**Docker – Private Registries**

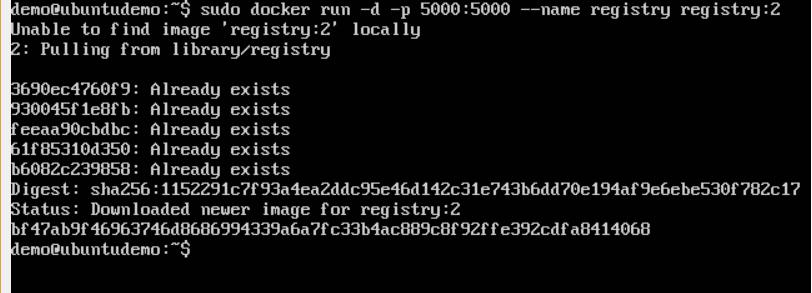
You might have the need to have your own private repositories. You may not want to host the repositories on Docker Hub. For this, there is a repository container itself from Docker. Let’s see how we can download and use the container for registry.

**Step 1**: Use the Docker **run** command to download the private registry. This can be doneusing the following command:

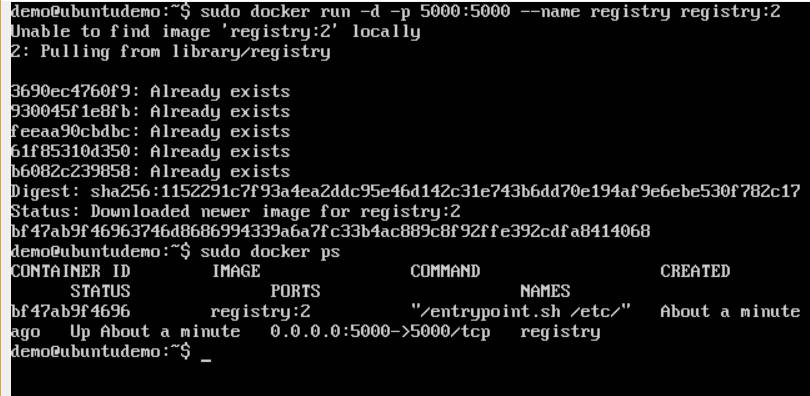
sudo docker run –d –p 5000:5000 –-name registry registry:2

The following points need to be noted about the above command:

* **Registry** is the container managed by Docker which can be used to host privaterepositories.
* The port number exposed by the container is 5000. Hence with the **–p command**, we are mapping the same port number to the 5000 port number on our localhost.
* We are just tagging the registry container as “2”, to differentiate it on the Docker host.
* The **–d** option is used to run the container in detached mode. This is so that the container can run in the background.



**Step 2**:Let’s do a **docker ps** to see that the registry container is indeed running.



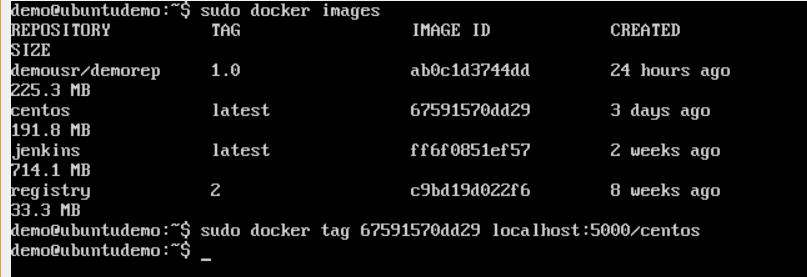
We have now confirmed that the registry container is indeed running.

