阿里云 × CLOUD NATIVE COMPUTING FOUNDATION

云原生技术公开课

第 02 讲

容器基本概念

傅伟 阿里巴巴高级开发工程师

关注“阿里巴巴云原生”公众号
获取第一手技术资料

1 容器与镜像

2 容器生命周期

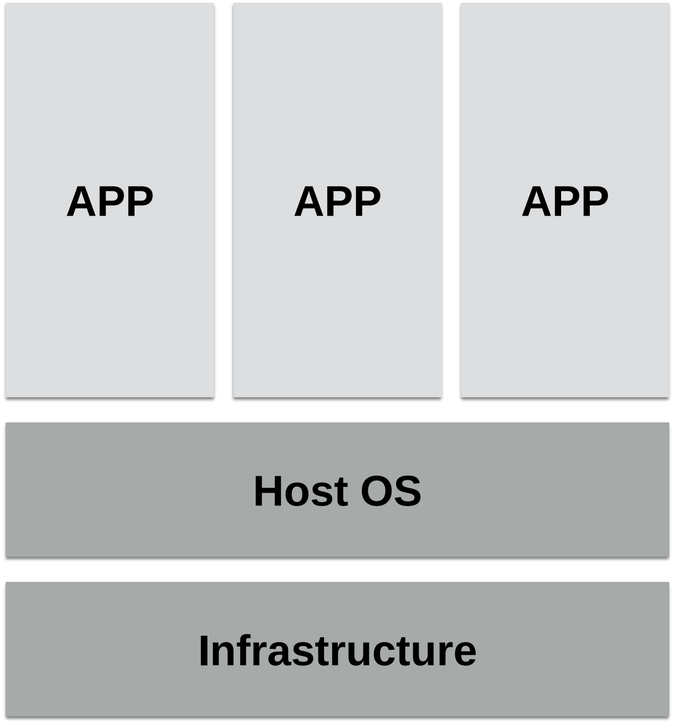
3 容器项目的架构

4 容器 vs VM

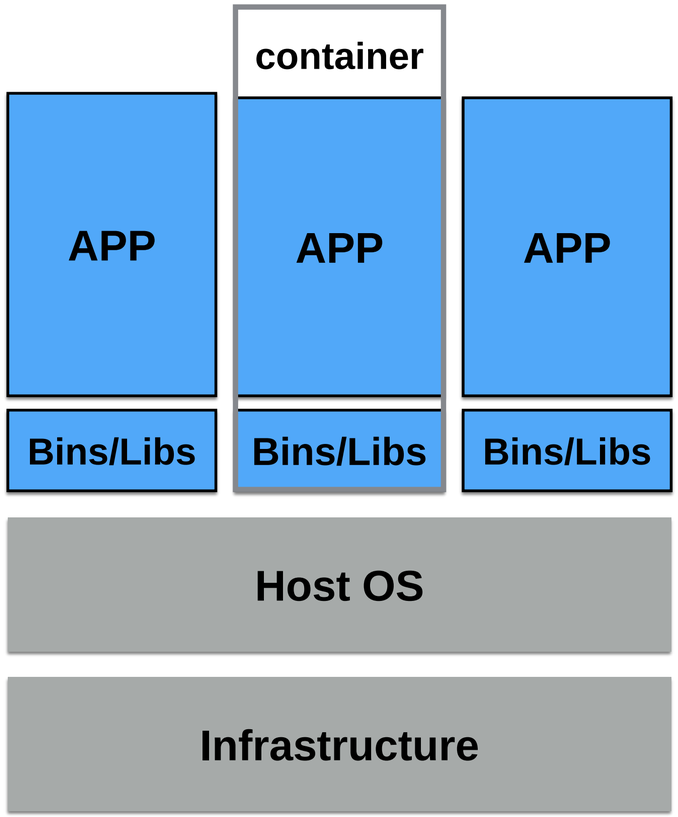
5 课后实践

1 容器与镜像

什么是容器?



* 进程可见、可相互通信
* 共享同一份文件系统



* 资源视图隔离 - namespace
* 控制资源使用率 - cgroup
* 独立的文件系统 - chroot

什么是容器?

