Docker Commands

Contents

Container & Image Management

To create a container in interactive mode:

docker run –d –it –name <New containername> < imame ID>

```
[root@nagios ~]# docker run -d -it centos
471d835d5afa1d833c67d51dbbf6e074f2a6f130e19fe9ce1c524c3241b958e7
[root@nagios ~]#
[root@nagios ~]#
```

To go inside the container:

docker exec -it <container ID> bash

```
[root@nagios ~]# docker exec -it caa bash
[root@caa028804d8c /]# ifconfig
```

Docker History will give all the changes in images:

```
[root@nagios ~]# docker history centos_new:v1
IMAGE               CREATED             CREATED BY                                      SIZE                COMMENT
6fffdf7076e7        2 weeks ago         bash                                            47.3MB              ifconfig & telnet package
5d0da3dc9764        4 weeks ago         /bin/sh -c #(nop)  CMD ["/bin/bash"]            0B
<missing>           4 weeks ago         /bin/sh -c #(nop)  LABEL org.label-schema.scâ€¦   0B
<missing>           4 weeks ago         /bin/sh -c #(nop) ADD file:805cb5e15fb6e0bb0â€¦   231MB
```

To create new image with comment with latest changes in container:

docker commit –m “message will give info” <NewImagename:tag> <Container ID>

```
[root@nagios ~]# docker commit -m "yum update,ping,ifconfig"  caa centos:v3
sha256:3245cd13499513c951e44b430530a5b5e92527a87155c3a4f597046a6cb43047
[root@nagios ~]# docker history centos:v3
IMAGE               CREATED             CREATED BY                                      SIZE                COMMENT
3245cd134995        About a minute ago  bash                                            300MB               yum update,ping,ifconfig
6fffdf7076e7        2 weeks ago         bash                                            47.3MB              ifconfig & telnet package
5d0da3dc9764        4 weeks ago         /bin/sh -c #(nop)  CMD ["/bin/bash"]            0B
<missing>           4 weeks ago         /bin/sh -c #(nop)  LABEL org.label-schema.scâ€¦   0B
<missing>           4 weeks ago         /bin/sh -c #(nop) ADD file:805cb5e15fb6e0bb0â€¦   231MB
[root@nagios ~]#
```

To take backup of Image:

docker save –output imagename-tag.tar.gz <imagename:tag>

==========

```
[root@nagios ~]# docker save --output centos-v3.tar.gz centos:v3
[root@nagios ~]# ll
total 583224
-rw-------. 1 root root      2598 Oct  4 22:30 anaconda-ks.cfg
-rw-------  1 root root 597173760 Oct 20 10:06 centos-v3.tar.gz
```

To take backup on particular path:

docker save –output /path/imagename-tag.tar.gz <imagename:tag>

```
[root@nagios ~]# docker save --output /u01/centos-v3.tar.gz centos:v3
[root@nagios ~]#
[root@nagios ~]#
[root@nagios ~]# cd /u01
[root@nagios u01]# ll
total 627212
-rw-r--r--.  1 root root       542 Oct  4 23:14 1
-rw-r--r--   1 root root         0 Oct 18 00:05 abc12a.txt
-rw-r--r--   1 root root         0 Oct 18 00:02 abc12.txt
-rw-r--r--   1 root root         0 Oct 18 00:02 abc1.txt
-rw-------   1 root root 597173760 Oct 20 10:08 centos-v3.tar.gz
```

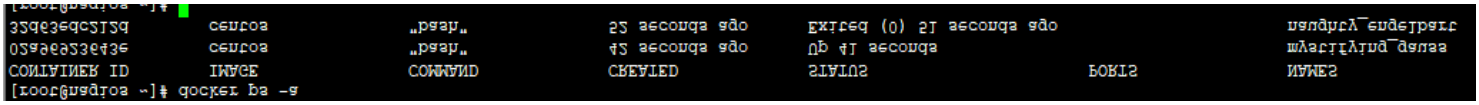
To See All the Running containers:

docker ps

```
[root@nagios ~]# docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
02a96923643e        centos              "bash"              3 seconds ago       Up 2 seconds                            mystifying_gauss
```

To See All the Running and stopped containers:

docker ps –a



To remove container:

docker rm <Container ID>

```
[root@nagios ~]# docker rm 32d
32d
[root@nagios ~]# docker rm mystifying_gauss
Error response from daemon: You cannot remove a running container 02a96923643eccb4224f8ela
e
[root@nagios ~]# docker rm -f mystifying_gauss
mystifying_gauss
[root@nagios ~]#
```

To see all the running and stopped container ID only:

docker ps –a –q

```
[root@nagios ~]# docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS                      PORTS               NAMES
9cc702b77aaf        centos              "/bin/bash"         8 seconds ago       Exited (0) 7 seconds ago                        frosty_elbakyan
a5fc76bfaabf        centos              "bash"              18 seconds ago      Up 17 seconds                               jayesh
d2148f4c0d33        centos              "bash"              26 seconds ago      Up 25 seconds                               determined_yalow
[root@nagios ~]#
[root@nagios ~]#
[root@nagios ~]#
[root@nagios ~]# docker ps -a -q
9cc702b77aaf
a5fc76bfaabf
d2148f4c0d33
```

