Contents

What is Docker Registry

داکر registry در حقیقت انباری(repository) از image های مختلف با ورژن های مختلف میباشد که این امکان را به کاربران میدهد تا با دسترسی به این انبار از image های مد نظر خود استفاده کرده و container خود را run کنند.

در حال حاضر DockerHub یکی از قوی ترین repository های موجود برای کاربران docker میباشد که میتوان به راحتی image هایی را که میخواهیم را از انجا بگیریم.حتی این امکان وجود دارد که با ساخت اکانت registry خود را در این repository راه بیندازیم.اما بیشتر سازمان ها و شرکت ها ترجیح میدهند بر روی registry خود کنترل قوی تری داشته باشند تا مدیریت تمامی image های خود را در دست بگیرند در اینجا بحث docker private registry مطرح میشود.شاید مهمترین و اصلی ترین مزیت private registry به public registry در این باشد که مدیریت Image ها کاملا به دست سازمان/شرکت میباشد.

Run a local registry

در حقیقت registry یک نمونه از registry image میباشد که به صورت container راه اندازی میشه

```
$ docker run -d -p 5000:5000 --restart=always --name registry_rayan registry:
2
```

حال برای تست registry_rayan یک image alpine رو از داکر hub گرفته و بعد از tag زدن آن را با pushکردن در رجیستری مورد نظر میریزیم:

```
$ docker pull alpine

$ docker tag alpine localhost:5000/alpine -> hostname:port/imagename:version

$ docker push localhost:5000/alpine
```

بعد از زدن دستور push با خروجی زیر مواجه میشویم :

```
[root@docker-mgr ~]# docker push localhost:5000/alpine
The push refers to repository [localhost:5000/alpine]
f1b5933fe4b5: Pushed
latest: digest: sha256:bf1684a6e3676389ec861c602e97f27b03f14178e5bc3f70dce198f9f160cce9 size: 528
```

با دستور docker images و لیست کردن همه ی image های موجود خروجی زیر را مشاهده میکنیم:

```
[root@docker-mgr ~]# docker images
REPOSITORY              TAG                 IMAGE ID            CREATED             SIZE
l_restapi               latest              eaaaa1830971        4 days ago          105MB
test                    latest              eaaaa1830971        4 days ago          105MB
restapi_ver1            latest              39168e6b517a        4 days ago          105MB
<none>                  <none>              871f46f2727a        4 days ago          5.53MB
<none>                  <none>              a8c399f70332        5 days ago          5.53MB
alpine                  latest              055936d39205        8 days ago          5.53MB
localhost:5000/alpine   latest              055936d39205        8 days ago          5.53MB
alpine                  <none>              cdf98d1859c1        5 weeks ago         5.53MB
webserver               ver1                32833a77dee6        5 weeks ago         157MB
friendlyhello           latest              9bb580aaa62d        5 weeks ago         131MB
python                  2.7-slim            48e3247f2a19        7 weeks ago         120MB
nginx                   latest              2bcb04bdb83f        7 weeks ago         109MB
centos                  latest              9f38484d220f        2 months ago        202MB
centos                  ver1                9f38484d220f        2 months ago        202MB
registry                2                   f32a97de94e1        2 months ago        25.8MB
tomcat                  latest              dd6ff929584a        2 months ago        463MB
tomcat                  8.0                 ef6a7c98d192        8 months ago        356MB
```

اما سوال اینجاست این image در کدام مسیر ریخته شده است..برای درک بهتر به container registry_rayan از طریق

/ bin/sh وصل میشویم و دستور زیر را اجرا میکنیم:

```
[root@docker-mgr ~]# docker exec -it d5 /bin/sh
/ # ls /var/lib/registry/docker/registry/v2/repositories/alpine/
_layers     _manifests  _uploads
/ #
```

همانطور که مشاهده میکنیم imageو لایه های مختلف آن در این مسیر ریخته شده است البته در سطح container. برای اینکه متوجه شویم در سطح hostدر چه مسیری ریخته شده است از دستور inspectاستفاده میکنیم:

```
[root@docker-mgr ~]# docker inspect d5 | grep -i mounts -A10
        "Mounts": [
            {
                "Type": "volume",
                "Name": "d48e631de65b8896d34587d0ae0827159d2ac927747a183668bfe03fb5e553c7",
                "Source": "/var/lib/docker/volumes/d48e631de65b8896d34587d0ae0827159d2ac927747a183668bfe03fb5e553c7/_data",
                "Destination": "/var/lib/registry",
                "Driver": "local",
                "Mode": "",
                "RW": true,
                "Propagation": ""
            }
[root@docker-mgr ~]#
```

همانطور که مشاهده میکنید مسیر مربوط به source/destination کاملا مشخص میباشد..میتوان در صورت نیاز مسیر های مورد نیاز را بر روی مسیر دیگری bind mountکرد.مثلا فرض کنید مسیری به اسم /mnt/registryدارین که برروی هارد SSD ساخته شده است و در نظر دارین از این مسیر برای نگهداری imageها استفاده کنید بدین منظور در زمان دستور docker run باید از پارامتر v /mnt/registry:/var/lib/registry– استفاده کرد.

