**docker云平台**

**摘要：**

Docker，一个新的容器技术，它能够在相同的旧服务器上运行的更多的应用程序，这也使得它很容易打包和发布程序。

它可以得到相同的硬件上比其他技术运行更多的应用（小的开销内存/ CPU/硬盘，这意味着更低成本）;

它使开发人员能够快速创建简单的，现成的运行容器化应用;

它使管理和部署应用程序更加容易。

**优点：**

https://www.docker.com/whatisdocker/

1.一些优势和VM一样，但不是所有都一样。

比VM小，比VM快，Docker容器的尺寸减小相比整个虚拟机大大简化了分布到云和从云分发。

2.对于在笔记本电脑，数据中心的虚拟机，以及任何的云上，运行相同的没有变化的应用程序，IT的发布速度更快。

Docker是一个开放的平台，构建，发布和运行分布式应用程序。

Docker使应用程序能够快速从组件组装和避免开发，QA和生产环境之间的摩擦。

3.您可以在部署在公司局域网或云或虚拟机上使用它。

4.开发人员并不关心具体哪个Linux操作系统

使用Docker，开发人员可以根据所有依赖关系构建相应的软件，针对他们所选择的操作系统。

然后，在部署时一切是完全一样的，因为一切都在DockerImage的容器在其上运行。

开发人员负责并且能够确保所有的相关性得到满足。

5.Google，微软，亚马逊，IBM等都支持Docker。

**缺点局限性：**

1.Docker支持Unix/Linux操作系统，不支持Windows或Mac（即使可以在其上安装，不过也是基于Linux虚拟机的）

2.Docker用于应用程序时是最有用的，但并不包含数据。日志，跟踪和数据库等通常应放在Docker容器外。

**前提条件：**

目前，CentOS 仅发行版本中的内核支持 Docker。

Docker 运行在 CentOS 7 上，要求系统为64位、系统内核版本为 3.10 以上。

Docker 运行在 CentOS-6.5 或更高的版本的 CentOS 上，要求系统为64位、系统内核版本为 2.6.32-431 或者更高版本

1，安装docker

2，docker使用

3，docker容器中安装tomcat及常见服务

最终实现 tomcat、nginx 、redis等服务在docker容器中的正常运行。

**安装Docker：**

Centos6

安装：

yum install epel-release -y

yum install docker-io -y

启动：

/etc/init.d/docker start

Centos7

安装：

yum install docker -y

启动：

systemctl start docker

docker包已经在Centos7的ca源中，无需再安装其他的扩展源。

**docker常见的操作**

docker images   //查看当前的docker镜像

docker ps          //查看当前运行的docker

docker ps -a      //查询所有的docker（包括停止的docker）

docker rmi  <REPOSITORY/IMAGE ID > latest

REPOSITORY：docker 的名称

IMAGE ID  : docker的编号

docker cp

传文件进docker   docker cp test.war IMAGE ID/webapps

docker传入宿主机   docker cp IMAGE ID/webapps/test.war  /data/

docker  top  b0c5183a4065  //查看docker中运行的进程

docker port b0c5183a4065   //输出容器端口与宿主机端口的映射情况

docker search centos:7.2.1511   //在仓库中搜索此镜像

docker pull centos:7.2.1511       //下载此镜像

通过docker的两个参数 -i -t，让docker运行的容器实现"对话"的能力

docker run -it centos:7.2.1511 /bin/bash    //登录此docker

参数解析：

-t:在新容器内指定一个伪终端或终端。

-i:允许你对容器内的标准输入 (STDIN) 进行交互。

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

- -d:让容器在后台运行。

- -P:将容器内部使用的网络端口映射到我们使用的主机上

--restart=always：开机自启动

**docker容器中部署 tomcat应用**

docker pull tomcat    //dao pull tomcat

docker images

docker tag library/tomcat  tomcat-1  克隆一个tomcat并重名为tomcat-1

docker run -p 5000:8080 --name container1 tomcat-1

-p 5000:8080的意思是把容器tomcat的8080端口隐射到宿主机的端口上，这样外网访问5000就能访问到docker容器中container1的8080 tomcat上面了.

此处容器container1 和 镜像tomcat-1，我的理解是镜像就是一个模板，container1就是根据这个模板创造的一个真正的盆子，这个盆子里面就跑着我们的tomcat. 所以我们可以用同一个镜像创建许多container。

**在tomcat上面部署我们的应用**

1.进入容器内部

docker exec -it container2 /bin/bash

2.查看tomcat webapp路径

/tomcat/webapps

3.传war

把war丢到宿主机 在丢到container里面丢到tomcat/webapps

docker cp test.war b0c5183a4065:/usr/local/tomcat/webapps

4.重启容器

docker自动重启tomcat项目

5.查看log日志

docker logs -f b0c5183a4065

至此，一个完整的docker部署tomcat及上线一个java web应用流程就走通了.

tomcat中war包自动解压并自己重启。

可以把dokcer创建的container当成新的linux主机

<http://blog.csdn.net/jinzhencs/article/details/51373852>

Docker部署Redis应用

1，下载redis

docker search redis

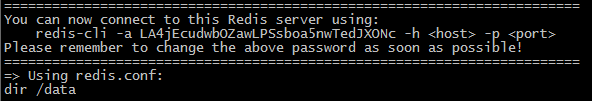
docker pull daocloud.io/daocloud/dao-redis & dao pull daocloud.io/daocloud/dao-redis