容器，是一个视图隔离、资源可限制、独立文件系统的进程集合。

* 视图隔离 - 如能看见部分进程；独立主机名 等等；
* 控制资源使用率 - 如 2G 内存大小；CPU 使用个数 等等；

什么是镜像?

运行容器所需要的所有文件集合 - 容器镜像

Dockerfile - 描述镜像构建步骤

构建步骤所产生出文件系统的变化 - changeset
★类似 disk snapshot
★提高分发效率，减少磁盘压力

如何构建镜像?

编写 Dockerfile - app:v1

$ docker build . -t app:v1

docker registry - 镜像数据的存储和分发

$ docker push app:v1

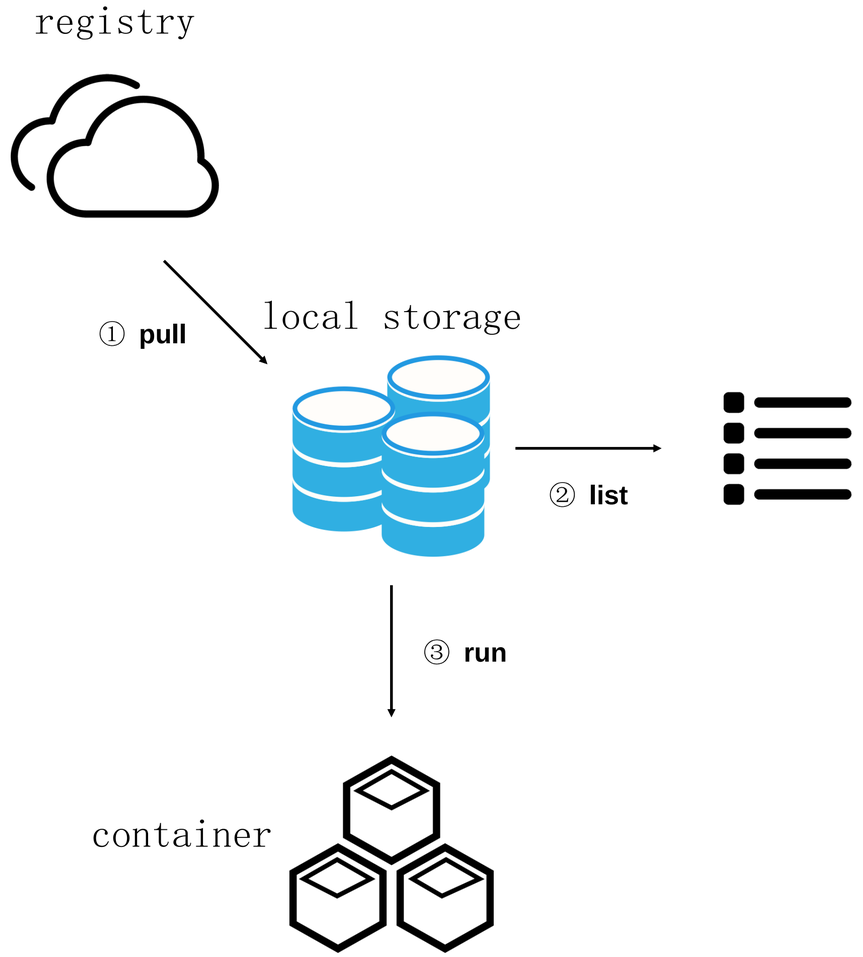
```
1  # base on golang:1.12-alpine image
2  FROM golang:1.12-alpine
3
4  # setting current working dir (PWD -> /go/src/app)
5  WORKDIR /go/src/app
6
7  # copy local files into /go/src/app
8  COPY . .
9
10 # get all the dependencies
11 RUN go get -d -v ./...
12
13 # build the application and install it
14 RUN go install -v ./...
15
16 # by default, run the app
17 CMD ["app"]
```

如何运行容器?

