1. Rancher的结构及简介

1.共分为五部分：

①rancher server&rancher agent

②Account

③Environment

④Stack

⑤Rancher Compose CLI

2.这五部分的关系为：

Rancher

Environment

Hosts

Stacks

Containers

Services

Account

1. 结构详细介绍
   1. rancher server&rancher agent

Rancher的各种容器管理理念均架构在由Rancher server和rancher agent构建的Infrastructure之上。Rancher server是Rancher的核心，其地位就类似于k8s、Docker swarm或mesos中的master，提供核心容器管理服务以及API服务。

Rancher的一个设计理念是所有组件都Containerized（容器化），这里的Rancher Server也不例外，极大的方便了我们的Install。通过rancher server 管理rancher agent

1.2搭建Rancher Server

下载rancher server镜像

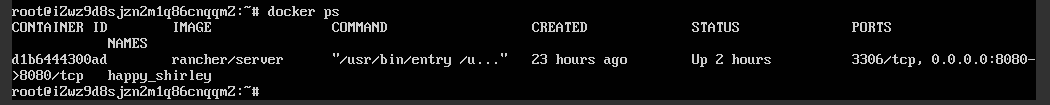
Cmd:docker pull rancher/server

运行rancher server

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

查看正在运行的容器：

Cmd:docker ps



/usr/bin/s6-svscan是容器的第一个启动进程，/service这个 路径作为其命令行参数，估计这是一个类似于supervisord的进程控制程 序，由其 负责启动和管理Rancher server的两个重要服务：MySQL和cattle。

单节点rancher server的数据都保存在其内部的MySQL中，而多节点rancher server则采用一个外部的MySQL存储数据。

2. Account

第一次启动Rancher后，Rancher的UI是没有访问控制的，所有人都可以访问这个地址并控制一切。因此在使用rancher之前需要创建一个account来控制访问的权限。

* + 1. 查看rancher server是否工作

切换到API菜单，可以看到当前默认Environment的API访问endpoint是：

<http://39.108.97.103:8080/v1>

我们可以用curl来访问一下这个url：



返回超过一屏的信息，这同时也说明Rancher Server在正常工作。

在使用rancher前首先得添加一个账户，否则任何人都能访问。

* + 1. 添加账号

添加权限控制，“ACCESS CONTROL”中。Rancher支持四种权限控制方案，分别是：Active Directory、GitHub、Local Auth和OpenLDAP（新版本又增加了Azure AD、SHIBBOLETH）。我们使用最简单的Local Auth，即设置一个用户名和密码，然后点击“Enable Local Auth”按钮即可。然后我们再回到”ACCOUNTS”页面.

这时如果你再用API访问当前默认环境的EndPoint的话，结果就会变成下面这样



提示错误：Unauthorized

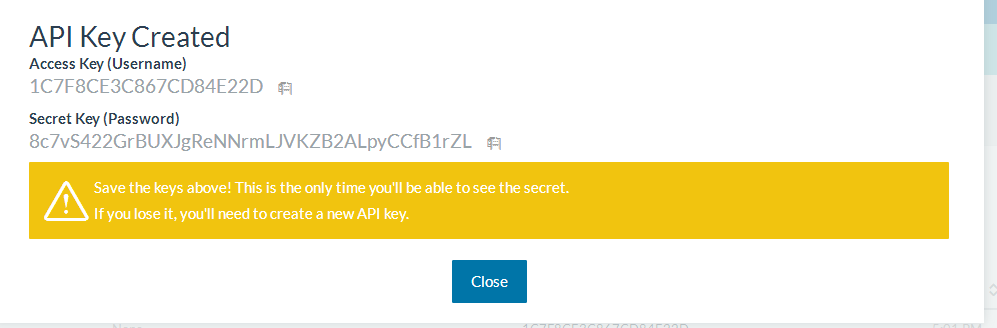
这时如果还想用API访问，就需要为该环境添加一个API Key了。在”API”页面下，点击 “Add Environment API Key”按钮，在弹出的窗口中输入key的name：tonybai-default-env-key，点击”Create”创建：

