Anwendungsvirtualisierung in Containern mit Docker

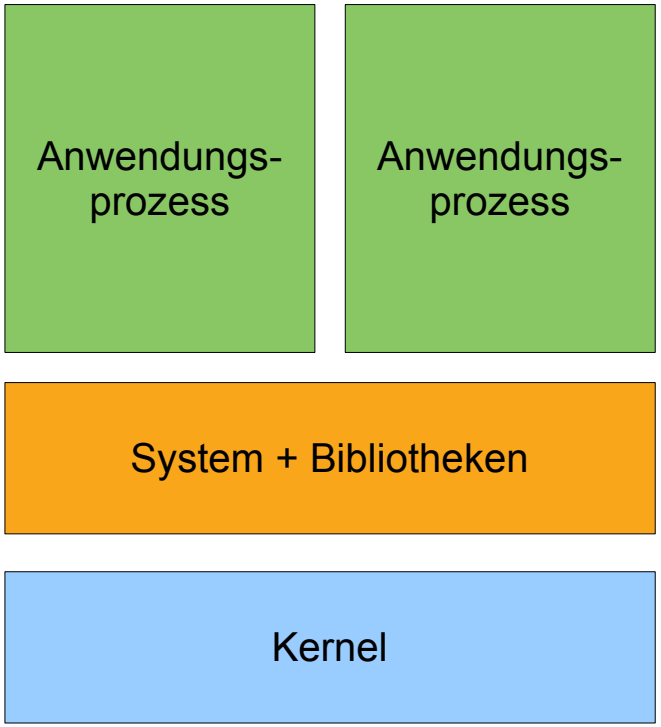
Troisdorfer Linux User Group
1. Februar 2018

Harald Weidner
hweidner@gmx.net

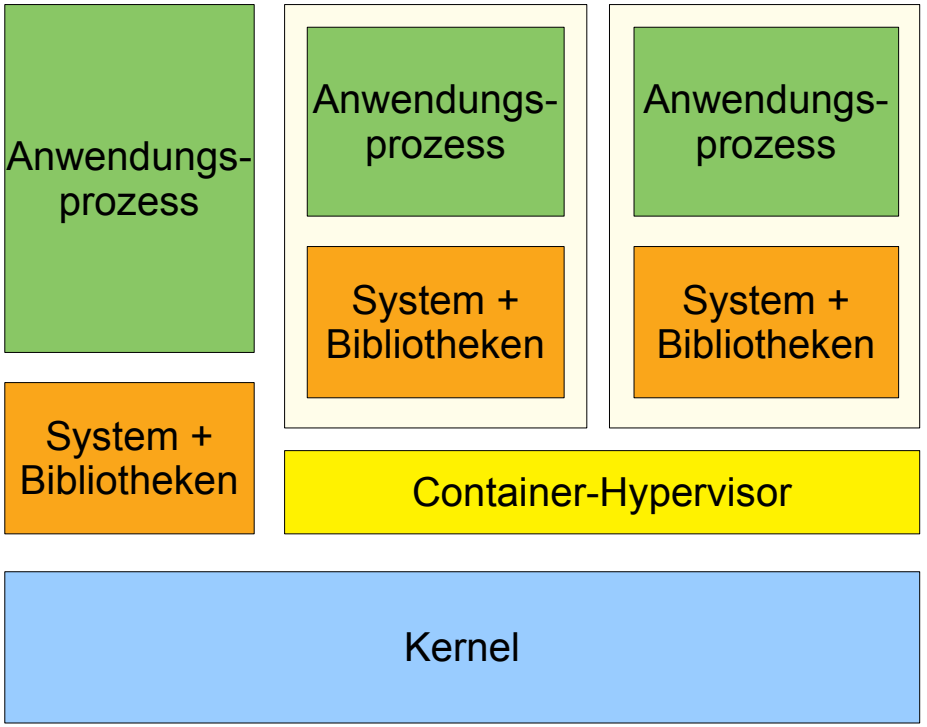
Container

- Mechanismus zur Prozess-Isolierung auf einem Betriebssystem
  - Filesystem
  - Prozesstabelle
  - Benutzerverwaltung
  - Netzwerk
- Aktuelle Linux-Container basieren auf
  - Linux-Namespaces: Isolation gegenüber Hostsystem
  - Cgroups: Ressourcenverteilung
  - Seccomp: Rechteeinschränkung

