docker私有仓库

原创 Mr大表哥 2017-03-14 16:17:39 评论(0) 220人阅读

博主QQ：819594300
博客地址：http://zpf666.blog.51cto.com/
有什么疑问的朋友可以联系博主，博主会帮你们解答，谢谢支持!

Docker仓库

仓库（Repository）是集中存放镜像的地方。
一个容易混淆的概念是注册服务器（Registry） 。实际上注册服务器是管理仓库的具体服务器，每个服务器上可以有多个仓库，而每个仓库下面有多个镜像。从这方面来说，仓库可以被认为是一个具体的项目或目录。例如对于仓库地址 docker.benet.com/centos:centos7 来说，docker.benet.com是注册服务器地址，centos 是仓库名，centos7 是仓库的 tag(标签)。

Docker Hub 官方仓库
目前 Docker 官方维护了一个公共仓库Docker Hub，其中已经包括了超过 15,000 的镜像。大部分需求，都可以通过在 Docker Hub 中直接下载镜像来实现。
注册&登录
可以通过命令行执行 docker login 命令来输入用户名、 密码和邮箱来完成注册和登录。注册成功后，本地用户目录的 .docker/config.json 中将保存用户的认证信息。

```
[root@localhost ~]# docker login
Username: zhengpengfei
Password:
Email: 819594300@qq.com
WARNING: login credentials saved in /root/.docker/config.json
Login Succeeded
[root@localhost ~]#
```

```
[root@localhost ~]# cat ~/.docker/config.json
{
        "auths": {
                "https://index.docker.io/v1/": {
                        "auth": "emhlbmdwZW5nZmVpOjEyMzQ1Ng==",
                        "email": "819594300@qq.com"
                }
        }
}[root@localhost ~]#
```

基本操作

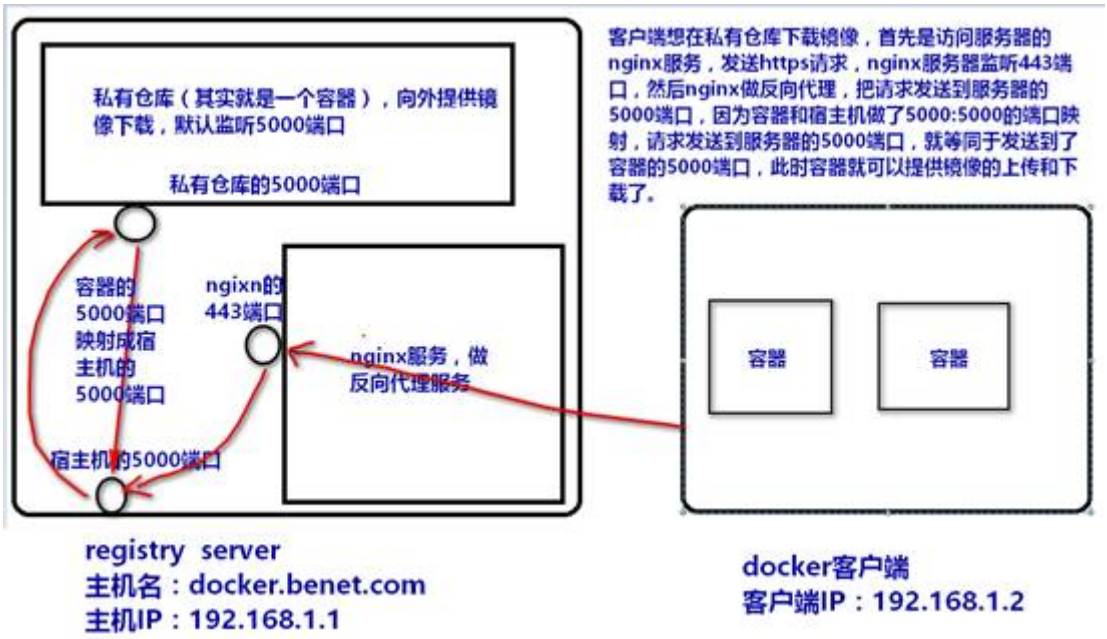
用户无需登录即可通过 “docker search 关键字”命令来查找官方仓库中的镜像，并利用 docker pull 命令来将它下载到本地。

