Docker 构建本地私有仓库

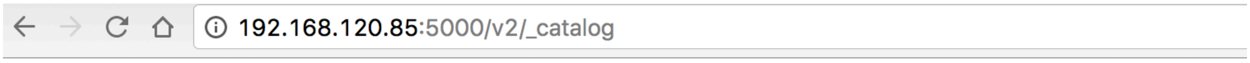
版本号	构建时间	备注
v1.0	2017.7.27	初稿完成

1. 搭建本地私有仓库

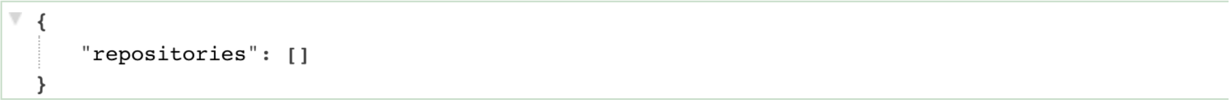
我们使用docker官方提供的*reigstry*镜像来搭建本地私有仓库环境

```
➜  ~ docker run -d -p 5000:5000 -v /opt/data/registry:/tmp/registry registry
Unable to find image 'registry:latest' locally
latest: Pulling from library/registry
90f4dba627d6: Pull complete
3a754cdc94a5: Pull complete
bf16d9b6d4c1: Pull complete
7eea83c9b7bb: Pull complete
23293c727551: Pull complete
Digest: sha256:f5552e60ffd56fecbe2f04b61a3089a9cd755bd9352b6b5ab22cf2208af6a3a8
Status: Downloaded newer image for registry:latest
2866677f454d110f4d78a572ecee5e542a84dc16179f5e7775da86407a017f4b
```

和搭建普通Docker镜像命令类似。由于本地不存在`registry`镜像，Docker会先下载该镜像，然后执行`run`操作，映射容器内的`5000`端口到本机的`5000`端口，映射`/tmp/registry registry`到本机的`/opt/data/registry`，表示上传的镜像存储目录为：`/opt/data/registry`。然后我们可以尝试使用浏览器访问私有仓库：



192.168.120.85:5000/v2/_catalog



```
{
    "repositories": []
}
```

或者是使用`curl`命令

```
➜  ~ curl -XGET 127.0.0.1:5000/v2/_catalog
{"repositories":[]}
```

查看镜像版本列表路径为：

```
➜  ~ curl -XGET 127.0.0.1:5000/v2/image_name/tags/list
```

> Tip:Docker Register HTTP API V2 在这里

2. 搭建Harbor运行环境

Docker官方并没有提供docker registry的用户界面，对权限的控制粒度也比较粗。所以我们需要使用第三方提供的web ui和相关的权限控制，其中相对优秀的第三方镜像有：SUSE提供的Portus 和Vmware提供的Harbor ,他们都提供了良好的安装文档，WEB UI界面，更细粒度的权限控制和用户认证等功能。由于两者没有明显的差别，此处我们选用的是Vmware公司提供的**Harbor**，因为它在github的star比较多。

因为**Harbor**中包含docker-registry，所以我们先把我们的docker-registry停止掉，避免出现端口冲突等不必要的麻烦。

我们跟随**Harbor**的installation_guide 完成安装。

2.1.安装前准备

2.1.1 下载

Harbor提供了两种安装方式，分别是*在线安装*和*离线安装*。区别如下：

> - **Online installer:** The installer downloads Harbor's images from Docker hub. For this reason, the installer is very small in size.
> - **Offline installer:** Use this installer when the host does not have an Internet connection. The installer contains pre-built images so its size is larger.

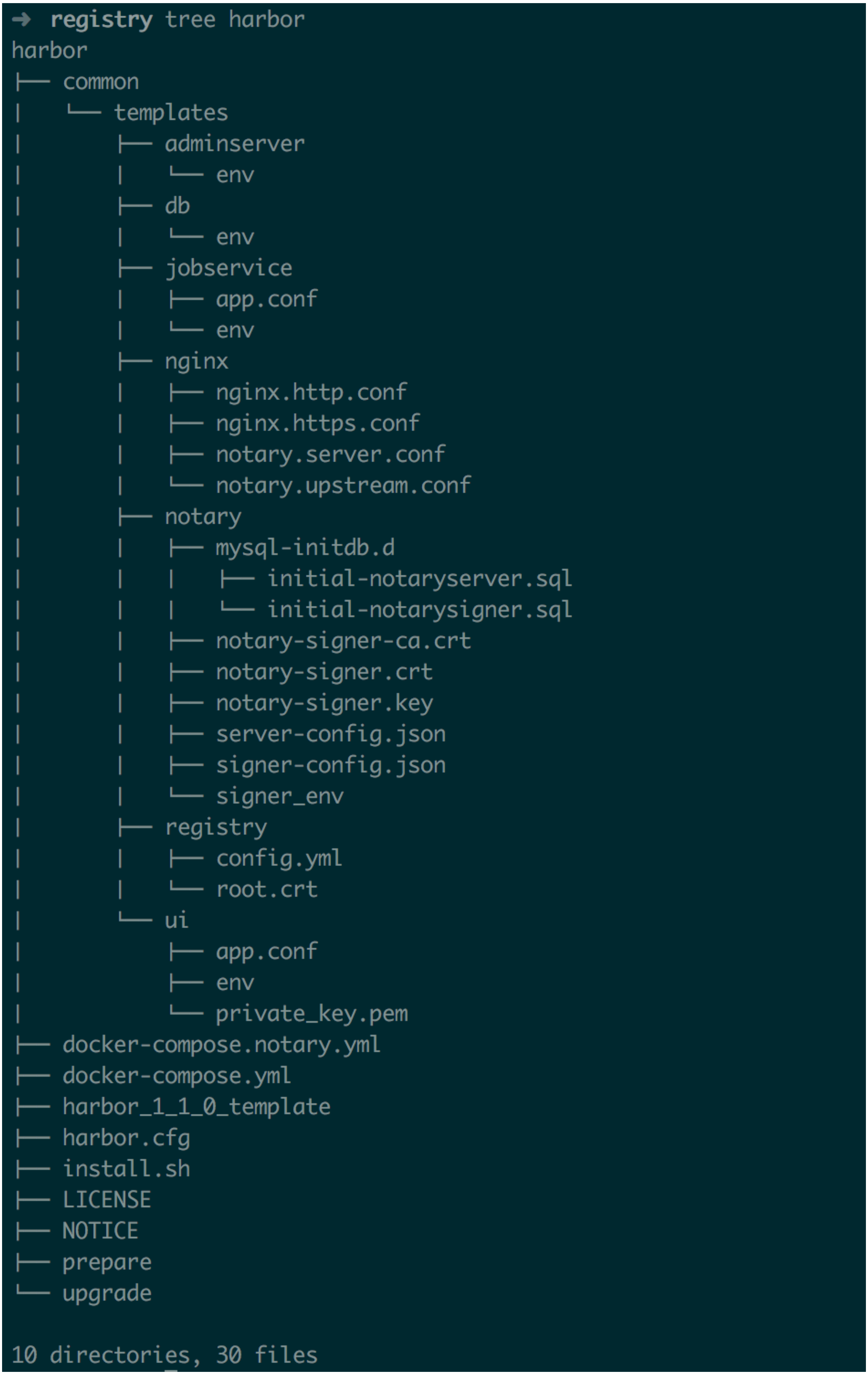
此处我们选择在线安装，首先我们去这里下载离线包，我们使用的版本是**v1.1.2**

```
➜ wget https://github.com/vmware/harbor/releases/download/v1.1.2/harbor-online-installer-v1.1.2.tgz
```