随便输入个名字，例如这里输入的是：default-key

然后会随机生成用户名和密码，密码需立即保存，之后无法查看，只能重新再生成一对。

1C7F8CE3C867CD84E22D

8c7vS422GrBUXJgReNNrmLJVKZB2ALpyCCfB1rZL

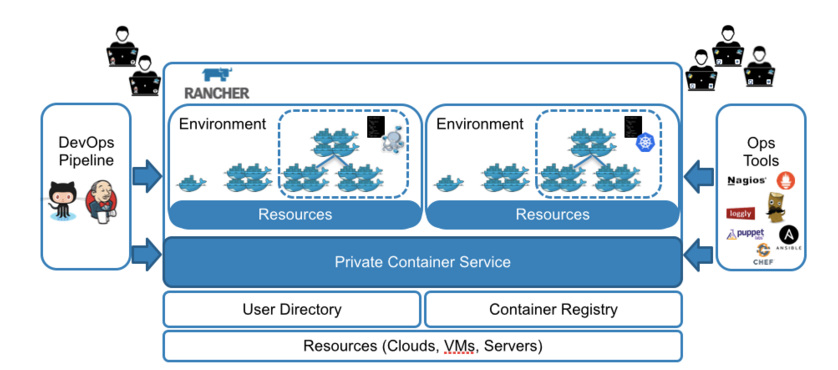


Cmd:curl –u 1C7F8CE3C867CD84E22D:8c7vS422GrBUXJgReNNrmLJVKZB2ALpyCCfB1rZL <http://39.108.97.103:8080/v1>



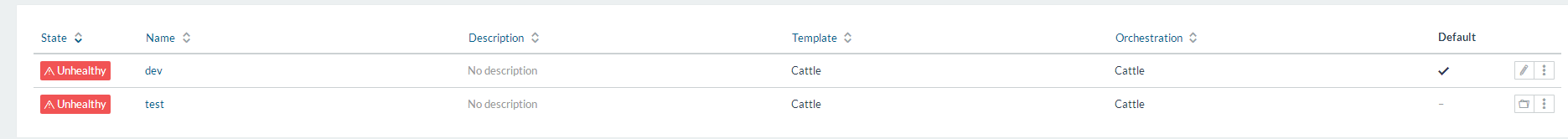
3. Environment

一个Environment就是一个resource group，每个Environment都有自己的服务和基础设施资源，并且通过Access Control来赋予每个Account访问该Environments的权限。比如建立dev、test、production environment，保证Environments间的隔离。



Rancher Server建立后，会建立一个”Default” Environment，我们可以Edit一下这个Environment的信息，可以修改它的Name、Container Orchestration引擎（cattle、k8s和swarm，默认cattle）以及Access Control.这里我们将Default的名字改为”dev”。

我们再添加一个Environment “test”，引擎用cattle.



我们看到dev environment后面有一个”对号”，说明dev environment是当前active environment，所有操作均针对该environment，你当然可以通过点击每个environment列表后面的切换图标来切换active environment。

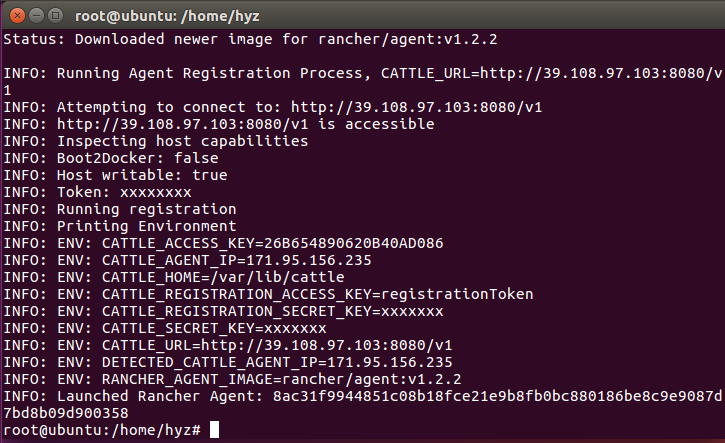
到目前为止，虽然Rancher Server建立ok了，environment这个逻辑实体也建立了，但dev environment仍处于“无米下炊”的状态。因为除了Rancher自身外，该Environment下没有任何Resources（主机、存储 等）可供使用（比如创建Containers）。

现在为dev添加主机资源：

在”INFRASTRUCTURE”-> HOSTS中点击”Add Host”按钮添加主机资源。Rancher支持多种主机资源，包括Custom（本地自定义）、Amazon EC2、 Azure 以及 DigitalOcean 等。

我们以本地Host资源(选择Custom)为例，在添加Host页面中，我们输入第一个Host的IP，Rancher UI会生成下面这段命令行：(查看内网ip：cmd:ifconfig 外网ip:cmd:curl ifconfig.me)

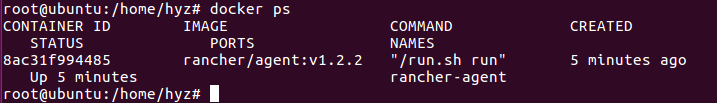
sudo docker run -e CATTLE\_AGENT\_IP="171.95.156.235" --rm --privileged -v /var/run/docker.sock:/var/run/docker.sock -v /var/lib/rancher:/var/lib/rancher rancher/agent:v1.2.2 <http://39.108.97.103:8080/v1/scripts/6666824B42FA442A2A5E:1483142400000:SoxXthqukDiZuz7XjcgAS3WOiw>



