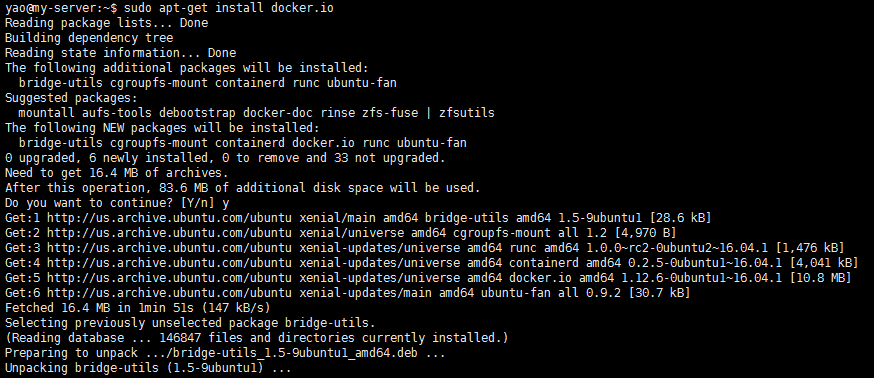
Docker安装部署使用

# 安装Docker

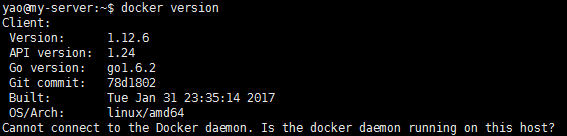
## **从官方的Ubuntu软件库安装**

sudo apt-get update

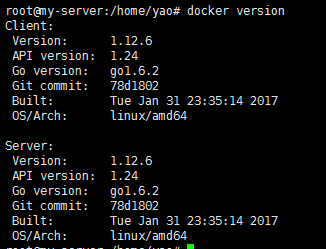
sudo apt-get install docker.io



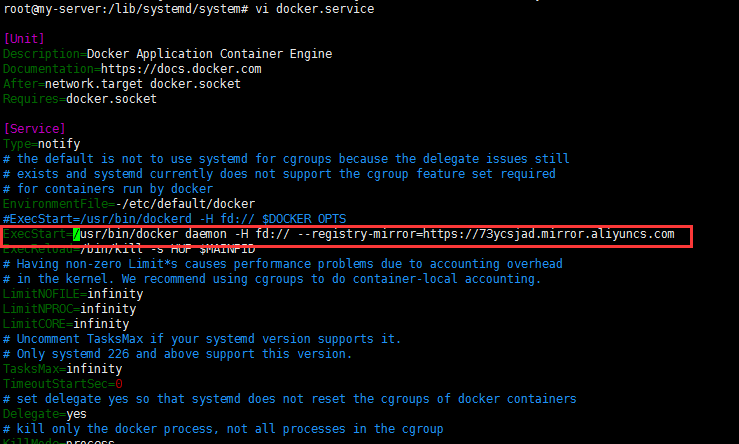
查看版本（注意这时候没有启动）



## 启动docker



## 镜像加速度（pull加速）



1. 注册阿里云开发账号，获取镜像加速地址（<https://dev.aliyun.com/search.html>）

ExecStart=/usr/bin/docker daemon -H fd:// --registry-mirror=https://73ycsjad.mirror.aliyuncs.com

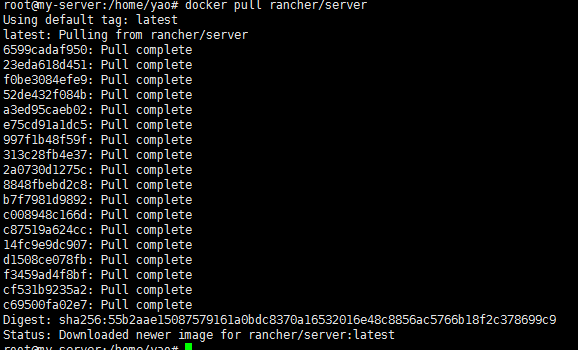
1. vi /lib/systemd/system/docker.service, 修改获取官方镜像地址(粘贴地址，Shift + Insert)
2. systemctl daemon-reload
3. systemctl restart docker

# 部署配置Rancher

## 下载并运行镜像

docker run -d --restart=unless-stopped -p 8080:8080 rancher/server

* 如果本地没有rancher镜像会自动从中心服务器拉去rancher镜像，并启动，可以通过 docker ps 进行查看，如下图：
* 如果下载失败，请多次尝试；或者换国内镜像