2.1.2 解压文件

```
➜ tar xvf harbor-online-installer-v1.1.2.tgz
```

解压之后的目录结构如下：



```
➜  registry tree harbor
harbor
├── common
│   └── templates
│       ├── adminserver
│       │   └── env
│       ├── db
│       │   └── env
│       ├── jobservice
│       │   ├── app.conf
│       │   └── env
│       ├── nginx
│       │   ├── nginx.http.conf
│       │   ├── nginx.https.conf
│       │   ├── notary.server.conf
│       │   └── notary.upstream.conf
│       ├── notary
│       │   ├── mysql-initdb.d
│       │   │   ├── initial-notaryserver.sql
│       │   │   └── initial-notarysigner.sql
│       │   ├── notary-signer-ca.crt
│       │   ├── notary-signer.crt
│       │   ├── notary-signer.key
│       │   ├── server-config.json
│       │   ├── signer-config.json
│       │   └── signer_env
│       ├── registry
│       │   ├── config.yml
│       │   └── root.crt
│       └── ui
│           ├── app.conf
│           ├── env
│           └── private_key.pem
├── docker-compose.notary.yml
├── docker-compose.yml
├── harbor_1_1_0_template
├── harbor.cfg
├── install.sh
├── LICENSE
├── NOTICE
├── prepare
└── upgrade

10 directories, 30 files
```

其中最外层有一个 `install.sh` 脚本，用于安装 Harbor，config 目录存放了一些配置信息，如 registry 和 ui 目录中存放了相关证书用于组件间加密通讯，`harbor.cfg` 是全局配置文件，里面主要包含了一些常用设置，比如是否启用 https 等，`prepare` 是一个 python 写的预处理脚本，主要负责初始化一些 `harbor.cfg` 的相关配置，`docker-compose.yml` 顾名思义，里面顶一个各个组件的依赖关系以及配置挂载、数据持久化等设置。

2.1.3修改基础配置

```
## Configuration file of Harbor

#The IP address or hostname to access admin UI and registry service.
#DO NOT use localhost or 127.0.0.1, because Harbor needs to be accessed by external clients.【服务器域名】
hostname = 192.168.120.85

#The protocol for accessing the UI and token/notification service, by default it is http.【UI组件访问协议 http/https,默认为http,启用SSL需要配置ngix,下面会详细介绍】
#It can be set to https if ssl is enabled on nginx.
ui_url_protocol = https

#The password for the root user of mysql db, change this before any production use.【数据库密码】
db_password = haichuang

#Maximum number of job workers in job service  【最大任务数量】
max_job_workers = 3

#Determine whether or not to generate certificate for the registry's token.
#If the value is on, the prepare script creates new root cert and private key
#for generating token to access the registry. If the value is off the default key/cert will be used.
#This flag also controls the creation of the notary signer's cert.【是否生成自定义证书】
customize_crt = on

#The path of cert and key files for nginx, they are applied only the protocol is set to https 【供nginx使用的证书和key的路径,见下方】
ssl_cert =  /root/cert/hcregistry.crt
ssl_cert_key = /root/cert/hcregistry.key

#The path of secretkey storage 【密钥存储路径】
secretkey_path = /data

#Admiral's url, comment this attribute, or set its value to NA when Harbor is standalone 【集群环境下的主节点URL,我们是单机运行，所以设置为NA】
admiral_url = NA

#NOTES: The properties between BEGIN INITIAL PROPERTIES and END INITIAL PROPERTIES
#only take effect in the first boot, the subsequent changes of these properties
```

```
#should be performed on web ui 【以下是初始化的参数】

#************************BEGIN INITIAL PROPERTIES************************

#Email account settings for sending out password resetting emails.

#Email server uses the given username and password to authenticate on TLS connections to host and act as identity.【邮件相关信息配置，如忘记密码发送邮件,此处我们就不配置了】
#Identity left blank to act as username.
email_identity =

email_server = smtp.mydomain.com
email_server_port = 25
email_username = sample_admin@mydomain.com
email_password = abc
email_from = admin <sample_admin@mydomain.com>
email_ssl = false

##The initial password of Harbor admin, only works for the first time when Harbor starts. 【管理员密码，用户名默认为admin】
#It has no effect after the first launch of Harbor.
#Change the admin password from UI after launching Harbor.
harbor_admin_password = haichuang

##By default the auth mode is db_auth, i.e. the credentials are stored in a local database. 【权限验证方式 默认为db_auth，支持ldap验证，需要在下方配置相关的参数，此处我们使用默认的db_auth】
#Set it to ldap_auth if you want to verify a user's credentials against an LDAP server.
auth_mode = db_auth

#The url for an ldap endpoint. 【以下为ladp的相关参数，我们就不配置了，因为我不会】
ldap_url = ldaps://ldap.mydomain.com

#A user's DN who has the permission to search the LDAP/AD server.
#If your LDAP/AD server does not support anonymous search, you should configure this DN and ldap_search_pwd.
#ldap_searchdn = uid=searchuser,ou=people,dc=mydomain,dc=com

#the password of the ldap_searchdn
#ldap_search_pwd = password

#The base DN from which to look up a user in LDAP/AD
ldap_basedn = ou=people,dc=mydomain,dc=com

#Search filter for LDAP/AD, make sure the syntax of the filter is correct.
#ldap_filter = (objectClass=person)
```

```
# The attribute used in a search to match a user, it could be uid, cn, email, sAMAccountName or other attributes depending on your LDAP/AD
ldap_uid = uid

#the scope to search for users, 1-LDAP_SCOPE_BASE, 2-LDAP_SCOPE_ONELEVEL, 3-LDAP_SCOPE_SUBTREE
ldap_scope = 3

#Timeout (in seconds)  when connecting to an LDAP Server. The default value (and most reasonable) is 5 seconds.
ldap_timeout = 5

#Turn on or off the self-registration feature
self_registration = on

#The expiration time (in minute) of token created by token service, default is 30 minutes 【token过期时间，默认为30分钟】
token_expiration = 30

#The flag to control what users have permission to create projects
#The default value "everyone" allows everyone to creates a project.
#Set to "adminonly" so that only admin user can create project. 【项目创建权限，默认为everyone,我们改成只有管理员有权限创建。】
project_creation_restriction = adminonly

#Determine whether the job service should verify the ssl cert when it connects to a remote registry.
#Set this flag to off when the remote registry uses a self-signed or untrusted certificate. 【是否验证远程证书】
verify_remote_cert = on
#************************END INITIAL PROPERTIES************************#############
```

2.1.4 生成CA证书

1. 生成CA证书

```
➜ openssl req \
>     -newkey rsa:4096 -nodes -sha256 -keyout ca.key \
>     -x509 -days 365 -out ca.crt
```

此处需要配置相关的组织结构信息。请自行设置。

2. 生成证书签名请求

```
➜ openssl req \
>     -newkey rsa:4096 -nodes -sha256 -keyout yourdomain.com.key \
>     -out yourdomain.com.csr
```

yourdomain.com 为你的域名，如果是IP地址可以随意设置。此处我们使用的是IP地址，所以设置为**hcregistry**

3. 生成注册机构证书

```
➜ echo subjectAltName = IP:192.168.120.85 > extfile.cnf
➜ openssl x509 -req -days 365 -in hcregistry.csr -CA ca.crt -CAkey ca.key \
 -CAcreateserial -extfile extfile.cnf -out hcregistry.crt
```

