Docker 容器与镜像

架构与机动组-杨逸（@cloverstd）

2017.11.27

Docker intro

- Docker 是 Docker 公司（曾经）开源的一个基于轻量级虚拟化技术的容器项目，现在改名叫 moby 基于 Golang 开发的
- 基于 Golang 开发的
- Linux cgroups && namespaces
- C/S 结构（docker cli/dockerd）

platform

- Windows 10（Hyper-V）
- MacOS（Hypervisor）
- Linux

适用场景

- CI/CD
- 快速开发环境
- PaaS

Docker image

- 相当于一个 Linux 的 root 文件系统
- 分层储存
- Dockerfile 用于描述镜像的生成规则
    - Dockerfile中的每一条命令，都在Docker镜像中以一个独立镜像层的形式存在

```
FROM debian:8
MAINTAINER @cloverstd <cloverstd@gmail.com>

RUN apt-get update -y && \
    apt-get install -y emacs

RUN apt-get install -y apache

CMD ["/bin/bash"]
```

```
docker build -t apache -f Dockerfile .
```

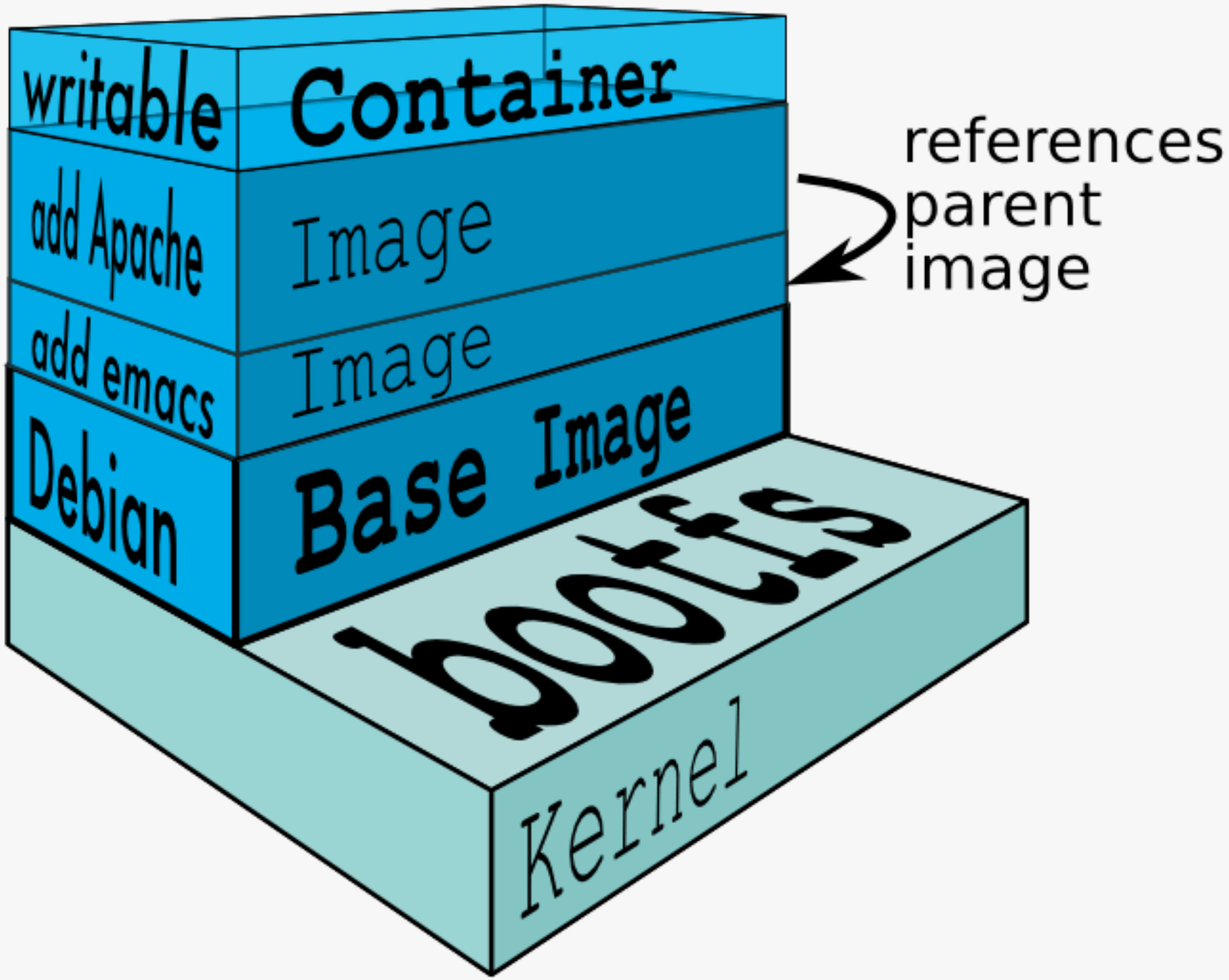
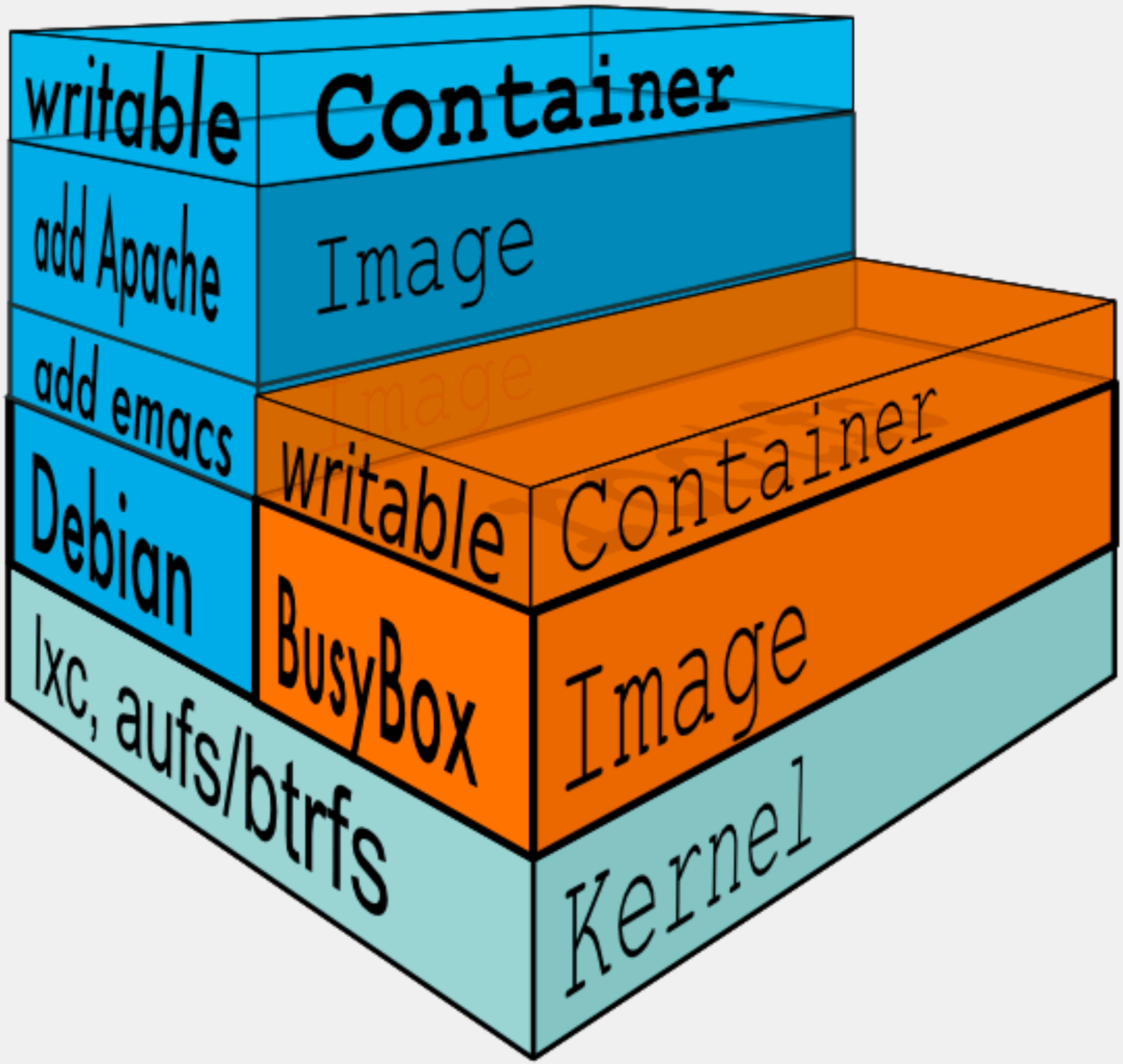

writable
Container
add Apache
Image
add emacs
Image
Debian
Base Image
bootfs
Kernel
references
parent
image

image storage

- $DOCKER_ROOT_PATH/image/STORAGE_DRIVER
  - /var/lib/docker/image/overlay2
- 分层储存
  - 节省空间
  - 多个镜像共用同一层

