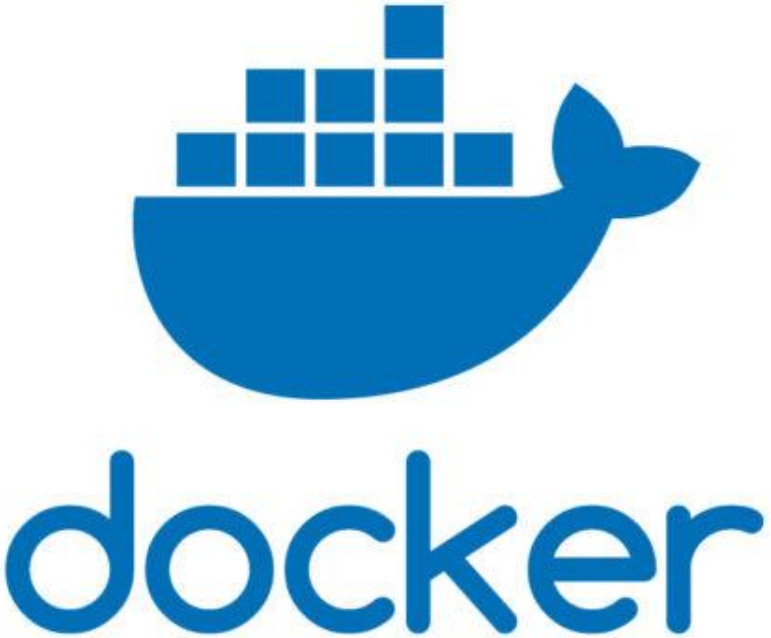


deep-dive

About me

– David Kröll, IT student

– Docker user since Jun `17

https://davidkroell.github.io

Preface

- Container technology highly used in cloud
- Must-have for Ops in 2018

- Presentation and snippets
  - https://github.com/davidkroell/docker-deepdive

Topics

- General
- Difference to VMs
- Image vs. Container
- Registry
- Dockerfile
- docker-compose

Topics

- Storage
- Networking
- Swarm
- Monitoring
- Outlook

General

- Collection of tools
  - runc, containerd
- Versions
  - CE, EE
- Builds
  - edge, stable
- written in Golang

General

- Save costs in application deployment
  - Same motivation as VMs
- Isolation
- „Runs on my machine“
  - Packaging of apps
  - Build, Ship, Run

