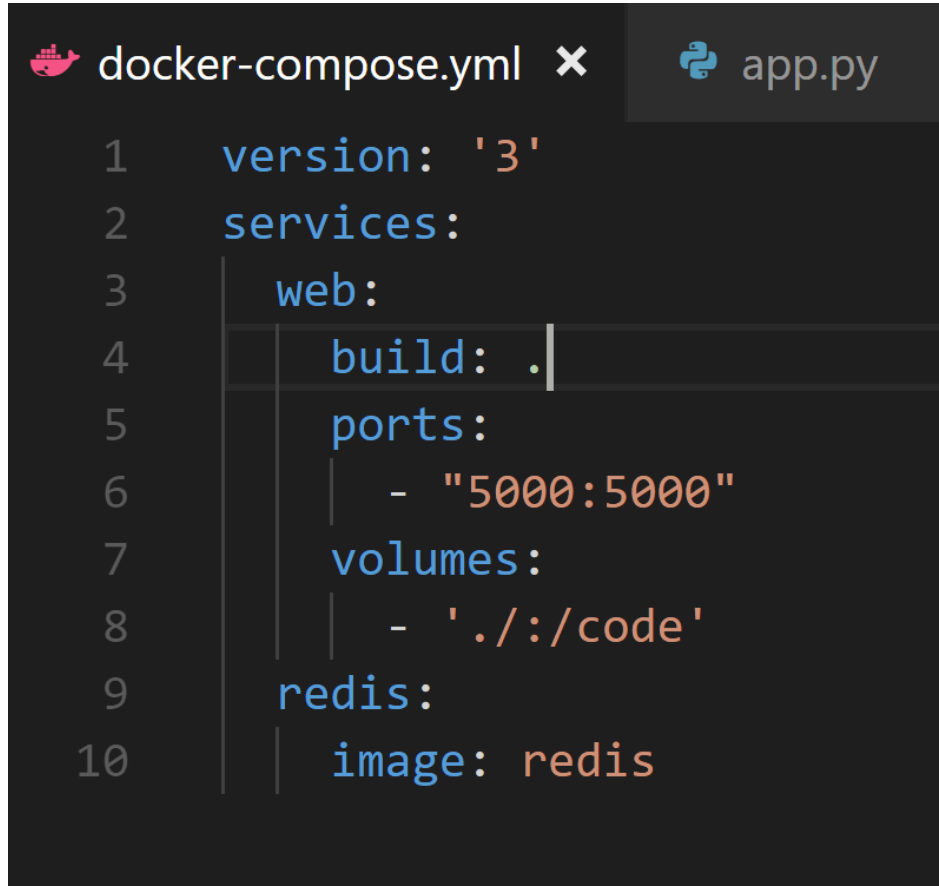
Docker compose

- docker-compose.yml bevat:
 - Te starten containers
 - Container networking
 - Open te stellen poorten
 - Container dependencies
 - Docker volumes
- Syntax: <https://docs.docker.com/compose/compose-file/>

Docker compose



Docker compose sample



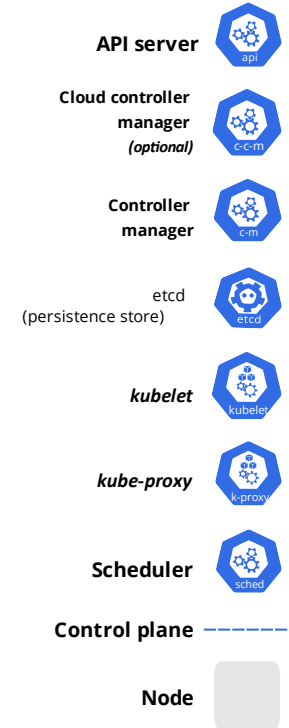
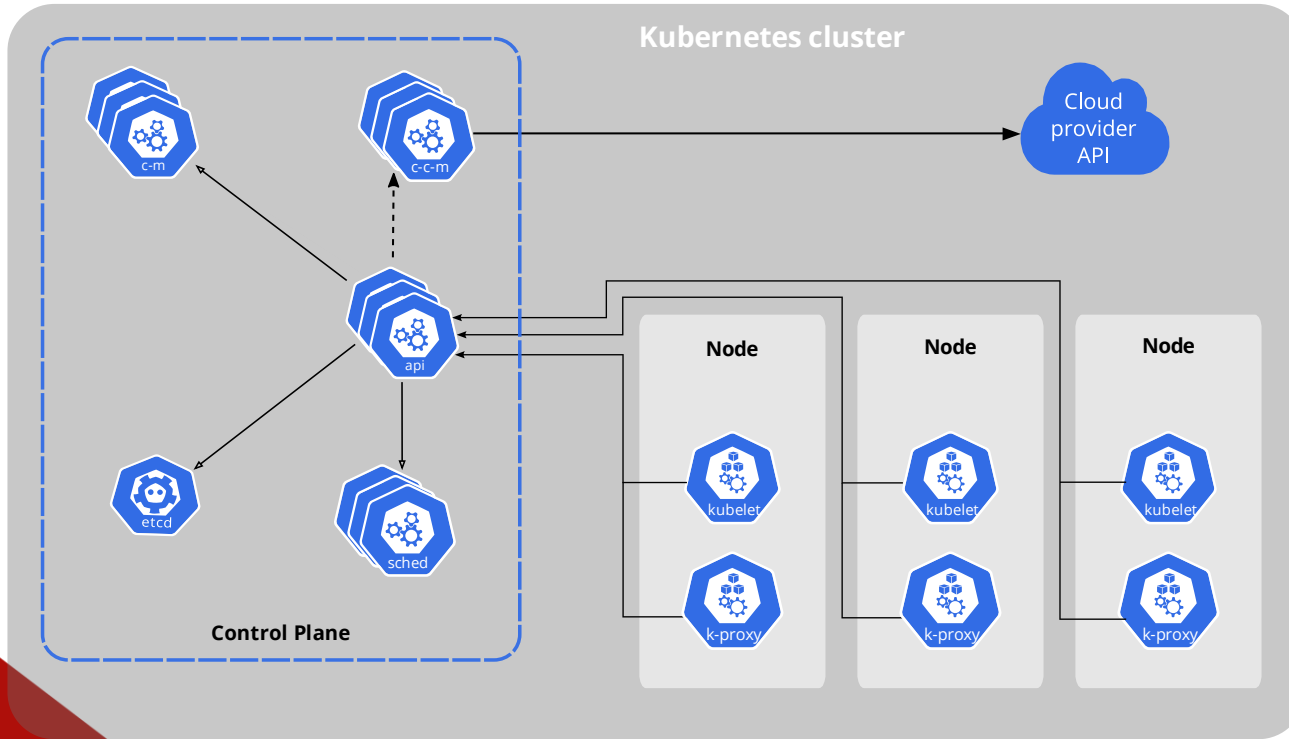
The screenshot shows a code editor with two tabs: 'docker-compose.yml' (with a Docker icon) and 'app.py' (with a Python icon). The 'docker-compose.yml' tab is active, displaying a YAML configuration for a Docker Compose setup. The code is as follows:

```
1  version: '3'
2  services:
3    web:
4      build: .
5      ports:
6        - "5000:5000"
7      volumes:
8        - './:/code'
9    redis:
10     image: redis
```