2，启动

docker run -p 6379:6379 -d 173a30377d85

3.查看

docker logs -f 173a30377d85    //可以查看到密码



4.进去redis容器中

docker exec -ti  173a30377d85 /bin/bash

可以用redis图形化工具连接redis测试是否可以正常使用。

Zookeeper

docker search zookeeper

docker pull docker.io/jplock/zookeeper

docker tag docker.io/jplock/zookeeper zookeeper

docker run -d -p 2181:2181  zookeeper

docker logs -f 40019c9162044131749cdd913f8123514044fce01330c02fc67560c9f1192678

docker exec -ti  40019c916204  /bin/bash

docker 加速

由于国内众所周知的原因，官方的DockerHub访问太慢，所以这里使用Daocloud的国内镜像加速

加速账号注册：https://dashboard.daocloud.io/nodes/new

注意：Daocloud加速后使用dao pull下载更快

**docker管理平台 ：**

推荐 Rancher，shipyard ，mesos，Kubernetes

**rancher 管理平台的安装**

dao pull rancher/server

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

docker ps

docker logs -f id

访问地址: http://ip:8080

**shipyard部署**

Shipyard也是完全基于Docker API，支持container管理、engine管理（一个engine就是监听tcp端口的docker daemon）。

优点：

- 支持多主机；

- 支持container及engine资源限制及图形展示；

- 支持container实例横向扩展；

- 支持批量创建；

- 支持创建时自动调度。

缺点：

- 不支持image管理；

- 不支持container批量操作。

由于官方支持一键安装脚本，这里我们使用一键安装。如下

$ docker pull alpine

$ docker pull library/shipyard

   docker pull  docker.io/dockerclub/shipyard            // docker.io/dockerclub/shipyard         shipyard中文版，一个开源的容器

$ docker pull library/rethinkdb

$ docker pull microbox/etcd

$ docker pull shipyard/docker-proxy

$ docker pull swarm

$ docker pull shipyard/shipyard

curl -s https://shipyard-project.com/deploy | bash -s

中文支持：

shipyard 中文静态插件

用法:

1.删除英文静态网页

  docker exec -it shipyard-controller rm -rf static/app

  docker exec -it shipyard-controller rm -f static/index.html

2.下载中文静态页面

 git clone https://github.com/StarWars-Team/shipyard\_zh.git

3.将汉化文件拷贝到Docker

  docker cp shipyard\_zh/static/app shipyard-controller:/static/

  docker cp shipyard\_zh/static/index.html shipyard-controller:/static/

中文汉化来自github  <https://github.com/StarWars-Team/shipyard_zh>

Shipyard 默认访问端口是8080，默认用户名和密码是admin 和 shipyard

Docker平台3- Mesos+Marathon

增加仓库

$ rpm -Uvh <http://repos.mesosphere.com/el/7/noarch/RPMS/mesosphere-el-repo-7-3.noarch.rpm>

$ yum -y update

安装Master

这里需要在Master上安裝Mesos、Marathon、ZooKeeper。

yum install -y mesos marathon mesosphere-zookeeper

详见：http://dockone.io/article/515

Docker平台4 - Kubernetes (from Google)

Rancher + k8s  基于rancher商店，在商店添加Kubernetes安装即可

扩展：docker仓库安装使用

Harbor 安装

1，Harbor介绍

  harbor 是由 vmware 中国团队开发并开源的企业级 docker registry 服务，是对官方开源的 Docker Registry 的扩展，增加了一些企业需要的功能，如安全、复制和管理。主要用于搭建私有 registry，提供了企业需要的安全和控制功能。同时，它也帮助减少带宽使用量，这对提供生产能力和性能有帮助。它的强大之处在于，同时提供了 registry 的存储功能、认证功能、web ui 浏览和管理功能，且搭建相对简单。

软件要求：

 目标主机需要安装 python 2.7或以上, docker 1.10或以上，docker-compose 1.6.0 或以上。

安装：

1.python-pip的安装

yum install -y python-pip

2. 下载Harbor安装包

#在线安装

wget https://github.com/vmware/harbor/releases/download/0.5.0/harbor-online-installer-0.5.0.tgz

#离线安装包

wget https://github.com/vmware/harbor/releases/download/0.5.0/harbor-offline-installer-0.5.0.tgz

tar -zxvf harbor-offline-installer-0.5.0.tgz

cd harbor

3.参数配置

vi harbor.cfg

#配置私有仓库的访问地址rul,ip或者域名,不要写localhost/127.0.0.1的地址，尽量能让外网访问到

hostname=172.16.51.65

4.安装

./install.sh

安装成功，通过之前在harbor.cfg配置的hostname即可以访问到前端了，默认登陆用户名密码是admin/Harbor12345

6. 提交项目到仓库

在前端项目管理中添加一个项目名叫 myproject的项目，然后通过docker push 提交到仓库

由于默认安装我们使用的是HTTP协议，需要在docker客户端加入信任私有仓库地址，编辑vi /etc/sysconfig/docker

修改配置INSECURE\_REGISTRY 加入地址 172.16.10.10,格式为--insecure-registry 仓库地址 多个以空格隔开

INSECURE\_REGISTRY='--insecure-registry 172.16.10.10:5000 --insecure-registry 172.16.10.10'

测试

#镜像仓库的用户

$ docker login 172.16.10.10

#将镜像打一个tag,格式必须为 docker push [私库地址]/[项目名]/镜像名称:tag

$ docker tag 镜像id reg.yourdomain.com/myproject/myrepo:mytag

$ docker push reg.yourdomain.com/myproject/myrepo:mytag