General

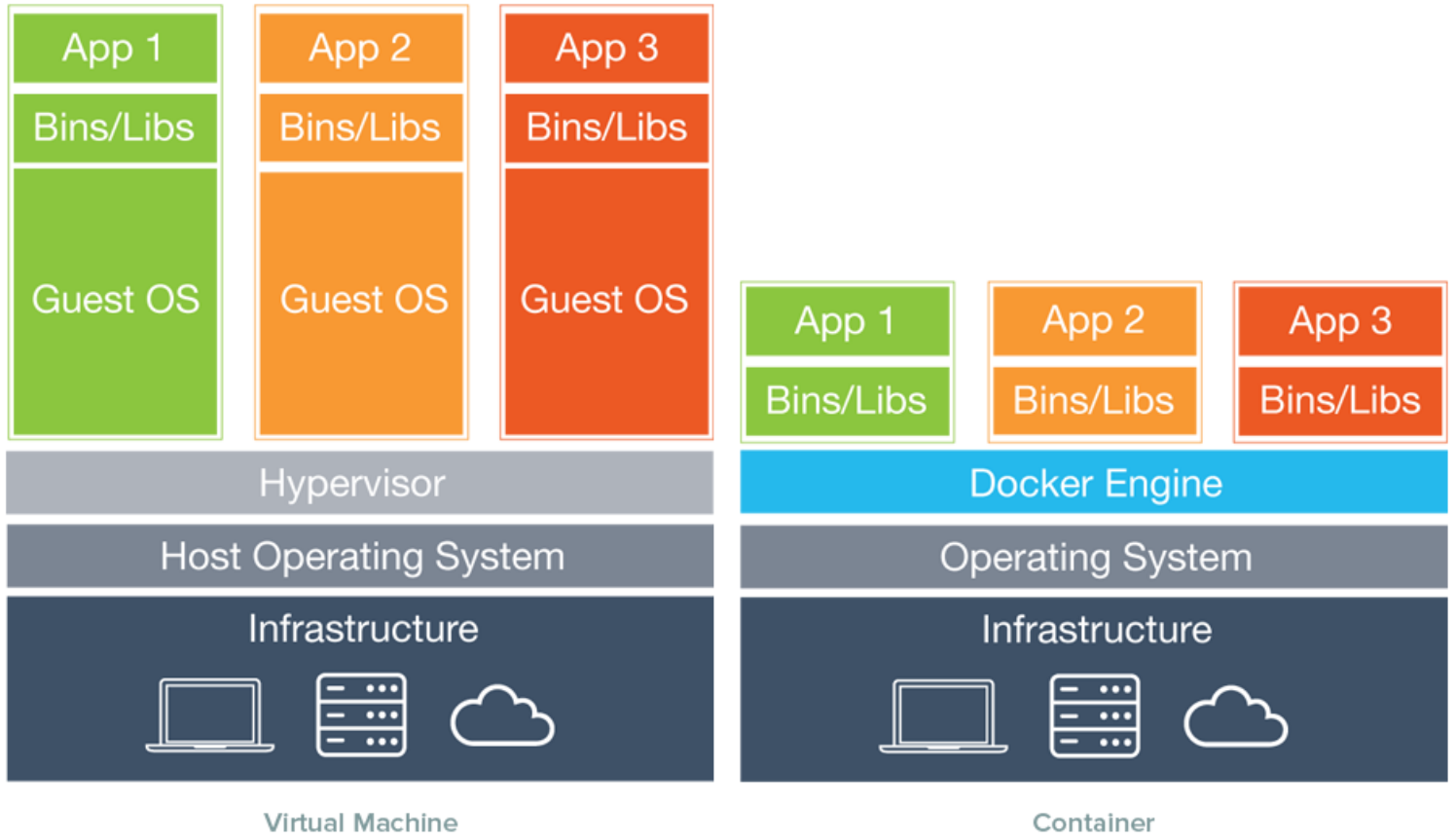
Three core principles

- Kernel Namespaces
  - Linux 2.6 (2009)
- Cgroups
- Chroot

Containers from Scratch – Liz Rice

https://www.youtube.com/watch?v=MHv6cWjvQjM&t

Difference to VMs



Difference to VMs

- Decrease virtualization overhead
- No syscall translations
- OS virtualization instead HW virtualization
- Nearly same performance as bare metal
  - Efficiency

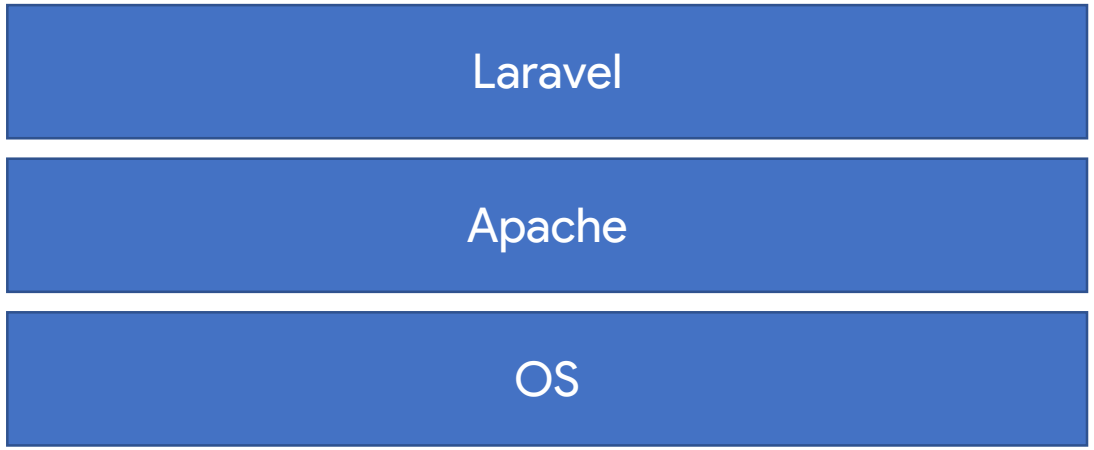
Kernel namespacing

```
[node1] (local) root@192.168.0.48 ~
$ docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS
    NAMES
bcc1df6c14b2        davidkroell/ytdl    "/usr/bin/env node /…"   19 seconds ago      Up 18 seconds       3000/tcp
    determined_agnesi
[node1] (local) root@192.168.0.48 ~
$ ps -ef | grep node
  404 root      0:00 node /usr/src/app/ytdl.js
  519 root      0:00 grep node
```