Docker container

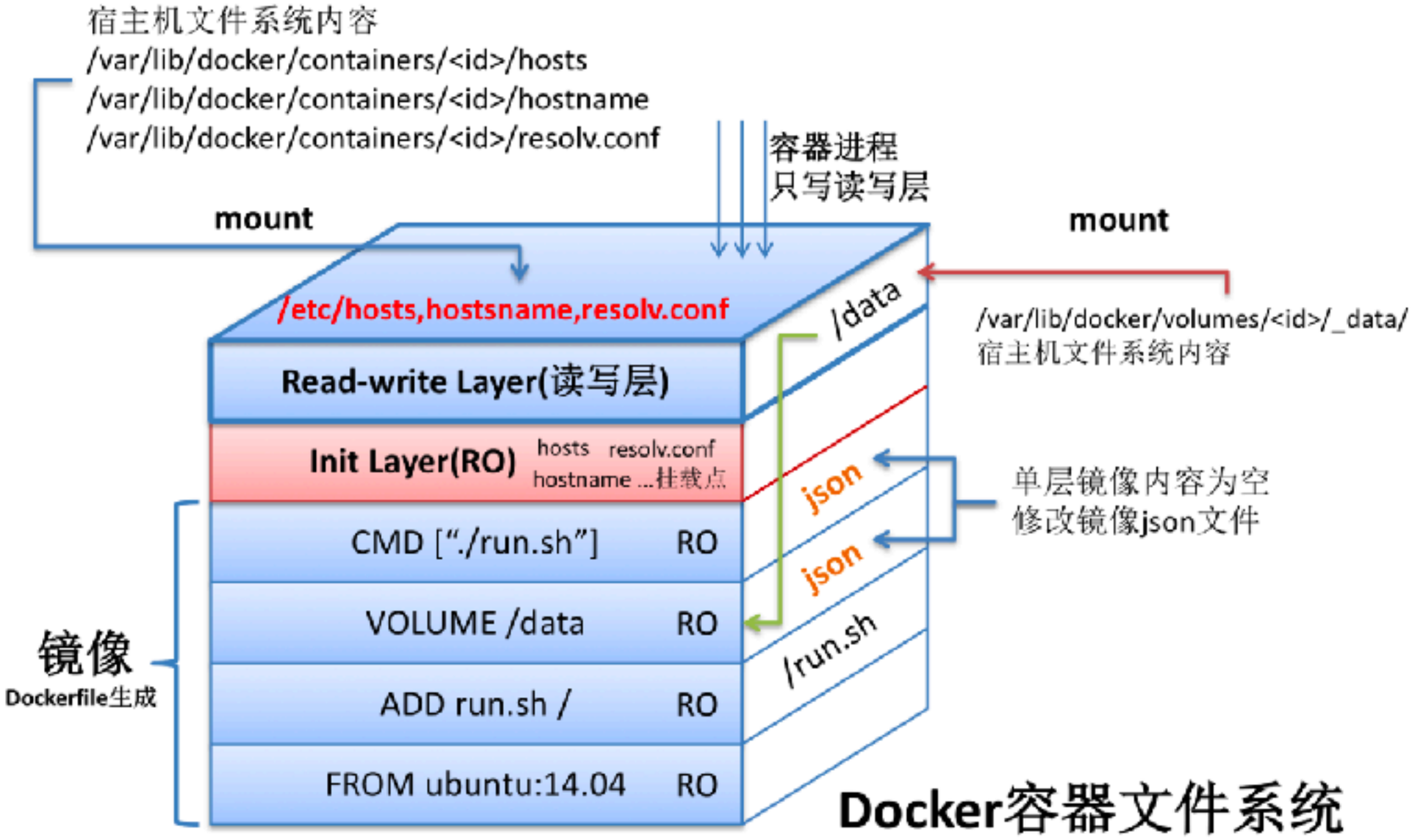
```
docker run -it --rm -v /data:/data:ro -p 8080:80 apache bash
```


writable
Container
add Apache
Image
add emacs
Debian
writable
Container
BusyBox
Image
lxc, aufs/btrfs
Kernel

UnionFS

- 应用场景：把一张 CD/DVD 和一个硬盘目录给联合 mount 在一起，然后，就可以对这个只读的 CD/DVD 上的文件进行修改（修改的文件存于硬盘上的目录里）

```
1 FROM ubuntu:14.04
2 ADD run.sh /
3 VOLUME /data
4 CMD ["./run.sh"]
```