可以看到返回了很多包含关键字的镜像， 其中包括镜像名字、 描述、 星级 （表示该镜像的受欢迎程度） 、是否官方创建、是否自动创建。官方的镜像说明是官方项目组创建和维护的，automated 资源允许用户验证镜像的来源和内容。

根据是否是官方提供，可将镜像资源分为两类。 一种是类似 centos 这样的基础镜像，被称为基础或根镜像。这些基础镜像是由 Docker 公司创建、验证、支持、提供。这样的镜像往往使用单个单词作为名字。 还有一种类型，比如 tianon/centos 镜像，它是由 Docker 的用户创建并维护的，往往带有用户名称前缀。可以通过前缀 user_name/ 来指定使用某个用户提供的镜像，比如 tianon 用户。另外，在查找的时候通过 -s N 参数可以指定仅显示评价为 N 星以上的镜像。

创建自己的仓库----镜像仓库

拓扑图：



说明:

docker.benet.com这是docker registry服务器的主机名称， ip是192.168.1.1;因为https的SSL证书要用到主机名，所以要设置主机名。

docker registry 服务器作为处理docker镜像的最终上传和下载，用的是官方的镜像registry。

nginx 1.6.x 是一个用nginx作为反向代理服务器。

准备工作:

关闭selinux（临时关闭selinux命令：setenforce 0,也可以永久关闭）

实验步骤:

1）私有仓库https支持:

A)修改主机名和hosts文件

```
[root@localhost ~]# vim /etc/hostname
docker.benet.com
~
```

```
[root@localhost ~]# bash
[root@docker ~]# hostname
docker.benet.com
[root@docker ~]#
```

```
[root@docker ~]# vim /etc/hosts
127.0.0.1   localhost localhost.localdomain
::1         localhost localhost.localdomain
192.168.1.1      docker.benet.com
~
```

B)安装依赖软件包

```
[root@docker ~]# mount /dev/sr0 /media
mount: /dev/sr0 写保护，将以只读方式挂载
[root@docker ~]# rm -rf /etc/yum.repos.d/CentOS-*
[root@docker ~]# vim /etc/yum.repos.d/docker.repo
```

```
[docker]
name=registry
baseurl=file:///media
enabled=1
gpgcheck=0
```

```
[root@docker ~]# yum -y install pcre-devel zlib-devel openssl openssl-devel
```

在Nginx编译需要pcre，因为Nginx的Rewrite模块和HTTP核心模块会使用到pcre正则表达式。需要安装pcre和pcre-devel用yum就能安装。
Zlib库提供了开发人员的压缩算法，在nginx的模块中需要使用gzip压缩。
需要安装zlib和zlib-devel用yum就可以安装
在Nginx中如果需要为服务器提供安全则需要用到OpenSSL库。
需要安装的是openssl和openssl-devel。用yum就可以安装。

C)配置SSL

(1)生成根密钥

如果有

/etc/pki/CA/cacert.pem
/etc/pki/CA/index.txt
/etc/pki/CA/index.txt.attr
/etc/pki/CA/index.txt.old
/etc/pki/CA/serial
/etc/pki/CA/serial.old

则先删除这些文件再继续，没有的就下面继续。

```
[root@docker ~]# cd /etc/pki/CA/
[root@docker CA]# openssl genrsa -out private/cakey.pem 2048
Generating RSA private key, 2048 bit long modulus
....+++
.....................+++
e is 65537 (0x10001)
[root@docker CA]#
```

cakey.pem就是根密钥

```
[root@docker CA]# ls private/
cakey.pem
[root@docker CA]#
```

(2) 生成根证书

```
[root@docker CA]# openssl req -new -x509 -key private/cakey.pem -out cacert.pem
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [XX]:CN
State or Province Name (full name) []:beijing
Locality Name (eg, city) [Default City]:beijing
Organization Name (eg, company) [Default Company Ltd]:bdqn
Organizational Unit Name (eg, section) []:benet
Common Name (eg, your name or your server's hostname) []:docker.benet.com
Email Address []:819594300@qq.com
[root@docker CA]#
```

通过根密钥来生成根证书。

下面会提示输入一些内容，因为是私有的，所以可以随便输入，最好记住能与后面保持一致,特别是"Common Name"，必须要和hostname显示的一致。

依次填写的是国家两个英文字母的代码、省份、城市、组织名字、组织单位名字、主机名字、邮箱。

```
[root@docker CA]# ls
cacert.pem  certs  crl  newcerts  private
[root@docker CA]#
```

