目录

[第一章 docker的基础 2](#_Toc6489695)

[1、安装docker 2](#_Toc6489696)

[2、卸载docker 2](#_Toc6489697)

[3、使用镜像 2](#_Toc6489698)

[4、镜像迁移 2](#_Toc6489699)

[5、docker Hub 2](#_Toc6489700)

[5.1、搭建私有镜像仓库 2](#_Toc6489701)

[6、使用容器 3](#_Toc6489702)

# 第一章 docker的基础

## 1、安装docker

1.安装依赖包

yum install -y yum-utils device-mapper-persistent-data lvm2

2.添加阿里yum源

yum-config-manager --add-repo http://mirrors.aliyun.com/docker-ce/linux/centos/docker-ce.repo

3.更新yum缓存

yum makecache fast

4.安装docker

yum -y install docker-ce

5.启动docker

systemctl start docker #开机自启动enable

6.启动一个容器

docker run -itd -p 80:80 --name webserver nginx /bin/bash #先从本地仓库查找nginx镜像,如果没有会从官网上查找

## 2、卸载docker

yum remove docker-ce

rm -rf /var/lib/docker

## 3、使用镜像

docker pull ubnutu #从互联网上下载一个镜像到本地仓库,默认使用latest这个标签

docker push 注册用户名/镜像名 #推送镜像到远程镜像仓库

docker search php #默认从官网查找php的镜像

docker images #列出本地镜像

docker inspect nginx:latest #获取docker镜像的详细信息

docker inspect -f '{{.NetworkSettings.IPAddress}}' mynginx #获取在运行的mynginx容器IP

docker rmi ubuntu:latest #删除本地仓库的镜像，如果正在使用可加上 -f 强制删除但不建议使用

Docker rmi $(docker images -a|grep none|awk '{print $3}') 删除没有打tag的镜像

## 4、镜像迁移

docker save nginx:latest > nginx.tar.gz #将镜像导出来，就是指保存到本地

docker save -o nginx.03.tar nginx:latest #或者使用这个命令

docker load < nginx.tar.gz #将导出的镜像重新导入镜像库

docker load -i nginx.tar.gz #或者使用这个命令

## 5、docker Hub

官方提供的镜像仓库网址：https://hub.docker.com/

docker login -u user -p password server\_url #登陆docker Hub

直接输入 docker login 也可以登录 默认登陆的是https://hub.docker.com/

### 5.1、搭建私有镜像仓库

需要基于docker环境,

**1、安装docker-compose**

版本下载地址：https://github.com/docker/compose/releases/

二进制安装：

curl -L https://github.com/docker/compose/releases/download/1.16.1/docker-compose-`uname -s`-`uname -m` -o /usr/local/bin/docker-compose

chmod +x /usr/local/bin/docker-compose

**1.1、根据自己的情况决定是否安装命令补全功能**

yum install bash-completion

或者

curl -L https://raw.githubusercontent.com/docker/compose/1.16.1/contrib/completion/bash/docker-compose -o /etc/bash\_completion.d/docker-compose

检查是否安装成功

docker-compose --version

其他安装方式docker-compose

yum install python-pip

pip install docker-compose

**2、安装harbor**

在线安装的方式

wget -P /usr/loca/src/ https://github.com/vmware/harbor/releases/download/v1.2.0/harbor-online-installer-v1.2.0.tgz

离线安装的方式

https://github.com/vmware/harbor/releases #下载地址

tar -zxvf harbor-1.7.5.tar.gz

cd /usr/local/harbor/

修改配置文件

1.harbor.cfg

#### **5.1.1、docker开源的镜像分发工具**--docker Registry

部署方式参考：<https://www.cnblogs.com/Eivll0m/p/7089675.html> https://www.cnblogs.com/feinian/p/7857430.html

GitHub网址：https://hub.docker.com/\_/registry/

Docker registry 是用于打包，传输，存储和分发的工具

1、安装docker-registry

docker run -d -p 5000:5000 --restart=always --name registry -v /opt/registry:/var/lib/registry registry:2

服务启动之后就可以向她推送和拉取镜像了

#### 5.1.2、harbor部署

## 6、使用容器

docker create debian:jessie #创建容器但不启动

docker run -it debian:jessie /bin/bash #运行容器,通过bash进入debian系统,退出容器后会关闭.

docker run --restart=always -itd centos:latest /bin/bash #后台运行容器,加上--restart=always参数随宿主机一同启动，其他可选参数如下