Remove all container forcefully:

docker rm `docker ps –a –q`

```
[root@nagios ~]# docker rm -f `docker ps -a -q`
9cc702b77aaf
a5fc76bfaabf
d2148f4c0d33
[root@nagios ~]# docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
[root@nagios ~]#
```

to see all images :

docker image

to see the Image ID only belong to all images :

docker image –q

```
[root@nagios u01]# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
centos              v3                  3245cd134995        8 minutes ago       579MB
centos_new          v1                  6fffdf7076e7        2 weeks ago         279MB
centos              latest              5d0da3dc9764        4 weeks ago         231MB
[root@nagios u01]#
[root@nagios u01]#
[root@nagios u01]# docker images -q
3245cd134995
6fffdf7076e7
5d0da3dc9764
```

to remove any images :

docker rmi <image ID>

```
[root@nagios u01]# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
centos              v3                  3245cd134995        10 minutes ago      579MB
ubuntu              latest              ba6acccedd29        4 days ago          72.8MB
centos_new          v1                  6fffdf7076e7        2 weeks ago         279MB
[root@nagios u01]#
[root@nagios u01]#
[root@nagios u01]# docker rmi ubuntu
Untagged: ubuntu:latest
Untagged: ubuntu@sha256:626ffe58f6e7566e00254b638eb7e0f3b11d4da9675088f4781a50ae288f3322
Deleted: sha256:ba6acccedd2923aee4c2acc6a23780b14ed4b8a5fa4e14e252a23b846df9b6c1
Deleted: sha256:9f54eef412758095c8079ac465d494a2872e02e90bf1fb5f12a1641c0d1bb78b
[root@nagios u01]#
```

To remove all images:

docker rmi `docker images –q`

To restore the backup images after removing all:

docker load --input </path/backupofimage>

```
[root@nagios u01]# docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
[root@nagios u01]# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
[root@nagios u01]#
[root@nagios u01]#
[root@nagios u01]#
[root@nagios u01]# docker load --input /root/centos-v3.tar.gz
74ddd0ec08fa: Loading layer [==================================================>]  238.6MB/238.6MB
8cc9a38f31ab: Loading layer [==================================================>]  47.36MB/47.36MB
4c64f1d29ef2: Loading layer [==================================================>]  311.2MB/311.2MB
Loaded image: centos:v3
```

History of New Restored Images will be same:

```
[root@nagios u01]# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
centos              v3                  3245cd134995        16 minutes ago      579MB
[root@nagios u01]# docker history centos:v3
IMAGE               CREATED             CREATED BY                                      SIZE                COMMENT
3245cd134995        16 minutes ago      bash                                            300MB               yum update,ping,ifconfig
<missing>           2 weeks ago         bash                                            47.3MB              ifconfig & telnet package
<missing>           4 weeks ago         /bin/sh -c #(nop)  CMD ["/bin/bash"]            0B
<missing>           4 weeks ago         /bin/sh -c #(nop)  LABEL org.label-schema.scâ€¦   0B
<missing>           4 weeks ago         /bin/sh -c #(nop) ADD file:805cb5e15fb6e0bb0â€¦   231MB
```

Port Exposer:

1. docker run httpd

2. docker run -it --name web1 httpd bash

3. docker run -d httpd : It open website only on container IP address , Not host system IP address

```
[root@nagios html]# docker run -d --name web2 httpd
74625243a5da0c4aac924746c7f87d45f7967ae60567675fac1506518401816b
[root@nagios html]#
[root@nagios html]#
[root@nagios html]#
[root@nagios html]# docker inspect web2 | grep IPA
            "SecondaryIPAddresses": null,
            "IPAddress": "172.17.0.4",
                    "IPAMConfig": null,
                    "IPAddress": "172.17.0.4",
[root@nagios html]#
[root@nagios html]# curl http://172.17.0.4
<html><body><h1>It works!</h1></body></html>
[root@nagios html]#
```

Through Port Expose Host machine will use the container website using random port : docker run -d -p 80 httpd

```
[root@nagios html]# docker run -d -p 80 --name web5 httpd
4e56eb9480870e3e8c113ce4f478a86e93bd1bf592736ea6cc57b73ade423d1f
[root@nagios html]#
[root@nagios html]# docker inspect web5|grep IPA
            "SecondaryIPAddresses": null,
            "IPAddress": "172.17.0.5",
                    "IPAMConfig": null,
                    "IPAddress": "172.17.0.5",
[root@nagios html]# curl http://
curl: (6) Could not resolve host: http; Unknown error
[root@nagios html]# curl http://172.17.0.5
<html><body><h1>It works!</h1></body></html>
[root@nagios html]#
[root@nagios html]#
[root@nagios html]# docker ps
CONTAINER ID        IMAGE               COMMAND              CREATED             STATUS              PORTS                     NAMES
4e56eb948087        httpd               "httpd-foreground"   36 seconds ago      Up 35 seconds       0.0.0.0:32769->80/tcp     web5
74625243a5da        httpd               "httpd-foreground"   4 minutes ago       Up 4 minutes        80/tcp                    web2
55c81a79b8d4        httpd               "httpd-foreground"   13 minutes ago      Up 13 minutes       0.0.0.0:32768->80/tcp     quirky_dubinsky
22286426fe14        httpd               "httpd-foreground"   17 minutes ago      Up 17 minutes       80/tcp                    heuristic_yonath
[root@nagios html]#
[root@nagios html]# curl http://192.168.56.101:32769
<html><body><h1>It works!</h1></body></html>
[root@nagios html]#
```

