Docker

一、Docker简介

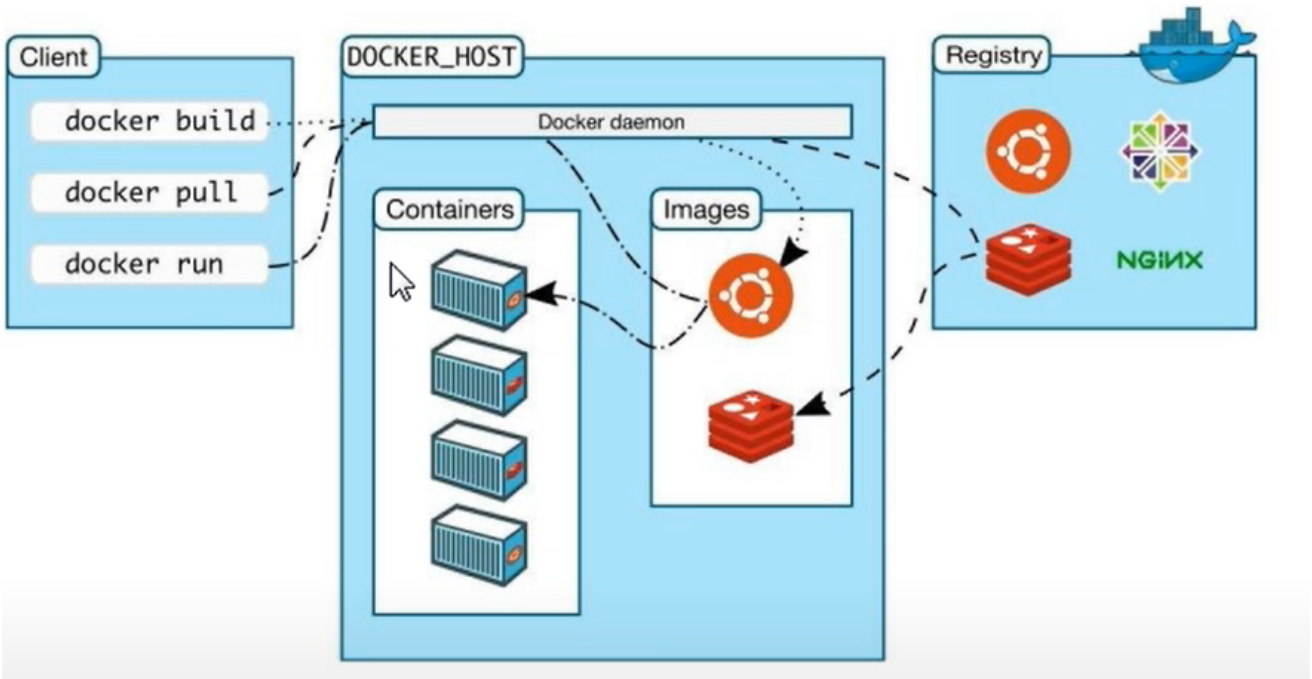
1、什么是Docker

> *“Docker是一个开源的应用容器引擎，基于GO，遵从Apache2.0协议开源。”*

Docker可以让开发者打包他们的应用以及依赖包到一个轻量级、可移植的容器中，然后发布到任何流行的Linux机器上，也可以实现虚拟化。

容器是完全的沙箱机制，互相之间不会有任何接口，性能开销极低。

2、Docker架构原理



> *“Docker三要素：**镜像**、**容器**、**仓库**。”*

1、镜像

镜像（Image）是一个只读的模板，它可以是一个可运行软件（tomcat,Mysql），也可以是一个系统（CentOS）。镜像可以用来创建容器，一个镜像可以创建很多容器。

2、容器

Docker利用容器（Container），可以独立运行一个或一组应用。容器是用镜像创建的运行实例。它可以被启动、开始停止删除。每个容器都是互相隔离的、保证安全的平台。