## 手动启动(已经pull rancher/server)

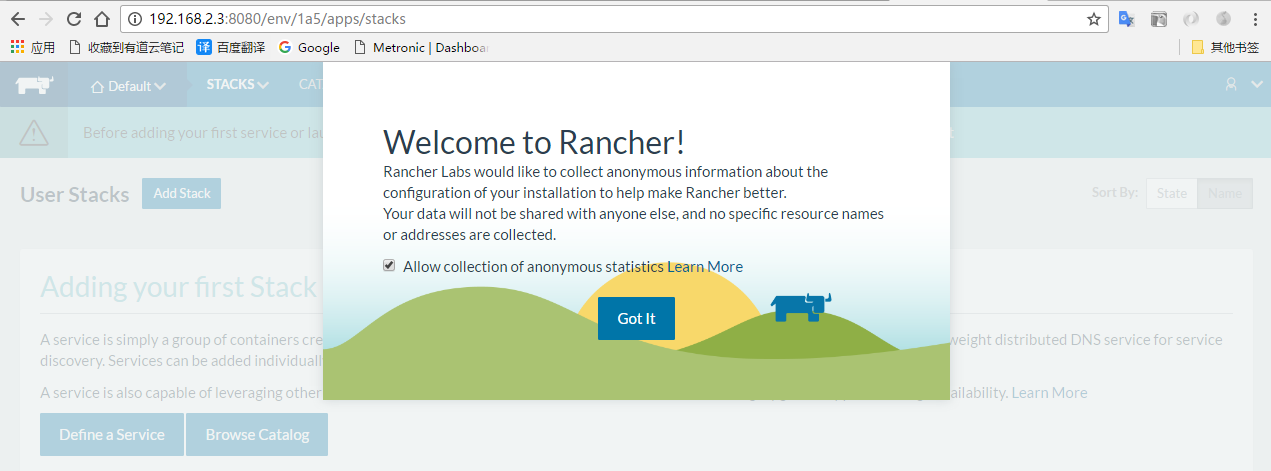
docker run -d --restart=always -p 8080:8080 rancher/server