-a stdin: 指定标准输入输出内容类型，可选 STDIN/STDOUT/STDERR 三项；

-d: 后台运行容器，并返回容器ID；

-i: 以交互模式运行容器，通常与 -t 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；

-P: 容器的80端口映射到主机的随机端口

-p: 容器的端口映射到主机的对应端口，例如： -p 80:80

-v: 主机的目录映射(挂载)到容器的目录，例如：-v /home/ubuntu/nginx/www:/www

-h "mars": 指定容器的hostname；

-e username="ritchie": 设置环境变量；

-c 或 --cpu-shares：设置容器使用cpu权重；

-m 或 --memory：设置内存使用限额。例如 -m 200M、--memory 300M；

--memory-swap：设置内存+swap的使用限额，当-m 200M --memory-swap=300M时，表示容器可以使用200M内存和100Mswap；

--vm：启动内存工作线程数。例如：--vm 1，启动1个内存工作线程；

--vm-bytes 280M：每个工作线程分配280M内存；

--dns 8.8.8.8: 指定容器使用的DNS服务器，默认和宿主一致；

--dns-search example.com: 指定容器DNS搜索域名，默认和宿主一致；

--env-file=[]: 从指定文件读入环境变量；

--cpuset="0-2" or --cpuset="0,1,2": 绑定容器到指定CPU运行；

--net="bridge": 指定容器的网络连接类型，支持 bridge/host/none/container:<name|id> 四种类型；

--link=[]: 添加链接到另一个容器；

--expose=[]: 开放一个端口或一组端口；

--name: 为容器指定一个名称；

docker exec -it CONTAINER ID(容器的ID) /bin/bash #进入正在运行中的容器

docker restart start stop 镜像名或容器ID # 重启 启动 关闭 容器

docker rm 容器id #删除容器

docker kill #发送信号给容器默认SIGKILL例如：docker kill -s KILL mycon (-s表示向容器发送一个信号)

docker attach #进入)正在运行的容器，例如：docker attach --sig-proxy=false mycon

docker wait #阻塞到一个容器，直到容器停止运行。例如：docker wait mycon

docker pause #暂停容器中所有的进程。 例如：docker pause mycon

docker unpause #恢复容器中所有的进程。 例如：docker unpause mycon

docker ps #查看容器的状态

-a 列出所有容器 包含未运行的

-l 列出最新创建的容器

-n=2 列出最近创建的2个容器

-q 仅列出容器ID

-s 显示容器大小

docker logs # 查看容器的日志(stdout/stderr)

　　　　-f : 跟踪日志输出,例如：docker logs -f mycon（查看容器mycon的日志输出）

　　　　--since :显示某个开始时间的所有日志

　　　　-t : 显示时间戳

　　　　--tail :仅列出最新N条容器日志，例如：docker logs --since="2017-05-01" --tail=10 mycon（查看容器mycon从2017年5月1日后的最新10条日志。）

docker events # 得到docker服务器的实时的事件

　　　　-f ：根据条件过滤事件；例如：docker events -f "image"="mysql:5.6" --since="1466302400" （显示docker 镜像为mysql:5.6 这个时间戳对应的日期之后的相关事件。）

　　　　--since ：从指定的时间戳后显示所有事件;例如：docker events --since="1466302400" （显示docker 在这个时间戳对应的日期之后的所有事件。）

　　　　--until ：流水时间显示到指定的时间为止；

docker port # 显示容器的端口映射，例如：docker port mycon

docker top # 显示容器的进程信息，支持ps参数。例如docker top mycon

docker diff # 显示容器文件系统的前后变化， 检查容器里文件结构的更改。例如：docker diff mycon

docker cp /www/test mycon:/www/ #将主机的/www/test目录拷贝到容器mycon的/www目录下

docker cp mycon:/www /tmp/test #将容器mycon中的/www目录拷贝到主机的/tmp/test目录中

### 6.1、容器保存和迁移

容器中的所有更改都是在沙盒环境下,当容器停止后所有的更改都会丢失,如果需要保存修改，可以用docker commit命令，把修改的容器保存成一个新的镜像,

docker ps -la == 找到CONTAINER ID 然后使用命令

docker commit -m "ifconfig" c738ec435db3 centos:7.4 #保存成镜像 -m 参数为镜像层写一条提交信息

**容器的导入和导出:**不管容器是否处于运行的状态,都可以使用docker export命令将容器保存到压缩文件里

docker ps #找出需要导出的容器ID （加上-a参考，可以列出所有的容器）

docker export 39ec3c12291d > centos.tar.gz #导出容器

docker import centos.tar.gz #导入容器