上面的自签证书cacert.pem应该生成在/etc/pki/CA下。

(3) 为nginx web服务器生成ssl密钥

```
[root@docker CA]# mkdir /etc/pki/CA/ssl
[root@docker CA]# cd !$
cd /etc/pki/CA/ssl
[root@docker ssl]# openssl genrsa -out nginx.key 2048
Generating RSA private key, 2048 bit long modulus
.....................................................+++
.....................+++
e is 65537 (0x10001)
[root@docker ssl]#
```

注：因为CA中心与要申请证书的nginx服务器是同一台主机，所以就在本机上执行，为nginx服务器生成ssl密钥，否则应该是在另一台需要用到证书的服务器上生成。

```
[root@docker ssl]# ls
nginx.key
[root@docker ssl]#
```

查看nginx服务器的密钥

(4) 为nginx生成证书签署请求（通过私钥文件生成请求文件）

```
[root@docker ssl]# openssl req -new -key nginx.key -out nginx.csr
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [XX]:CN
State or Province Name (full name) []:beijing
Locality Name (eg, city) [Default City]:beijing
Organization Name (eg, company) [Default Company Ltd]:bdqn
Organizational Unit Name (eg, section) []:benet
Common Name (eg, your name or your server's hostname) []:docker.benet.com
Email Address []:819594300@qq.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
[root@docker ssl]#
```

.csr是请求文件。
这里同样会提示输入一些内容（和前面的一定要一样，就是那个common Name可以不一样），Commone Name一定要是你要授予证书的服务器域名或主机名（即nginx本地主机名），其他内容一定要和前面保持一致
challenge password和optional company name不填。

(5) 私有CA根据请求来签发证书（通过请求文件让CA给颁发证书）

```
[root@docker ssl]# touch /etc/pki/CA/index.txt
[root@docker ssl]# touch /etc/pki/CA/serial
[root@docker ssl]# echo 00 > /etc/pki/CA/serial
[root@docker ssl]# openssl ca -in nginx.csr -out nginx.crt
```

.csr是请求文件
.crt是证书文件

```
Certificate is to be certified until Feb 16 13:48:14 2018 GMT (365 days)
Sign the certificate? [y/n]:y


1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
[root@docker ssl]#
```

```
[root@docker ssl]# ls
nginx.crt  nginx.csr  nginx.key
[root@docker ssl]#
```

查看nginx的证书

C) 安装,配置,运行nginx

(1) 添加组和用户

```
[root@docker ssl]# groupadd www -g 58
[root@docker ssl]# useradd -u 58 -g www www
[root@docker ssl]#
```

(2) 下载nginx源文件并解压缩、安装
（我这里有下载好的直接解压缩、安装即可）

```
[root@docker ~]# umount /dev/cdrom
[root@docker ~]# umount /dev/cdrom
[root@docker ~]# cd /run/media/root/20170216_113104/
[root@docker 20170216_113104]# tar zxf nginx-1.11.2.tar.gz -C /usr/src/
[root@docker 20170216_113104]# cd /usr/src/nginx-1.11.2/
[root@docker nginx-1.11.2]# ./configure --prefix=/opt/nginx --user=www --group=www --with-pcre --with-http_stub_status_module --with-http_ssl_module --with-http_addition_module --with-http_realip_module --with-http_flv_module && make && make install
```

上述选项的解释：

--user=USER 设定程序运行的用户环境(www)

--group=GROUP 设定程序运行的组环境(www)

--prefix=PATH 设定安装目录

--with-pcre 启用pcre库，Nginx的Rewrite模块和HTTP核心模块会使用到PCRE正则表达式

--with-http_stub_status_module 是为了启用 nginx 的 NginxStatus 功能，用来监控 Nginx 的当前状态

--with-http_ssl_module 开启SSL模块，支持使用HTTPS协议的网页

--with-http_realip_module 开启Real IP的支持，该模块用于从客户请求的头数据中读取Real Ip地址

--with-http_addition_module 开启Addtion模块，该模块允许你追加或前置数据到相应的主体部分

--with-http_flv_module模块ngx_http_flv_module为Flash Video(FLV)文件 提供服务端伪流媒体支持

(3) 编辑/opt/nginx/conf/nginx.conf文件

```
[root@docker nginx-1.11.2]# vim /opt/nginx/conf/nginx.conf
```