4. Copy证书到指定路径

```
➜  cp hcregistry.crt /root/cert
➜  cp hcregistry.key /root/cert
```

5. 设置SSL的路径

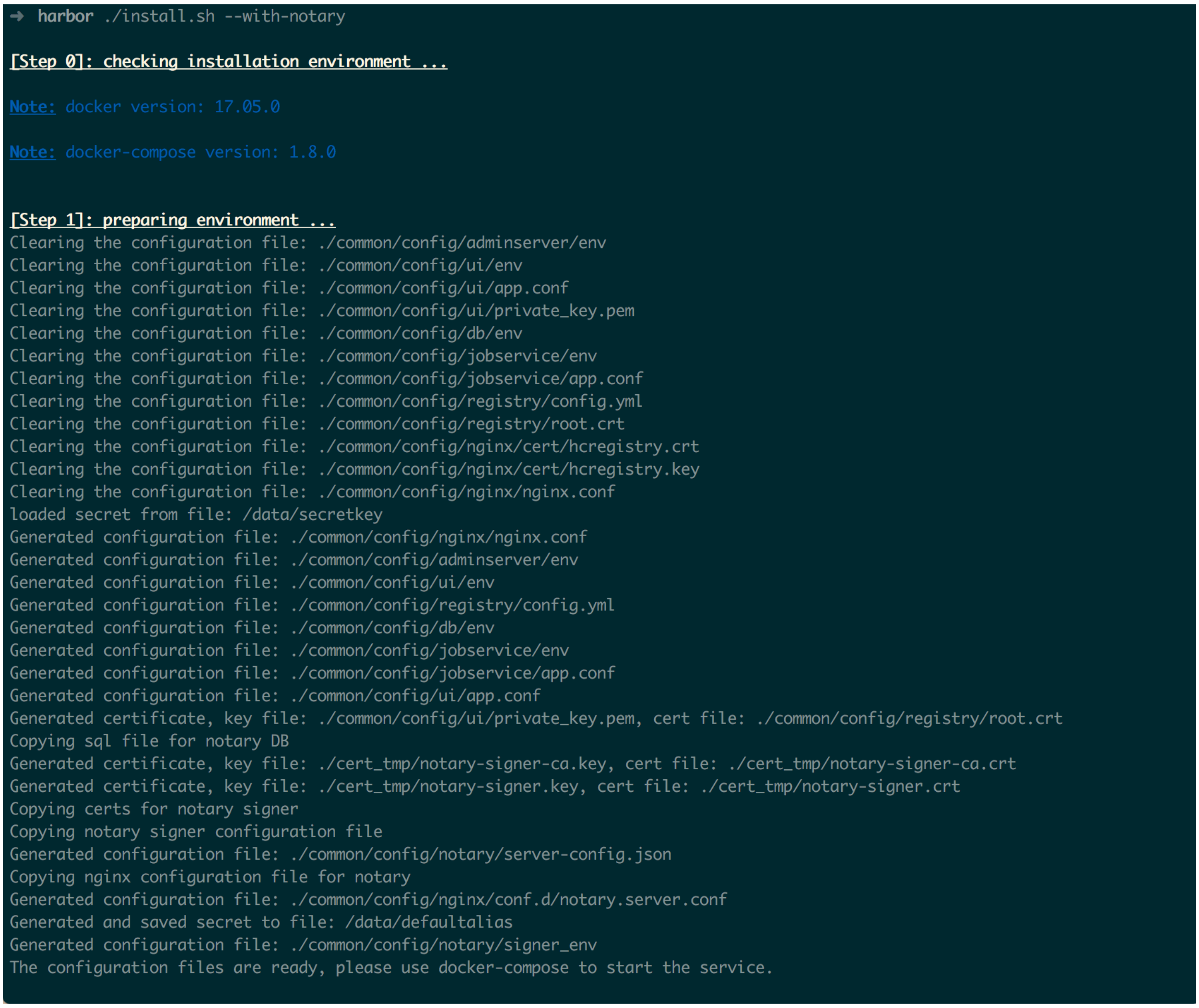
> 配置上方的**ssl_cert**和**ssl_cert_key**

2.1.5 生成相关的配置文件

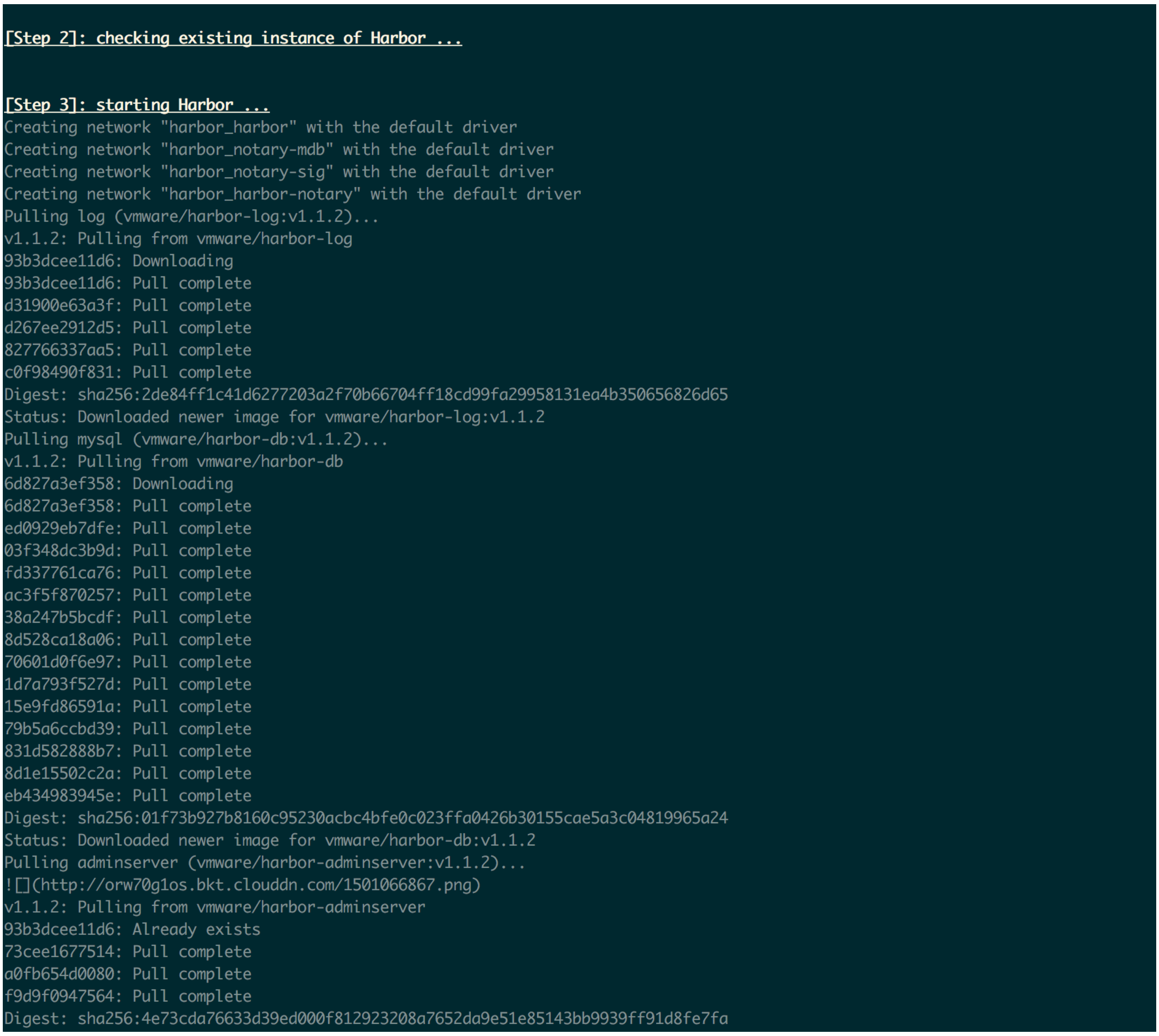
```
➜  ./prepare
```

2.2 安装

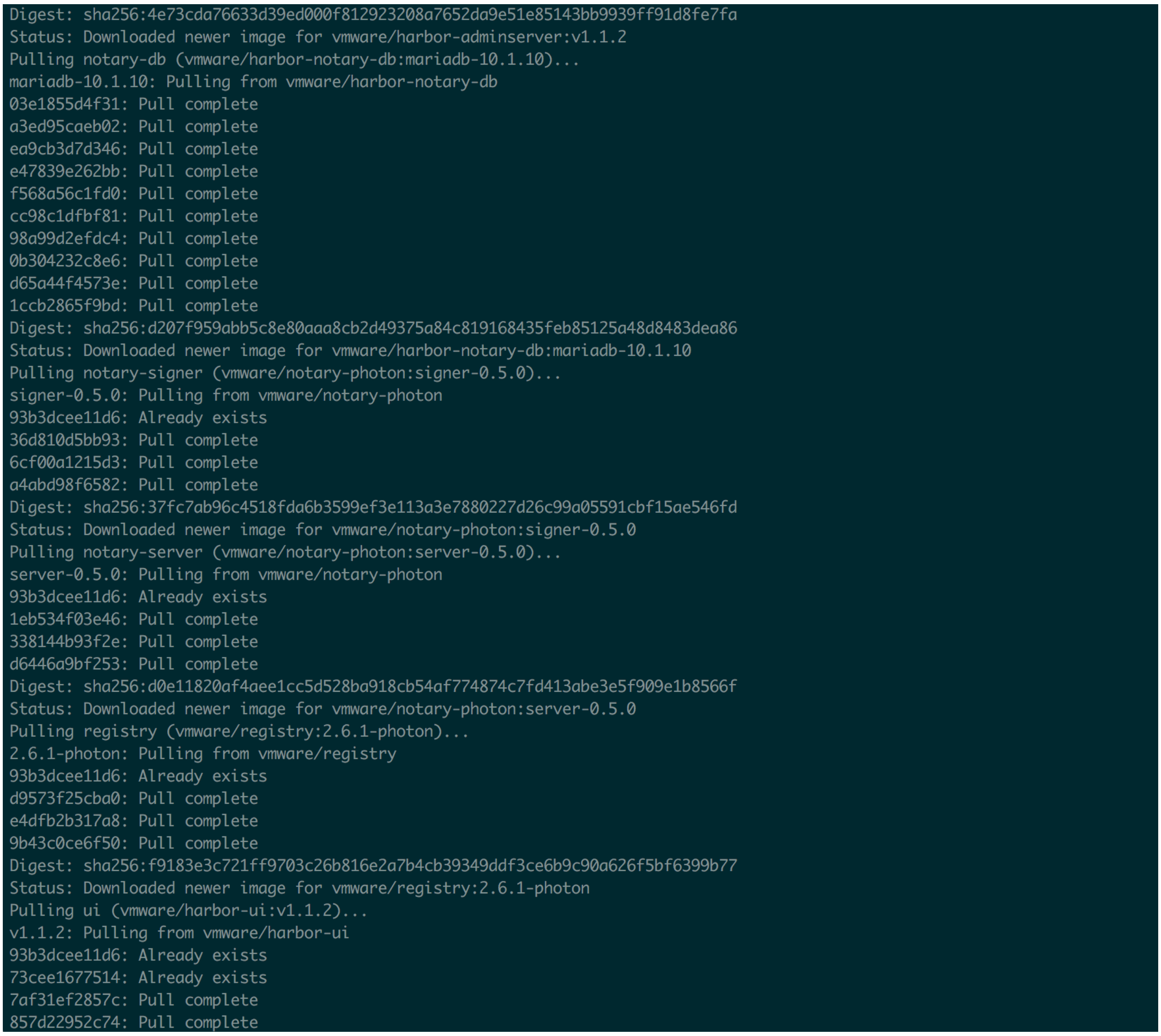
```
➜ ./install.sh --with-notary
```



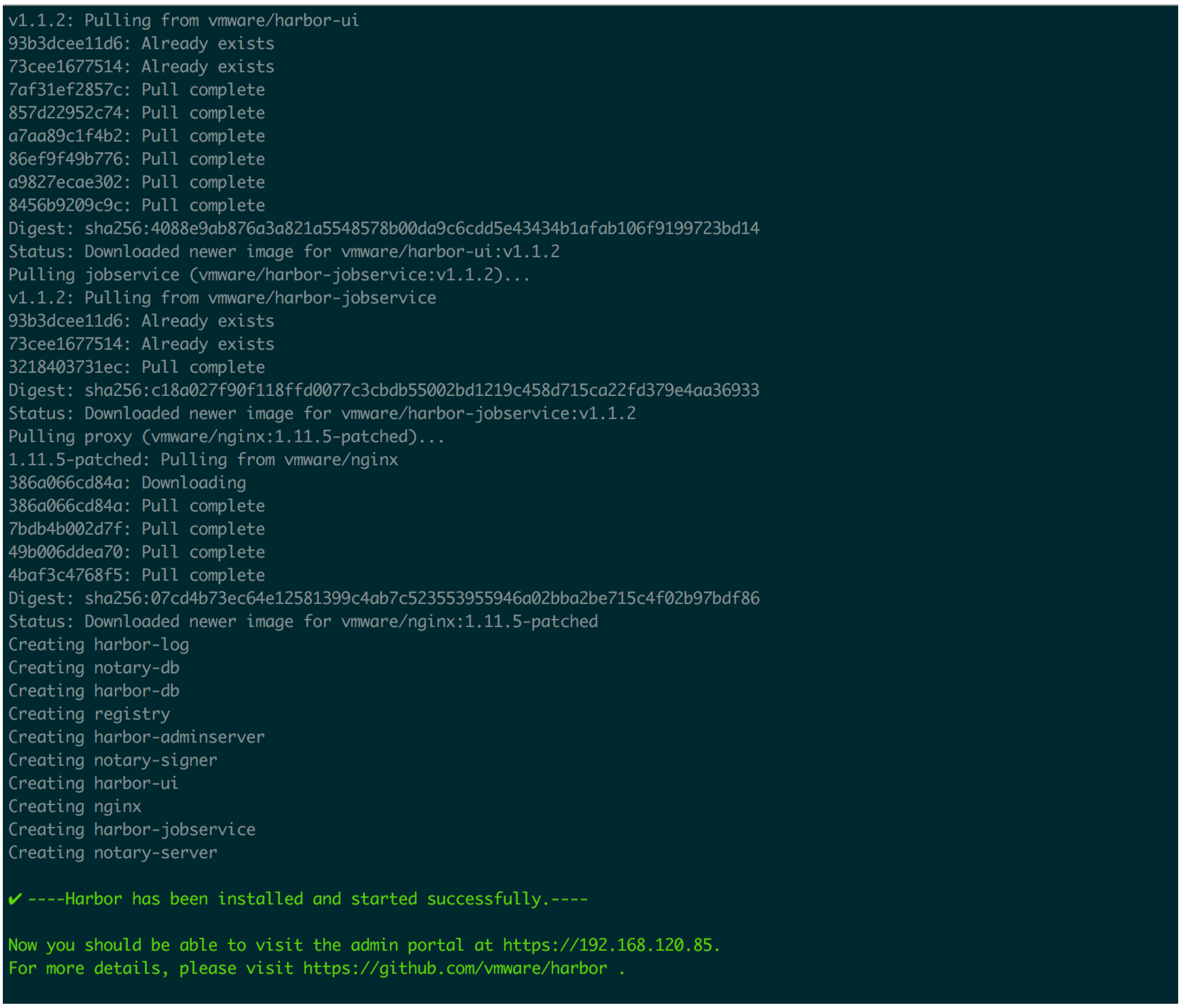
```
➜ harbor ./install.sh --with-notary

[Step 0]: checking installation environment ...

Note: docker version: 17.05.0

Note: docker-compose version: 1.8.0


[Step 1]: preparing environment ...
Clearing the configuration file: ./common/config/adminserver/env
Clearing the configuration file: ./common/config/ui/env
Clearing the configuration file: ./common/config/ui/app.conf
Clearing the configuration file: ./common/config/ui/private_key.pem
Clearing the configuration file: ./common/config/db/env
Clearing the configuration file: ./common/config/jobservice/env
Clearing the configuration file: ./common/config/jobservice/app.conf
Clearing the configuration file: ./common/config/registry/config.yml
Clearing the configuration file: ./common/config/registry/root.crt
Clearing the configuration file: ./common/config/nginx/cert/hcregistry.crt
Clearing the configuration file: ./common/config/nginx/cert/hcregistry.key
Clearing the configuration file: ./common/config/nginx/nginx.conf
loaded secret from file: /data/secretkey
Generated configuration file: ./common/config/nginx/nginx.conf