- VOLUME /data 和 CMD ["./run.sh"] 没有添加任何文件，只是在镜像的 meta info 文件中加了相关的配置
- ADD run.sh / 在这里一层增加了文件内容
- init layer: 初始化容器的一些配置信息，hostname, host, dns 等
- read-write layer: 容器运行时对容器内的任何修改都会储存在这一层

Example(overlay2)

```
1 FROM alpine:3.4
2 RUN mkdir -p /data/layer
3 COPY layer1 /data/layer
4 COPY layer2 /data/layer
5 COPY layer3 /data/layer
6 RUN echo 'echo "hello"' >> /etc/profile
```

```
docker build -t layer .
```

```
docker run -it --rm -v `pwd`:/code-r:ro -v `pwd`:/code-w layer sh
docker inspect ed87a88a8384
```

```
{
    // ...
    "GraphDriver": {
        "Data": {
            "LowerDir": "/data/docker/overlay2/971bf559559d65b686483fd3a7c2fcc25ed6826b8c50e62e86983036424f35a6-init/diff:/data/docker/overlay2/b2a005eb51a3ee2f94dd247665649b1ba8b00d34826d18f9b09952d234b311d1/diff:/data/docker/overlay2/7de52726234f64fa09a4c9cae26c0b93e17f257f2f2fbcbe53640de4e9631b2d/diff:/data/docker/overlay2/4c21baa722402857d4de72cdd45d02d3e538a65f0d69186d7a0609d6761bf2ba/diff:/data/docker/overlay2/3b9867ae0f17b189e86dab3624a870fa9944e4eb579358bd7028f7f1b2b3aa72/diff:/data/docker/overlay2/0f97da2aa3b51aa414ad8bb0409023491faf76e17b59a12e0f8b4ef5d536376f/diff:/data/docker/overlay2/94084d1feab115441442023e520bd89e7fe86aef06c6c9b474e085efa1f5a8e1/diff",
            "MergedDir": "/data/docker/overlay2/971bf559559d65b686483fd3a7c2fcc25ed6826b8c50e62e86983036424f35a6/merged",
            "UpperDir": "/data/docker/overlay2/971bf559559d65b686483fd3a7c2fcc25ed6826b8c50e62e86983036424f35a6/diff",
            "WorkDir": "/data/docker/overlay2/971bf559559d65b686483fd3a7c2fcc25ed6826b8c50e62e86983036424f35a6/work"
        },
        "Name": "overlay2"
    }
    // ...
}
```

```
1 FROM alpine:3.4
2 RUN mkdir -p /data/layer
3 COPY layer1 /data/layer
4 COPY layer2 /data/layer
5 COPY layer3 /data/layer
6 RUN echo 'echo "hello"' >> /etc/profile
```

- 6: /data/docker/overlay2/b2a00…/diff
- 5: /data/docker/overlay2/7de52…/diff
- 4: /data/docker/overlay2/4c21b…/diff
- 3: /data/docker/overlay2/3b986…/diff
- 2: /data/docker/overlay2/0f97d…/diff
- 1: /data/docker/overlay2/94084…/diff

Dockerfile

- 合理分层（最多127层）
- 利用缓存
- 优先使用 alpine 作为基础镜像

能写多少层就写多少层

```
1  FROM alpine:3.4
2  RUN apk update
3  COPY big-file.tar.gz /code
4  RUN tar zxvf big-file.tar.gz
5  RUN sh big-file.sh
6  RUN apk add nodejs make gcc git
7  RUN apk add nginx
8  RUN apk add supervisor
9  COPY package.json /code
10 WORKDIR /code
11 RUN npm install
12 RUN rm -rf big-file.tar.gz
13 RUN apk del make gcc git && rm /var/cache/*
14 CMD ["supervisord", "-n"]
```

优点

- 适合开发阶段
- 写起来快，调试方便
- 每一层尽量充分复用
- 可以加速镜像的 build

缺点

- 镜像体积过大

能一层搞定的绝对不分成两层

```
FROM alpine:3.4
COPY package.json /code
RUN set -ex && \
    apk update && \
    apk add --virtual .build-dependencies make gcc ca-certificates wget && \
    apk add nodejs nginx supervisor && \
    wget -q -O /tmp/big-file.tar.gz https://hui.lu/big-file.tar.gz && \
    cd /tmp && tar zxvf big-file.tar.gz && sh big-file.sh && \
    cd /code && npm install && \
    apk del .build-dependencies && \
    rm -rf /var/cache/* /tmp/*
WORKDIR /code
CMD ["supervisord", "-n"]
```

优点

- 镜像体积小
- 适合下载分发

缺点

- 失去了镜像分层的意义
- 调试麻烦，build 慢

完