相关选项含义：

ssl_session_cache 会话缓存用于保存SSL会话，这些缓存在工作进程间共享，可以使用ssl_session_cache指令进行配置。1M缓存可以存放大约4000个会话。

ssl_session_timeout 缓存超时，默认的缓存超时是5分钟。

ssl_ciphers HIGH:!aNULL:!MD5 使用高强度的加密算法

ssl_prefer_server_ciphers on 依赖SSLv3和TLSv1协议的服务器密码将优先于客户端密码。即：在SSLv3或这是TLSv1握手时选择一个密码，通常是使用客户端的偏好。如果这个指令是启用的，那么服务器反而是使用服务器的偏好。

client_max_body_size 即允许上传文件大小的最大值

proxy_set_header Host $host和proxy_set_header X-Forward-For $remote_addr的作用描述：

nginx为了实现反向代理的需求而增加了一个ngx_http_proxy_module模块。其中proxy_set_header指令就是该模块需要读取的配置文件。在这里，所有设置的值的含义和http请求同中的含义完全相同，除了Host外还有X-Forward-For。

Host的含义是表明请求的主机名，因为nginx作为反向代理使用，而如果后端真实的服务器设置有类似防盗链或者根据http请求头中的host字段来进行路由或判断功能的话，如果反向代理层的nginx不重写请求头中的host字段，将会导致请求失败【默认反向代理服务器会向后端真实服务器发送请求，并且请求头中的host字段应为proxy_pass指令设置的服务器】。

同理，X_Forward_For字段表示该条http请求是有谁发起的？如果反向代理服务器不重写该请求头的话，那么后端真实服务器在处理时会认为所有的请求都来自反向代理服务器，如果后端有防攻击策略的话，那么机器就被封掉了。因此，在配置用作反向代理的nginx中一般会增加两条配置，修改http的请求头：

proxy_set_header Host $ host;

proxy_set_header X-Forward-For $remote_addr;
这里的$ host和$remote_addr都是nginx的导出变量，可以再配置文件中直接使用。

（4）验证配置，并启动nginx，查看nginx是否监听443端口

```
[root@docker nginx-1.11.2]# ln -s /opt/nginx/sbin/nginx /usr/local/sbin
[root@docker nginx-1.11.2]# nginx -t
nginx: the configuration file /opt/nginx/conf/nginx.conf syntax is ok
nginx: configuration file /opt/nginx/conf/nginx.conf test is successful
[root@docker nginx-1.11.2]# nginx
[root@docker nginx-1.11.2]# netstat -anpt | grep nginx
tcp        0      0 0.0.0.0:443             0.0.0.0:*               LISTEN      59531/nginx: master
[root@docker nginx-1.11.2]#
```

```
[root@docker nginx-1.11.2]# ps -ef | grep -i "nginx"
root      59531      1  0 22:11 ?        00:00:00 nginx: master process nginx
www       59533  59531  0 22:11 ?        00:00:00 nginx: worker process
www       59534  59531  0 22:11 ?        00:00:00 nginx: worker process
www       59535  59531  0 22:11 ?        00:00:00 nginx: worker process
www       59536  59531  0 22:11 ?        00:00:00 nginx: worker process
root      59572  55856  0 22:13 pts/0    00:00:00 grep --color=auto -i nginx
[root@docker nginx-1.11.2]#
```

（5）防火墙开启443/tcp端口例外

```
[root@docker ~]# firewall-cmd --permanent --add-service=https
success
[root@docker ~]# firewall-cmd --reload
success
[root@docker ~]#
```

2）配置，运行Docker

（1）停止docker

```
[root@docker ~]# systemctl stop docker
[root@docker ~]#
```

（2）编辑/etc/sysconfig/docker文件，加上如下一行

```
[root@docker ~]# vim /etc/sysconfig/docker
```

```
1 # /etc/sysconfig/docker
2 DOCKER_OPTS="--insecure-registry docker.benet.com --tlsverify --tlscacert /etc/pki/CA/cacert.pem"
3 # Modify these options if you want to change the way the docker daemon runs
4 OPTIONS='--selinux-enabled --log-driver=journald'
5 DOCKER_CERT_PATH=/etc/docker
```

