Workshop

Docker

For Software Developers

Rainer Stropek

software architects gmbh

Web http://www.timecockpit.com
Mail rainer@timecockpit.com
Twitter @rstropek

time cockpit
Saves the day.

Your Host

Rainer Stropek

Developer, Entrepreneur
Azure MVP, MS Regional Director
IT-Visions

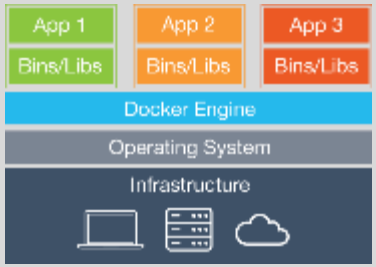
Contact

software architects gmbh
rainer@timecockpit.com
Twitter: @rstropek

What is Docker?

Introduction

Virtual Machines



Docker Container

What is Docker?

Virtual machines vs. Docker

Each VM runs its own guest operating system

Container reuse the host operating system

Container run in user space

https://www.docker.com/whatisdocker/

What's Docker?

Container virtualization

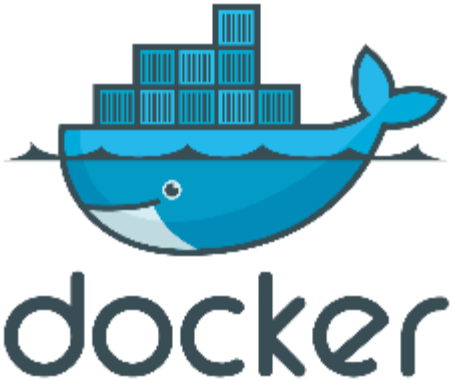
Container run in user space and use kernel of host
Has been existing in Linux for quite a while
Docker builds on Linux Containers (LXC) and makes it easy to use and consume

Advantages?

Fast, small, and agile (e.g. Docker in Docker)

Disadvantages?

Security (less isolated)



What's Docker?

Command line tool, REST services
Docker client can manage remote Docker daemon

Container packaging format

Dockerfiles for image creation from source code

Version management for images
Images can be based on images

Docker Hub: Platform to exchange images and Dockerfiles
Publishing on Docker Hub is not in scope of this talk

Docker Tools

Introduction

Docker and Microsoft

Docker Toolbox

Docker environment for Windows and Mac incl. VirtualBox
Docker Machine: Support for Hyper-V and Microsoft Azure

Container virtualization in Windows

Announced for next version of Windows Server
Windows Containers Quick Start

Use Azure to play with Docker

Existing VM image (Docker on Ubuntu server) in Azure marketplace
Use Docker container to run Azure tools (e.g. https://hub.docker.com/r/microsoft/azure-cli/)

Visual Studio DevOps Tooling

Docker Extension for Visual Studio Code

https://code.visualstudio.com/Docs/languages/dockerfile

Visual Studio 2015 Tools for Docker

https://visualstudiogallery.msdn.microsoft.com/0f5b2caa-ea00-41c8-b8a2-058c7da0b3e4

Step-by-step description for deploying an ASP.NET Web App

Demo

Docker in Azure

Ubuntu server with Docker in Microsoft Azure
Azure Docker Extension

ARM Template
https://github.com/rstropek/DockerVS2015Intro/tree/master/dockerDemos/00-AzureARM

Docker Machine

Documentation

https://docs.docker.com/machine/overview/

Important Commands for Docker Machine

`docker-machine create` – Create a machine
`docker-machine ls` – Lists machines
`docker-machine config` – Print the connection config
`docker-machine start`/`stop` – Restarts/stops a machine
`docker-machine rm` – Removes a machine
`docker-machine ssh` – Log into or run a command on a machine using SSH
`docker-machine scp` – Copy files using *scp*
`docker-machine env` – Set environment variables to make Docker use a machine

Demo

Docker in Azure

List Docker Machines

Create machines
- Hyper-V
- Azure

Create containers on machine

Remove machines

Docker Cluster Solutions

Docker Swarm

https://docs.docker.com/swarm/overview/

Native clustering for Docker, turns a pool of Docker hosts into a single, virtual Docker host