Container



Betriebssystem ohne Container



Betriebssystem mit Containern

Container vs. virtuelle Maschinen

Container

- Ein gemeinsamer Kernel
- Keine Hardware-Emulation
- Gleiches Betriebssystem (aber verschiedene Distributionen möglich)

Virtuelle Maschinen

- Eigener Kernel pro virtueller Maschine
- Ressourcenverbrauch durch Emulation von Hardware
- Verschiedene Betriebssysteme möglich

BS-/Anwendungscontainer

BS-Container

- Vollständiger Satz an Bibliotheken und Tools
- Init-System, Syslog, Cron, SSH-Daemon
- Ggf. mehrere Prozesse im Hintergrund
- Explizites Herunterfahren

Anwendungscontainer

- Eine Anwendung pro Container
- Nur die benötigten Bibliotheken
- Prozess im Vordergrund, Prozess-Ende = Container-Ende

Container – History

1979 chroot (Filesystembeschränkung)

1999 Linux-VServer.org

2000 BSD Jails

2001 Virtuozzo / OpenVZ (SWsoft/Parallels)

2005 Zones (Solaris 10)

2008 LXC (im offiziellen Kernel)

2013 Docker (Docker Inc.)

2014 rkt (CoreOS)

2015 LXD (Canonical)

Docker

- Software zum Erstellen, Verwalten und Ausführen von Anwendungscontainern
- Docker, Inc. (San Francisco)
- Erste Veröffentlichung 2013
- Geschrieben in Go
- Lizenz: Apache 2.0
- Basis großer Cloud-Infrastrukturen

Docker

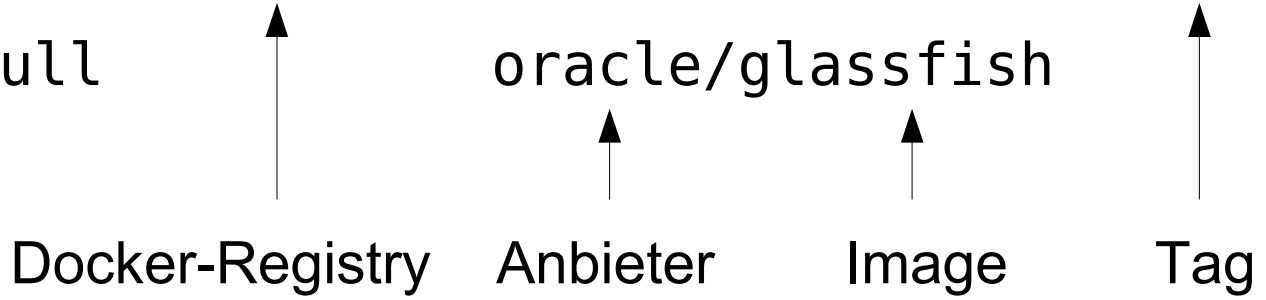
- „Build, Ship and Run Any App, Anywhere“
- „Build Once, Run Anywhere“

- Isolation: Namespaces, Cgroups, Seccomp
- Effizienz durch Caching und FS-Layering
- Anbindung von Ressourcen
  - Port Forwarding
  - Volume Mounts
  - Logging

