Docker 落地踩过的坑

前言

接触 Docker 是在大四毕业设计中，当时选的课题是：基于 Docker 的自动化运维研究。也很幸运，毕业从事的工作也是 Docker 相关的工作。第一份工作是在杭州铜板街，杭州的朋友可能听过，在那里工作了近 2 年时间。主要的工作是将 Docker 落地于公司，提高整个交付流程的效率。第二份工作是涂鸦智能，主要工作是，实现公司的 DevOps 。在整个 Docker 落地过程中，踩过无数的坑，这当中，包括技术的选型，如何将现有的开发习惯都 Docker 化，以及最大的困难和挑战就是让大家相信 Docker 能在 DevOps 中带来更好的帮助较于传统的自动化运维。本文，主要介绍几个在Docker 落地过程中的一些建议和常见的几个坑。
包括：

- Registry 使用 Harbor 对镜像仓库管理
- Registry 使用 oss 或者 s3 作为存储

目前主流的 Registry 有 Portus、Harbor、docker-registry-web 等。本文建议使用 VMware 的开源 Harbor 作为 Docker 镜像管理仓库。Harbor 相关文档较全面，社区相对活跃，而且功能强大。

Harbor 主要的功能有：

- 基于角色的访问控制
- 镜像同步复制
- 图形界面管理镜像
- 漏洞扫描
- 支持 LDAP/AD
- 镜像删除和垃圾回收
- 对所有操作进行审计
- 提供 RESTful API

详细介绍请参考官方文档 Harbor。

需要注意的一点是，尽量使用 https，安装文档：https 访问安装文档

使用 oss 或者 s3 作为存储

Docker 提供了丰富的镜像仓库存储方式，默认我们是使用本地的文件系统。这种方式有很多弊端，一个就是本地磁盘有限，因为镜像仓库是很耗存储的。如果咱们用的是 aws 或者 aliyun 机器，那么使用 s3 或 oss 作为镜像仓库的存储是超级便利的，也就再也不用担心磁盘不够，磁盘损坏等问题。这里介绍一下使用 aliyun 的 oss 最为镜像仓库的存储。

先看看使用 oss 最为存储的配置项：

```
storage:
  oss:
    accesskeyid: accesskeyid
    accesskeysecret: accesskeysecret
    region: OSS region name
```

Parameters

Parameter	Required	Description
`accesskeyid`	yes	Your access key ID.
`accesskeysecret`	yes	Your access key secret.
`region`	yes	The name of the OSS region in which you would like to store objects (for example `oss-cn-beijing`). For a list of regions, you can look at https://docs.aliyun.com/#/oss/product-documentation/domain-region.
`endpoint`	no	An endpoint which defaults to `[bucket].[region].aliyuncs.com` or `[bucket].[region]-internal.aliyuncs.com` (when `internal=true`). You can change the default endpoint by changing this value.
`internal`	no	An internal endpoint or the public endpoint for OSS access. The default is false. For a list of regions, you can look at [https://docs.aliyun.com/#/oss/product-documentation/domain-region] (https://docs.aliyun.com/#/oss/product-documentation/domain-region).
`bucket`	yes	The name of your OSS bucket where you wish to store objects (needs to already be created prior to driver initialization).
`encrypt`	no	Specifies whether you would like your data encrypted on the server side. Defaults to false if not specified.
`secure`	no	Specifies whether to transfer data to the bucket over ssl or not. If you omit this value, `true` is used.
`chunksize`	no	The default part size for multipart uploads (performed by WriteStream) to OSS. The default is 10 MB. Keep in mind that the minimum part size for OSS is 5MB. You might experience better performance for larger chunk sizes depending on the speed of your connection to OSS.
`rootdirectory`	no	The root directory tree in which to store all registry files. Defaults to an empty string (bucket root).