Host will use the container web using Specific port : docker run -d --name web5 -p 8080:80 httpd

-p 8080:80 means
-p hostport:container port
It open website on container port 80 but assign specific port to host system Ip address

```
[root@nagios ~]# docker run -d --name Nweb1 -p 8080:80  httpd
7ba25ae2c2f0f42013f81c4711307aaadc7a1cda91212900988d109a8ed3eda5
[root@nagios ~]#
[root@nagios ~]# docker inspect 7ba | grep IPA
            "SecondaryIPAddresses": null,
            "IPAddress": "172.17.0.2",
                    "IPAMConfig": null,
                    "IPAddress": "172.17.0.2",
[root@nagios ~]# curl http://172.17.0.2
<html><body><h1>It works!</h1></body></html>
[root@nagios ~]#
[root@nagios ~]#
[root@nagios ~]# curl http://192.168.56.101:8080
<html><body><h1>It works!</h1></body></html>
[root@nagios ~]#
[root@nagios ~]#
[root@nagios ~]# docker ps
CONTAINER ID        IMAGE               COMMAND              CREATED             STATUS              PORTS                    NAMES
7ba25ae2c2f0        httpd               "httpd-foreground"   About a minute ago   Up About a minute   0.0.0.0:8080->80/tcp     Nweb1
```

Docker files and image creation implementation

==========

1. Docker file should have name “dockerfile”
2. For default path no need to do anything
3. For specific path use “–f /path” to add configuration
4. All steps run at the time of image creation.

```
[root@nagios ~]# vi dockerfile
[root@nagios ~]# cat dockerfile
#This is my first docker file

FROM centos:7
LABEL Maintainer="Jayesh.shah@stl.tech"
RUN yum update -y
RUN yum install httpd -y
RUN yum install telnet -y
RUN useradd jayesh
USER jayesh
[root@nagios ~]#
```

Build Image using Docker file with automate changes:

docker build -t <image:version> <path of docker file>

docker build -t centosnew:v9 .

```
Successfully built 6ffac0cea239
Successfully tagged centosnew:v9
[root@nagios ~]# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED                  SIZE
httpd               latest              ad17c88403e2        Less than a second ago   143MB
centosnew           v9                  6ffac0cea239        45 seconds ago           799MB
centos              v3                  3245cd134995        2 hours ago              579MB
centos              7                   eeb6ee3f44bd        4 weeks ago              204MB
```

docker build -t centosnewwithpath:v10 -f dockerfilewithpath .

```
[root@nagios u01]# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED                  SIZE
httpd               latest              ad17c88403e2        Less than a second ago   143MB
centosnew           v9                  6ffac0cea239        5 minutes ago            799MB
centosnewwithpath   v10                 6ffac0cea239        5 minutes ago            799MB
centos              v3                  3245cd134995        2 hours ago              579MB
centos              7                   eeb6ee3f44bd        4 weeks ago              204MB
```

Tags :

CMD: it will execute at the time of Container creationm

if you define multiple CMD than it will execute last Command only.

if you define command during container creation than last CMD will be overwrite

ENTRYPOINT : it will execute at the time of Container creationm

if you define multiple CMD than it will execute last Command only.

but you define command during container creation than last CMD will not be overwrite and last line will be executed

USER: it will used to login container through particular user

LABEL : to maintain the change name

FROM : It is used to get image from docker hub.

