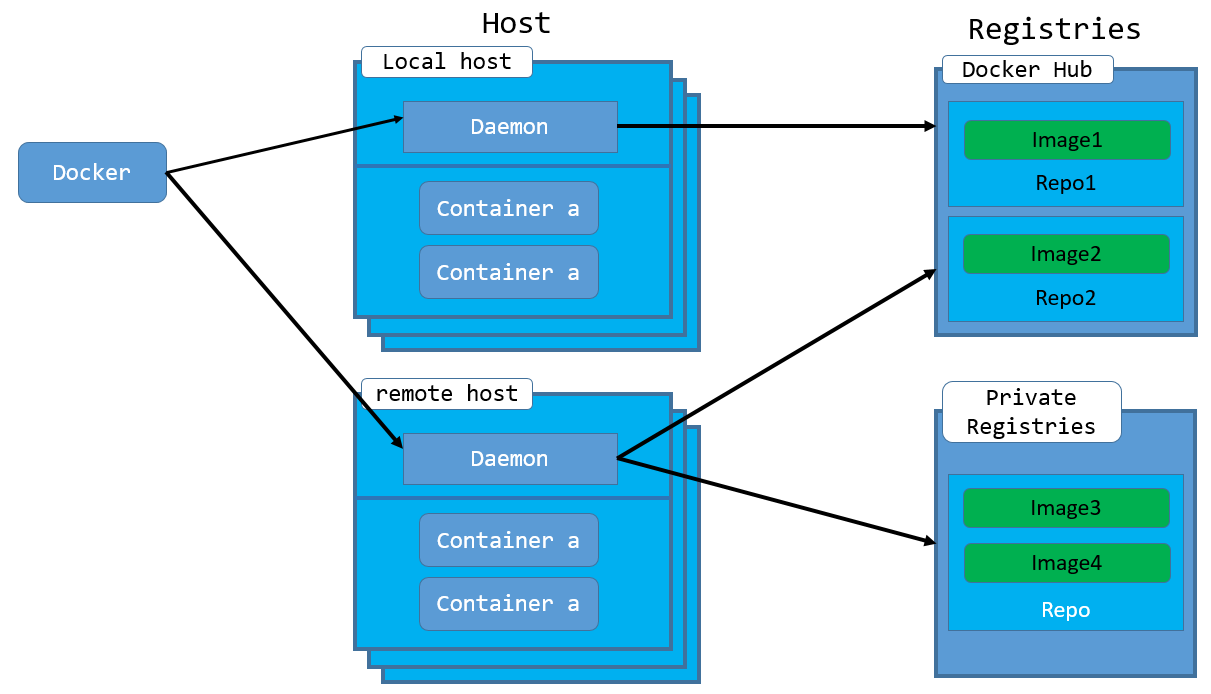
Docker 改变了什么？

1. 运维，保证开发环境，测试环境，生产环境一致
2. 产品，应用隔离
3. 开发，开发环境不需要管理环境问题
4. 测试，多版本并行测试

Docker架构



1.Docker简介和KVM区别

**历史简介**

Docker是PaaS供应商dotCloud开源的一个基于LXC 的高级容器引擎，源代码托管在 GitHub 上, 基于Go语言开发并遵从Apache 2.0协议开源.Docker 是通过内核虚拟化技术 （namespaces及cgroups等，这里的内核技术指的是Linux内核哦）来提供容器的资源 隔离与安全保证等。由于docker通过操作系统层的虚拟化实现隔离，所以Docker容器 在运行时，不需要额外的虚拟化管理程序（VMM(Virtual Machine Monitor)，以及 hyperisor）支持，他是内核级虚拟化，可以实现更高的性能，同时对资源的额外需求 很低。最本质特征docker是通过隔离来进行创建容器，而KVM等均为通过模拟方式创 建虚拟机

**优势说明**

更快的交付和部署

使用docker 开发人员可以使用镜像来快速构建一套标准的开发环境；开发完成之后， 测试和原味人员可以直接使用完全相同的环境来部署代码，只要开发测试过的代码， 就可以确保在生产环境无缝运行。docker可以快速创建和删除容器，实现快速迭代， 大量节约开发、测试、部署的时间

更高效的资源利用

docker 容器不需要二外的徐牛啊管理程序支持，他是内核级的虚拟化，可以实现更高 级的性能，同事对资源的额外需求很低

更轻松的迁移和扩展

docker容器几乎可以再任意的平台上运行，包括物理机、虚拟机、公有云、私有云、 个人电脑、服务器等，同事支持主流的操作系统发行版本。这种兼容性让用户可以再 不同平台之间轻松地迁移应用

