**[Docker学习笔记之一，搭建一个JAVA Tomcat运行环境](http://blog.csdn.net/junjun16818/article/details/34845613)**

**前言**

Docker旨在提供一种应用程序的自动化部署解决方案，在 Linux 系统上迅速创建一个容器（轻量级虚拟机）并部署和运行应用程序，并通过配置文件可以轻松实现应用程序的自动化安装、部署和升级，非常方便。因为使用了容器，所以可以很方便的把生产环境和开发环境分开，互不影响，这是 docker 最普遍的一个玩法。更多的玩法还有大规模 web 应用、数据库部署、持续部署、集群、测试环境、面向服务的云计算、虚拟桌面 VDI 等等。

主观的印象：Docker 使用 Go 语言编写，用 cgroup 实现资源隔离，容器技术采用 LXC. 提供了能够独立运行Unix进程的轻量级虚拟化解决方案。它提供了一种在安全、可重复的环境中自动部署软件的方式。LXC命令有些复杂，若感兴趣，这里有一篇我以前写的基于LXC，（从无到有，搭建一个简单版的JAVA PAAS云平台），可以提前复习一下。

有关实现原理、相关理论、运用场景等，会在本系列后面书写，这里先来一个浅尝辄止，完全手动，基于Docker搭建一个Tomcat运行环境。先出来一个像模像样Demo，可以见到效果，可能会让我们走的更远一些。

**环境**

本文所有环境，VMware WorkStation上运行ubuntu-13.10-server-amd64,注意是64位系统，理论上其它虚拟机也是完全可行的。

**安装Docker**

Docker 0.7版本需要linux内核 3.8支持，同时需要AUFS文件系统。

# 检查一下AUFS是否已安装

sudo apt-get update

sudo apt-get install linux-image-extra-`uname -r`

# 添加Docker repository key

sudo sh -c "wget -qO- https://get.docker.io/gpg | apt-key add -"

# 添加Docker repository，并安装Docker

sudo sh -c "echo deb http://get.docker.io/ubuntu docker main > /etc/apt/sources.list.d/docker.list"

sudo apt-get update

sudo apt-get install lxc-docker

# 检查Docker是否已安装成功

sudo docker version

# 终端输出 Client version: 0.7.1

Go version (client): go1.2

Git commit (client): 88df052

Server version: 0.7.1

Git commit (server): 88df052

Go version (server): go1.2

Last stable version: 0.7.1

**去除掉sudo**

在Ubuntu下，在执行Docker时，每次都要输入sudo，同时输入密码，很累人的，这里微调一下，把当前用户执行权限添加到相应的docker用户组里面。

# 添加一个新的docker用户组

sudo groupadd docker

# 添加当前用户到docker用户组里，注意这里的yongboy为ubuntu server登录用户名

sudo gpasswd -a yongboy docker

# 重启Docker后台监护进程

sudo service docker restart

# 重启之后，尝试一下，是否生效

docker version

#若还未生效，则系统重启，则生效

sudo reboot

**安装一个Docker运行实例-ubuntu虚拟机**

Docker安装完毕，后台进程也自动启动了，可以安装虚拟机实例（这里直接拿官方演示使用的learn/tutorial镜像为例）：

docker pull learn/tutorial

安装完成之后，看看效果

docker run learn/tutorial /bin/echo hello world

交互式进入新安装的虚拟机中

docker run -i -t learn/tutorial /bin/bash

会看到：

root@51774a81beb3:/#

说明已经进入交互式环境。

安装SSH终端服务器，便于我们外部使用SSH客户端登陆访问

apt-get update

apt-get install openssh-server

which sshd

/usr/sbin/sshd

mkdir /var/run/sshd

passwd #输入用户密码，我这里设置为123456，便于SSH客户端登陆使用

exit #退出

获取到刚才操作的实例容器ID

#docker ps -l

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

51774a81beb3 learn/tutorial:latest /bin/bash 3 minutes ago Exit 0 thirsty\_pasteur

可以看到当前操作的容器ID为：51774a81beb3。注意了，一旦进行所有操作，都需要提交保存，便于SSH登陆使用：

docker commit 51774a81beb3 learn/tutorial

以后台进程方式长期运行此镜像实例：

docker run -d -p 22 -p 80:8080 learn/tutorial /usr/sbin/sshd -D

ubuntu容器内运行着的SSH Server占用**22**端口，-p 22进行指定。**-p 80:8080** 指的是，我们ubuntu将会以8080端口运行tomcat，但对外（容器外）映射的端口为80。

这时，查看一下，是否成功运行。

#docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

871769a4f5ea learn/tutorial:latest /usr/sbin/sshd -D About a minute ago Up About a minute 0.0.0.0:49154->22/tcp, 0.0.0.0:80->8080/tcp focused\_poincare

注意这里的分配随机的SSH连接端口号为49154：

ssh root@127.0.0.1 -p 49154

输入可以口令，是不是可以进入了？你一旦控制了SSH，剩下的事情就很简单了，安装JDK，安装tomcat等，随你所愿了。以下为安装脚本：

# 在ubuntu 12.04上安装oracle jdk 7

apt-get install python-software-properties

add-apt-repository ppa:webupd8team/java

apt-get update

apt-get install -y wget

apt-get install oracle-java7-installer

java -version

# 下载tomcat 7.0.47

wget http://mirror.bit.edu.cn/apache/tomcat/tomcat-7/v7.0.47/bin/apache-tomcat-7.0.47.tar.gz

# 解压，运行

tar xvf apache-tomcat-7.0.47.tar.gz

cd apache-tomcat-7.0.47

bin/startup.sh

默认情况下，tomcat会占用**8080**端口，刚才在启动镜像实例的时候，指定了 -p 80:8080，ubuntu镜像实例/容器，开放8080端口，映射到宿主机端口就是80。知道宿主机IP地址，那就可以自由访问了。在宿主机上，通过curl测试一下即可：

curl http://192.168.190.131

当然，你也可以使用浏览器访问啦。

真实情况，可能不会让tomcat直接对外开放80端口，一般都会位于nginx/apache或者防火墙的后面，上面仅为演示。

**小结**

在Docker帮助下搭建一个Tomcat运行时环境，总体很简单，让我们看到了PAAS的身影。不错，使用Docker作为PAAS底层服务，本身就不复杂。 下面有时间，会谈一谈如何使用脚本文件构建一个镜像实例，同时会谈一谈Docker的实现原理和机制等。