等一会进入INFRASTRUCTURE-> HOSTS页面，就能看到该主机已经添加上去了。

注意：

上面的命令启动了两个Container，image虽然都是rancher/agent:v0.11.0，但执行的命令行参数略有 不同（其中一个Container为临时Container，一段时间后会自动退出）。



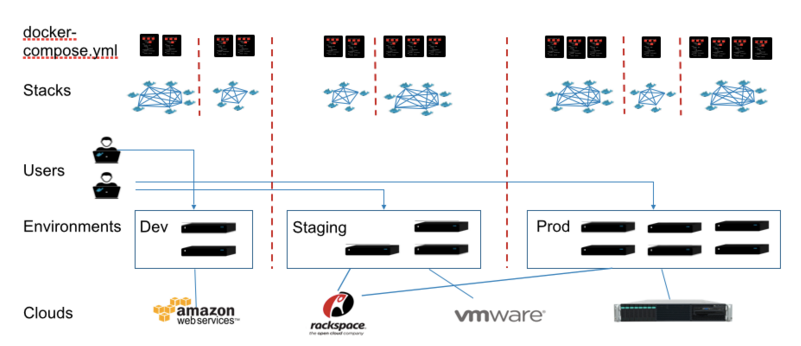
我们点击Rancher UI右上角的下拉箭头，将当前Environment从dev切换到test，我们发现test Environment下的Hosts又为空了

4. Stack

由一系列services组成的一个application

A Stack <=> An Application <=> A group of services(由类docker-compose的工具rancher-compose管理)

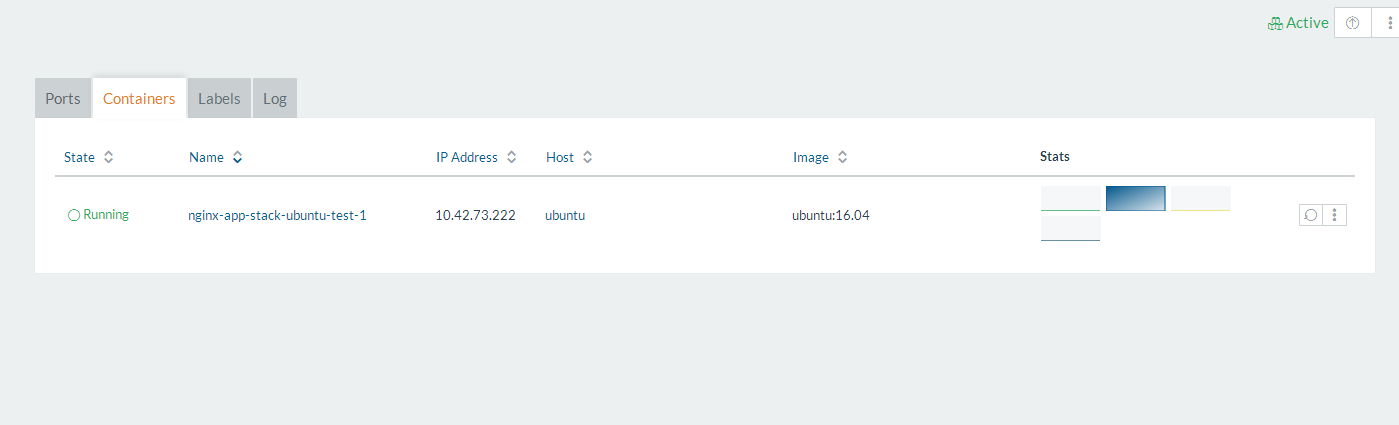
下面这幅图直观描述了user account, environment与stacks之间的关系：



我们在dev environment下添加一个Service。下面支持两种添加Service的方式，一种是手工添加，一种是从Catalog添加。Catalog类似于一个Rancher App Market，里面有Rancher预定义好的service template。我们这次采用手工添加的方式，便于控制。

创建stack名为nginx-app-stack。

我们基于nginx:1.8-alpine创建单一实例的service: nginx-alpine-service，端口映射：10086->80。其他采用默认配置。启动后，我们看到第一个nginx-alpine-service的Container运行在 171.95.156.235（即已有的一个host）上。(这里由于是参考网上的教程，使用nginx:1.8-alpine镜像会产生timeout的erro，所以我换成了ubuntu:16.04，可以正常运行)



5. Rancher Compose CLI

Rancher除了提供UI工具外，还提供了一个名为rancher-compose的CLI工具，用于在一个stack的范围内管理各个services。rancher-compose的灵感来源于docker-compose，兼容docker-compose的配置文件格式，并有自己的扩展。此外与docker-compose不同的是rancher-compose支持跨多主机管理。

在Rancher UI的右下角有一个Rancher-compose的下载链接，支持Linux，Windows和Mac。