下面是一个具体例子：


```
storage:
    cache:
        layerinfo: inmemory
    oss:
      accesskeyid: xxxx
      accesskeysecret: xxxx
      region: oss-cn-hongkong
      endpoint: xxx.oss-cn-hongkong.aliyuncs.com
      bucket: xxx
    maintenance:
        uploadpurging:
```

【公共云】中国大陆区域--价格详情

中国大陆区域包括：华东1、华东2、华北1、华北2、华北3、华北5、华南1；全国统一价，各资源使用单价详情如下：

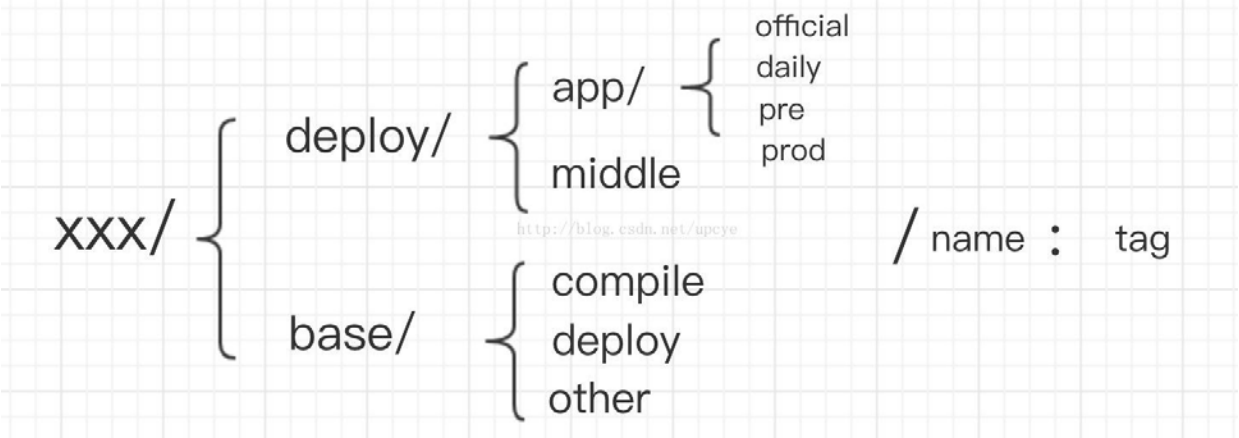
按量付费-资费详情　包年包月-资费详情

资费项	计费项	标准型单价	低频访问型单价	归档型单价
存储费用 (注①)	数据存储	0.148元/GB/月	0.08元/GB/月	0.033/GB/月
流量费用 (注①)	内/外网流入流量（数据上传到OSS）	免费	免费	免费
	内网流出流量（通过ECS云服务器下载OSS的数据）	免费	免费	免费
	外网流出流量	00:00-08:00（闲时）：0.25元/GB 8:00-24:00（忙时）：0.50元/GB	00:00-08:00（闲时）：0.25元/GB 8:00-24:00（忙时）：0.50元/GB	00:00-08:00（闲时）：0.25元/GB 8:00-24:00（忙时）：0.50元/GB
	CDN回源流出流量	0.15元/GB	0.15元/GB	0.15元/GB
	跨区域复制流量	0.50元/GB	0.50元/GB	0.50元/GB
请求费用	所有请求类型	0.01元/万次	0.1元/万次	0.1元/万次
数据处理费用 (注②)	图片处理	每月0-10TB：免费 >10TB：0.025元/GB	每月0-10TB：免费 >10TB：0.025元/GB	无
	视频截帧	0.1元/千张	0.1元/千张	无
	数据取回	免费	0.0325元/GB	0.06元/GB

我们可以看到，外网流出的费用是需要收费的，所以如果 oss 和 开的 ECS 都是在同一个 Region ，建议配置 endpoint : true ，然后 endpoint 配置内网的地址。可以节省成本。

- 如果公司使用的是 aliyun ECS ，而且机器涉及到跨 Region 。那么建议搭建多个镜像仓库。比如开发环境或者日常环境搭使用一个仓库，线上使用一个仓库。线上的仓库只存储预发环境或者是线上环境的镜像。具体仓库部署可以后面一起讨论。