**对比KVM**

docker容器很快，启动和停止可以再秒级实现，而传统的虚拟机需要数分钟

docker容器对系统资源需求很少，一台主机上可以同时运行数百甚至上千个

docker容器 docker 通过类似于git的设计理念的操作来方便用户获取、分发和更新应用镜像，存储复用，增量更新

docker通过dockerfile支持灵活的自动化创建和部署机制，提高工作效率，使流程标准化

**Docker的特性**

Docker的三大特性为BUILD（构建）、SHIP（运输）、RUN（运行）。（一次构建多处运 行，像不像JAVA呢）。docker是传统的CS架构分为docker client和docker server，主要分 为三大组件 镜像（image）、容器（container）、仓库（Repository）。那么docker和我们 之前接触的openstack又有什么区别呢？他们的区别如下所示。

|  |  |  |
| --- | --- | --- |
| 类别 | Docker | OpenStack |
| 部署难度 | 非常简单 | 组件多，部署负责 |
| 启动速度 | 秒级 | 分钟级 |
| 执行性能 | 和物理系统几乎一致 | VM会占用一些资源 |
| 镜像体积 | 镜像MB级别 | 虚拟机镜像GB |
| 管理效率 | 管理简单 | 组件互相依赖 |
| 隔离性 | 隔离性高 | 彻底隔离 |
| 网络连接 | 弱 | 借助Neutron可以灵活配置网络 |

**Docker操作指南**

Docker安装

$ sudo apt-get install docker

使用apt-get

# step 1：安装必要的系统工具

$ sudo apt-get update

$ sudo apt-get -y install apt-transport-https ca-certificates curl software-properties-common

# step 2：安装GPG证书

$ curl -fsSL http://mirrors.aliyun.com/docker-ce/linux/ubuntu/gpg | sudo apt-key add –

$ sudo add-apt-repository "deb [arch=amd64] http://mirrors.aliyun.com/docker-ce/linux/ubuntu $(lsb\_release -cs) stable"

$ sudo apt-get update

$ sudo apt-get install docker-ce –y

查看docker-ce版本

$ sudo apt-cache madison docker-ce

安装指定版本的Docker-CE

$ sudo apt-get install docker-ce=17.06.0~ce-0~ubuntu

# docker -v

镜像仓库：[https://cr.console.aliyun.com](https://cr.console.aliyun.com/)

**Docker镜像操作**

查找Nginx

# docker search nginx

拉取Docker 官方image

# docker pull nginx

Docker Image导出

# docker save nginx > ./nginx.tar.gz

Docker Image导入

# docker load < nginx.tar.gz

Docker Image删除

# docker rmi 719cd2e3ed04

**Docker容器操作**

启动docker,前台运行

# docker run -it 719cd2e3ed04

启动docker,后台运行

# docker run –it –d --name mynginx 719cd2e3ed04

推出容器，并不关闭  ctl+p+q

容器删除

# docker rm mynginx

查看容器信息

# docker inspect mynginx

进入容器

# docker attach mynginx

进入容器

# docker exec -it mynginx sh

Docker Hub操作

# docker login

> Username :

> Password :

镜像上传

# docker images

# docker tag c7109f74d339 docker.io/wangbojing/mysql

# docker push docker.io/wangbojing/mysql

Docker 端口映射

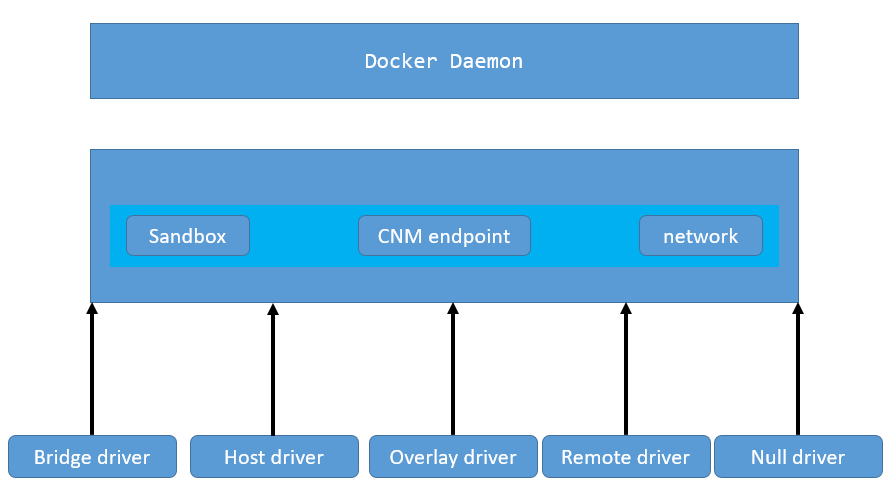
Redis镜像启动

# docker run --name myredis -d -p 127.0.0.1:6379:6379 3c41ce05add9

Nginx镜像启动

# docker run --name mynginx -d -p 127.0.0.1:80:80 719cd2e3ed04

Docker网络虚拟化架构



查看网络类型

# docker network ls

**Docker磁盘操作**

文件夹挂载

# docker run -it -d --name nginx\_node --rm -v /home/wangbojing/share:/opt/ 719cd2e3ed04

文件挂载

# docker run -it -d --name nginx\_node --rm -v /etc/hosts:/etc/hosts 719cd2e3ed04

**Dockerfile 构建镜像**

Dockerfile编写

Docker镜像编排

# curl -L https://github.com/docker/compose/releases/download/1.21.2/docker-compose-$(uname -s)-$(uname -m) -o /usr/local/bin/docker-compose