Apache Mesos and Docker

http://mesos.apache.org/documentation/latest/docker-containerizer/

Azure Container Service

https://azure.microsoft.com/en-us/services/container-service/

Set of templates to deploy *Apache Mesos* or *Docker Swarm* into Azure

Access Docker Remotely

Default: Docker runs on non-networked Unix socket

TCP socket can be enabled (see Docker docs)

Docker available on the network → enable TLS

Docker docs

Container

Working with containers

Docker CLI

Documentation

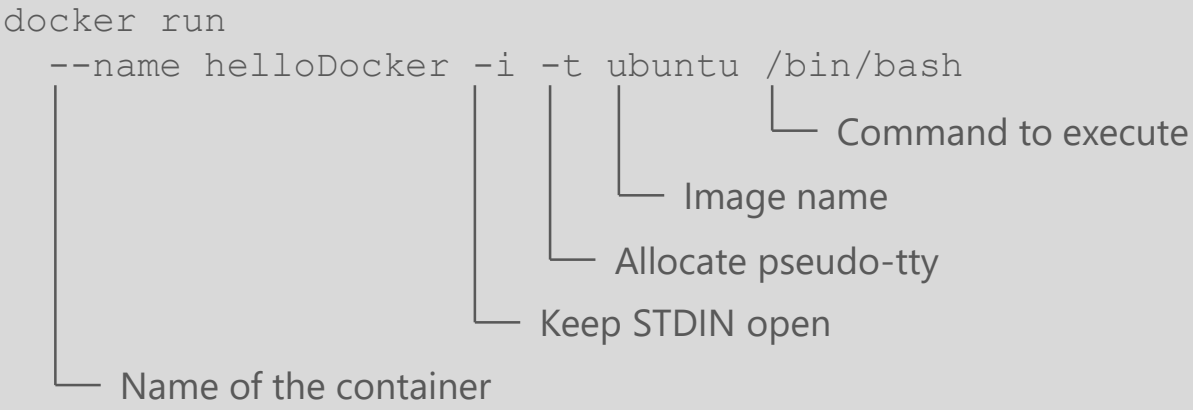
http://docs.docker.com/reference/commandline/cli

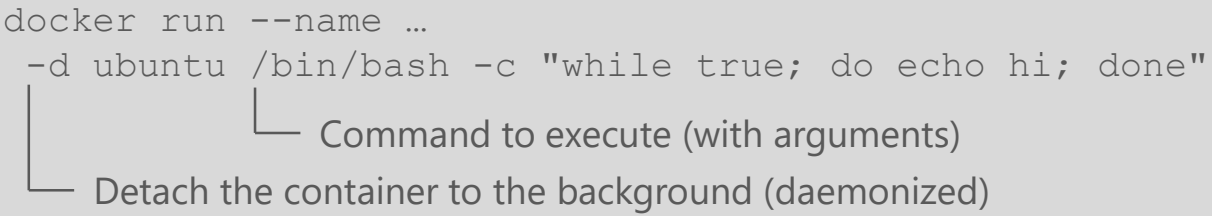
Important Commands for Containers

`docker run` – Run a command in a new container
`docker ps` – List containers
`docker start/stop` – Restarts/stops a container
`docker rm` – Removes container(s)
`docker attach` – Attach to running container
`docker top` – Display processes running in container
`docker exec` – Run a command in a container

Docker CLI

Starting Containers





Interactive container

Daemonized container

Running in the background

`--rm` removes container when it exits

Docker Events

Docker reports real time events from the server

`docker events`

Usages

Admin and monitoring purposes

Triggering auto-configurations (e.g. load balancer configuration with Interlock and Nginx)

Networking

Docker Networking

Networks

By default, three networks

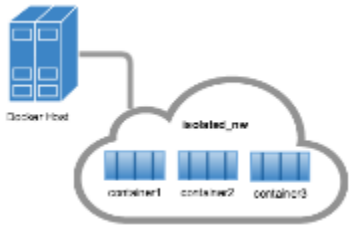
none, *host*, *bridge* (default)