Port Exposed in container

==========

a. Expose port is used to identify the port used in container

```
[root@nagios u01]# vi dockerfilewithpath
[root@nagios u01]# cat dockerfilewithpath
#This is my first docker file

FROM centos:7
LABEL Maintainer="Jayesh.shah@stl.tech"
RUN yum update -y
RUN yum install httpd -y
RUN yum install telnet -y
RUN useradd jayesh
CMD echo "Remember to check Container IP address"
EXPOSE 80 443
ENTRYPOINT apachectl "-DFOREGROUND"
USER jayesh
[root@nagios u01]#
```

```
[root@nagios u01]# docker build -t myimage:v1 -f dockerfilewithpath .
Sending build context to Docker daemon  1.013GB
Step 1/8 : FROM centos
 ---> 5d0da3dc9764
Step 2/8 : LABEL Maintainer="Jayesh@STL.Tech"
 ---> Running in c04ec81ele25
Removing intermediate container c04ec81ele25
 ---> 9669e0f964de
Step 3/8 : RUN yum update -y
 ---> Running in 416cf6548334
CentOS Linux 8 - AppStream                      2.1 MB/s | 8.1 MB     00:03
CentOS Linux 8 - BaseOS                         1.2 MB/s | 3.5 MB     00:02
CentOS Linux 8 - Extras                          15 kB/s |  10 kB     00:00
```

```
Installed:
  telnet-1:0.17-76.el8.x86_64

Complete!
Removing intermediate container 39d817326ce3
 ---> 0d8f10cb97de
Step 6/8 : CMD echo "Remember to check Container IP address"
 ---> Running in 76524c7aae18
Removing intermediate container 76524c7aae18
 ---> 2d59fd7d0a92
Step 7/8 : EXPOSE 80 443
 ---> Running in 3ed8b1ccb4c1
Removing intermediate container 3ed8b1ccb4c1
 ---> c5ac5131b291
Step 8/8 : ENTRYPOINT apachectl "-DFOREGROUND"
 ---> Running in d95b39d77910
Removing intermediate container d95b39d77910
 ---> 6799fb244ff9
Successfully built 6799fb244ff9
Successfully tagged myimage:v1
```

```
[root@nagios u01]# docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS                                                  NAMES
b20cc78dfecb        myimage:v1          "/bin/sh -c 'apachecâ€¦"   About a minute ago   Up About a minute   0.0.0.0:32769->80/tcp, 0.0.0.0:32768->443/tcp   sad_lichterman
[root@nagios u01]# docker exec -it b20 bash
[root@b20cc78dfecb /]# cd /var/www/html/
[root@b20cc78dfecb html]# ll
bash: ll: command not found
[root@b20cc78dfecb html]# ls
[root@b20cc78dfecb html]# vi index.html
[root@b20cc78dfecb html]# exit
exit
[root@nagios u01]# curl http://172.17.0.2
<html>
<h1> This is my birthday </h1>
</html>
[root@nagios u01]# curl http://192.168.56.101:32769
<html>
<h1> This is my birthday </h1>
</html>
[root@nagios u01]# ^C
[root@nagios u01]#
```

Update in docker file with CP command:
copy index.html /var/www/html

```
[root@nagios u01]# vi dockerfilewithpath
[root@nagios u01]# docker build -t myimage:v2 -f dockerfilewithpath .
Sending build context to Docker daemon  1.013GB
Step 1/9 : FROM centos
 ---> 5d0da3dc9764
Step 2/9 : LABEL Maintainer="Jayesh@STL.Tech"
 ---> Using cache
 ---> 9669e0f964de
Step 3/9 : RUN yum update -y
 ---> Using cache
 ---> 40203e193cb0
Step 4/9 : RUN yum install *http* -y
 ---> Using cache
 ---> ad59e04984f5
Step 5/9 : RUN yum install telnet -y
 ---> Using cache
 ---> 0d8f10cb97de
Step 6/9 : CMD echo "Remember to check Container IP address"
 ---> Using cache
 ---> 2d59fd7d0a92
Step 7/9 : EXPOSE 80 443
 ---> Using cache
 ---> c5ac5131b291
Step 8/9 : copy index.html /var/www/html
 ---> Using cache
 ---> 35a2a5f2d367
Step 9/9 : ENTRYPOINT apachectl "-DFOREGROUND"
 ---> Running in 7a55a1cde2da
Removing intermediate container 7a55a1cde2da
 ---> d01007e12cb8
Successfully built d01007e12cb8
Successfully tagged myimage:v2
```

Volume Management or Volume Mapping

==========

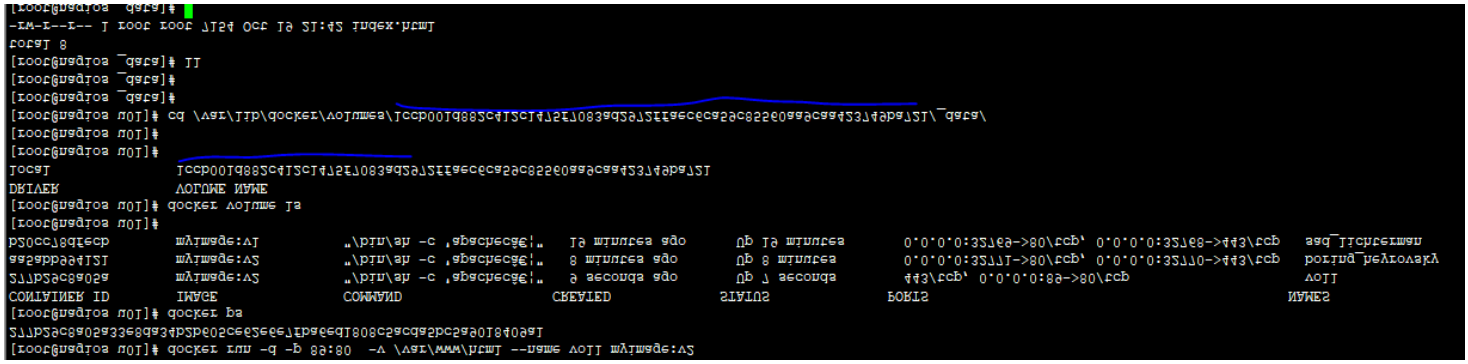
Approach 1 : only “–v <container path>”
Approach 2: map “<host path > -v <container path>”
Approach 3: map Volume ID “VolumeName –v <container path>”