Kubernetes (K8s)

- Container orchestration cluster
- Minstens 3 servers
- Remote management API
- Zeer uitgebreide keuze ondersteunde volumes
- Aangeboden als PAAS oplossing

Kubernetes basic concepts



Kubernetes sample

```
kind: Deployment
apiVersion: extensions/v1beta1
metadata:
  name: hostname-101-deployment
spec:
  replicas: 3
  selector:
    # Like saying "Make sure there are three pods running
    # with the label app = hostname and version = v101"
    matchLabels:
      app: hostname
      version: v101
  template:
    metadata:
      labels:
        # The `app` label is used by both the service
        # and the deployment to select the pods they operate on.
        app: hostname
        # The `version` label is used only by the deployment
        # to control replication.
        version: v101
    spec:
      containers:
        - name: nginx-hostname
          image: kubegoldenguide/nginx-hostname:1.0.1
          ports:
            - containerPort: 80
```

Docker compose

A practical guide

Docker compose basics

- docker-compose + windows = weirdness
- Mapstructuur
 - docker-compose.yml
 - container1/
 - Dockerfile
 - ...
 - container2/
 - Dockerfile
 - ...

docker-compose.yml

- Opbouw in code
 - Definitie netwerken en poorten
 - Definitie volumes
 - Koppeling volumes en containers
 - Koppeling tussen containers
 - Container dependencies
- Indentatie is belangrijk!! 4 x space != tab

docker-compose.yml instructies

version: "X.X"

Bevat de versie van de gebruikte syntax in het docker-compose.yml bestand.

docker-compose.yml instructies

services:

Onder deze blok kan je services definiëren. Een service is een container instantie die kan gestart worden van een bestaande image of kan gebouwd worden van een Dockerfile.

docker-compose.yml instructies

<service-naam>:

Definieert een service onder de services blok en geeft deze een naam. Aan de hand van deze naam kan je in andere services naar deze service verwijzen. Deze naam is ook beschikbaar voor alle services binnen deze deployment via DNS.

Elke service bevat op zijn minst een build of image instructie.

docker-compose.yml instructies

image: <image-naam>

Selecteert de image die wordt gebruikt om de service op te bouwen.

docker-compose.yml instructies

build: <path/to/build/dir>

Geeft aan welke map er moet gebouwd worden om de service te bouwen. Deze map moet een Dockerfile bevatten.

docker-compose.yml instructies

volumes:

- `<volume-name>:/map/in/container`
- `/map/op/host:/map/in/container`

Als deel van een service:

Koppelt volumes/mappen op de host aan mappen in de container

Als top level blok:

Definieert volumes voor gebruik met containers

docker-compose.yml instructies

ports:

- **<address>:<port>:<container-port>**

Forward een poort van de container naar een adres op de host. Voor security purposes best localhost adres meegeven.

docker-compose.yml instructies

depends_on:

- **<service-name>**

Geeft aan dat een andere service eerst moet gestart zijn voor deze service online kan komen

docker-compose.yml voorbeeld

```
version: "3.9"
services:
  web:
    image: nginx:alpine
    volumes:
      - mydata:/data
      - ./static:/opt/app/static/
    depends_on:
      - db

  db:
    image: postgres:latest
    volumes:
      - "/var/run/postgres/postgres.sock:/var/run/postgres/postgres.sock"
      - "dbdata:/var/lib/postgresql/data"

volumes:
  mydata:
  dbdata:
```

docker-compose command

docker-compose up

Bouwt en start alle services

Opties:

-d voer uit in de achtergrond

docker-compose command

docker-compose build

Bouwt alle services

Opties:

--force-rm verwijder alle tussencontainers

--no-cache herbouw alles

docker-compose command

docker-compose kill

Stopt alle services

docker-compose command

docker-compose rm

Verwijdert alle services

Opdracht

What to do

opdracht

- Basis app:
 - <https://github.com/DevOpsAP-22-23/BaseApp>

opdracht

- Voorzie een werkende docker-compose file voor de app. Zorg ervoor dat de juiste poorten geforward worden.
- Bij gebruik voorziene code zie architectuur volgende slide
 - nginx.conf is voorzien

architectuur

Container orchestration