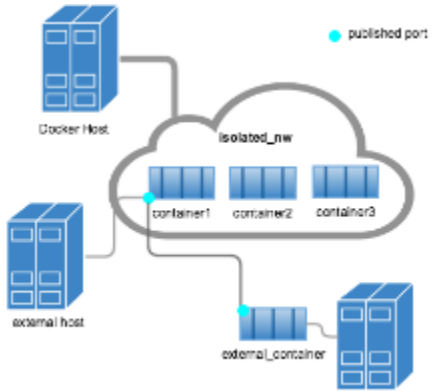
Additional networks can be created

Bridge network = single host

Overlay network (advanced topic, see Docker docs) can include multiple hosts

Network isolation





For details see https://docs.docker.com/engine/userguide/networking/dockernetworks/

Networks

```
# List all networks
docker network ls

# Inspect network details
docker network inspect bridge

# Disconnect a container from network
docker network disconnect bridge mycontainer
```

Container name

Network name

```
# Connect a container to a network
docker network connect mynetwork mycontainer

# Create own network
docker network create -d bridge mynetwork
```

Network name

Driver name

```
# Start container in a specific network
docker run -it --net=mynetwork ubuntu
```

For details about network security, see Docker docs

```
# Start nginx web server on a custom network
docker run -d --net mynetwork --name web nginx
```

Container name in DNS

```
# Start Ubuntu client in same network
docker run -it --net mynetwork --name client ubuntu

  # Ping web server
  ping web

  # Install curl and access web server
  apt-get install curl
  curl web

# Start Ubuntu container and link it using alias
docker run -it --net mynetwork --link=server3:nginx ubuntu
```

Container-specific link

DNS

Docker daemon contains embedded DNS server

For details see
https://docs.docker.com/engine/userguide/networking/configure-dns/

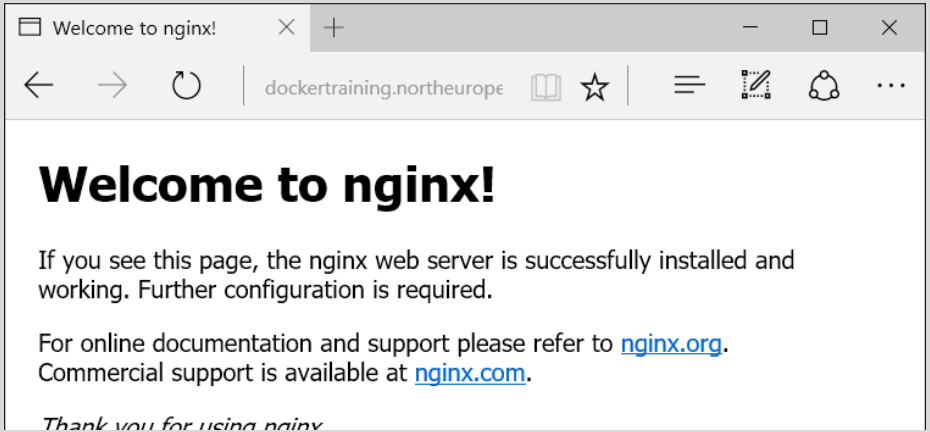
```
docker run -d --net bridge -p 8080:80 nginx
```

Host port — Container port

```
# Start nginx web server on host network
docker run -d --net host nginx
```

Assign container to *host* network

```
# Nginx is now available on the public internet:
```