Run an externally-accessible registry

اما یک docker registry برای محیط عملیاتی باید به وسیله ی پروتوکل TLS کاملا حفاظت بشه و مکانیزم های دسترسی به docker registry کاملا رعایت بشه.باید در نظر داشت در محیط عملیاتی که از چند node استفاده میشود همه ی این nodeها باید به این registryدسترسی داشته .برای ایجاد بستری که همه ی node های داکر به private registry دسترسی پیدا کنند باید از TLS استفاده کرد .در ادامه به چگونگی کانفیگ کردن docker registry در محیط عملیاتی میپردازیم.

STEP 1 -> Generate a self-signed certificate

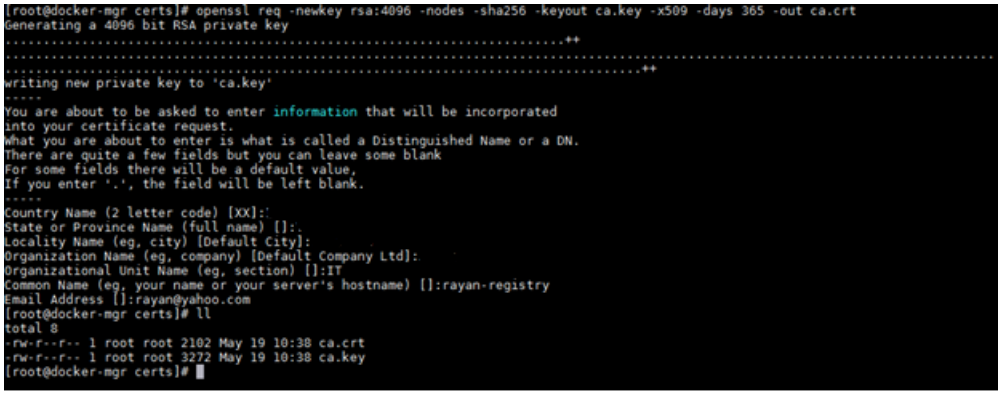
از آنجایی که ارتباط همه ی nodeهای مربوط به docker با registry برای upload/download کردن image باید روی بستر امن TLS باشد برای همین باید با openssl یک self-signed certificate درست کنیم و certificate رو به node های مربوط بدهیم.

```
[root@docker-mgr ~]# mkdir /etc/certs
```

```
[root@docker-mgr ~]# cd /etc/certs
```

```
[root@docker-mgr certs]# openssl req -newkey rsa:4096 -nodes -sha256 -keyout ca.key -x509 -days 365
-out ca.crt
```

بعد از اجرای دستور فوق سوالاتی از شما پرسیده میشه.به عکس زیر توجه کنید:



STEP 2 -> Run registry container on port 443

```
docker run -d --restart=always --name rayan_registry -v /etc/certs:/certs -e
REGISTRY_HTTP_ADDR=0.0.0.0:443 -e REGISTRY_HTTP_TLS_CERTIFICATE=/certs/ca.crt
-e REGISTRY_HTTP_TLS_KEY=/certs/ca.key -p 443:443 registry:2
```

دستور ساخت Container در این مرحله با local registry بسیار متفاوت است.همانطور که مشاهده میکنید دو environment variable در دستور داریم که وظیفه ی معرفی certificate , certificate key رو به کانتینر دارن. شایان ذکر است از bind

mount هم برای دیده شدن دایرکتوری /etc/certs استفاده میکنیم تا self-signed certificate را به کانتینر معرفی کنیم.حال با registry server یک image را تگ میزنیم و بعد به داخل رجیستری push میکنیم:

```
root@docker-mgr certs# docker tag tomcat docker-mgr:443/tomcat:rayan_v1

root@docker-mgr certs# docker push docker-mgr:443/tomcat:rayan_v1
```

`

```
The push refers to repository [docker-mgr:443/tomcat]
Get https://docker-mgr:443/v2/: x509: certificate signed by unknown authority
```

خوب همانطور که در عکس بالا مشاهده میکنید همچنان certificate ساخته شده ما قابل قبول نیست..اما چرا؟به صورت default در داکر TLSفعال نیست و ارتباطات بین docker daemon & client در بستر امن HTTPS نمی باشد.کانتینری که ما در چند خط بالا run کردیم اولین قدم در راستای مهیا کردن بستر امن HTTPSبین docker daemon & client هست در حقیقت در دستور run ما فقط فایل های مربوط به certificateرا فقط به container معرفی کردیم تا client ما با دستوراتی که میزند از آن استفاده کند سوال اینجاست آیا docker daemon هم باید از certificate ساخته شده خبر داشته باشد؟بله حتما!