① 从 docker registry 下载镜像 - docker pull busybox:1.25

② 查看本地镜像列表 - docker images

③ 选择相应的镜像并运行 - docker run [-d] —name demo busybox:1.25 top



小节

容器 - 和系统其他部分隔离开的进程集合

镜像 - 容器所需要的所有文件集合 - Build once, Run anywhere

1 容器与镜像

2 容器生命周期

3 容器项目的架构

4 容器 vs VM

5 课后实践

2 容器生命周期

容器运行时的生命周期

单进程模型

① Init 进程生命周期 = 容器生命周期
② 运行期间可运行 exec 执行运维操作

数据持久化

① 独立于容器的生命周期
② 数据卷 - docker volume vs bind

```
# bind host dir into container
$ docker run -v /tmp:/tmp busybox:1.25 sh -c "date > /tmp/demo.log"

# check result
$ cat /tmp/demo.log
Tue Apr  9 02:17:55 UTC 2019

# let it handled by docker container engine
$ docker create volume demo

# demo is volume name
$ docker run -v demo:/tmp busybox:1.25 sh -c "date > /tmp/demo.log"

# check result
$ docker run -v demo:/tmp busybox:1.25 sh -c "cat /tmp/demo.log"
Tue Apr  9 02:19:57 UTC 2019
```