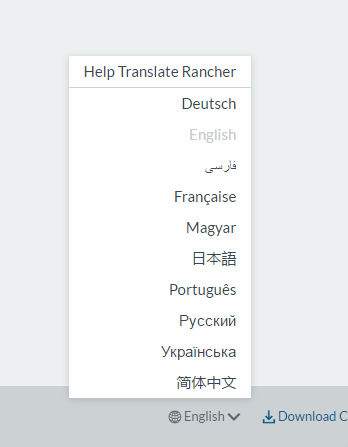


## 测试访问

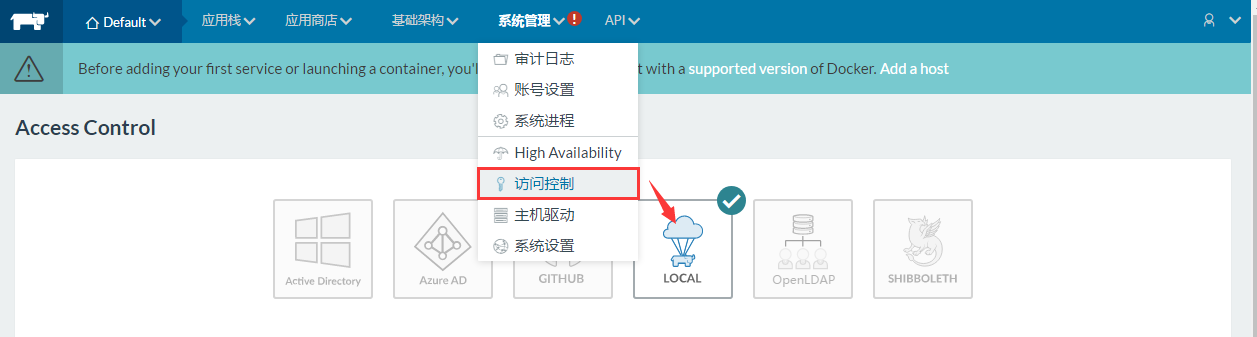
地址：<http://192.168.2.3:8080>

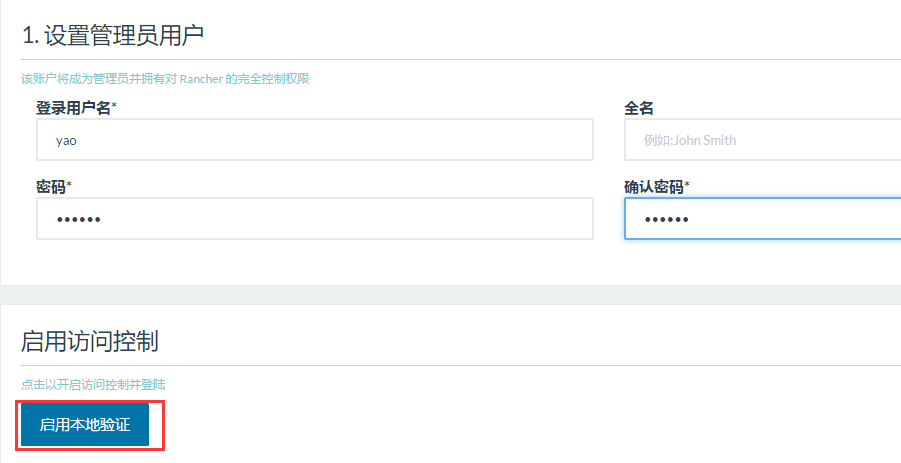


## 设置中文(右下角)

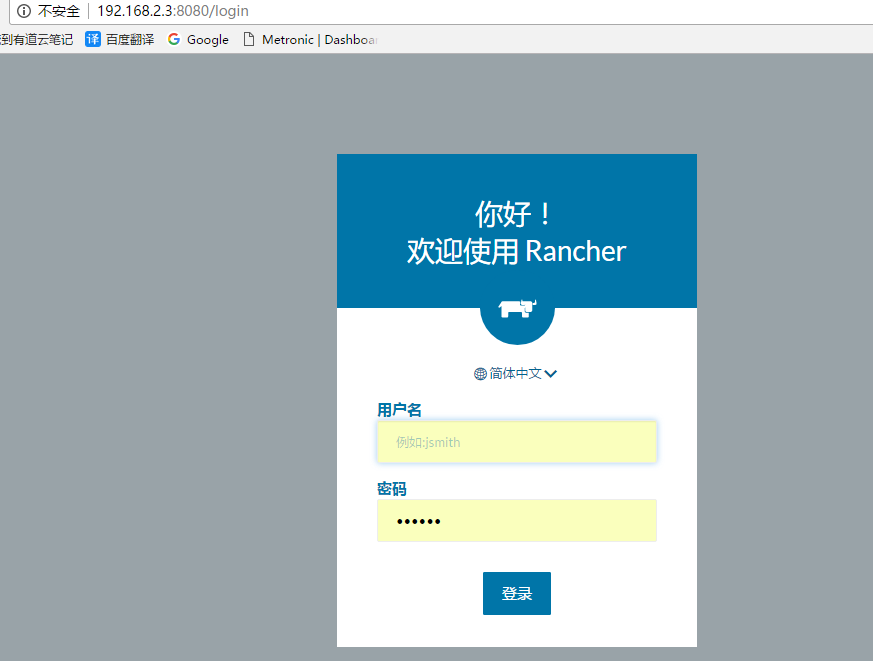


## 用户配置





**添加对应的用户名，密码即可（建议与用户一致）**



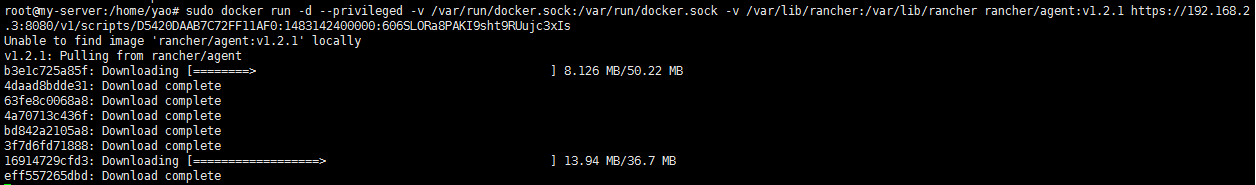