3、仓库

仓库是集中存放镜像文件的场所。

3、Docker用处

1 、简化环境搭建

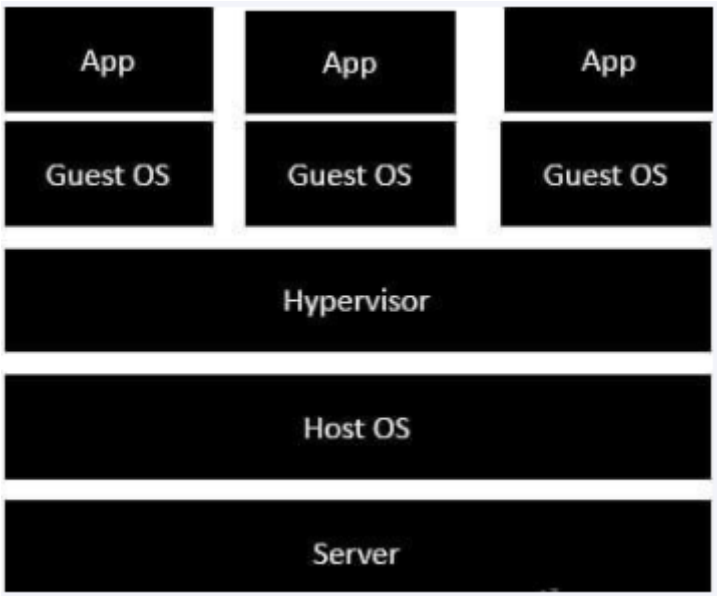
2、 简化运维工作量

3、 微服务利器

4、 Docker与虚拟机的区别

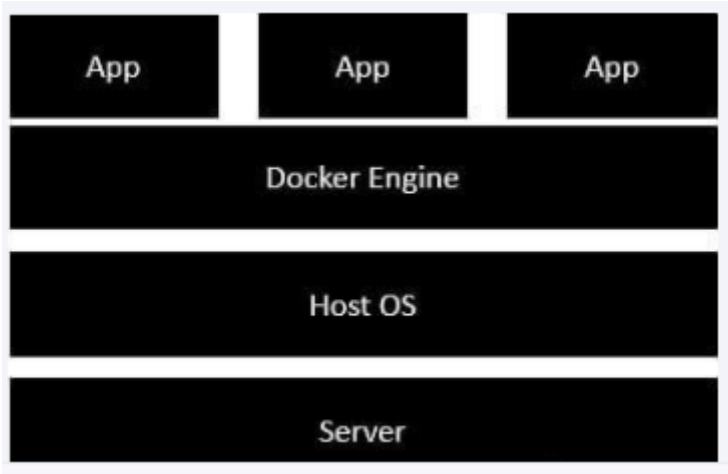
> *“Docker是一个轻量级的虚拟化技术，比传统的虚拟机性能更好。”*

虚拟机体系结构：



- Server：真实的电脑。
- Host OS：真实电脑操作系统。
- Hypervisor：虚拟平台。
- Guest OS:虚拟平台上的系统。
- APP：虚拟平台操作系统中的应用。

Docker体系结构：



- Server：真实的电脑。
- Host OS：真实电脑操作系统。
- Docker Engine：Docker虚拟化技术。

- APP：所有的应用程序现在都作为Docker容器运行。

`这种体系结构的明显优势是，不需要为虚拟机操作系统提供硬件模拟。所有应用程序都作为**Docker**容器工作，性能更好。`

	Docker容器	虚拟机（VM）
操作系统	与宿主机共享OS	宿主机OS上运行宿主机OS
存储大小	镜像小，便于存储与传输	镜像庞大（vmdk等）
运行性能	几乎无额外性能损失	操作系统额外的cpu、内存消耗
移植性	轻便、灵活、适用于Linux	笨重、与虚拟化技术耦合度高
硬件亲和性	面向软件开发者	面向硬件运维者

二、 Docker 安装

> *“官方安装教程CentOS7。”*

> *“更换阿里云镜像”*

三、Docker基本命令

1、启动：`systemctl start docker`

2、重启：`systemctl restart docker`

3、停止：`systemctl stop docker`

4、开机启动：`systemctl enable docker`

5、查看概要信息：`docker info`

6、查看帮助文档：`docker --help`

7、查看版本信息：`docker version`

四、Docker镜像

1、`docker images`列出Docker本地所有镜像。

REPOSITORY	镜像的仓库源
TAG	镜像的标签（版本）
IMAGE ID	镜像的ID
CREATE	创建日期
SIZE	镜像大小

OPTIONS：可选参数：

-a	显示所有镜像（包括中间层）
-q	只显示镜像ID
-qa	可以组合
--digests	显示镜像的摘要信息
--no-trunc	显示完整的镜像信息

2、`docker search`搜索镜像

OPTIONS：可选参数：

--no-trunc	显示完整的镜像描述

--no-trunc	显示完整的镜像描述
-s	列出收藏数不小于指定值的镜像
--automated	只列出Docker Hub自动构建类型的镜像

3、`docker pull`[:TAG]下载镜像

不加TAG默认下载latest。

4、`docker rmi`删除镜像

不加TAG默认删除latest。

有镜像生成的容器在运行时候会报错，删除失败。

删除多个：docker rmi -f 镜像名称1:[TAG] 镜像名称2:[TAG]