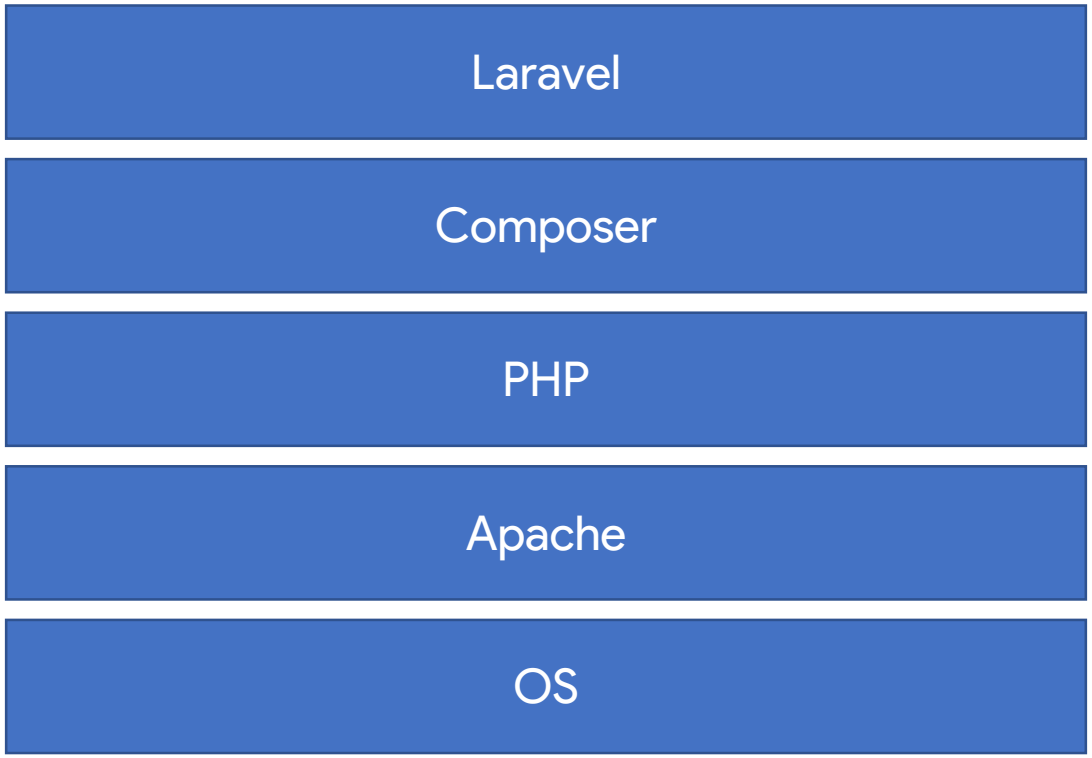
Image

- Package of software
  - with dependencies (libs ...)
  - Includes anything to run
- Application with OS (if needed)
- Storage driver

Example: Laravel Stack

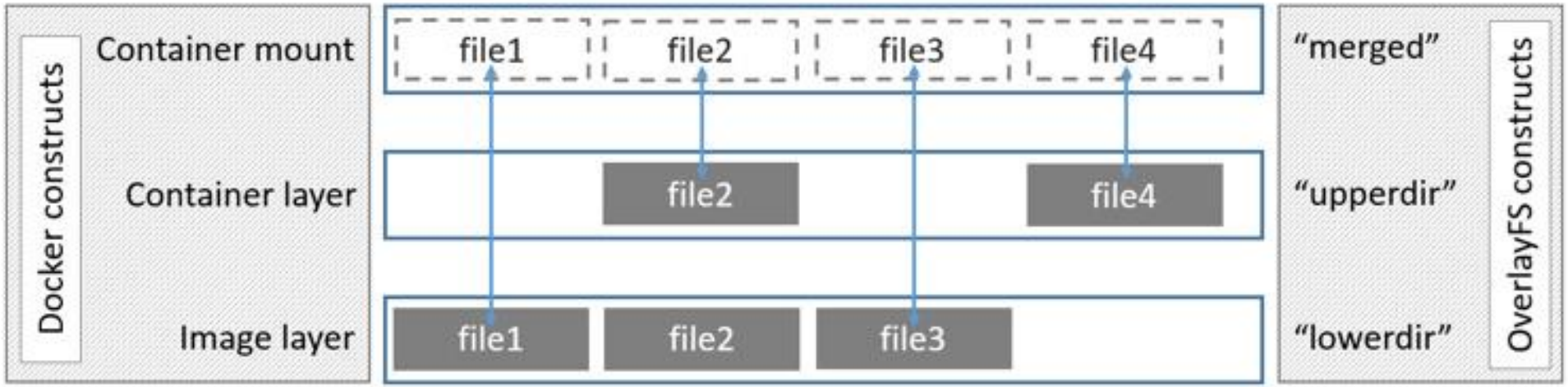


Example: Laravel Stack



Images architecture

- Different Storage Drivers
  - Overlay2 (default)