**右上角注销，即可弹出登录页面**

## 添加rancher节点

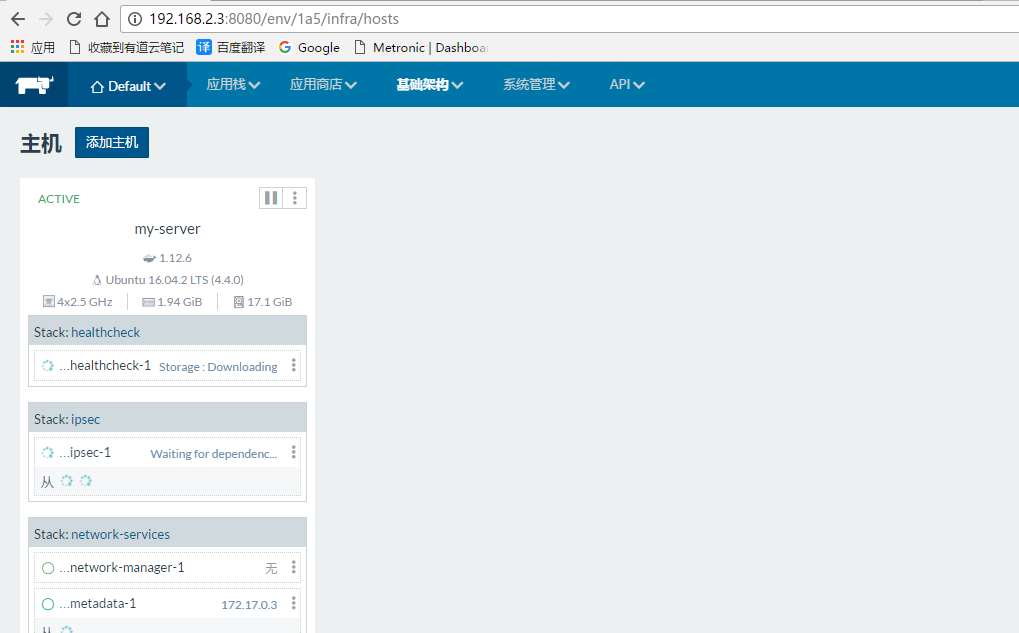




**复制以上文本，在shell端口录入**

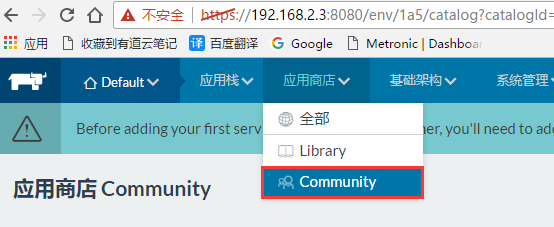
****

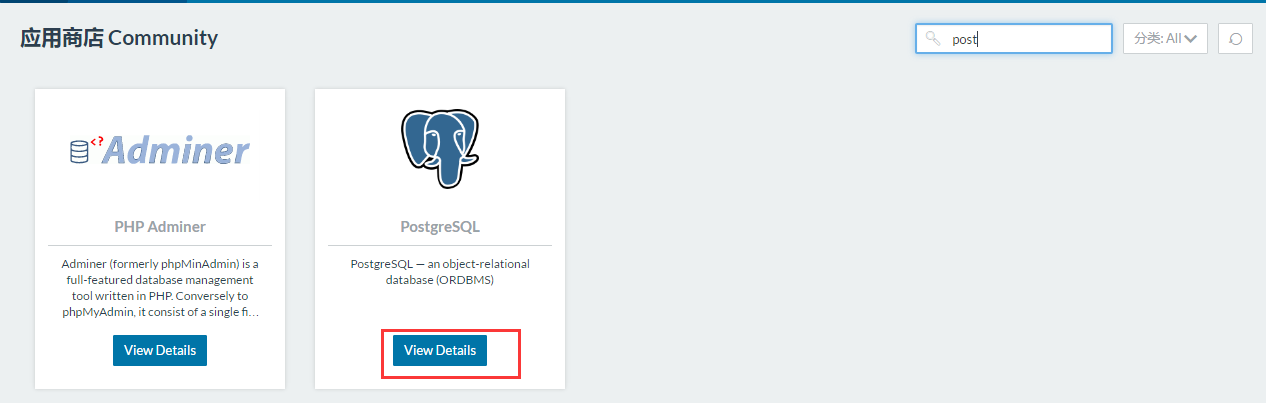
**等命令执行完成【自动下载 rancher/agent 镜像并运行】，在添加主机的页面，点击关闭**