App1 :

docker run -d -p 89:80 -v /var/www/html --name vol1 myimage:v2

This will map container path “/var/www/html” to default path with random volume ID, So all the data present in container path will be stored at below path

/var/lib/docker/volumes/**<Random Volume ID>**/_data/



App2:

mkdir -p /u01/data; docker run -d -p 91:80 -v /u01/data:/var/www/html --name vol3 myimage:v2

a. Mapped the host path “/u01/data” to “/var/www/html” So all the data belong to container at container path will be saved at given host path but no volume ID generated for the same.
b. We can map same path with multiple container to shared path

```
[root@nagios ~]# mkdir -p /u01/data
[root@nagios ~]#
[root@nagios ~]# docker run -d -p 91:80  -v /u01/data:/var/www/html --name vol3 myimage:v2
0c0ce05f5351f7558bfcafb823d5a1148da66358f443bb335658c3ec5a28a9e9
[root@nagios ~]#
[root@nagios ~]#
[root@nagios ~]# docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS                                NAMES
0c0ce05f5351        myimage:v2          "/bin/sh -c 'apachecâ€¦"   4 seconds ago       Up 3 seconds        443/tcp, 0.0.0.0:91->80/tcp          vol3
```

App3:

docker run -d -p 92:80 -v web_vol3:/var/www/html --name vol4 myimage:v2

a. Here mapped the docker default volume with fixed Volume ID to Container path So all the data belong to container will be stored in docker default with defined Volume ID generated.
/var/lib/docker/volumes/<Volume ID>/_data/
b. We can check with volume ID with below command :
docker volume ls
docker volume inspect <volumeID>

```
[root@nagios ~]# docker run -d -p 92:80  -v web_vol3:/var/www/html --name vol4 myimage:v2
0920307a61fe47208797a2dd1a2d815273ae7178407c92bae64f249464048f88
[root@nagios ~]#
[root@nagios ~]# docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS                          NAMES
0920307a61fe        myimage:v2          "/bin/sh -c 'apachecâ€¦"   5 seconds ago       Up 4 seconds        443/tcp, 0.0.0.0:92->80/tcp    vol4
[root@nagios ~]#
[root@nagios ~]#
[root@nagios ~]# docker volume ls
DRIVER              VOLUME NAME
local               1ccb001d882c412c1475f7083ad2972ffaec6ca59c85560aa9caa423749ba721
local               web_vol3
[root@nagios ~]#
[root@nagios ~]#
[root@nagios ~]# cd /var/lib/docker/volumes/
1ccb001d882c412c1475f7083ad2972ffaec6ca59c85560aa9caa423749ba721/ web_vol3/
[root@nagios ~]# cd /var/lib/docker/volumes/web_vol3/_data/
[root@nagios _data]# ll
total 8
-rw-r--r-- 1 root root 7154 Oct 19 21:42 index.html
[root@nagios _data]#
```

```
[root@nagios _data]# docker volume ls
DRIVER              VOLUME NAME
local               1ccb001d882c412c1475f7083ad2972ffaec6ca59c85560aa9caa423749ba721
local               web_vol3
[root@nagios _data]# docker volume inspect web_vol3
[
    {
        "CreatedAt": "2021-10-19T22:04:58+05:30",
        "Driver": "local",
        "Labels": null,
        "Mountpoint": "/var/lib/docker/volumes/web_vol3/_data",
        "Name": "web_vol3",
        "Options": null,
        "Scope": "local"
    }
]
```

Folder Sharing

==========

Task : We have to share particular volume in host system “/u01/data” to 3 container and in sync to all of the container

docker run -d -p 92:80 -v /u01/data:/var/www/html --name vol3 myimage:v2

docker run -d -p 93:80 -v /u01/data:/var/www/html --name vol4 myimage:v2
docker run -d -p 94:80 -v /u01/data:/var/www/html --name vol5 myimage:v2

Three Container **vol3,vol4,vol5** created using same volume **/u01/data** mapped :

```
[root@nagios data]# docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
[root@nagios data]#
[root@nagios data]#
[root@nagios data]#
[root@nagios data]# docker run -d -p 92:80  -v /u01/data:/var/www/html --name vol3 myimage:v2
5b9f43111306d4874d398215445f3575d65f6304e83174b1ef79a0f0d63de51c
[root@nagios data]#
[root@nagios data]#
[root@nagios data]# docker run -d -p 93:80  -v /u01/data:/var/www/html --name vol4 myimage:v2
013d4142791a84d451ca50d1423la6cec957116bd7d8f47dd3a82c048a30aeee
[root@nagios data]#
[root@nagios data]#
[root@nagios data]# docker run -d -p 94:80  -v /u01/data:/var/www/html --name vol5 myimage:v2
fbc5f60641bd29aa374167bd2c3440c7e061a76e5790242c33e93c614d08294e
[root@nagios data]#
[root@nagios data]#
[root@nagios data]# docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS                           NAMES
fbc5f60641bd        myimage:v2          "/bin/sh -c 'apachecâ€¦"   4 seconds ago       Up 3 seconds        443/tcp, 0.0.0.0:94->80/tcp     vol5
013d4142791a        myimage:v2          "/bin/sh -c 'apachecâ€¦"   12 seconds ago      Up 11 seconds       443/tcp, 0.0.0.0:93->80/tcp     vol4
5b9f43111306        myimage:v2          "/bin/sh -c 'apachecâ€¦"   20 seconds ago      Up 19 seconds       443/tcp, 0.0.0.0:92->80/tcp     vol3
```