Exploring Images

- Inspect Images with Docker Toolset
- Use wagoodman/dive

Container

- Image which runs on a Docker host (dockerd)
- Network and storage attached
- Consists of an Image
  - Adds container storage layer

Run a container

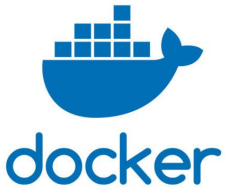
```
docker run mongo
```

Run a container

```
docker run --name mysql57 -d \
      --restart=always
      -p 127.0.0.1:3306:3306 \
      -v mysql:/var/run/mysql \
      -e MYSQL_ROOT_PASSWORD=my-secret-pw
      mysql:5.7
```

https://docs.docker.com/engine/reference/run

Docker Client & Daemon



- JSON API under the hood
  - `/var/run/docker.sock`
- May be called from any HTTP client
- SSH Transport support (v18.09)
- Various integrations
  - Docker2go – https://davidkroell.github.io/docker2go

Docker on windows

- Needs HyperV, better use forwarding
- `docker -H ssh://user@host ps`
  - ps on remote server
- Set `DOCKER_HOST` environment variable

Composing an Image

- `docker build`
- Parse Dockerfile
- Set of statements

```
1    FROM scratch
2    ADD busybox.tar.xz /
3    CMD ["sh"]
```

https://github.com/docker-library/busybox/blob/master/uclibc/Dockerfile

Dockerfile

```
FROM bitnami/minideb-extras:stretch-r225
LABEL maintainer "Bitnami <containers@bitnami.com>"

# Install required system packages and dependencies
RUN install_packages ghostscript imagemagick libbz2-1.0 libc6 libcomerr2 libcurl3 libffi6 libfreetype6 libgcc1 libgcrypt20 lib
RUN bitnami-pkg unpack php-7.1.25-20 --checksum b4350f7370f196def8ff56462d7efb4961c15f97a705b96211006a16cec02cac
RUN bitnami-pkg install node-8.14.0-20 --checksum 408efbaecc9a5d5aa93f0e1755a15f5fe29c08d37baaac3900c9f02551d6da2b
RUN bitnami-pkg install laravel-5.7.15-20 --checksum 19e49651d46ceaa1e62c9d03a1d0d0fc369adc6a95867bd88819c1de26e901a2
RUN mkdir /app && chown bitnami:bitnami /app

COPY rootfs /
ENV BITNAMI_APP_NAME="laravel" \
    BITNAMI_IMAGE_VERSION="5.7.15-debian-9-r22" \
    NODE_PATH="/opt/bitnami/node/lib/node_modules" \
    PATH="/opt/bitnami/php/bin:/opt/bitnami/php/sbin:/opt/bitnami/node/bin:/opt/bitnami/laravel/bin:$PATH"

EXPOSE 3000

WORKDIR /app
USER bitnami
ENTRYPOINT [ "/app-entrypoint.sh" ]
CMD [ "php", "artisan", "serve", "--host=0.0.0.0", "--port=3000" ]
```