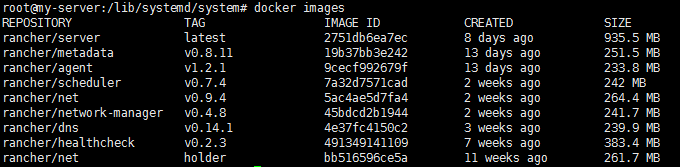


## 添加应用示例

### 根据应用商店分类，搜索对应的docker服务

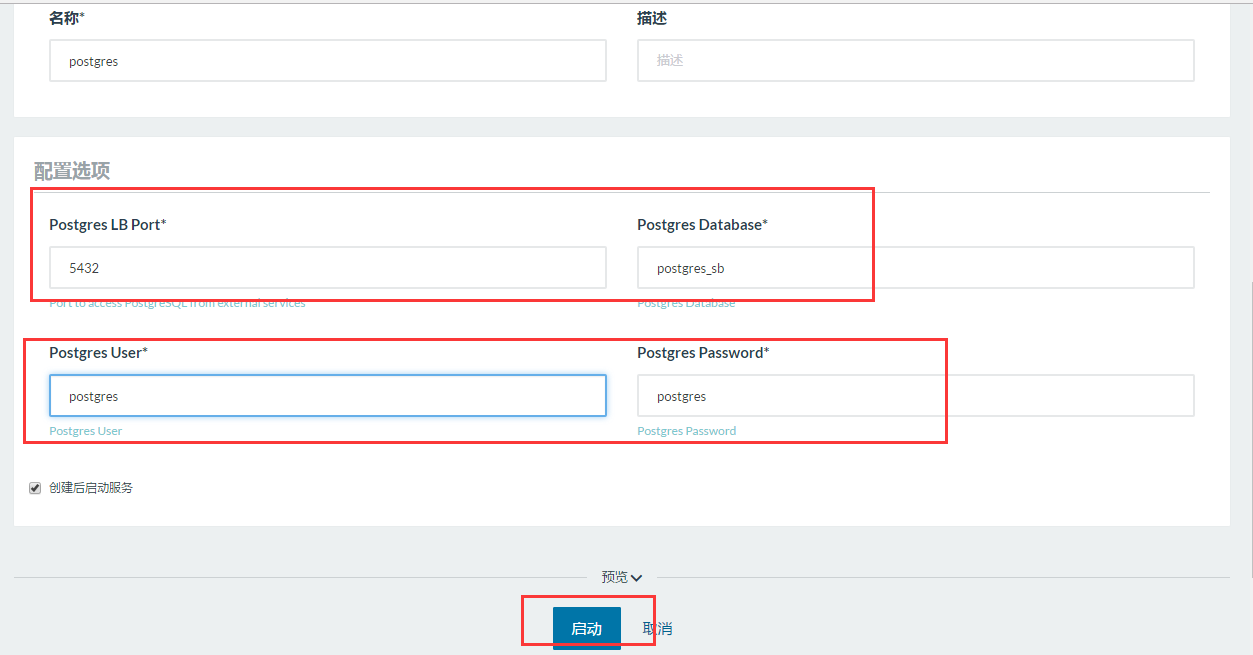


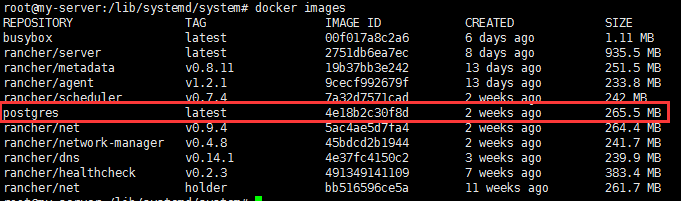




**启动postgre之前所有的镜像**

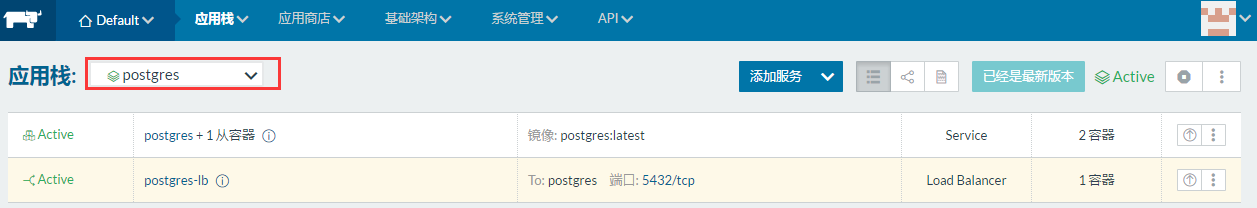
### 配置数据库端口等信息



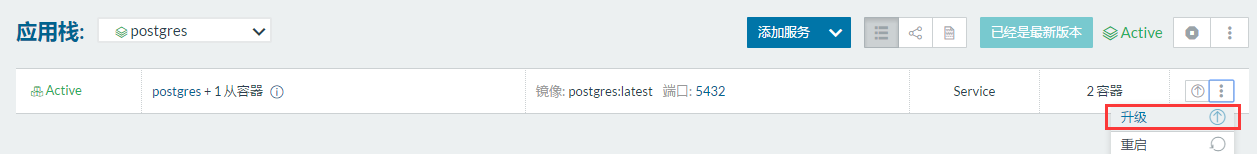


**启动后所有的镜像（原docker没有，会自动下载）**

### 启动（官方支持负载均衡）

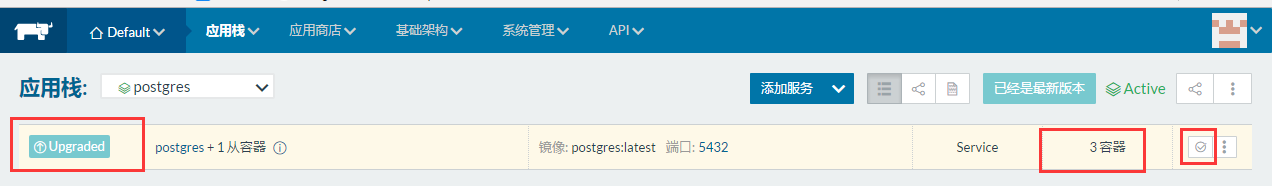


### 删除postgres-lb, 升级镜像postgres:latest，修改端口映射

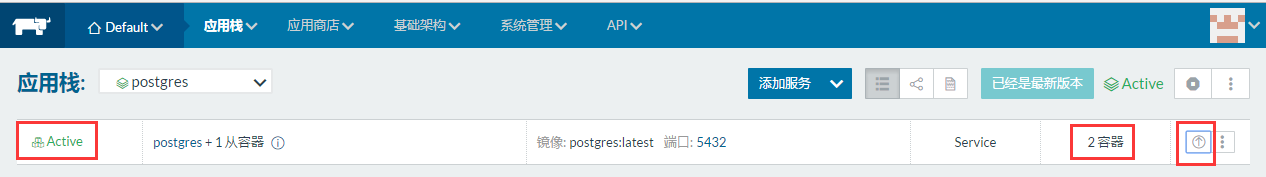


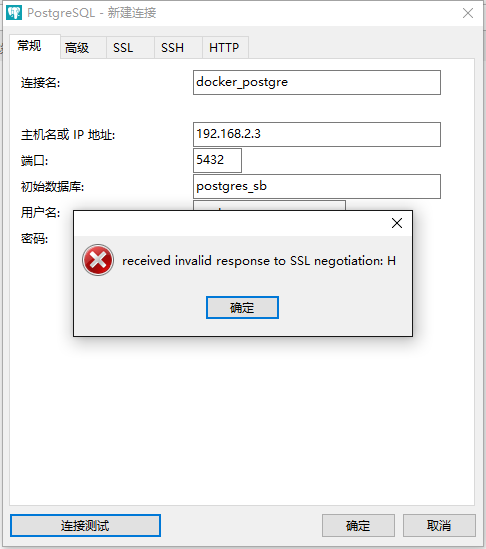


点击最下面的升级按钮（停止当前容器，并新创建一个容器）

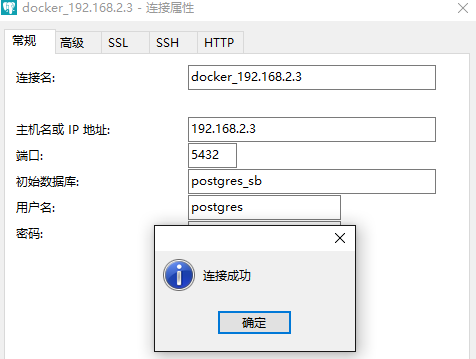


点击最右侧的升级（删除之前的容器）





**使用Navicat连接docker版的postgre失败，请检查负责均衡、端口映射**



# Rancher上使用自己的镜像

官方参照地址：<https://github.com/rancher/rancher-catalog>

# 搭建私有仓库

## 搭建最简单的仓库

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

docker run -d -p 5000:5000 -v /home/yao/registry:/tmp/registry registry

* 测试访问

<http://192.168.2.3:5000/v2/>

* 查看所有镜像

<http://192.168.2.3:5000/v2/_catalog>

* 注意https错误，请将私有仓库加入到docker.service中

## 配置带用户权限的registry

* 创建用户信息

mkdir /home/yao/registry/auth/

docker run --entrypoint htpasswd registry:latest -Bbn yao ubuntu >> /home/yao/registry /auth/htpasswd

* 再次启动

docker run -d -p 5000:5000 --restart=always -v /opt/registry-var/auth/:/auth/ -e "REGISTRY\_AUTH=htpasswd" -e "REGISTRY\_AUTH\_HTPASSWD\_REALM=Registry Realm" -e REGISTRY\_AUTH\_HTPASSWD\_PATH=/auth/htpasswd -v /opt/registry-var/:/var/lib/registry/ registry:latest

