Docker介绍以及Registry的安装

【编者的话】本文介绍了Docker与Registry，作者说Docker是一个application hosting框架，亮点是简化应用的部署以及应用部署的版本控制。同时，作者介绍了Docker Registry的安装以及一个可以通过网页浏览Registry的镜像项目docker-registry-web。

Docker是一个应用托管框架（application hosting framework），它可以通过类似虚拟机一样的容器来部署、管理应用，容器又可以通过API创建和控制它们。

Docker允许你把依赖/服务器和应用打包成一个叠加在其他镜像（如Ubuntu，或专为需求准备的内容）之上的瘦小镜像。有别于虚拟机的是，尽管使用了LXC和cgroups（前面文章中提到的Linux概念）将它们与其它系统中的程序严格地隔离开来，它们却共享着相同的资源，并且几乎没有额外开销。当你启动一个虚拟机，你最终得到的是一个用于安装或运行应用的提示符或UI。当你启动一个应用容器，你只要运行一个用于启动应用及其依赖的脚本，仅此而已。你能在一个系统上运行数个虚拟机，却可以运行上千个应用容器。如果你想要流线分布，可以考虑在系统层面上使用CoreOS来托管你的镜像。

Docker的另一个功能是版本控制。你可以把容器里做的任何改变提交成一个新的镜像。当然，你也可以使用同一个镜像（镜像本身是不可变的）启动任意多个容器。

在分发镜像给其他团队或公司的过程中，可能需要在你的当前系统之外找个地方发布或定位你的镜像。这可通过Registry实现。尽管Docker提供了公共的Docker Hub Registry，你或许想要一个用于自己公司或团队的私有的Registry。

因为Docker的组件/附件自身经常是通过Docker镜像发布的，本示例也侧面展示了启动一个以Docker为基础的应用是多么简单（如果你之前并不熟悉这块）。除了服务占用的端口之外，你无须知道客户机应用的任何东西。实际上，你可以启动其他Docker镜像需要的镜像（之后就被称为容器），让Docker映射随机的本地端口给它，然后将提供服务的容器的端口自动转发到依赖这些服务的容器上（通过“链接”功能）。

使用类似Registry项目首页示例的命令来启动你的Registry：

```
$ docker run
    -e SETTINGS_FLAVOR=s3
    -e AWS_BUCKET=mybucket
    -e STORAGE_PATH=/registry
    -e AWS_KEY=myawskey
    -e AWS_SECRET=myawssecret
    -e SEARCH_BACKEND=sqlalchemy
    -p 5000:5000
    registry
```

这主要是为应用设置六个环境变量，让它保存到S3上，并将宿主（本地）系统的5000端口转发到客户机（Registry）的5000端口上。“registry”是运行的镜像名称（如果它是由某个用户拥有的，那看起来像是“/”）。如果本地尚不存在这个镜像，它将被定位并拉取（pull）下来。如果没有使用registry前缀做限定，将会假定它位于Docker Hub上。

这个示例中，我们从Hub上将Ubuntu镜像拉取下来，然后推送（push）到我们的Registry里。值得注意的是，我们通过添加Registry的主机名/端口前缀来限定“推送”和“拉取”请求到我们的registry中。

```
$ sudo docker pull ubuntu:14.04
$ sudo docker tag 826544226fdc yourregistry.net:5000/ubuntu
$ sudo docker push yourregistry.net:5000/ubuntu
$ sudo docker pull yourregistry.net:5000/ubuntu
```

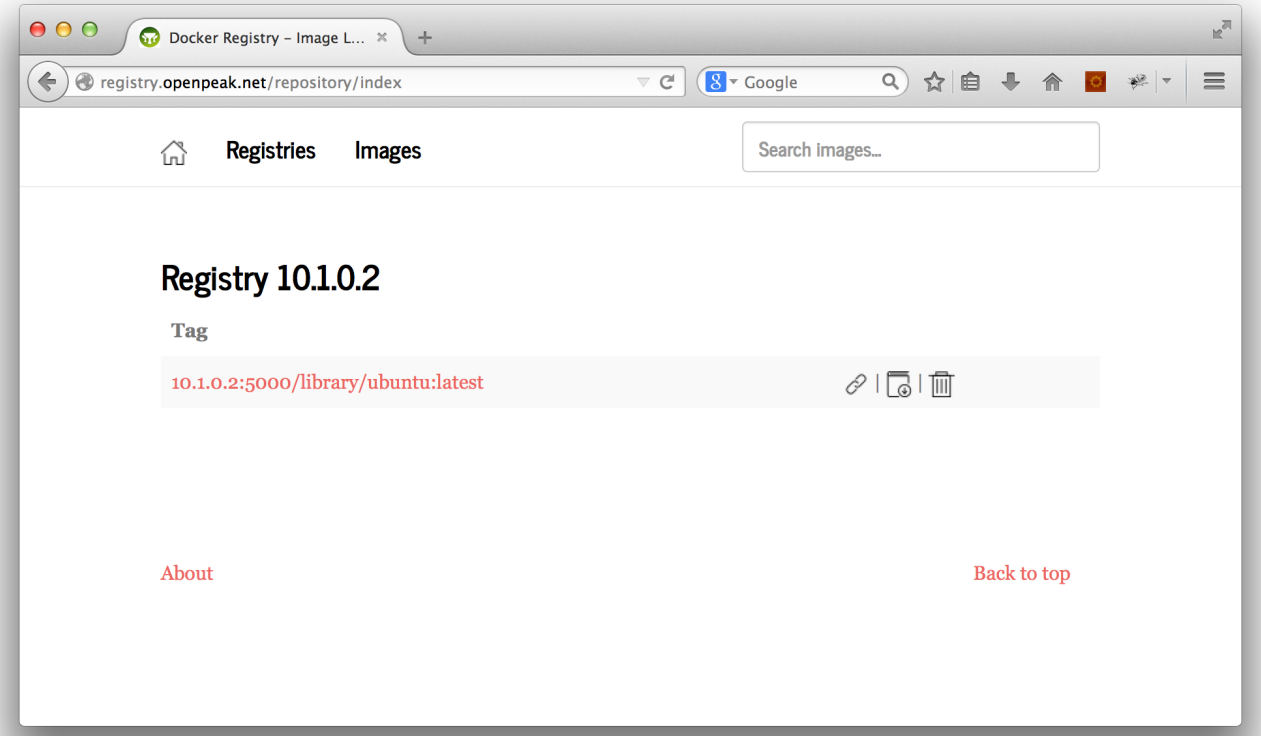
`tag`命令在我们的registry中为给定的其他地方的镜像保留了一个新的位置。你可以在本地列表中得到它的ID字符串。

默认情况下，Registry只与Docker socket直接通讯或通过REST进行管理。如果你想更容易地浏览镜像，请安装docker-registry-web项目：

```
$ docker run -p 8080:8080 -e REG1=http://<system hostname>:5000/v1/ atcol/docker-registry-ui
```

请记住，它需要与你的Registry实例联系，所以要确保你提供的registry主机名在docker-registry-web容器里可被解析。

截图如下：



docker-registry-web实际上是一个Java应用，然而它是个设计不佳的镜像（如果知道这点对你很重要的话）。

最后，在你玩够Registry实例后，记得将它隐藏在Nginx代理之后，并添加认证（双向、HTTP等）。

原文链接：Intro to Docker, and Private Image Registries（翻译：Sean 审校：林仁）