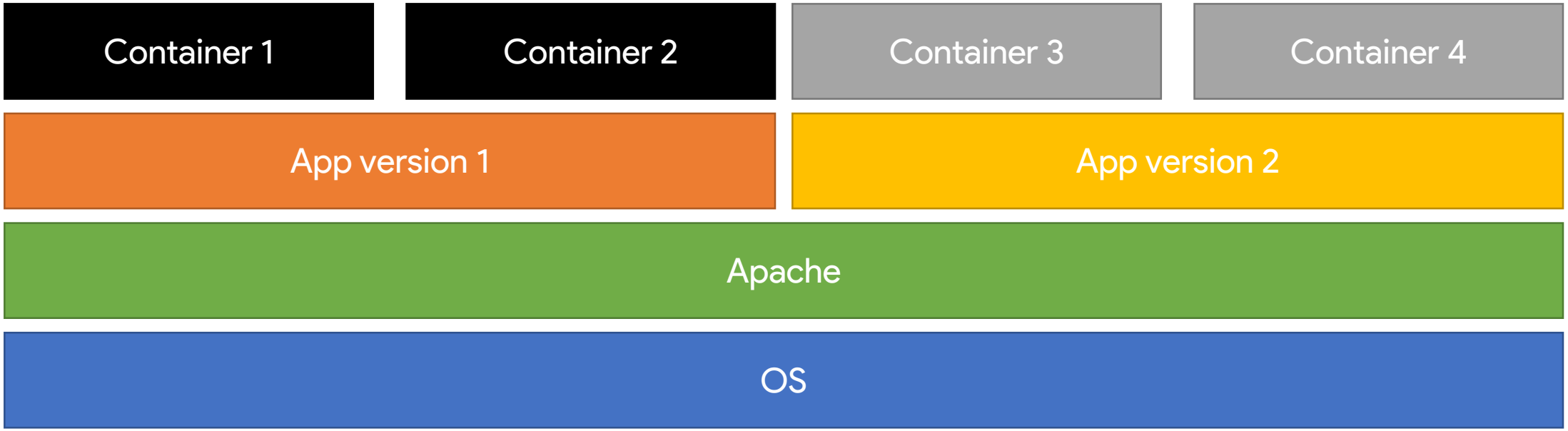
https://github.com/bitnami/bitnami-docker-laravel/blob/master/5/debian-9/Dockerfile

Multi-staged build

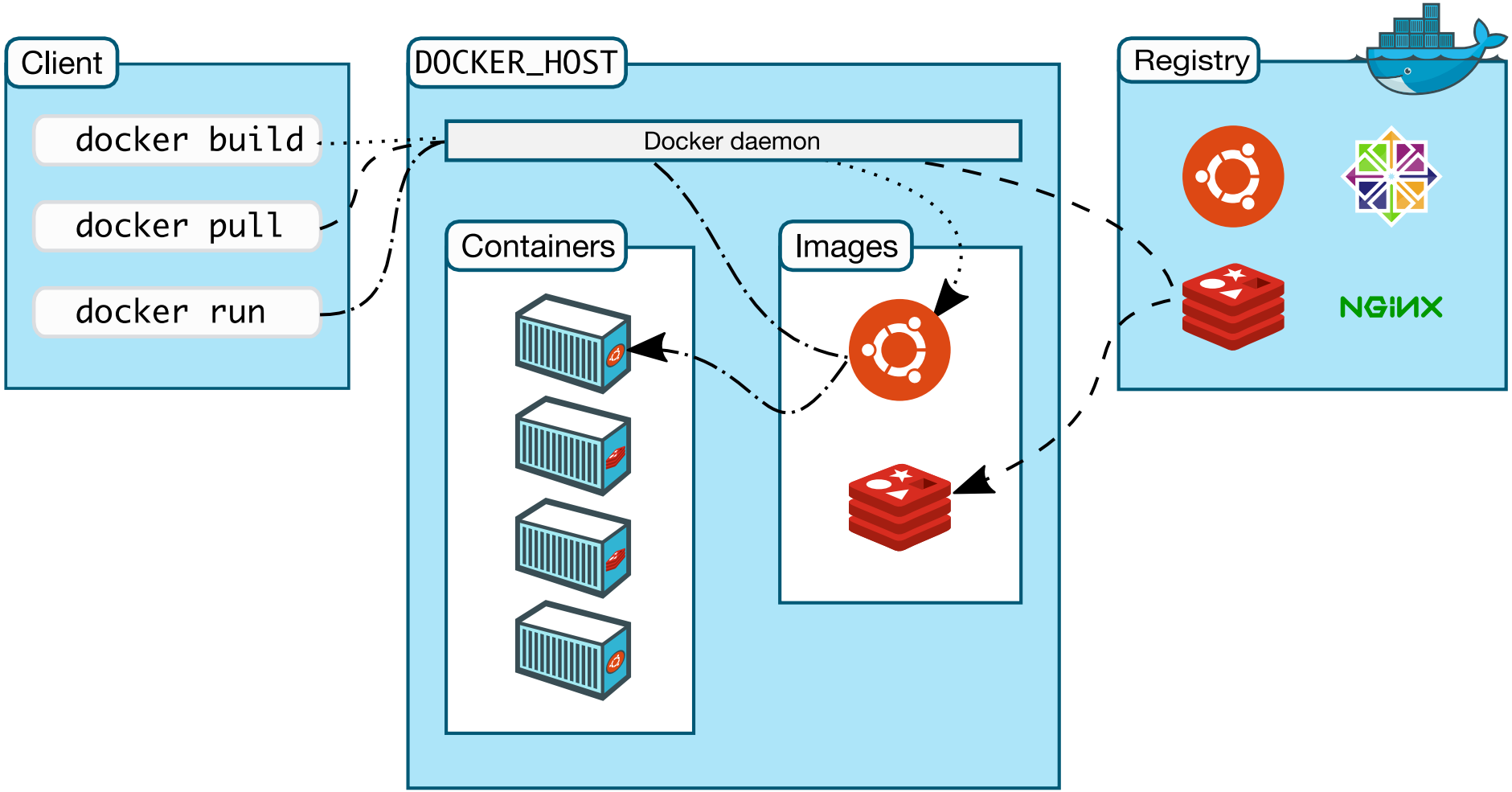
```
1   # base image for building the binary
2   FROM golang:1.11-alpine AS base
3   # $GOPATH is /go
4   COPY . /go/src/github.com/ironarachne/namegen
5   # the output is located in the working directory without fileextension
6   # binary path: /go/main
7   RUN go build /go/src/github.com/ironarachne/namegen/cmd/namegen/main.go
8
9   # add the binary to an empty image
10  FROM scratch
11  # copy from build-image
12  COPY --from=base /go/main /namegen
13  # set namegen as
14  ENTRYPOINT ["/namegen"]
```