（3）把根证书复制到/etc/docker/certs.d/docker.**.com/目录下

```
[root@docker ~]# mkdir -p /etc/docker/certs.d/docker.benet.com
[root@docker ~]# cp /etc/pki/CA/cacert.pem /etc/docker/certs.d/docker.benet.com/ca-certificates.crt
[root@docker ~]#
```

把根证书拷贝一份给nginx服务，并起一个固定的名字（名字必须叫那个名字），让nginx根据其根证书检验私有仓里容器发来的证书。

（4）启动docker

```
[root@docker ~]# systemctl start docker
[root@docker ~]#
```

3）运行私有仓库容器

通过获取官方 registry 镜像来运行

（我这里已经下载好了，是归档文档，我直接解压缩即可）

```
[root@docker ~]# cd /run/media/root/20170216_113104/
[root@docker 20170216_113104]# docker load -i registry.tar
[root@docker 20170216_113104]# docker images
REPOSITORY              TAG                 IMAGE ID            CREATED             SIZE
docker.io/registry      latest              bca04f698ba8        12 months ago       422.8 MB
[root@docker 20170216_113104]#
```

```
[root@docker ~]# mkdir -p /opt/data/registry
[root@docker ~]#
```

使用官方的 registry 镜像来启动本地的私有仓库，用户可以通过指定参数来配置私有仓库位置。

```
[root@docker ~]# docker run -dit --name=registry -p 5000:5000 -v /opt/data/registry/:/tmp/registry docker.io/registry
95ef7b6b9aa3e67736fe95c486df9d36d6de47de5afbdb5b3c40a25687d89b51
[root@docker ~]# docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS                    NAMES
95ef7b6b9aa3        docker.io/registry  "docker-registry"        7 seconds ago       Up 4 seconds        0.0.0.0:5000->5000/tcp   registry
[root@docker ~]#
```

创建私有仓库容器，让其后台运行，并且把容器的5000端口映射到宿主机的5000端口，并挂载数据卷，其中容器内的挂载点/tmp/registry必须是这个，不能改变。

4）验证registry:

用浏览器输入: https://docker.benet.com

看见这个即为成功

或者:curl -i -k https://docker.benet.com

```
[root@docker ~]# curl -ik https://docker.benet.com
HTTP/1.1 200 OK
Server: nginx/1.11.2
Date: Thu, 16 Feb 2017 14:42:27 GMT
Content-Type: application/json
Content-Length: 28
Connection: keep-alive
Expires: -1
Pragma: no-cache
Cache-Control: no-cache

"\"docker-registry server\""[root@docker ~]#
```

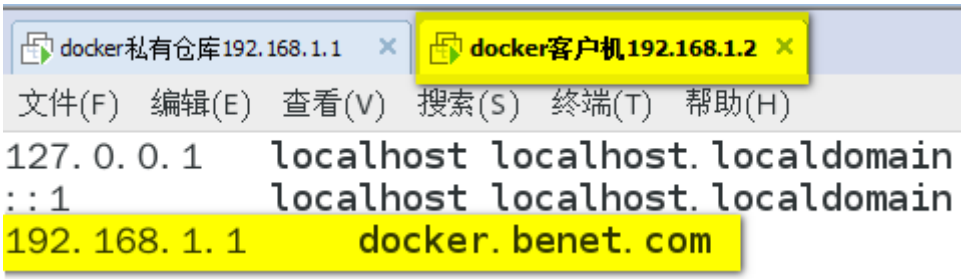
curl是通过url语法在命令行下上传或下载文件的工具软件，它支持http,https,ftp,ftps,telnet等多种协议，常被用来抓取网页和监控Web服务器状态。

返回的信息是“200 OK”即为成功

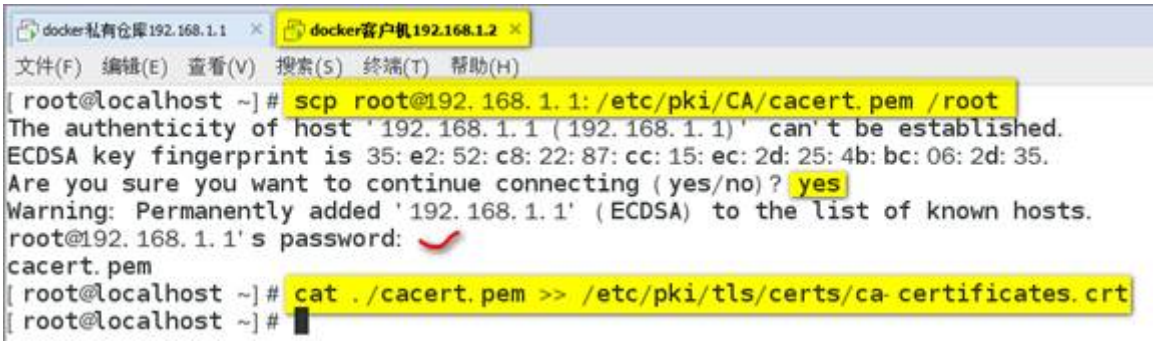
服务端的配置就到此完成!