Checked Vol3 and created one file in shared partition

```
[root@nagios data]# docker exec -it vol3 bash
[root@5b9f43111306 /]# df -kh
Filesystem               Size  Used Avail Use% Mounted on
overlay                   19G   14G  3.9G  78% /
tmpfs                     64M     0   64M   0% /dev
tmpfs                    2.0G     0  2.0G   0% /sys/fs/cgroup
/dev/mapper/centos-root   19G   14G  3.9G  78% /etc/hosts
shm                       64M     0   64M   0% /dev/shm
/dev/mapper/centos-u01   6.3G 1022M  4.9G  17% /var/www/html
tmpfs                    2.0G     0  2.0G   0% /proc/asound
tmpfs                    2.0G     0  2.0G   0% /proc/acpi
tmpfs                    2.0G     0  2.0G   0% /proc/scsi
tmpfs                    2.0G     0  2.0G   0% /sys/firmware
[root@5b9f43111306 /]# cd /var/www/html
[root@5b9f43111306 html]# touch crestedin_VOL3
[root@5b9f43111306 html]#
```

Checked vol4 and found files created under vol3 is visible and new file created in vol4

```
[root@nagios data]# docker exec -it vol4 bash
[root@013d4142791a /]# df -kh
Filesystem               Size  Used Avail Use% Mounted on
overlay                   19G   14G  3.9G  78% /
tmpfs                     64M     0   64M   0% /dev
tmpfs                    2.0G     0  2.0G   0% /sys/fs/cgroup
/dev/mapper/centos-root   19G   14G  3.9G  78% /etc/hosts
shm                       64M     0   64M   0% /dev/shm
/dev/mapper/centos-u01   6.3G 1022M  4.9G  17% /var/www/html
tmpfs                    2.0G     0  2.0G   0% /proc/asound
tmpfs                    2.0G     0  2.0G   0% /proc/acpi
tmpfs                    2.0G     0  2.0G   0% /proc/scsi
tmpfs                    2.0G     0  2.0G   0% /sys/firmware
[root@013d4142791a /]# cd /var/www/html
[root@013d4142791a html]# ll
bash: ll: command not found
[root@013d4142791a html]# ls
crestedin_VOL3
[root@013d4142791a html]# touch "crestedin_is_Shownin_VOL4"
[root@013d4142791a html]# exit
```

Checked in vol5 and found files created under vol3 and vol4 is visible and new file created in vol5

```
[root@nagios data]# docker exec -it vol5 bash
[root@fbc5f6064lbd /]# df -kh
Filesystem               Size  Used Avail Use% Mounted on
overlay                   19G   14G  3.9G  78% /
tmpfs                     64M     0   64M   0% /dev
tmpfs                    2.0G     0  2.0G   0% /sys/fs/cgroup
/dev/mapper/centos-root   19G   14G  3.9G  78% /etc/hosts
shm                       64M     0   64M   0% /dev/shm
/dev/mapper/centos-u01   6.3G 1022M  4.9G  17% /var/www/html
tmpfs                    2.0G     0  2.0G   0% /proc/asound
tmpfs                    2.0G     0  2.0G   0% /proc/acpi
tmpfs                    2.0G     0  2.0G   0% /proc/scsi
tmpfs                    2.0G     0  2.0G   0% /sys/firmware
[root@fbc5f6064lbd /]# cd /var/www/html/
[root@fbc5f6064lbd html]# ls
crestedin_is_Shownin_VOL4  crestedin_VOL3
[root@fbc5f6064lbd html]#
[root@fbc5f6064lbd html]# touch "both_VOL3_and_VOL4_shown_in_VOL5"
[root@fbc5f6064lbd html]# exit
exit
```

Check the host partition and found all the files present in this :

```
[root@nagios data]# ll
total 0
-rw-r--r-- 1 root root 0 Oct 19 22:23 both_VOL3_and_VOL4_shown_in_VOL5
-rw-r--r-- 1 root root 0 Oct 19 22:22 crestedin_is_Shownin_VOL4
-rw-r--r-- 1 root root 0 Oct 19 22:21 crestedin_VOL3
[root@nagios data]# pwd
/u01/data
[root@nagios data]#
```

Removed all three container vol3 and vol4 and vol5 but data remain present at host shared partition Conclusion: after deleting the container we can save the configuration in it

```
[root@nagios data]#
[root@nagios data]# docker rm -f vol3
vol3
[root@nagios data]# docker rm -f vol4
vol4
[root@nagios data]# docker rm -f vol5
vol5
[root@nagios data]#
[root@nagios data]#
[root@nagios data]# docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
[root@nagios data]#
[root@nagios data]#
[root@nagios data]# ll
total 0
-rw-r--r-- 1 root root 0 Oct 19 22:23 both_VOL3_and_VOL4_shown_in_VOL5
-rw-r--r-- 1 root root 0 Oct 19 22:22 crestedin_is_Shownin_VOL4
-rw-r--r-- 1 root root 0 Oct 19 22:21 crestedin_VOL3
```