**Step 3**:Now let’s tag one of our existing images so that we can push it to our localrepository. In our example, since we have the **centos** image available locally, we are going to tag it to our private repository and add a tag name of **centos**.

sudo docker tag 67591570dd29 localhost:5000/centos

The following points need to be noted about the above command:

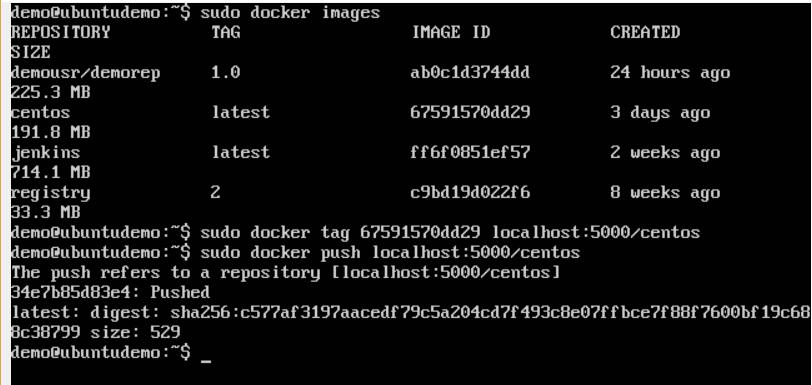
* **67591570dd29** refers to the Image ID for the **centos** image.
* **localhost:5000** is the location of our private repository.
* We are tagging the repository name as **centos** in our private repository.



**Step 4**:Now let’s use the Docker **push** command to push the repository to our privaterepository.

sudo docker push localhost:5000/centos

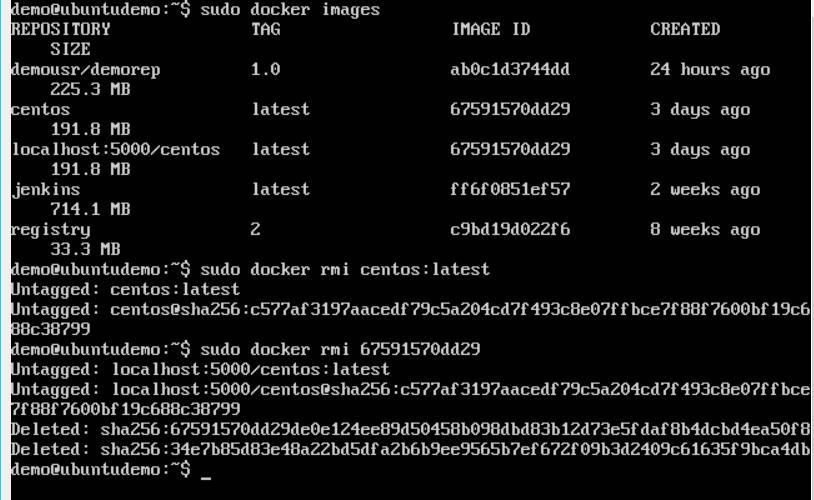
Here, we are pushing the **centos** image to the private repository hosted at **localhost:5000**.



**Step 5**:Now let’s delete the local images we have for **centos** using the **docker rmi** commands. We can then download the required **centos** image from our private repository.

sudo docker rmi centos:latest

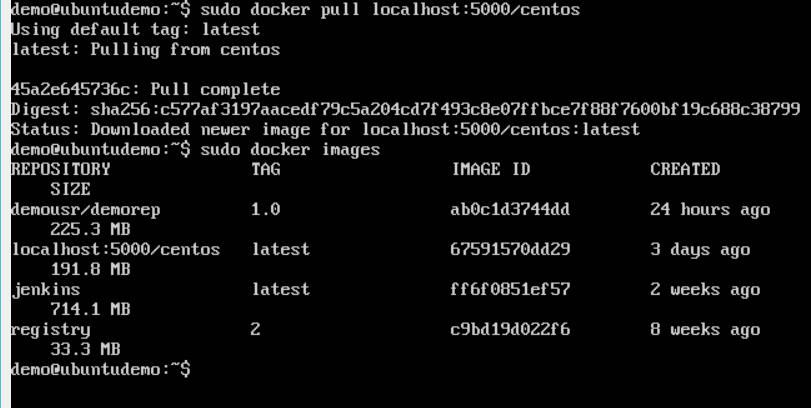
sudo docker rmi 67591570dd29



**Step 6**:Now that we don’t have any **centos** images on our local machine, we can nowuse the following Docker **pull** command to pull the **centos** image from our private repository.

sudo docker pull localhost:5000/centos

Here, we are pulling the **centos** image to the private repository hosted at **localhost:5000**.