https://github.com/ironarachne/namegen/blob/master/Dockerfile

Efficiency of Images



Registry Overview



Registry

- Storage for Images
  - Repositories, Tags
- Is delivered as Docker Image
- https://hub.docker.com -> official Registry

Registry

```
[manager1] (local) root@192.168.0.26 ~
$ docker run mongo
Unable to find image 'mongo:latest' locally
latest: Pulling from library/mongo
7b8b6451c85f: Extracting [==============================>                    ]  26.15MB/43.41MB
ab4d1096d9ba: Download complete
e6797d1788ac: Download complete
e25c5c290bde: Download complete
45aa1a4d5e06: Download complete
b7e29f184242: Download complete
ad78e42605af: Download complete
1f4ac0b92a65: Download complete
55880275f9fb: Download complete
bd0396c9dcef: Download complete
28bf9db38c03: Download complete
3e954d14ae9b: Download complete
cd245aa9c426: Download complete
```

Networking

- Docker implements „virtual“ Networks
- Port mappings
- Inter-container communication
- Address containers with hostnames
  - rrDNS

```
docker network ls
```

Storage

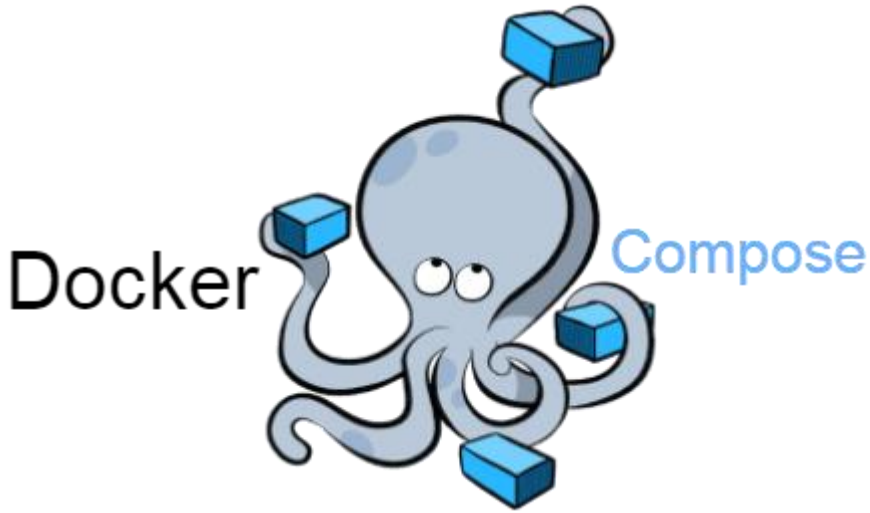
- Containers should be stateless
- Volumes
- Directory mounts
- Volume plugins for HA storage
  - REX-Ray for S3/GCE/Azure