## 仓库UI

<https://hub.docker.com/r/hyper/docker-registry-web/>

<https://hub.docker.com/r/konradkleine/docker-registry-frontend/>

sudo docker run \

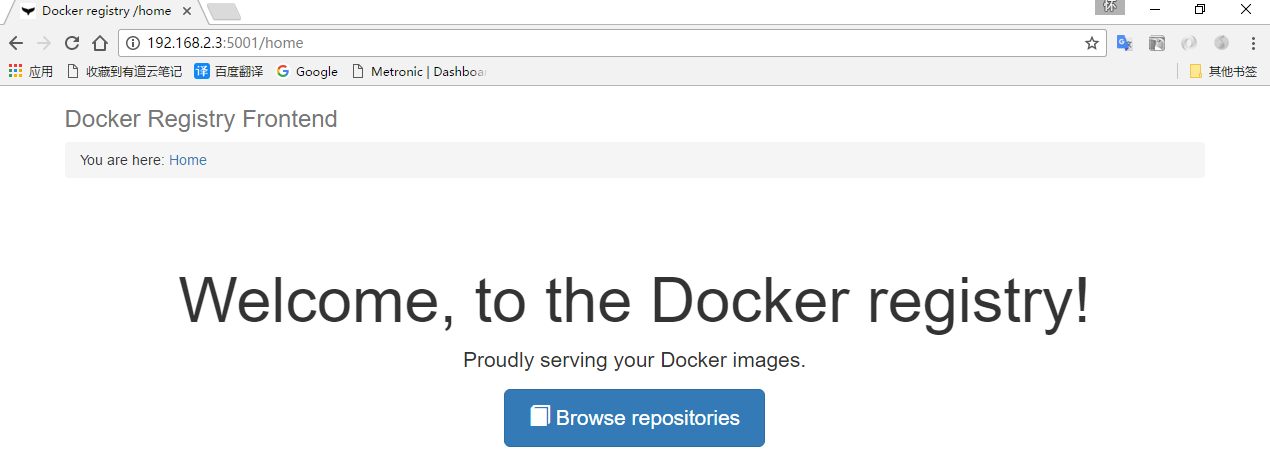
-d \

-e ENV\_DOCKER\_REGISTRY\_HOST=192.168.2.3 \

-e ENV\_DOCKER\_REGISTRY\_PORT=5000 \

-p 5001:80 \

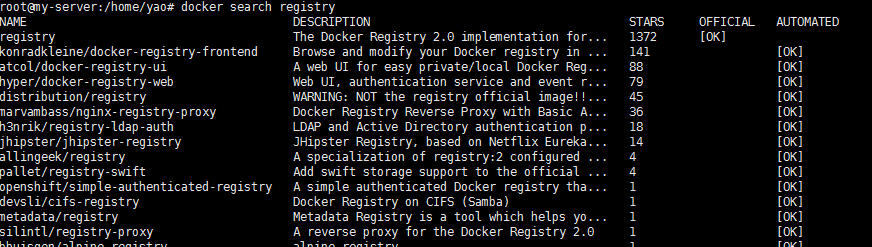
konradkleine/docker-registry-frontend:v2



# 常用命令

## docker search registry

搜索可用docker镜像



## docker pull

从远端拉取一个image

### 从默认的[Docker Hub](https://hub.docker.com/)拉取镜像

### 从指定的服务器拉取镜像

docker  pull docker.orientge.cn:5000/oge/vzdb

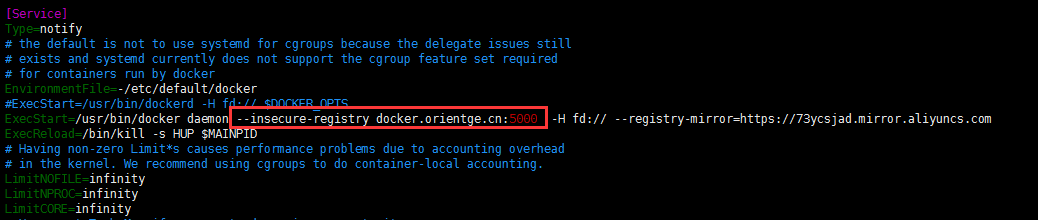


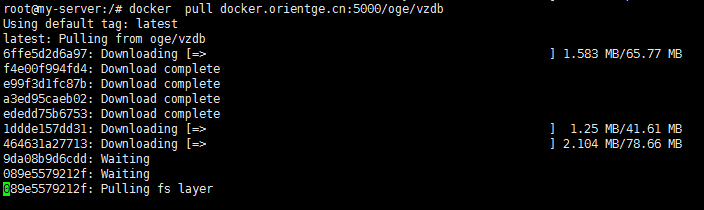
默认使用https客户端拉取，但私有仓库是http的，增加http仓库，修改docker.service， 命令如下：--insecure-registry docker.orientge.cn:5000

vi /lib/system/system/docker.service

ExecStart=/usr/bin/docker daemon --insecure-registry docker.orientge.cn:5000

-H fd:// --registry-mirror=https://73ycsjad.mirror.aliyuncs.com





docker  pull docker.orientge.cn:5000/oge/kdm-rdp

## docker build

### 编写Dockerfile

FROM fiadliel/java8-jre

VOLUME /logs

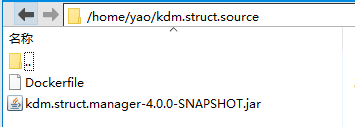
EXPOSE 8079

ADD kdm.struct.manager-4.0.0-SNAPSHOT.jar kdm.struct.manager-4.0.0-SNAPSHOT.jar

ENV JAVA\_OPTS=""