If you now see the images on your system, you will see the **centos** image as well.

**Docker Private registry on Ubuntu 14.04**

## **Prerequisites :**

## 1. 2 ubuntu host one for docker registry and one for client.

2. Docker installed on both Hosts.

3. Domain name or hosts name both having entry into each one.

**Running On LocalHost**

**Steps 1** : Start Registery

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

**Steps 2** : Use it to get image

$ docker pull ubuntu && docker tag ubuntu localhost:5000/ubuntu

**Steps 3** : push it to your registry

$ docker push localhost:5000/ubuntu

**Steps 4** : pull it back from your registry

$ docker pull localhost:5000/ubuntu

**Steps 5** : stop your registry

$ docker stop registry && docker rm -v registry

**Creating a Docker Registry For Widely used with Secure with TLS**

Make The following Entry with your Doamin-Name

1. Open the /etc/default/docker file , your Engine daemon start options.
2. Edit (or add) the DOCKER\_OPTS line and add the –insecure-registry flag. This flag takes the URL of your registry, for example.

DOCKER\_OPTS="--insecure-registry www.synoa.local:5000"

Note : Here u can put your domain name or host name

1. Close and save the configuration file.
2. Restart your Docker daemon

Commands to start and stop docker

$ service docker stop

$ service docker start

1. Repeat this configuration on every Engine host that wants to access your registry.

## **Using self-signed certificates**

This is more secure than the insecure registry solution. You must configure every docker daemon that wants to access your registry

**1.** Generate your own certificate:

$ mkdir -p /opt/certs

$ openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /opt/certs/registry.key -out /opt/certs/registry.crt

Be sure to use the name www.synoa.local  as a CN.

**2.**Restricting access :

$ mkdir /opt/auth

$ docker run --entrypoint htpasswd registry:2 -Bbn synoa synoa321 > /opt/auth/htpasswd

**4.** Don’t forget to restart the Engine daemon.

Make sure you stopped your registry from the previous steps, then start your registry again with TLS enabled:

$ docker run -d -p 5000:5000 --restart=always --name registry \

-v `pwd`/registoryData/:/var/lib/registry \

-v `pwd`/auth:/auth \

-e "REGISTRY\_AUTH=htpasswd" \

-e "REGISTRY\_AUTH\_HTPASSWD\_REALM=Registry Realm" \

-e REGISTRY\_AUTH\_HTPASSWD\_PATH=/auth/htpasswd \

-v `pwd`/certs:/certs \

-e REGISTRY\_HTTP\_TLS\_CERTIFICATE=/certs/registry.crt \

-e REGISTRY\_HTTP\_TLS\_KEY=/certs/registry.key \

registry:2

**Accessing Repository From Client side :**

Step 1 :-

make entry to resolved domain name of repository server

$ nano /etc/hosts

192.168.1.204 www.synoa.local

Step 2 :-

$ nano /usr/local/share/ca-certificates/www.synoa.local.crt

and copy certificate containts from previous step into this file

$ update-ca-certificates

Step 3 :-

$ nano /etc/default/docker

DOCKER\_OPTS="--insecure-registry www.synoa.local:5000"

Step 4 :-

$ service docker restart

Step 5 :-

Testing :

$ docker login www.synoa.local:5000

$ docker pull ubuntu:14.04

$ docker tag ubuntu:14.04 www.synoa.local:5000/synoa:ubuntu-01

$ docker push www.synoa.local:5000/synoa:ubuntu-01