Binding container ports to host

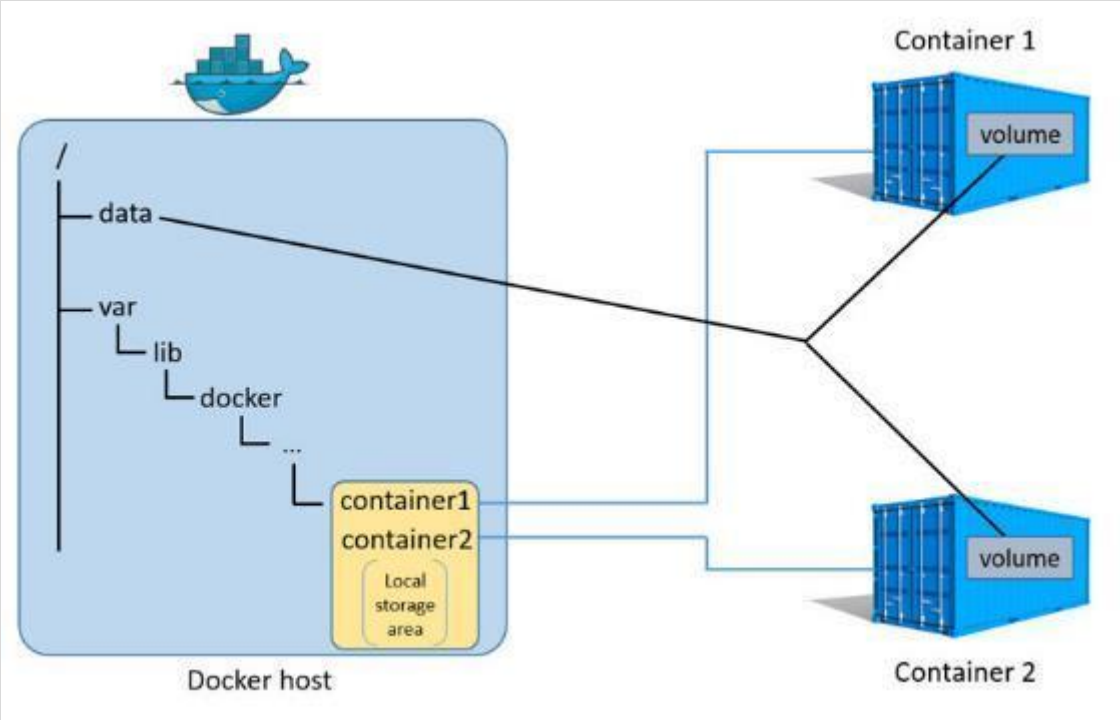
Port mapping

EXPOSE in Dockerfiles

See Docker docs

Use *host* network

Data Volumes



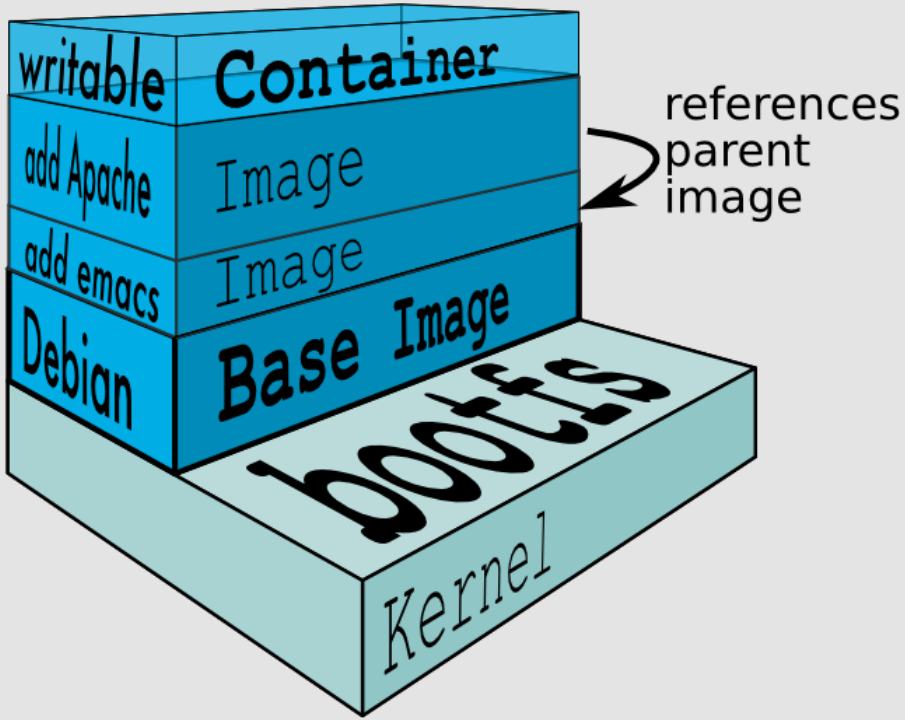
Directory or file in the Docker host's filesystem that is mounted directly into a container

Details see [Docker docs]

Source:
https://docs.docker.com/engine/userguide/storagedriver/imagesandcontainers/

Images

Working with images



File System Layers

Rootfs stays read-only

Union-mount file system
- over the read-only file system
  - Multiple file systems stacked on top of each other