Generated configuration file: ./common/config/adminserver/env
Generated configuration file: ./common/config/ui/env
Generated configuration file: ./common/config/registry/config.yml
Generated configuration file: ./common/config/db/env
Generated configuration file: ./common/config/jobservice/env
Generated configuration file: ./common/config/jobservice/app.conf
Generated configuration file: ./common/config/ui/app.conf
Generated certificate, key file: ./common/config/ui/private_key.pem, cert file: ./common/config/registry/root.crt
Copying sql file for notary DB
Generated certificate, key file: ./cert_tmp/notary-signer-ca.key, cert file: ./cert_tmp/notary-signer-ca.crt
Generated certificate, key file: ./cert_tmp/notary-signer.key, cert file: ./cert_tmp/notary-signer.crt
Copying certs for notary signer
Copying notary signer configuration file
Generated configuration file: ./common/config/notary/server-config.json
Copying nginx configuration file for notary
Generated configuration file: ./common/config/nginx/conf.d/notary.server.conf
Generated and saved secret to file: /data/defaultalias
Generated configuration file: ./common/config/notary/signer_env
The configuration files are ready, please use docker-compose to start the service.
```



```
[Step 2]: checking existing instance of Harbor ...


[Step 3]: starting Harbor ...
Creating network "harbor_harbor" with the default driver
Creating network "harbor_notary-mdb" with the default driver