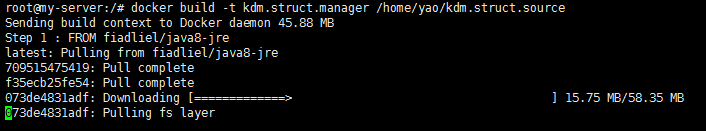
CMD ["java", "-jar", "kdm.struct.manager-4.0.0-SNAPSHOT.jar"]

### 文件目录



### 打包命令

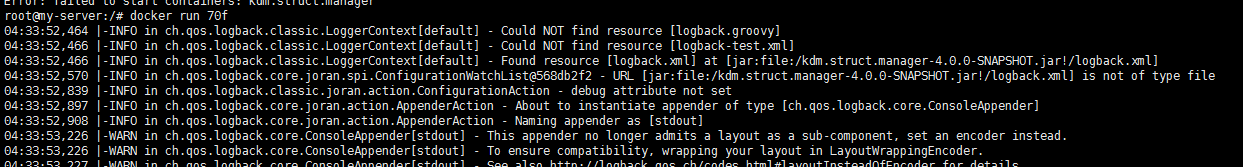
docker build -t kdm.struct.manager /home/yao/kdm.struct.source





### 测试运行

可以使用image id简写，使用这个镜像，启动新的容器：docker run 70f



## docker load

## docker save

docker save > "debian.tar"

将指定镜像保存成 tar 归档文件， docker load 的逆操作。保存后再加载（saved-loaded）的镜像不会丢失提交历史和层，可以回滚。

## docker push

将镜像推送至远程仓库，默认为 Docker Hub

## docker rmi

docker rmi  镜像名/ID

删除所有未运行Docker容器

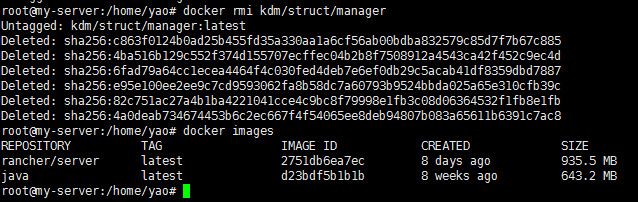
docker rm $(docker ps -a -q)

删除所有未打 tag 的镜像

docker rmi $(docker images | grep "^<none>" | awk "{print $3}")

删除所有镜像

docker rmi $(docker images -q)





注意：在删除镜像之前要先用 docker rm 删掉依赖于这个镜像的所有容器（哪怕是已经停止的容器），否则无法删除该镜像

## docker rm

删除一个或多个容器



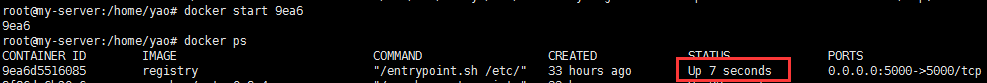
## docker run

docker run [options "o">] <image> [ "nb">command] "o">[arg...]

启动一个容器，在其中运行指定命令

## docker start|stop|restart

启动、停止和重启一个或多个指定**容器**。



## docker ps

列出所有运行中容器

-a 显示所有容器，包括当前没有运行的容器

--before="nginx" 列出在某一容器之前创建的容器，接受容器名称和ID作为参数；

--since="nginx" 列出在某一容器之后创建的容器，接受容器名称和ID作为参数；

-f [exited=<int>] 列出满足

exited=<int> 条件的容器；

-l 仅列出最新创建的一个容器；

--no-trunc 显示完整的容器ID；

-n=4 列出最近创建的4个容器；

-q 仅列出容器ID；

-s 显示容器大小。

## docker pause/ unpause

暂停/恢复某一容器的所有进程

## docker kill

杀死一个或多个指定容器进程。

## docker inspect

检查镜像或者容器的参数，默认返回 JSON 格式。

## docker logs

获取容器运行时的输出日志

## docker export

将指定的容器保存成 tar 归档文件， docker import 的逆操作。导出后导入（exported-imported)）的容器会丢失所有的提交历史，无法回滚。

## docker import

从归档文件（支持远程文件）创建一个镜像， export 的逆操作，可为导入镜像打上标签。导出后导入（exported-imported)）的容器会丢失所有的提交历史，无法回滚。

## docker top

查看一个正在运行容器进程，支持 ps 命令参数

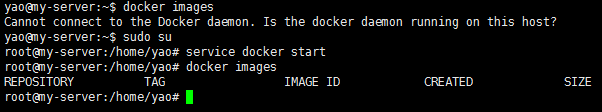
## docker images

docker images [options "o">] [name]

列出本地所有镜像。其中 [name] 对镜像名称进行关键词查询。

# 常见问题处理

## Cannot connect to the Docker daemon. Is the docker daemon running on this host?



权限问题，当前用户没权限，root用户可以运行

解决方法：

sudo su //切换到root

service docker start //启动docker service

docker images //显示所有images