Registry 批量清理镜像



这样做的好处是能够方便的做到批量删除镜像。比如想删除日常环境下的镜像，只需要匹配 xxx/deploy/app/daily 就行。

借助 Harbor Restful API 删除镜像或 Tag

主要有下面几个步骤：

- 如果想删除 xxx/deploy/app/daily 下的所有镜像。

> 1. 使用 GET /api/repositories ,该 API 有个 filter 参数，设置 filter=xxx/deploy/app/daily ，则可以匹配查找出xxx/deploy/app/daily 下的所有 repositories。
> 2. 遍历上一步拿到的 repositories， 使用 DELETE /api/repositories/${repoName}

- 如果想删除某个 repo 的 tag，同理，也可以调用 API 删除

- 上述删除动作 ，实际上并没有删除，只是删除了 Registry 的索引。实际文件并没有删除。

最后还需要执行镜像的垃圾回收：

```
docker run -it –name gc –rm –volumes-from registry vmware/registry:2.6.2-photon garbage-collect /etc/registry/config.yml
```

在容器部署应用和常规部署应用混合使用一套 zk 或者是在 Docker 集群下容器部署应用，将出现 dubbo 服务调用失败的问题。

原因

在容器中部署应用时，应用中的 dubbo 服务，获取容器 hostname 的 IP 地址和配置文件中的端口号向注册中心中注册。由于不同网段不能互相通信访问，这就导致了，容器部署应用和常规部署应用混合共用一套注册中心或是在 Docker 集群下容器部署应用时，将出现 dubbo 消费者不能访问服务提供者。

分析

由于 dubbo 服务注册，是去拿 hostname 的 IP 向，通过这点，hostname 必须为宿主机的 IP 地址。为了解决上述问题，首要解决的问题是网络通信问题。

解决方法

方法一：启用容器时，指定容器的网络模式为 host 模式，因为 host 模式容器共享主机的网络。

```
docker run –net=host
```


优点：简单，容易配置。

缺点：一个机器相同应用只能起一份。

方法二：想办法让主机 hostname 的 IP 为宿主机的 IP，这样通过宿主机的 IP 和映射的 dubbo 端口号，就能避免这个问题。这里推荐一个比较挫的方法。

启动容器时，指定 hostname，并且将宿主机的 /etc/hosts 挂载至容器中，如：/tmp/hosts，容器启动的时候，读取/tmp/hosts 中宿主机的 IP，写入到容器的 /etc/hosts 中。

具体步骤如下：

- 容器启动时执行脚本，步骤如下：

  1. 首先从 /tmp/hosts 中获取宿主机的 IP

     ```
     hostIP= grep -F hostname /home/tomcat/hosts |awk '{print $1}'
     ```

  2. 注释掉容器自身的host 配置(由于容器中的 /etc/hosts 权限问题，需采用如下曲折方法)

     ```
     cp /etc/hosts /etc/hosts.temp
     sed -i 's/^[^#].*dubbo-service-provider/#&/' /etc/hosts.temp
     cat /etc/hosts.temp >/etc/hosts
     ```

  3. 将宿主机IP追加到容器/etc/hosts

     ```
     echo "${hostIP} dubbo-service-provider" >> /etc/hosts
     ```

     以上两种方法可以解决 dubbo docker 化中网络通信问题。如果是在线上，建议使用第一种。不仅仅是 dubbo 服务注册会遇到这种网络通信问题，很多中间件都有同样的问题，比如 rocketMq、kafka 等。下面介绍一下 RocketMQ broker 注册 IP 问题。

RocketMQ broker 注册 IP 问题

问题描述

RocketMQ 装入容器中时，Broker 注册地址将使用容器自身的 IP ，导致 consumer 端不能从 broker 中拿到消费消息。

分析

- 加载修改过的配置文件：nohup sh mqbroker -c broker.p

总结

由于篇幅和时间有限，以上整理的几个建议和踩过的坑，分享给在 Docker 化落地的朋友，希望能够在这条路上少踩点坑。未来不久，我们会计划开源我们的 DevOps 平台，和大家一起探索和进步。