Creating network "harbor_notary-sig" with the default driver
Creating network "harbor_harbor-notary" with the default driver
Pulling log (vmware/harbor-log:v1.1.2)...
v1.1.2: Pulling from vmware/harbor-log
93b3dcee11d6: Downloading
93b3dcee11d6: Pull complete
d31900e63a3f: Pull complete
d267ee2912d5: Pull complete
827766337aa5: Pull complete
c0f98490f831: Pull complete
Digest: sha256:2de84ff1c41d6277203a2f70b66704ff18cd99fa29958131ea4b350656826d65
Status: Downloaded newer image for vmware/harbor-log:v1.1.2
Pulling mysql (vmware/harbor-db:v1.1.2)...
v1.1.2: Pulling from vmware/harbor-db
6d827a3ef358: Downloading
6d827a3ef358: Pull complete
ed0929eb7dfe: Pull complete
03f348dc3b9d: Pull complete
fd337761ca76: Pull complete
ac3f5f870257: Pull complete
38a247b5bcdf: Pull complete
8d528ca18a06: Pull complete
70601d0f6e97: Pull complete
1d7a793f527d: Pull complete
15e9fd86591a: Pull complete
79b5a6ccbd39: Pull complete
831d582888b7: Pull complete
8d1e15502c2a: Pull complete
eb434983945e: Pull complete
Digest: sha256:01f73b927b8160c95230acbc4bfe0c023ffa0426b30155cae5a3c04819965a24
Status: Downloaded newer image for vmware/harbor-db:v1.1.2
Pulling adminserver (vmware/harbor-adminserver:v1.1.2)...
![](http://orw70g1os.bkt.clouddn.com/1501066867.png)
v1.1.2: Pulling from vmware/harbor-adminserver
93b3dcee11d6: Already exists
73cee1677514: Pull complete
a0fb654d0080: Pull complete
f9d9f0947564: Pull complete
Digest: sha256:4e73cda76633d39ed000f812923208a7652da9e51e85143bb9939ff91d8fe7fa
```