STEP 3 -> Introduce Certificate to Docker Daemon

برای معرفی کردن custom certificate به docker daemon باید در مسیر زیر تمامی node ها رفته و فولدری به اسم registry hostname سا خته و فایل ca.crt را در آن قرار داد و در اخر docker را restart کرد:

```
mkdir -p /etc/docker/certs.d/docker-mgr:443/

cp /etc/certs/ca.crt /etc/docker/certs.d/docker-mgr:443/

systemctl restart docker
```

```
root@docker-mgr certs# docker push docker-mgr:443/tomcat:rayan_v1
```

```
[root@docker-mgr certs]# docker push docker-mgr:443/tomcat:rayan_v1
The push refers to repository [docker-mgr:443/tomcat]
445c9e4ffc46: Pushed
7c4a50087c15: Pushed
a00a5a5b6d7c: Pushed
40cde936fa58: Pushed
36e051842720: Pushed
d1646aaa6540: Pushed
19382582b926: Pushed
41715d8d7d2b: Pushed
f3a38968d075: Pushed
a327787b3c73: Pushed
5bb0785f2eee: Pushed
rayan_v1: digest: sha256:7d6f60c13d83d4cb7b68d41c818175ed04de43c3fef44f31a9abe15432d71ea7 size: 2626
[root@docker-mgr certs]#
```

در عکس زیر مشاهده میکنید image به اسم centos:ver1 از registry server docker-mgr گرفته شد.

```
[root@docker-worker1 ~]# docker pull docker-mgr:443/centos:ver1
ver1: Pulling from centos
8ba884070f61: Pull complete
Digest: sha256:ca58fe458b8d94bc6e3072f1cfbd334855858e05e1fd633aa07cf7f82b048e66
Status: Downloaded newer image for docker-mgr:443/centos:ver1
[root@docker-worker1 ~]#
```

Restricting Access to registry server

با بالا رفتن تعداد نود های docker شاید لازم باشد دسترسی بعضی از آنها به سرور registry بسته شود.برای این کار راه های زیادی وجود دارد اما ساده ترین آن احراز هویت (authentication) میباشد.برای این کار باید از htpasswd استفاده کرد.

```
mkdir /auth

docker run --entrypoint htpasswd registry:2 -Bbn rayan rayanpassword >> /auth/htpasswd
```

بعد از اجرای دستور فوق:

```
[root@docker-mgr ~]# docker run --entrypoint htpasswd registry:2 -Bbn rayan rayanpassword > /auth/htpasswd
[root@docker-mgr ~]# cat /auth/htpasswd
rayan:$2y$05$zJb/3cs0slxbW/Q0DVGNjuDXHD3XxconH2M2ZQg1AgNkaAZUKxEDG
```

حال باید کانتینر مربوط به رجیستری را از دوباره و با پارامتر های جدید بسازیم:

```
docker run -d --restart=always -v /auth:/auth -e "REGISTRY_AUTH=htpasswd" -e

"REGISTRY_AUTH_HTPASSWD_REALM=Registry Realm" -e

REGISTRY_AUTH_HTPASSWD_PATH=/auth/htpasswd --name rayan_registry -v /etc/certs:/certs -e

REGISTRY_HTTP_ADDR=0.0.0.0:443 -e REGISTRY_HTTP_TLS_CERTIFICATE=/certs/ca.crt -e

REGISTRY_HTTP_TLS_KEY=/certs/ca.key -p 443:443 registry:2
```