Deploying Services

- Deployment of dependent containers
- Networking
- Storage
- Environment variables
- Clustering (see Swarm)
- Scaling

docker-compose

- Defines a Service as a whole
- Add-on for Docker
- https://github.com/docker/compose



Compose file

```yaml
version: '3'

services:
  web-api:
    image: davidkroell/keylog.rest
    container_name: keylog.rest-api
    restart: always
    environment:
      NODE_ENV: production
      # use this if you are behind a proxy, this is the header, which the proxy sets
      PROXY_HEADER_REAL_IP_KEY: x-real-ip # for jwilder/nginx-proxy
    ports:
      - 3000:3000
```

https://github.com/davidkroell/keylog.rest/blob/master/docker-compose.yml

Using a Compose file

- `docker-compose up`
- `docker stack deploy`

Docker Swarm

- Docker-based cluster orchestration
- Multiple Nodes for HA
- Manger/Worker principle (up to 1/1000 ratio)
- Shared nothing
- Fully encrypted overlay networking
- Service discovery
- Rolling updates

Using Docker Swarm

- Enable Swarm mode in dockerd
  - `docker swarm init`
- Add node to cluster
  - `docker swarm join`
- Smallest unit is now a Service, not a container
- Deployment modes and constraints

Deploy to Docker Swarm

- Deployment based on Compose-file
- `docker stack deploy`
- Updates without Downtime
  - Rolling Updates

Scaling your app

- Keeping it stateless makes it simple
- Scale storage => Volume plugins
- Shared-nothing clustering

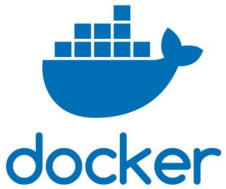
Monitoring

`docker container stats`

- Streaming usage API -> JSON

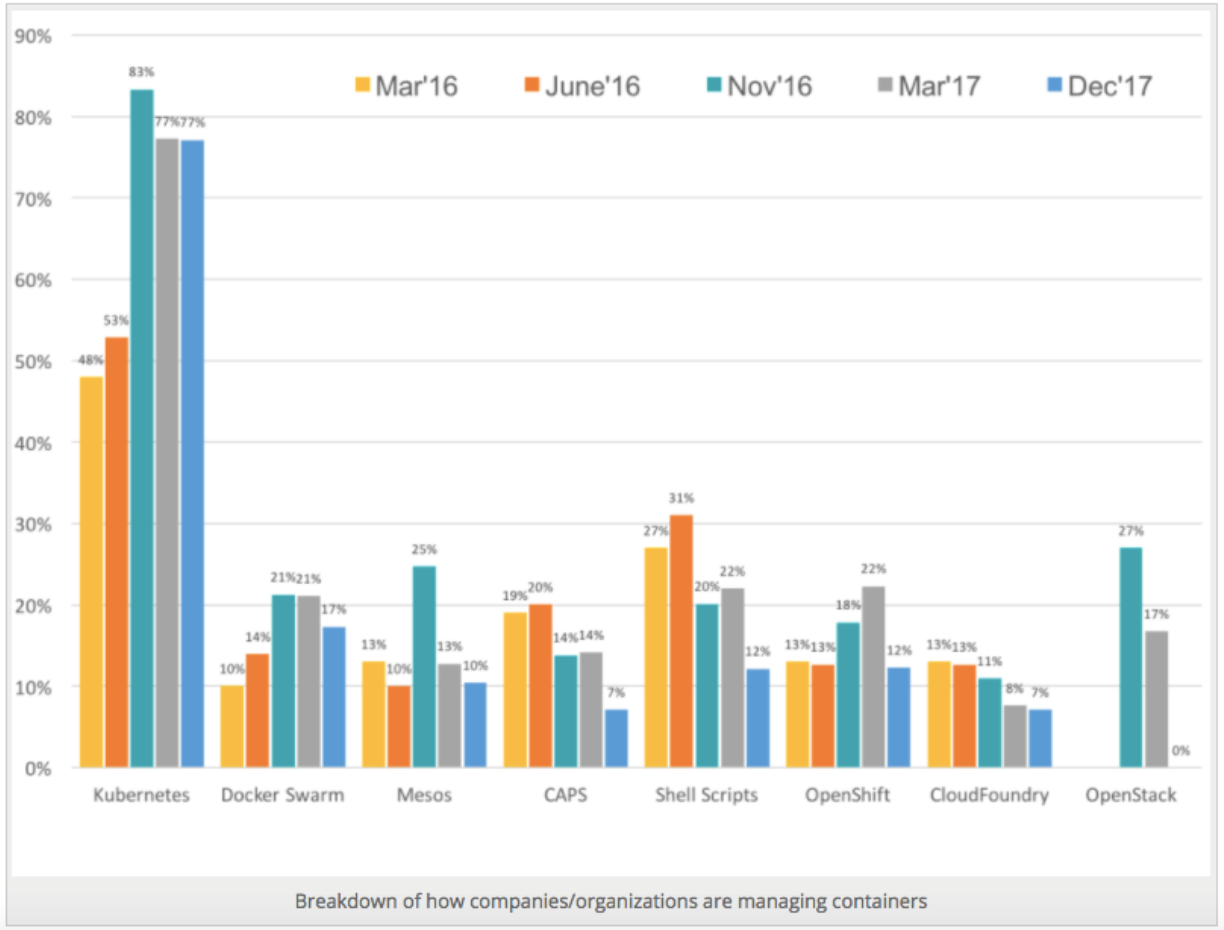
- Cloud-native Monitoring
  - Prometheus
  - cAdvisor
- Limit container resources
  - cgroups