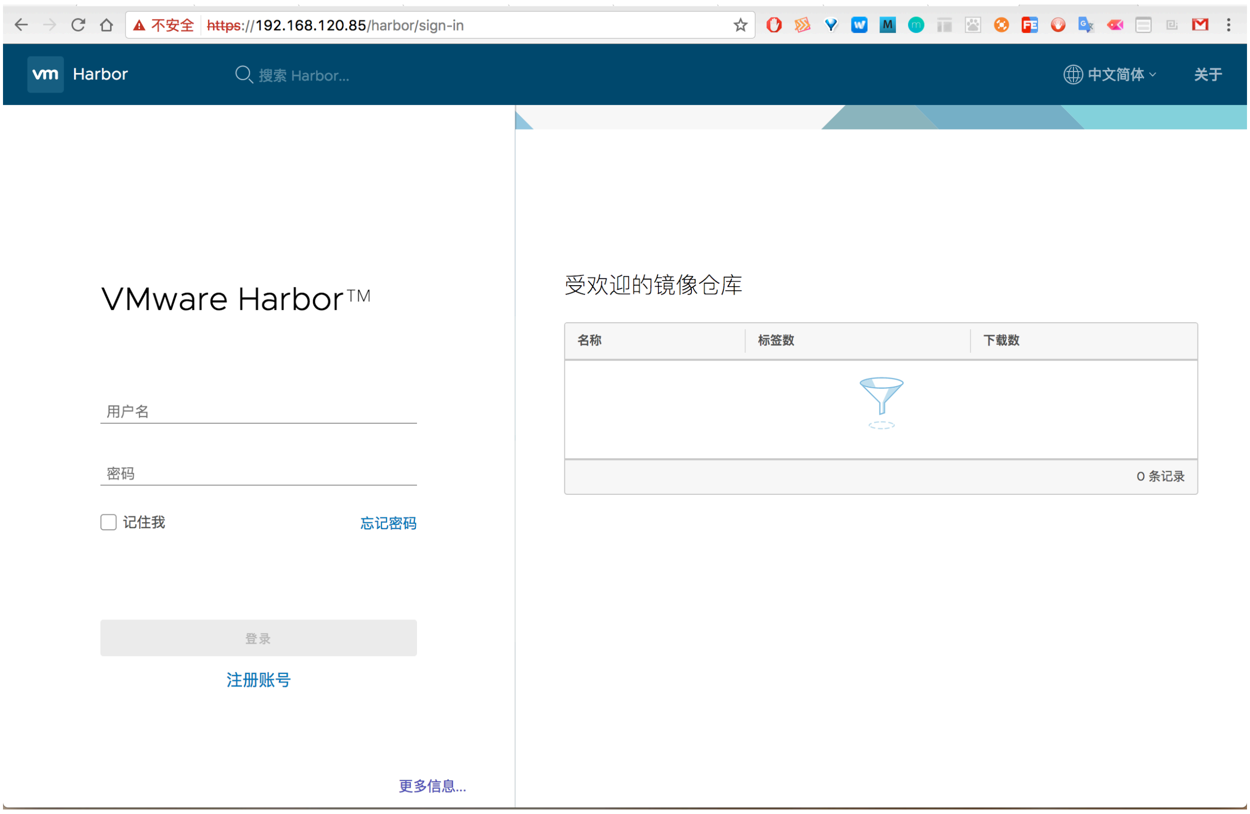


```
Digest: sha256:4e73cda76633d39ed000f812923208a7652da9e51e85143bb9939ff91d8fe7fa
Status: Downloaded newer image for vmware/harbor-adminserver:v1.1.2
Pulling notary-db (vmware/harbor-notary-db:mariadb-10.1.10)...
mariadb-10.1.10: Pulling from vmware/harbor-notary-db
03e1855d4f31: Pull complete
a3ed95caeb02: Pull complete
ea9cb3d7d346: Pull complete
e47839e262bb: Pull complete
f568a56c1fd0: Pull complete
cc98c1dfbf81: Pull complete
98a99d2efdc4: Pull complete
0b304232c8e6: Pull complete
d65a44f4573e: Pull complete
1ccb2865f9bd: Pull complete
Digest: sha256:d207f959abb5c8e80aaa8cb2d49375a84c819168435feb85125a48d8483dea86
Status: Downloaded newer image for vmware/harbor-notary-db:mariadb-10.1.10
Pulling notary-signer (vmware/notary-photon:signer-0.5.0)...
signer-0.5.0: Pulling from vmware/notary-photon
93b3dcee11d6: Already exists
36d810d5bb93: Pull complete
6cf00a1215d3: Pull complete
a4abd98f6582: Pull complete
Digest: sha256:37fc7ab96c4518fda6b3599ef3e113a3e7880227d26c99a05591cbf15ae546fd
Status: Downloaded newer image for vmware/notary-photon:signer-0.5.0
Pulling notary-server (vmware/notary-photon:server-0.5.0)...
server-0.5.0: Pulling from vmware/notary-photon
93b3dcee11d6: Already exists
1eb534f03e46: Pull complete
338144b93f2e: Pull complete
d6446a9bf253: Pull complete
Digest: sha256:d0e11820af4aee1cc5d528ba918cb54af774874c7fd413abe3e5f909e1b8566f
Status: Downloaded newer image for vmware/notary-photon:server-0.5.0
Pulling registry (vmware/registry:2.6.1-photon)...
2.6.1-photon: Pulling from vmware/registry
93b3dcee11d6: Already exists
d9573f25cba0: Pull complete
e4dfb2b317a8: Pull complete
9b43c0ce6f50: Pull complete
Digest: sha256:f9183e3c721ff9703c26b816e2a7b4cb39349ddf3ce6b9c90a626f5bf6399b77
Status: Downloaded newer image for vmware/registry:2.6.1-photon
Pulling ui (vmware/harbor-ui:v1.1.2)...
v1.1.2: Pulling from vmware/harbor-ui
93b3dcee11d6: Already exists
73cee1677514: Already exists
7af31ef2857c: Pull complete
857d22952c74: Pull complete
```



```
v1.1.2: Pulling from vmware/harbor-ui
93b3dcee11d6: Already exists
73cee1677514: Already exists
7af31ef2857c: Pull complete
857d22952c74: Pull complete
a7aa89c1f4b2: Pull complete
86ef9f49b776: Pull complete
a9827ecae302: Pull complete
8456b9209c9c: Pull complete
Digest: sha256:4088e9ab876a3a821a5548578b00da9c6cdd5e43434b1afab106f9199723bd14
Status: Downloaded newer image for vmware/harbor-ui:v1.1.2
Pulling jobservice (vmware/harbor-jobservice:v1.1.2)...
v1.1.2: Pulling from vmware/harbor-jobservice
93b3dcee11d6: Already exists
73cee1677514: Already exists
3218403731ec: Pull complete
Digest: sha256:c18a027f90f118ffd0077c3cbdb55002bd1219c458d715ca22fd379e4aa36933
Status: Downloaded newer image for vmware/harbor-jobservice:v1.1.2
Pulling proxy (vmware/nginx:1.11.5-patched)...
1.11.5-patched: Pulling from vmware/nginx
386a066cd84a: Downloading
386a066cd84a: Pull complete
7bdb4b002d7f: Pull complete
49b006ddea70: Pull complete
4baf3c4768f5: Pull complete
Digest: sha256:07cd4b73ec64e12581399c4ab7c523553955946a02bba2be715c4f02b97bdf86
Status: Downloaded newer image for vmware/nginx:1.11.5-patched
Creating harbor-log
Creating notary-db
Creating harbor-db
Creating registry
Creating harbor-adminserver
Creating notary-signer
Creating harbor-ui
Creating nginx
Creating harbor-jobservice
Creating notary-server

✔ ----Harbor has been installed and started successfully.----

Now you should be able to visit the admin portal at https://192.168.120.85.
For more details, please visit https://github.com/vmware/harbor .
```

我们可以访问一下给出的URL：https://192.168.120.85,由于我们本地没有安装相关的证书，所以链接仍然显示为不安全的。



VMware Harbor™

用户名

密码

□ 记住我　　忘记密码

登录

注册账号

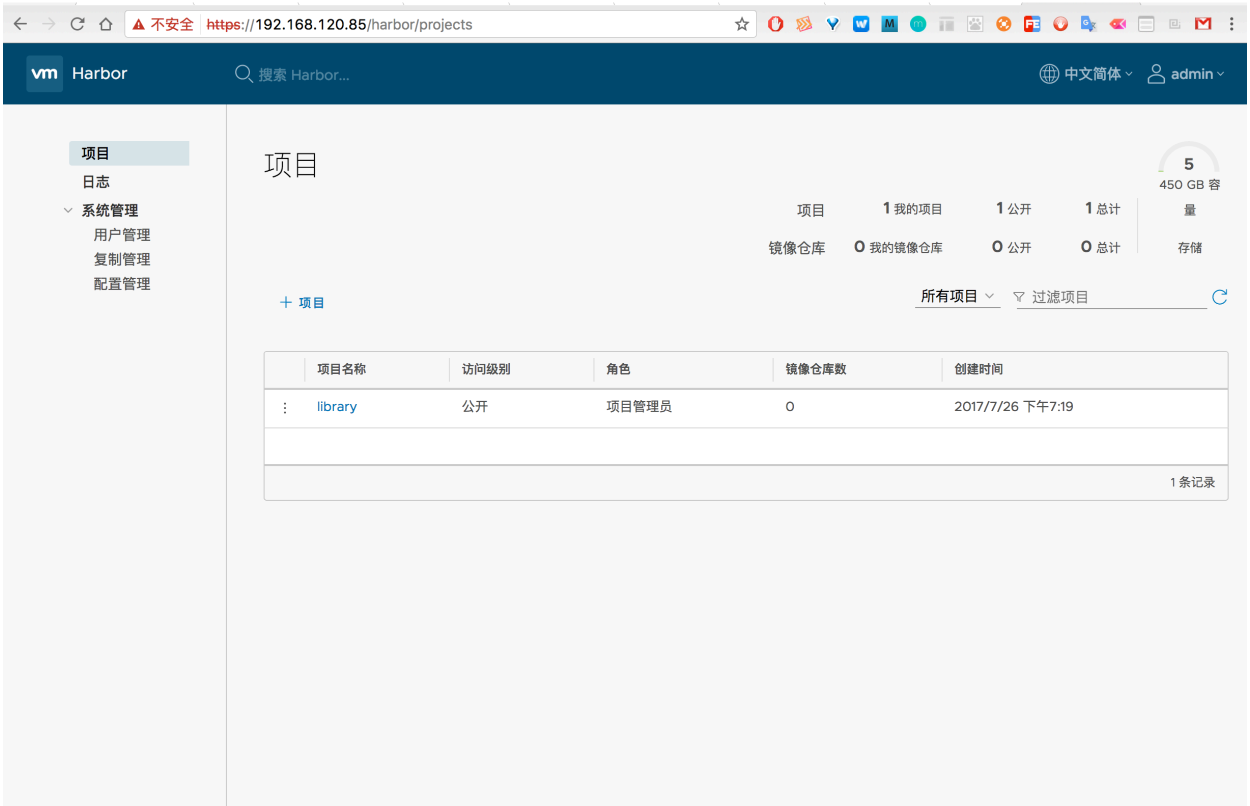
更多信息...