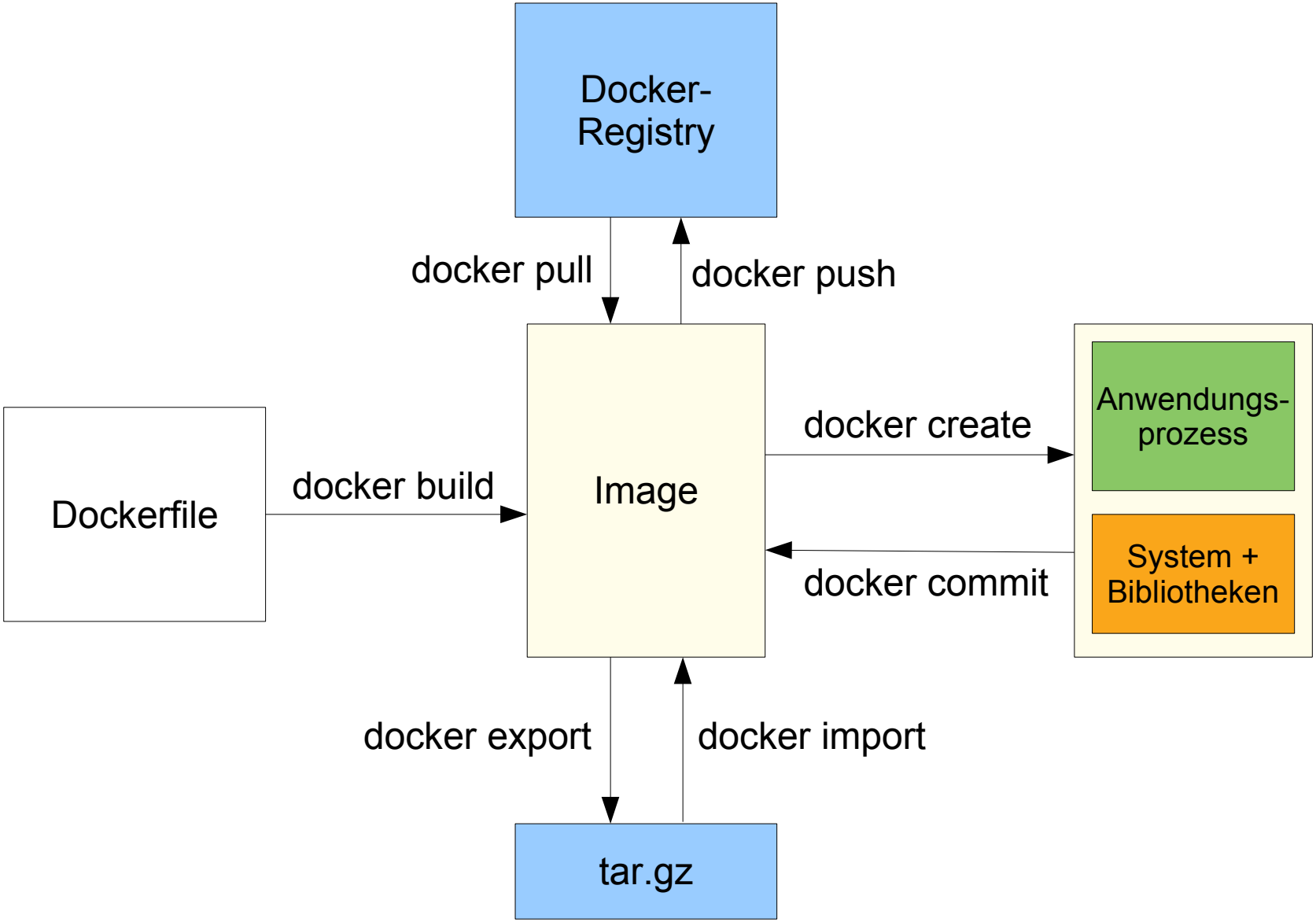
Docker Images

- Vorlagen für Container
- Speicherung in Docker Registry
- Vorgefertigte Images vom Docker Hub