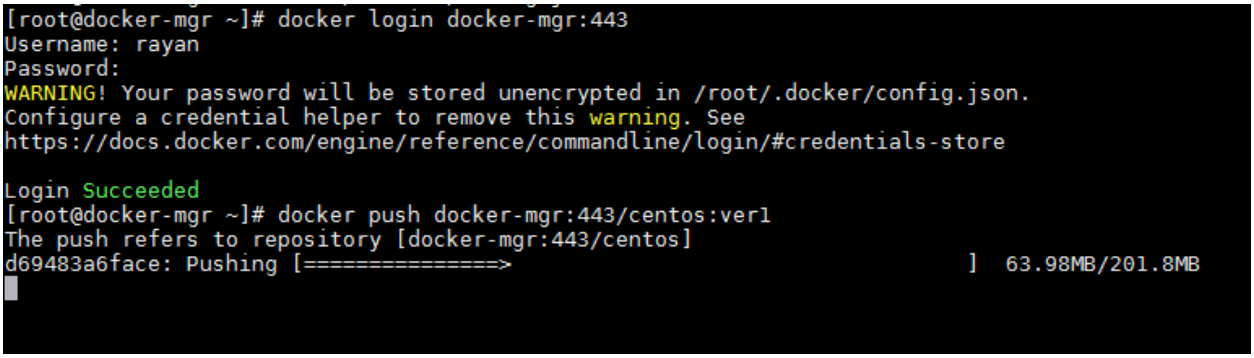
بعد از ران شدن registry اگر دستوری push/pull رو برای registry خود اجرا کنیم با مشکل مواجه میشیم:

```
[root@docker-mgr ~]# docker push docker-mgr:443/centos:ver1
The push refers to repository [docker-mgr:443/centos]
d69483a6face: Preparing
unauthorized: authentication required
[root@docker-mgr ~]#
```

```
[root@docker-worker1 ~]# docker pull docker-mgr:443/centos:ver1
Error response from daemon: unauthorized: authentication required
[root@docker-worker1 ~]#
```

برای حل مشکل باید در رجیستری خود login کنیم:

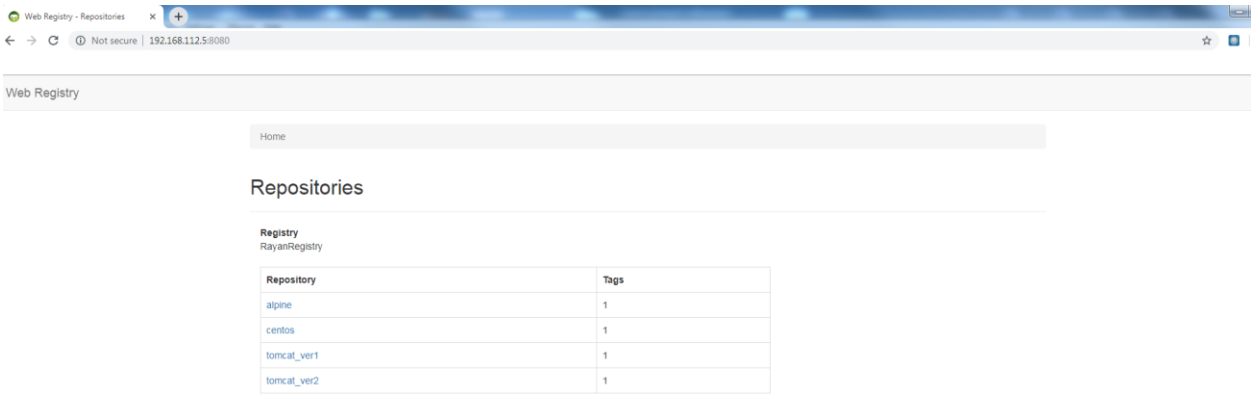
root@docker-mgr ~]# docker login docker-mgr:443



Using hyper/docker-registry-web

داشتن یک web-application خیلی ساده برای دیدن image های موجود یکی از نیاز های مهم در زمان راه اندازی registry یک سازمان میباشد.استفاده از hyper/docker-registry-web یکی از راه کار ها میباشد.

```
docker run -it -d -p 8080:8080 \

--name rayan_registry_web \

--link rayan_registry \

-e REGISTRY_BASIC_AUTH="cmF5YW46cmF5YW5wYXNzd29yZA==" \

-e REGISTRY_TRUST_ANY_SSL=true \

-e REGISTRY_URL=https://rayan_registry:443/v2 \

-e REGISTRY_NAME=RayanRegistry hyper/docker-registry-web
```



Resources

https://docs.docker.com/registry/deploying/

https://www.digitalocean.com/community/tutorials/how-to-set-up-a-private-docker-registry-on-ubuntu-18-04

https://medium.com/@dataq/membuat-docker-private-registry-6efe534df4d5

https://hackernoon.com/create-a-private-local-docker-registry-5c79ce912620

در پایان امیدوارم مطالب این داکیومنت به شما کمک کرده باشد.

Contact me on linkedin: https://www.linkedin.com/in/mohammad-ali-kamalian-7a3a72124/