删除全部：docker rmi -f $(docker images -qa)

五、Docker容器

1、创建并启动

> *“`dcoker run [OPTIONS] IMAGE [COMMAND] [ARG...]`”*

`--name "IMAGE NAME"`	为容器指定一个名称
`-i`	以交互模式运行容器，通常与-t或者-d同时使用
`-t`	为容器重新分配一个伪输入终端，通常与-i同时使用
`-d`	后台运行容器，并返回容器ID

`--name "IMAGE NAME"`	**为容器指定一个名称**
`-P`	**随机端口映射，容器内部端口随机映射到主机的端口**
`-p`	**指定端口映射，格式为：主机(宿主)端口:容器端口**

- **常用命令：**
  - 启动普通容器：`docker run --name 别名 镜像ID`
  - 启动交互式容器：`docker run -it --name 别名  镜像ID`
  - 守护式方式创建并启动容器：`docker run -di --name 别名 镜像ID`
  - 启动容器，并执行/bin/bash命令：`docker run -it --name 别名 镜像ID /bin/bash命令`
  - 端口映射：`docker run -it -p 8888:8080 tomcat`或`docker run -it -P tomcat`

2、列出容器

> “`docker ps [OPTIONS]`”

`-a`	**列出所有容器**
`-f`	根据条件筛选内容
`--format`	指定返回值的模板文件
`-l`	显示最近创建的容器
`-n`	显示最近创建的n个容器
`--no-trunc`	不截断输出
`-q`	静默模式，只显示镜像ID

`-a`	列出所有容器
`-s`	显示总的文件大小

- **常用命令：**
  - `docker ps` 查看正在运行的容器
  - `docker ps -a` 查看所有容器
  - `docker ps -n 2` 显示最近创建的2个容器
  - `docker ps -f status=exited` 查看停止的容器

3、退出容器

- `exit`退出容器并停止
- `CTRL + P + Q`容器不停止退出

4、进入容器

- `docker attach 容器ID或容器名`

5、启动容器

- `docker start 容器ID或容器名`

6、重启容器

- `docker restart 容器ID或容器名`

7、停止容器

- `docker stop 容器ID或容器名`
- 暴力删除：`docker kill 容器ID或容器名`(不推荐)

8、删除容器

- `docker rm -f 容器ID`强制删除
- `docker rm -f 容器ID1 容器ID2`删除多个容器，中间空格隔开
- `docker rm -f $(docker ps -qa)`删除所有容器

9、宿主机和容器之间的文件拷贝

1、宿主机文件拷贝到容器

- `docker cp 需要拷贝的文件或者目录  容器名称：容器目录`

2、容器文件拷贝到宿主机

- `docker cp 容器名称：容器目录  宿主机目录`

10、查看容器日志

日志文件记录在`var\lib\docker\containers\容器ID\XX.log`

11、查看容器进程

`docker top 容器ID`

12、进入容器执行命令

`docker exec -it [容器名称 或者 容器ID] 执行命令`

13、提交运行时容器成为镜像

`docker commit -a='作者' -m='备注' 运行时容器ID 新镜像名称`

14、推送镜像到Hub服务器

15、推送镜像到阿里云

16、查看容器元信息

六、容器目录挂载

1、简介

我们可以在创建容器的时候，将宿主机的目录与容器内的目录进行映射，这样我们就可以实现宿主机和容器目录的双向数据自动同步；

2、实现

- 单目录挂载

  `docker run -it -v /宿主机目录:/容器目录 镜像名`

- 多目录挂载

  `docker run -it -v /宿主机目录:/容器目录 -v /宿主机目录2:/容器目录2 镜像名`

- 挂载目录为只读

  `docker run -it -v /宿主机目录:/容器目录:ro 镜像名`