Only top-most file system is writable
- Copy-on-write

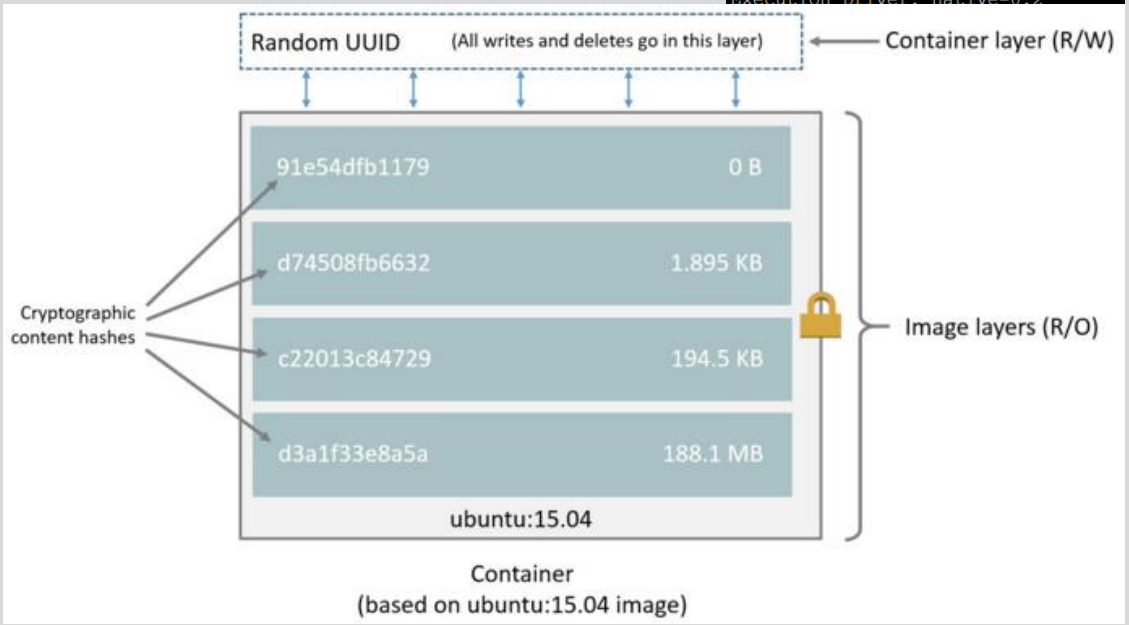
Images

```
# Pull image from docker hub
docker pull ubuntu

# Look for image directories on disk
ls /var/lib/docker/aufs/layers
```

└── Docker data directory

```
training@Docker:~$ docker info
Containers: 0
 Running: 0
 Paused: 0
 Stopped: 0
Images: 0
Server Version: 1.10.1
Storage Driver: aufs
 Root Dir: /var/lib/docker/aufs
 Backing Filesystem: extfs
 Dirs: 0
 Dirperm1 Supported: true
Execution Driver: native-0.2
```



More about storage drivers see [Docker docs]

Source:
https://docs.docker.com/engine/userguide/storagedriver/imagesandcontainers/

Docker CLI

Important Commands for Images

`docker images` – List all images
`docker search` – Search for image on Docker Hub
`docker pull` – Pulls an image from the registry (Docker Hub)
`docker commit` – Create image from container
`docker inspect` – Get low-level information on container or image

Docker CLI

Building Images from Containers

```
docker commit
  -m="Demo image" --author="Rainer Stropek"
```

Message

Author of the image

```
  templateContainer rstropek/ubuntu:withFile
```

Target repository:tag

Name of the container

Dockerfiles

Creating images from source

Dockerfiles

```
# Version 0.0.1
FROM nginx
MAINTAINER Rainer Stropek "rainer@timecockpit.com"
ENV REFRESHED_AT 2014-02-22
RUN apt-get -qq update
```

Execute command in new layer on top of the image and commit the result

```
COPY *.html /usr/share/nginx/html/
```

Copy files to the filesystem of the container

```
docker build -t staticweb .
```

Tag for the image

Dockerfile location

Documentation

https://docs.docker.com/reference/builder/
https://registry.hub.docker.com/_/nginx/