Outlook



- Kubernetes
  - Global container orchestration platform
- Docker Cloud
- Baremetal cloud
  - CoreOS

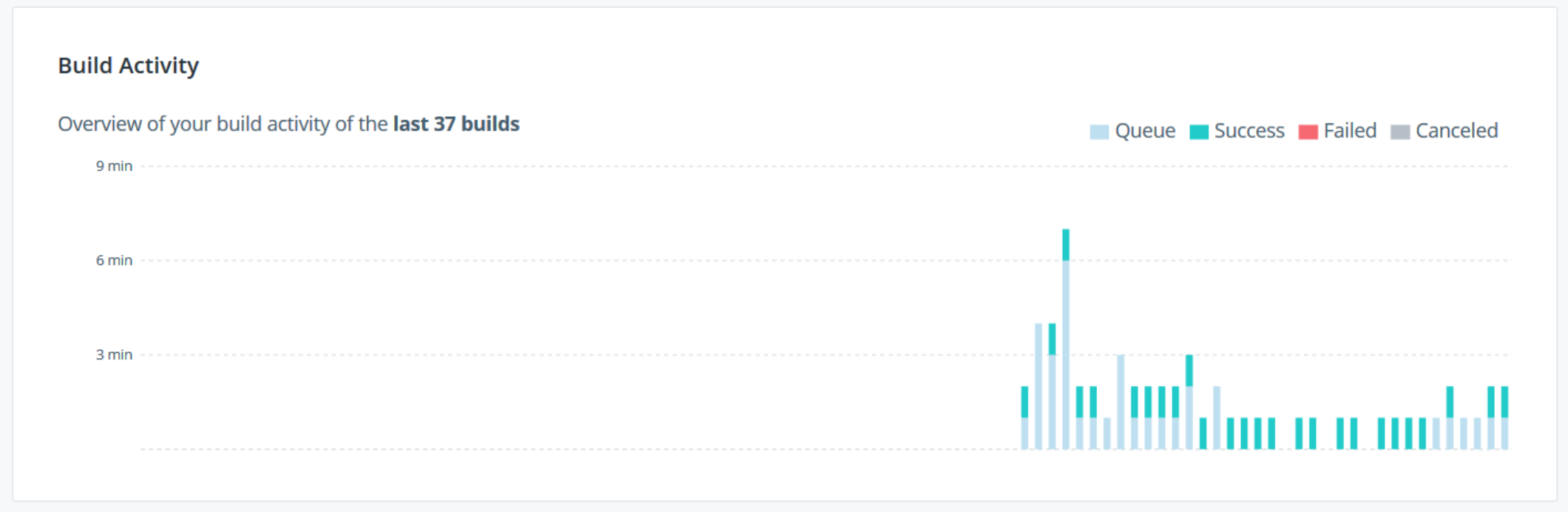
Outlook – Kubernetes



Breakdown of how companies/organizations are managing containers

Outlook – Docker Cloud

– Automated builds



Outlook – Baremetal cloud

- No virtual infrastructure
- Containers
- Orchestrated
  - Docker Swarm
  - K8s or
- CoreOS

Sources

- https://hackathon.bz.it
- https://www.porttechnology.org
- https://docs.docker.com
- https://www.sdxcentral.com