Same thing we can map the Volume ID with multiple container for file sharing

docker run -d -p 92:80 -v web_vol3:/var/www/html --name vol3 mypage:v2
docker run -d -p 93:80 -v web_vol3:/var/www/html --name vol4 mypage:v2
docker run -d -p 94:80 -v web_vol3:/var/www/html --name vol5 mypage:v2

Memory Management

==========

docker stats <container ID> :

a. it is used to check the utilization in the container

b. by default all the memory of the host system is assigned to the container

c. docker to <container ID> give the processes running in container

```
[root@nagios data]# free -g
              total        used        free      shared  buff/cache   available
Mem:              3           0           0           0           2           2
Swap:             3           0           3
[root@nagios data]#
[root@nagios data]#
[root@nagios data]# docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS                           NAMES
69fe0e713d8b        myimage:v2          "/bin/sh -c 'apachecâ€¦"   47 seconds ago      Up 45 seconds       443/tcp, 0.0.0.0:94->80/tcp   vol5
[root@nagios data]#
[root@nagios data]#
[root@nagios data]# docker stats vol5

CONTAINER ID        NAME                CPU %               MEM USAGE / LIMIT     MEM %               NET I/O             BLOCK I/O           PIDS
69fe0e713d8b        vol5                0.09%               6.688MiB / 3.858GiB   0.17%               656B / 0B           0B / 0B             214

CONTAINER ID        NAME                CPU %               MEM USAGE / LIMIT     MEM %               NET I/O             BLOCK I/O           PIDS
69fe0e713d8b        vol5                0.09%               6.688MiB / 3.858GiB   0.17%               656B / 0B           0B / 0B             214
```

d. For Setting the specific memory/CPU to one container
vi /etc/default/grub : add GRUB_CMDLINE_LINUX="cdgroup_enable=memory swapaccount=1"

```
[root@nagios data]# vi /etc/default/grub

GRUB_TIMEOUT=5
GRUB_DISTRIBUTOR="$(sed 's, release .*$,,g' /etc/system-release)"
GRUB_DEFAULT=saved
GRUB_DISABLE_SUBMENU=true
GRUB_TERMINAL_OUTPUT="console"
GRUB_CMDLINE_LINUX="rd.lvm.lv=centos/root rd.lvm.lv=centos/swap rhgb quiet"
GRUB_DISABLE_RECOVERY="true"
~
```

```
[root@nagios data]# vi /etc/default/grub

GRUB_TIMEOUT=5
GRUB_DISTRIBUTOR="$(sed 's, release .*$,,g' /etc/system-release)"
GRUB_DEFAULT=saved
GRUB_DISABLE_SUBMENU=true
GRUB_TERMINAL_OUTPUT="console"
#GRUB_CMDLINE_LINUX="rd.lvm.lv=centos/root rd.lvm.lv=centos/swap rhgb quiet"
GRUB_CMDLINE_LINUX="cdgroup_enable=memory swapaccount=1"
GRUB_DISABLE_RECOVERY="true"
```

e. Save the changes and exit the file

f. Update the grub file using below command
Update-grub

g. Finally, reboot your machine for the changes to take place
sudo docker run -it --memory="[memory_limit]" [docker_image]

(The value of memory_limit should be a positive integer followed by the suffix b, k, m, or g (short for bytes, kilobytes, megabytes, or gigabytes).

docker run -d --memory="1g" -p 94:80 -v /u01/data:/var/www/html --name vol5 myimage:v2

docker run -d --memory="1g" –cpu =”.5” -p 95:81 -v /u01/data:/var/www/html --name vol6 myimage:v2

```
[root@nagios ~]# docker run -d --memory="1g" -p 94:80  -v /u01/data:/var/www/html --name vol5 myimage:v2
fb0c49127ab8729484f9dc16b2d0fbf4b698c21dad1a17a261033da2f70d7c0d
[root@nagios ~]#
[root@nagios ~]#
[root@nagios ~]# docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS                          NAMES
fb0c49127ab8        myimage:v2          "/bin/sh -c 'apachecâ€¦"   8 seconds ago       Up 7 seconds        443/tcp, 0.0.0.0:94->80/tcp    vol5
[root@nagios ~]#
[root@nagios ~]#
[root@nagios ~]#
[root@nagios ~]# docker stats vol5

CONTAINER ID        NAME                CPU %               MEM USAGE / LIMIT   MEM %               NET I/O             BLOCK I/O           PIDS
fb0c49127ab8        vol5                0.09%               22.63MiB / 1GiB     2.21%               656B / 0B           0B / 0B             214

CONTAINER ID        NAME                CPU %               MEM USAGE / LIMIT   MEM %               NET I/O             BLOCK I/O           PIDS
fb0c49127ab8        vol5                0.09%               22.63MiB / 1GiB     2.21%               656B / 0B           0B / 0B             214
^C
[root@nagios ~]# free -g
              total        used        free      shared  buff/cache   available
Mem:              3           0           2           0           0           3
Swap:             3           0           3
[root@nagios ~]#
```