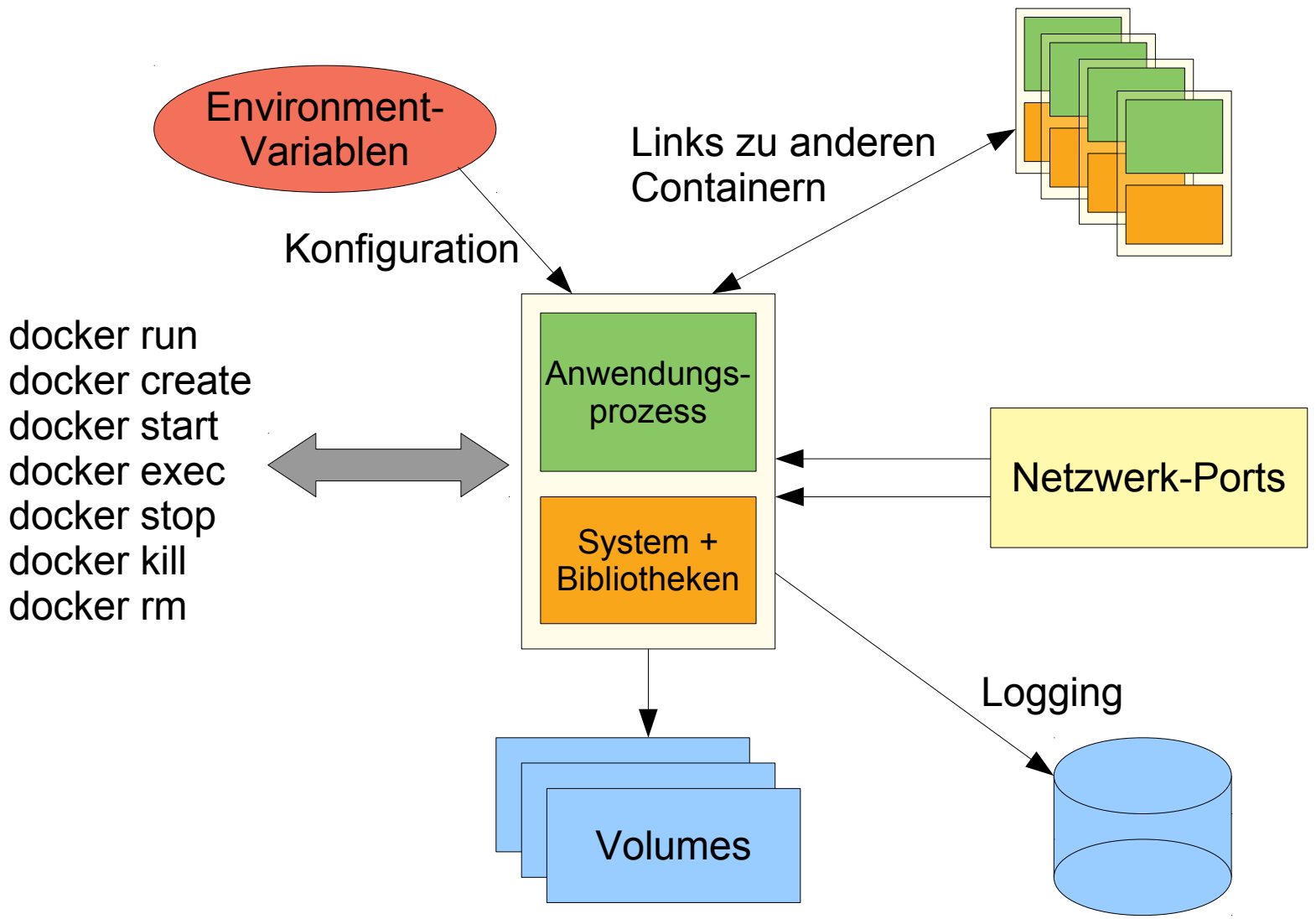
```
docker pull docker.io/oracle/glassfish:latest
docker pull                 oracle/glassfish
```



Docker Workflow



Docker Container



Docker Image bauen

Dockerfile

```
FROM alpine
MAINTAINER Harald Weidner <hweidner@gmx.net>

RUN apk update && apk add php5-apache2 && mkdir /run/apache2
COPY myapp.php /var/www/localhost/htdocs/index.php
EXPOSE 80

CMD ["/usr/sbin/httpd", "-D", "FOREGROUND"]
```

myapp.php

```
<?php
echo "Hello, World!"
?>
```

Image erstellen:

```
$ docker build -t phpapp .
```

Container ausführen:

```
$ docker run --rm -p 8080:80 phpapp
```

Docker-Erweiterungen

- Docker Compose
  - Beschreiben von Anwendungen auf Basis mehrerer Container
- Docker Swarm Mode
  - Ablaufumgebung für Container auf mehreren Hosts
- Produkte anderer Hersteller, z.B.
  - Kubernetes
  - Juju (Canonical)
  - OpenShift (RedHat)