受欢迎的镜像仓库

名称	标签数	下载数

0 条记录

这是登陆后的界面



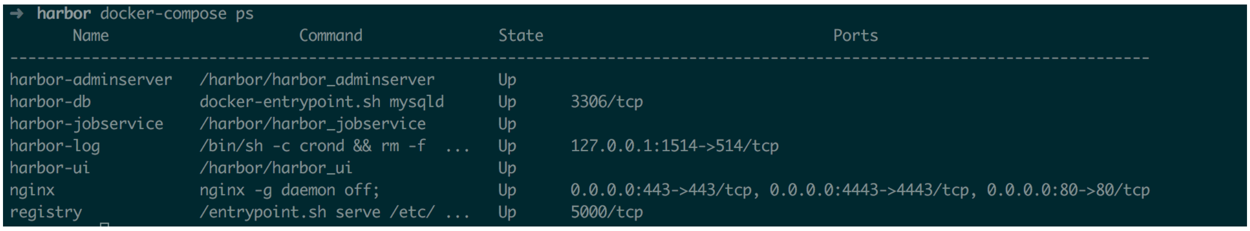
项目

项目	1 我的项目	1 公开	1 总计	5 450 GB 容量
镜像仓库	0 我的镜像仓库	0 公开	0 总计	存储

	项目名称	访问级别	角色	镜像仓库数	创建时间
⋮	library	公开	项目管理员	0	2017/7/26 下午7:19

1 条记录

我们使用`docker-compose ps`命令可以查看一下harbor启动了哪些容器。



```
➜ harbor docker-compose ps
        Name                     Command               State                   Ports
-------------------------------------------------------------------------------------------------------------------------------
harbor-adminserver   /harbor/harbor_adminserver       Up
harbor-db            docker-entrypoint.sh mysqld      Up      3306/tcp
harbor-jobservice    /harbor/harbor_jobservice        Up
harbor-log           /bin/sh -c crond && rm -f  ...   Up      127.0.0.1:1514->514/tcp
harbor-ui            /harbor/harbor_ui                Up
nginx                nginx -g daemon off;             Up      0.0.0.0:443->443/tcp, 0.0.0.0:4443->4443/tcp, 0.0.0.0:80->80/tcp
registry             /entrypoint.sh serve /etc/ ...   Up      5000/tcp
```

因为harbor是使用docker-compose进行编排的，所以关闭，重启等都可以使用docker-compose的命令进行操作。具体命令参见docker-compose章节

> Tip:推荐使用离线安装包，在线安装需要去墙外下载大量的镜像，整个过程极其缓慢。

到此**Harbor**的相关配置就结束了。

2.3 登陆

首先我们登陆到私有仓库

```
➜  tomcat8.0-jre8 docker login 192.168.120.85
Username: admin
Password:
Error response from daemon: Get https://192.168.120.85/v1/users/: x509: certificate
signed by unknown authority
```

输入账号和密码后，提示未知的证书签发机构，我们需要将证书导入。

```
➜  registry cp ca.crt /etc/docker/certs.d/192.168.120.85/ca.crt
```

`192.168.120.85`:domain:port

文件夹不存在请自行创建，然后我们执行登陆

```
➜  registry docker login 192.168.120.85
Username: admin
Password:
Login Succeeded
```

输入账号密码后提示成功。

2.4 push镜像

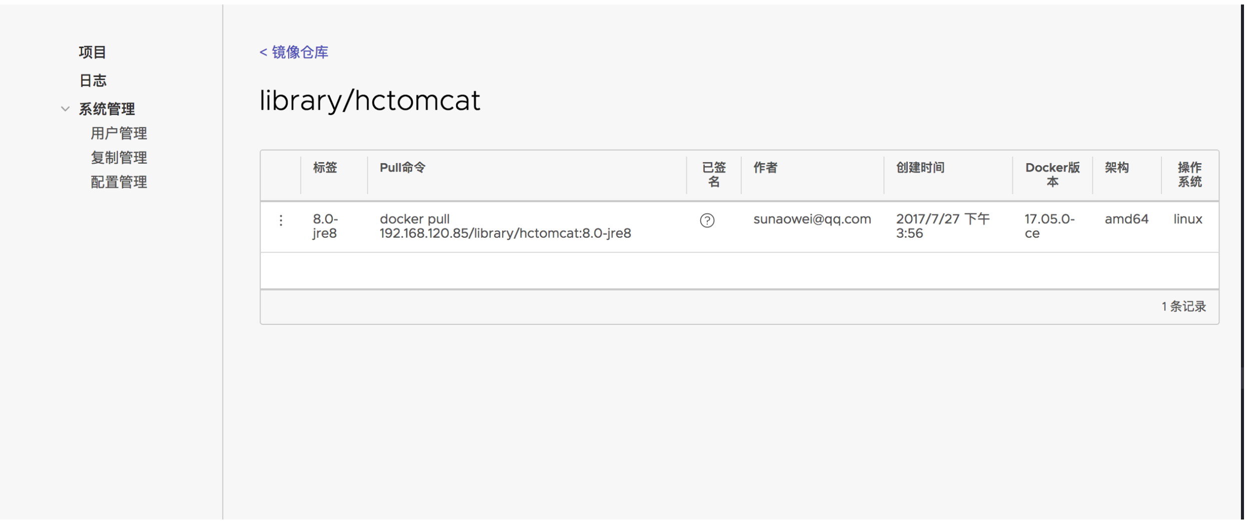
push镜像之前，我们需要先给镜像打一个tag

```
➜  registry docker tag hctomcat:8.0-jre8 192.168.120.85/library/hctomcat:8.0-jre8
➜  registry docker push 192.168.120.85/library/hctomcat:8.0-jre8
The push refers to a repository [192.168.120.85/library/hctomcat]
1af1a4ed7ea8: Pushed
049fa24a600c: Pushed
887b58b2ccb0: Pushed
c1ac78de2350: Pushed
26b126eb8632: Pushed
220d34b5f6c9: Pushed
8a5132998025: Pushed
aca233ed29c3: Pushed
e5d2f035d7a4: Pushed
8.0-jre8: digest:
sha256:dcfddd42443f2b0bc273425034a103d68ddcda6e7e81918bb83bb1381207d928 size: 2195
```

> Tip：tag的要求是 `docker tag imagename {docker-hub-domain}/{default-repo-folder-name}/imagename`