5）Docker客户端配置

(1)编辑/etc/hosts,把docker.benet.com的ip地址添加进来

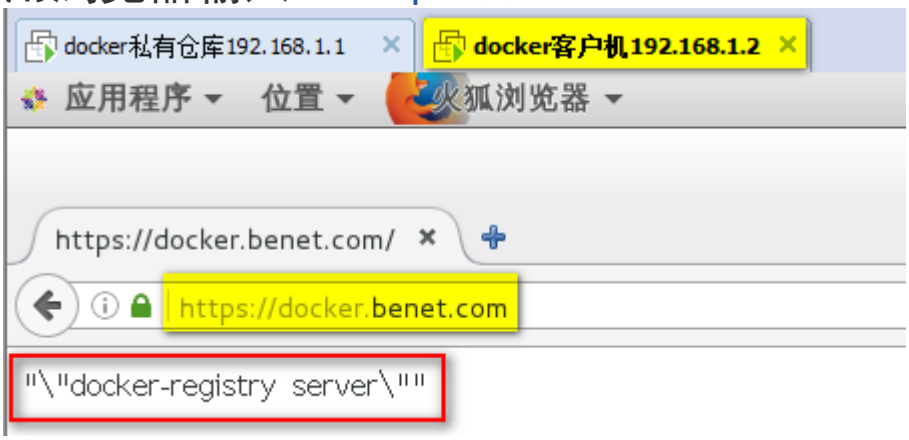


(2)把docker registry服务器端的根证书追加到ca-certificates.crt文件里



(3) 验证docker.benet.com下的registry:

用浏览器输入: https://docker.benet.com



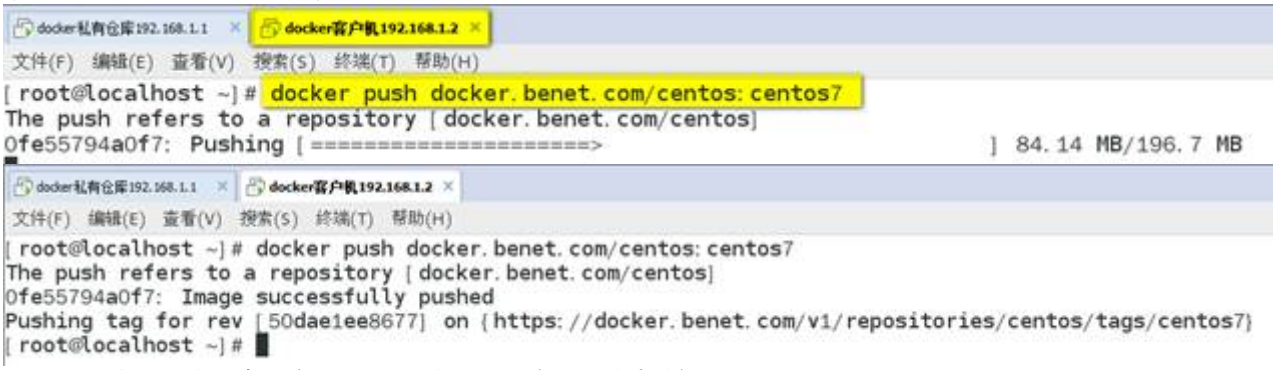
或者:curl -i -k https://docker.benet.com

(4) 使用私有registry步骤:

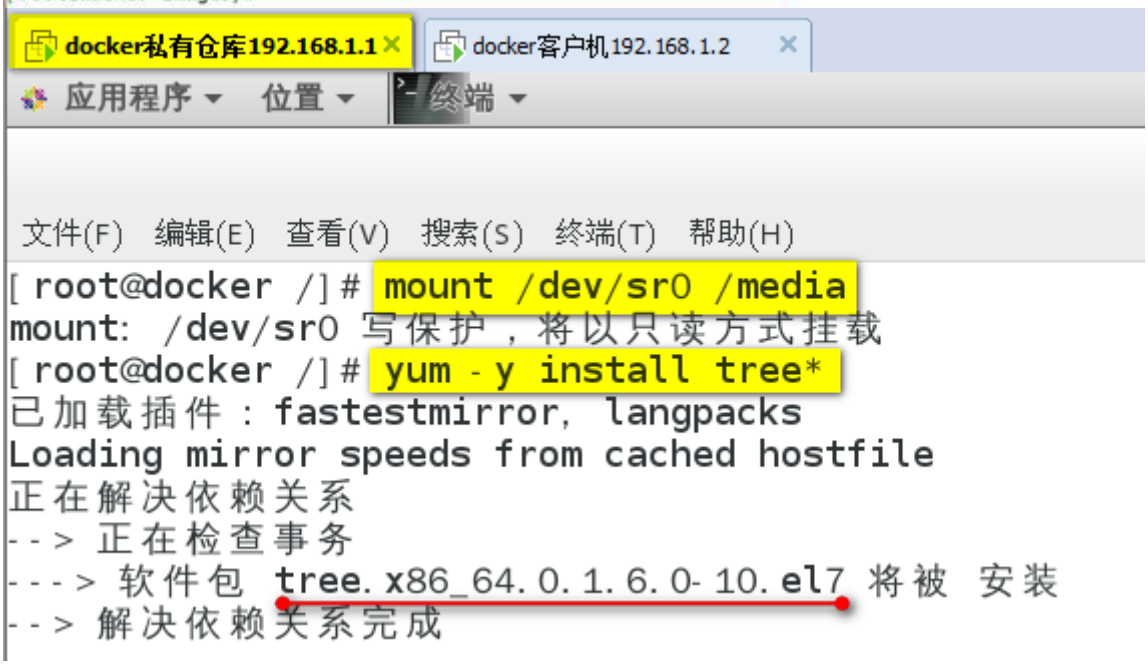
登录: docker login -utestuser -p pwd123 -e "test@benet.com" https://docker.benet.com

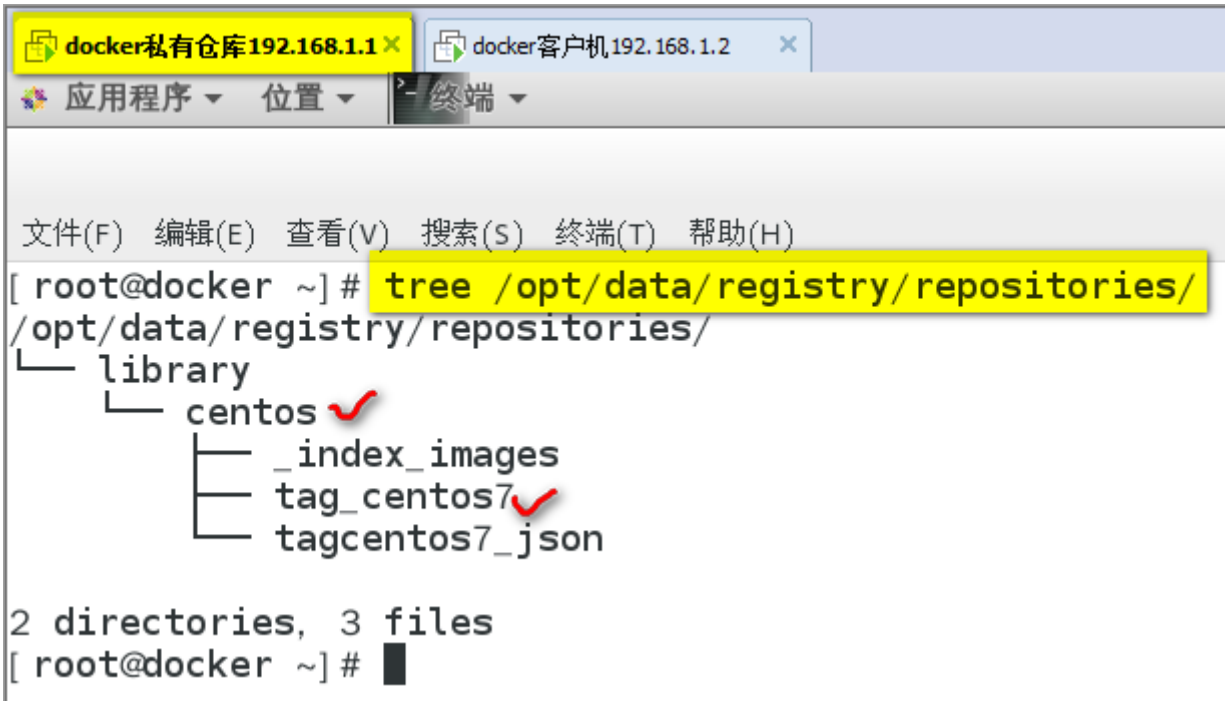
从Docker HUB 上拉取一个镜像测试，为基础镜像打个标签（我已下载好镜像，直接修改标签即可）