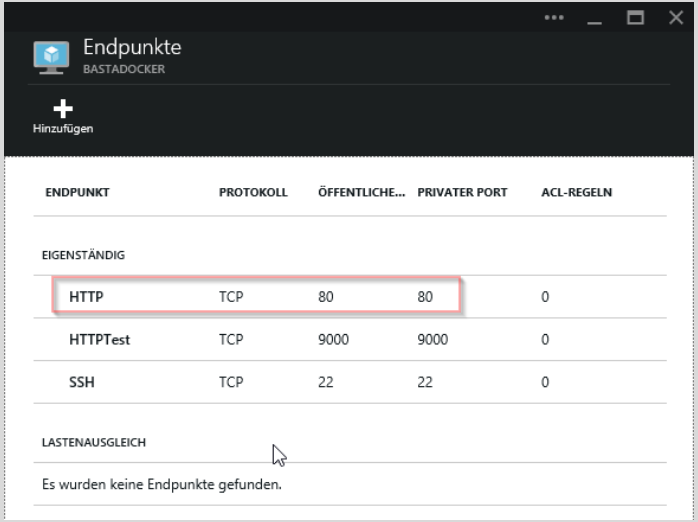
Docker CLI

Exposing ports

```
docker run --name staticwebcontainer \
  -d -p 80:80 staticweb
```

Expose port 80

Run daemonized



```
# Run grunt inside a docker container
docker run --rm
```

└ Remove the container when it exists

```
  -v ~/DockerVS2015Intro/dockerDemos/01-staticWeb/app:/data
```

└ Mount host volume (`host:container`)

```
  dockerfile/nodejs-bower-grunt
```

└ Use existing image

```
  grunt
```

└ Run `grunt`

Demo

Run Grunt (build) in Container

Docker Compose

Tool for running multi-container applications

```
printer:
  build: .
```

Build local Dockerfile

```
  links:
   - dependent-service
```

Link to other containers (e.g. Redis, MongoDB)

```
dependent-service:
  image: dependent-service
```

Run service container depends on based on an existing image

Demo

For more info visit https://docs.docker.com/compose/

ASP.NET in Docker

Running ASP.NET in Docker

```
FROM microsoft/aspnet

RUN apt-get install -y curl
RUN curl -sL https://deb.nodesource.com/setup_5.x | bash -
RUN apt-get install -y nodejs

COPY ./my-web /src

RUN cd /src && dnu restore

EXPOSE 5000

WORKDIR /src
CMD ["dnx", "web"]
```

Simple ASP.NET

Dockerfile

Sample files see
https://github.com/rstropek/DockerVS2015Intro/tree/master/dockerDemos/04-aspnet

```
# Generate an ASP.NET web app
yo aspnet webbasic "my-web"

# Add “--server.urls=http://*:5000/“ to project.json so
# that ASP.NET listens not only on localhost

# Build image with sample app
docker build -t rainer:myweb .

# Run ASP.NET container
docker run -d -p 80:5000 rainer:myweb
```

Simple ASP.NET

Sample files see
https://github.com/rstropek/DockerVS2015Intro/tree/master/dockerDemos/04-aspnet

```
FROM microsoft/aspnet
MAINTAINER Rainer Stropek "rainer@timecockpit.com"
ENV REFRESHED_AT 2015-01-02

ENV SOURCE_DIR /app/src

RUN mkdir -p $SOURCE_DIR
WORKDIR $SOURCE_DIR

COPY refreshAndRunSample.sh $SOURCE_DIR/
RUN chmod a+x $SOURCE_DIR/refreshAndRunSample.sh

RUN apt-get -qqy install git
RUN git init \
 && git pull https://github.com/aspnet/Home.git \
 && cd  samples/HelloMvc/ \
 && kpm restore

ENTRYPOINT ["/app/src/refreshAndRunSample.sh"]
```

Dockerfile

Base image:
https://registry.hub.docker.com/u/microsoft/aspnet/

Run container

```
docker run -d -t
  -p 80:5004 aspnet-beta8
```

Application Scenarios

Running continuous integration in containers

Rebuild complex runtime environment on my laptop

Identical environment for dev, test, and prod

Cost reduction in the cloud

High density hosting (e.g. multiple versions)

Split software into multiple, independent services

Micro-services, see Manfred's session tomorrow

Q&A

Thank you for attending!

Rainer Stropek

software architects gmbh

Web http://www.timecockpit.com
Mail rainer@timecockpit.com
Twitter @rstropek

time cockpit
Saves the day.