上面两条命令的意思就是将`hctomcat:8.0-jre8`这个镜像push到`192.168.120.85`这个domain下的`library`这个项目。

推送成功后，我们可以在harbor的项目管理中看到该镜像的信息



library/hctomcat

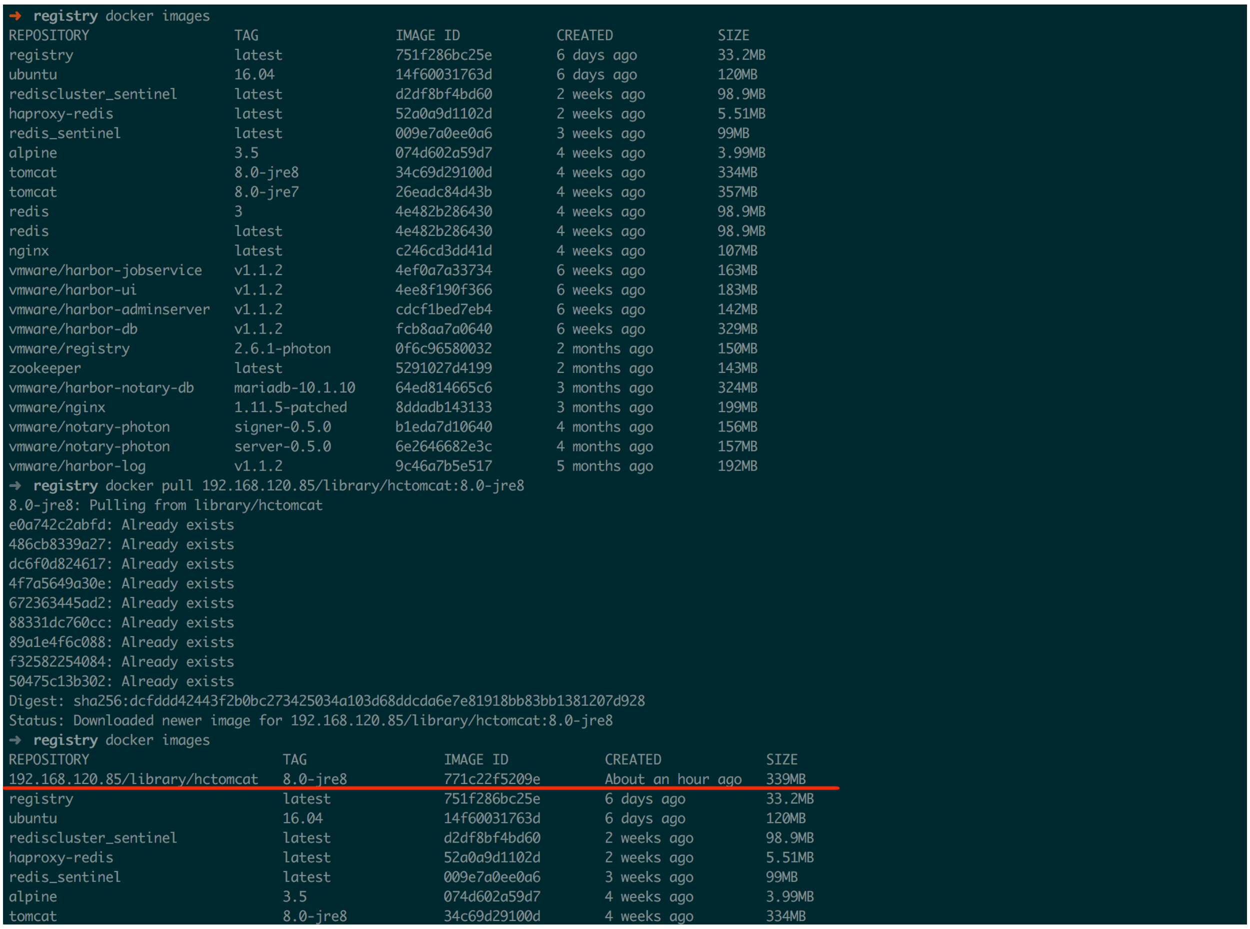
	标签	Pull命令	已签名	作者	创建时间	Docker版本	架构	操作系统
⋮	8.0-jre8	docker pull 192.168.120.85/library/hctomcat:8.0-jre8	?	sunaowei@qq.com	2017/7/27 下午3:56	17.05.0-ce	amd64	linux

1 条记录

2.5 pull镜像

我们把刚才的那个镜像从私有仓库中pull下来。

```
➜  registry docker pull 192.168.120.85/library/hctomcat:8.0-jre8
```



```
➜ registry docker images
REPOSITORY                    TAG                 IMAGE ID            CREATED             SIZE
registry                      latest              751f286bc25e        6 days ago          33.2MB
ubuntu                        16.04               14f60031763d        6 days ago          120MB
rediscluster_sentinel         latest              d2df8bf4bd60        2 weeks ago         98.9MB
haproxy-redis                 latest              52a0a9d1102d        2 weeks ago         5.51MB
redis_sentinel                latest              009e7a0ee0a6        3 weeks ago         99MB
alpine                        3.5                 074d602a59d7        4 weeks ago         3.99MB
tomcat                        8.0-jre8            34c69d29100d        4 weeks ago         334MB
tomcat                        8.0-jre7            26eadc84d43b        4 weeks ago         357MB
redis                         3                   4e482b286430        4 weeks ago         98.9MB
redis                         latest              4e482b286430        4 weeks ago         98.9MB
nginx                         latest              c246cd3dd41d        4 weeks ago         107MB
vmware/harbor-jobservice      v1.1.2              4ef0a7a33734        6 weeks ago         163MB
vmware/harbor-ui              v1.1.2              4ee8f190f366        6 weeks ago         183MB
vmware/harbor-adminserver     v1.1.2              cdcf1bed7eb4        6 weeks ago         142MB
vmware/harbor-db              v1.1.2              fcb8aa7a0640        6 weeks ago         329MB
vmware/registry               2.6.1-photon        0f6c96580032        2 months ago        150MB
zookeeper                     latest              5291027d4199        2 months ago        143MB
vmware/harbor-notary-db       mariadb-10.1.10     64ed814665c6        3 months ago        324MB
vmware/nginx                  1.11.5-patched      8ddadb143133        3 months ago        199MB
vmware/notary-photon          signer-0.5.0        b1eda7d10640        4 months ago        156MB
vmware/notary-photon          server-0.5.0        6e2646682e3c        4 months ago        157MB
vmware/harbor-log             v1.1.2              9c46a7b5e517        5 months ago        192MB
➜ registry docker pull 192.168.120.85/library/hctomcat:8.0-jre8
8.0-jre8: Pulling from library/hctomcat
e0a742c2abfd: Already exists
486cb8339a27: Already exists
dc6f0d824617: Already exists
4f7a5649a30e: Already exists
672363445ad2: Already exists
88331dc760cc: Already exists
89a1e4f6c088: Already exists
f32582254084: Already exists
50475c13b302: Already exists
Digest: sha256:dcfddd42443f2b0bc273425034a103d68ddcda6e7e81918bb83bb1381207d928
Status: Downloaded newer image for 192.168.120.85/library/hctomcat:8.0-jre8
➜ registry docker images
REPOSITORY                          TAG                 IMAGE ID            CREATED              SIZE
192.168.120.85/library/hctomcat     8.0-jre8            771c22f5209e        About an hour ago    339MB
registry                            latest              751f286bc25e        6 days ago           33.2MB
ubuntu                              16.04               14f60031763d        6 days ago           120MB
rediscluster_sentinel               latest              d2df8bf4bd60        2 weeks ago          98.9MB
haproxy-redis                       latest              52a0a9d1102d        2 weeks ago          5.51MB
redis_sentinel                      latest              009e7a0ee0a6        3 weeks ago          99MB
alpine                              3.5                 074d602a59d7        4 weeks ago          3.99MB
tomcat                              8.0-jre8            34c69d29100d        4 weeks ago          334MB
```

我们可以看到pull之前和之后本地镜像的对比，很明显，我们成功的从私有仓库中pull下来了我们所需要的镜像。

Q：为什么要谁用https进行配置，可以不使用https么？

A：https是未来的主流，相对于http更加安全，Docker默认http的连接是不安全的，如果需要访问http的仓库需要修改docker中的相关配置。步骤如下：

> 修改/lib/systemd/system/docker.service文件，添加--insecure-registry 你的IP，重启docker daemon 和service。
>
> (命令：systemctl daemon-reload 和 systemctl restart docker.service)。
>
> ```
> ExecStart=/usr/bin/docker daemon -H fd:// --insecure-registry 你的IP
> ```
>
> 其中 IP 地址要指向 `harbor.cfg` 中的 `hostname` ，然后执行 `docker-compose stop` 停掉所有 Contianer，再执行 `service docker restart` 重启 Dokcer 服务，最后执行 `docker-compose start` 即可。
>
> 注意 : Docker 服务重启后，执行 `docker-compose start` 时有一定几率出现如下错误(或者目录已存在等错误)，此时在 `docker-compose stop` 一下然后在启动即可，实在不行再次重启 Dokcer 服务，千万不要手贱的去删文件(别问我怎么知道的)