**搭建私有仓库**

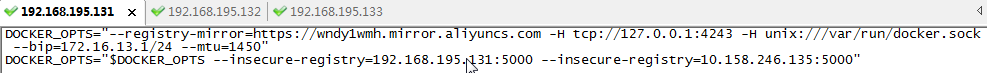
**1.**在master节点创建镜像仓库

docker pull registry

**2.**配置镜像仓库地址，需要在master节点和node节点都添加如下修改。

vi /etc/default/docker

修改成以下格式：

哈哈

修改完成后需要service docker restart（restart docker）

**3.**通过该镜像启动容器，在master上操作

docker run -d -p 5000:5000 registry（不常用）

默认情况下，会将仓库存放于容器内的/tmp/registry目录下，这样如果容器被删除，则存放于容器中的镜像也会丢失，所以我们一般情况下会指定本地一个目录挂载到容器内的/tmp/registry下，如下：

docker run -d -p 5000:5000 -v /opt/data/registry:/tmp/registry registry(有点不太好使)

docker run -d -p 5000:5000 --restart=always --name registry \

-v `pwd`/data:/var/lib/registry \

registry:2

（官网指令，亲测可行，将registry数据保留在`pwd`/data下）

通过执行docker ps查看镜像仓库是否启动，如果启动成功能看到如下，则成功C:\Users\123\Desktop\2.png

**4.测试**，在node节点执行 docker pull hello-world拉去新镜像。

执行命令docker tag hello-world 192.168.195.131:5000/hello-world给镜像添加新标识。

Docker push 192.168.195.131:5000/hello-world将镜像push到master节点的镜像仓库中。

Docker pull 192.168.195.131:5000/hello-world从master节点的镜像仓库获取镜像。

注：pull下来的镜像默认保存在/var/lib/docker/aufs/diff/下

当push到本地仓库时，保存在/var/lib/docker/aufs/mnt/下，挂载目录是由用户自己设置的。

**5.常用命令：**docker images 查看docker 镜像

docker ps 查看当前运行的容器 (docker ps –a)

-l：显示最后启动的容器

-a：同时显示停止的容器，默认只显示启动状态

docker stop [id]stop容器

docker restart [id] restart容器

docker rm [id] 删除容器（-f删除正在运行中容器）

docker rmi [id] 删除镜像(-f强制删除)

docker logs [CONTAINER\_ID]  : 输出容器日志

-f：实时输出

kubectl get pod –all-namespaces 查看所有pod

kubectl delete –namespace=[name] 删除pod

**6.运行镜像：**

<http://www.tuicool.com/articles/yiIVNz>讲创建镜像，容器的。

docker run -t -i ubuntu:14.04 /bin/bash

-i:标准输出给容器

-t:分配一个虚拟终端

-d：以守护进程方式运行（后台）

-p：默认匹配docker容器的5000端口号到宿主机的49153 to 65535端口

-p <HOT\_PORT>:<CONTAINER\_PORT>：指定端口号

docker run 192.168.195.133:5000/wdx-whale cowsay hello

**创建应用(manage)**

应用在k8s-install/k8s-install/docker-apps/下

启动前需要先修改yaml文件，然后执行./build.sh或者手动执行kubectl create -f manage-svc.yaml

1. **启动系统应用**

当前状态下执行kubectl get pod --all-namespaces发现kube-system

的容器并未启动，因为有一些应用的配置文件需要修改,三个节点也需要先下载镜像。

先在master，node1，node2节点上分别下载镜像:

docker pull 10.158.246.135:5000/gcr.io/google\_containers/pause:2.0

docker tag 10.158.246.135:5000/gcr.io/google\_containers/pause:2.0 gcr.io/google\_containers/pause:2.0

进入到k8s-install/k8s-install/system-apps/heapster下:

vi heapster-rc.yaml将其中的source的ip地址改为本机ip，退出。

Kubectl create –f heapster-rc.yaml

Kubectl create –f heapster-svc.yaml

执行kubectl get pod --all-namespaces发现应用都正常启动了。

**体系结构**

Docker下的概念有容器，镜像等。

Kubernetes下的概念有namespace，pod，service,replicationController等**。**（service和repl

icationController只是建立在pod之上的抽象，最终是要作用于pod的）

**docker镜像**

在node节点上安装docker后，需要下载镜像文件，镜像文件就相当于一个微型的操作系统，是一个只读的模板。一个镜像可以包含一个完整的操作系统环境，里面只安装了tomcat和其他用户需要的程序。用户可以从官方或者其他人那里下载已经制作好的镜像。

Docker images 查看当前所有镜像

**Docker容器**

docker利用容器来运行应用，容器是从镜像创建的应用实例，每个容器都是被隔离的，可以单独进行开始，暂停，删除等操作。容器启动时在镜像上建立了一层可写层。

Docker ps 查看当前运行容器

Docker ps -a 查看当前所有容器

**Docker常用指令：**http://www.open-open.com/lib/view/open1423703640748.html

**Kubernetes**

Kubernetes是一个用来管理docker集群的系统，在master节点安装kubernetes后，可以使用kubectl工具来管理docker集群。

Pod是kubernetes最基本的调度单元，一个pod可以包含一个或者多个容器，例如，一个web项目包含前端，后端，数据库，就可以创建成一个包含三个容器的pod。Pod可以对资源进行限定，比如CPU，Memory等。

Namespace是隔离不同用户进程的，一个namespace可以包含很多pod。