1 容器与镜像

2 容器生命周期

3 容器项目的架构

4 容器 vs VM

5 课后实践

3 容器项目的架构

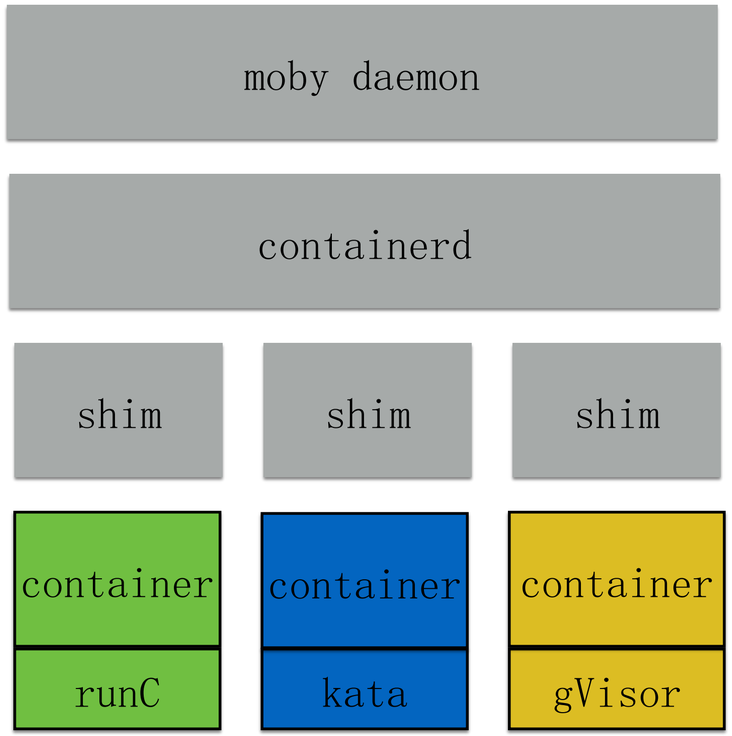
moby 容器引擎架构

containerd

① 容器运行时管理引擎，独立于 moby daemon
② containerd-shim 管理容器生命周期，可被 containerd 动态接管

容器运行时

① 容器虚拟化技术方案
② runC kata gVisor



1 容器与镜像

2 容器生命周期

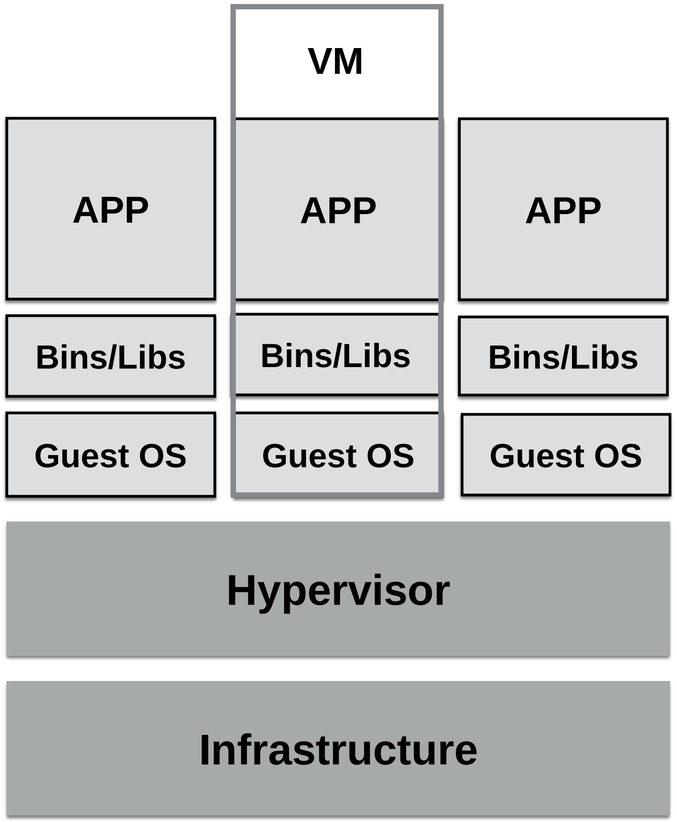
3 容器项目的架构

4 容器 vs VM

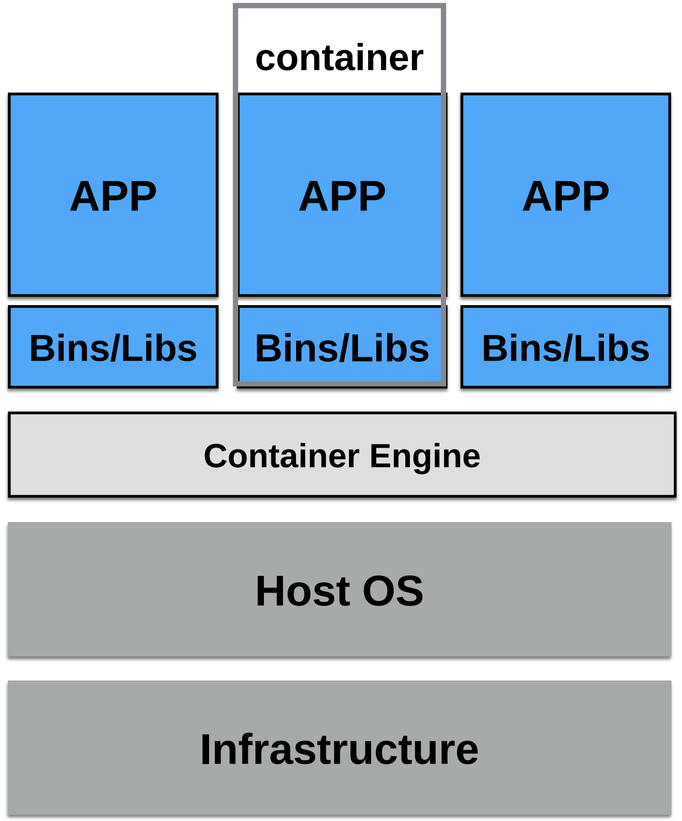
5 课后实践

4 容器 vs VM

容器和VM之间的差异



① 模拟硬件资源，需要 Guest OS
② 应用拥有 Guest OS 所有资源
③ 更好地隔离效果 - Hypervisor 需要消耗更多地资源



① 无 Guest OS，进程级别的隔离
② 启动时间更快
③ 隔离消耗资源少 - 隔离效果弱于 VM

1 容器与镜像

2 容器生命周期

3 容器项目的架构

4 容器 vs VM

5 课后实践

预告

- 容器镜像的本质
- 深入了解容器隔离(namespace) 和 资源控制(cgroup)
- 容器数据卷的实现方式
- containerd 架构分析和工作原理解析

谢谢观看

THANK YOU

关注“阿里巴巴云原生”公众号
获取第一手技术资料