发布：上传镜像到本地私有仓库

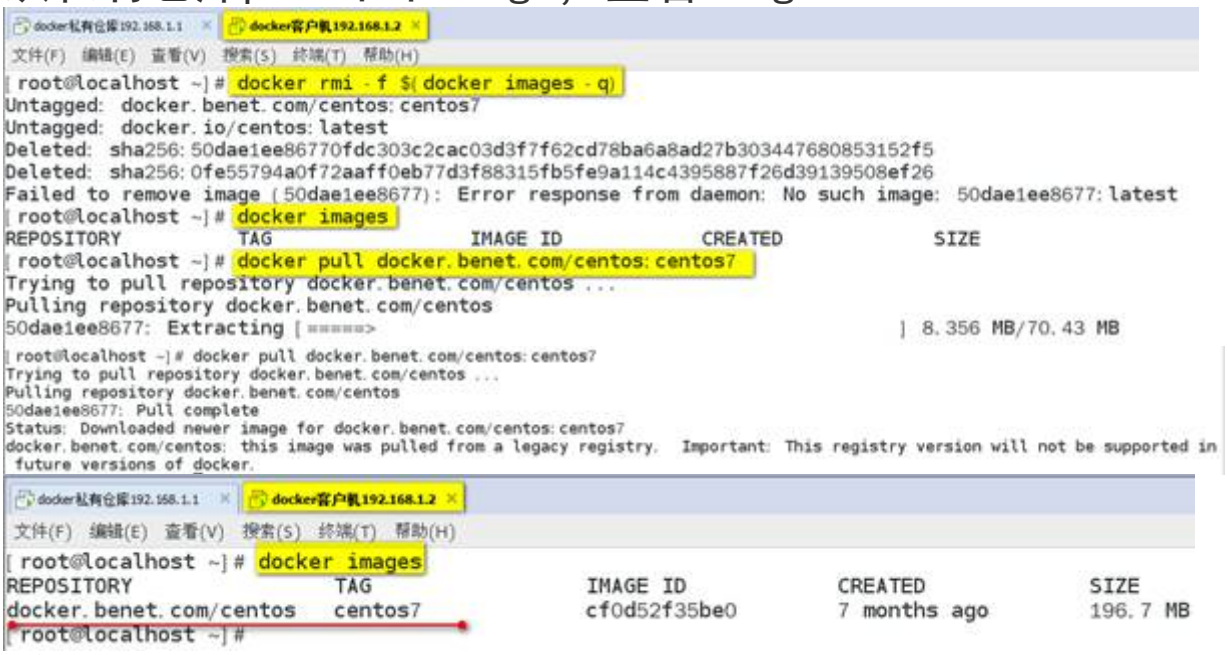


查看私有仓库是否有对应的镜像





从私有仓库pull下来image，查看image



说明：

(1) 弊端:

server端可以login到官方的DockerHub,可以pull,push官方和私有仓库!

client端只能操作搭设好的私有仓库!

私有仓库不能search!

(2) 优点:

所有的build,pull,push操作只能在私有仓库的server端操作,降低企业风险!

(3)报错：当client端docker login到官方的https://index.docker.io/v1/网站,出现x509: certificate signed by unknownauthority错误时

重命名根证书mv /etc/pki/tls/certs/ca-certificates.crt/etc/pki/tls/certs/ca-certificates.crt.bak
重启docker服务！servicedocker restart！