NETWORK ADMINISTRATION

==========

1. There is three type of Network Type : Bridge,host & none

```
[root@nagios ~]# docker network ls
NETWORK ID          NAME                DRIVER              SCOPE
589898378cbf        bridge              bridge              local
64455b641954        host                host                local
380fda14fc77        none                null                local
[root@nagios ~]#
```

Bridge Default details :

```
[root@nagios ~]# docker network inspect bridge
[
    {
        "Name": "bridge",
        "Id": "589898378cbf9d4b985bf48efad60b523c4fda2f7e0d7283f120032ffd8a2ea9",
        "Created": "2021-10-19T21:27:40.813834599+05:30",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": null,
            "Config": [
                {
                    "Subnet": "172.17.0.0/16",
                    "Gateway": "172.17.0.1"
                }
            ]
        },
```

For Creating Network :

docker network create --subnet <Subnet Range> --gateway <Gateway IP> --driver=<Network Type bridge/host/null> <network Name>

```
[root@nagios ~]# docker network create --subnet 192.168.10.0/24 --gateway 192.168.10.254 --driver=bridge mybridge
e671cfcc81f6ccc29497045e10130c3d509a9e6a500ddfee8cc37c87189ea4a0
[root@nagios ~]# docker network ls
NETWORK ID          NAME                DRIVER              SCOPE
589898378cbf        bridge              bridge              local
64455b641954        host                host                local
e671cfcc81f6        mybridge            bridge              local
380fda14fc77        none                null                local
[root@nagios ~]#
```

```
[root@nagios ~]# docker network inspect mybridge
[
    {
        "Name": "mybridge",
        "Id": "e671cfcc81f6ccc29497045e10130c3d509a9e6a500ddfee8cc37c87189ea4a0",
        "Created": "2021-10-19T21:40:48.238175444+05:30",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": {},
            "Config": [
                {
                    "Subnet": "192.168.10.0/24",
                    "Gateway": "192.168.10.254"
                }
            ]
        },
```

Create Container using new network :

docker run –d –it –name <containername> --net <networkname> <image:tag>

```
[root@nagios ~]# docker run -d --name testl --net mybridge --privileged centos:all /usr/sbin/init
07ae3d3ad89blef2b84fblbc050ea880e299532c77dldelc958eacb975f5184e
[root@nagios ~]#
[root@nagios ~]# docker inspect testl |grep IPA
            "SecondaryIPAddresses": null,
            "IPAddress": "",
                    "IPAMConfig": null,
                    "IPAddress": "192.168.10.1",
[root@nagios ~]#
```

Connect & Disconnect the Network in Existing Container :

docker network connect <Network name> <container name/id>

docker network disconnect <Network name> <container name/id>

```
[root@nagios ~]# docker run -d --name test3 --net bridge --privileged centos:all /usr/sbin/init
44c1840d126bcabd9ed5101d957492a2ae0cc10df51654fb768dba0d34990505
[root@nagios ~]#
[root@nagios ~]#
[root@nagios ~]#
[root@nagios ~]# docker inspect test3|grep IPA
            "SecondaryIPAddresses": null,
            "IPAddress": "172.17.0.2",
                    "IPAMConfig": null,
                    "IPAddress": "172.17.0.2",
[root@nagios ~]#
[root@nagios ~]#
[root@nagios ~]# docker network connect mybridge test3
[root@nagios ~]# docker inspect test3|grep IPA
            "SecondaryIPAddresses": null,
            "IPAddress": "172.17.0.2",
                    "IPAMConfig": null,
                    "IPAddress": "172.17.0.2",
                    "IPAMConfig": {},
                    "IPAddress": "192.168.10.1",
[root@nagios ~]#
```

```
[root@nagios ~]#
[root@nagios ~]#
[root@nagios ~]# docker network disconnect bridge test3
[root@nagios ~]#
[root@nagios ~]# docker inspect test3|grep IPA
            "SecondaryIPAddresses": null,
            "IPAddress": "",
                    "IPAMConfig": {},
                    "IPAddress": "192.168.10.1",
[root@nagios ~]#
```

How to add specific IP to container from any of the network:

note : Specific IP can be assigned in personalize network only not on public network

docker run -d --name <container name> --net <network name> --ip <IP address> <Image:tag>

we can also add the IP during container is already running

docker connect <networkname> --ip <IPaddress belong to network> <container ID/name>

```
[root@nagios ~]# docker network connect bridge --ip 172.17.0.9 test3
Error response from daemon: user specified IP address is supported on user defined networks only
[root@nagios ~]# docker network disconnect mybridge test3
[root@nagios ~]# docker network connect mybridge --ip 192.168.10.9 test3
[root@nagios ~]# docker inspect test3 |grep ipa
[root@nagios ~]# docker inspect test3 |grep IPA
            "SecondaryIPAddresses": null,
            "IPAddress": "",
                    "IPAMConfig": {
                    "IPAddress": "192.168.10.9",
[root@nagios ~]#
```

We can give the hostname while creating the container and if the container is running using personalize network then all container can communicate using container name itself

docker run -d --name container1 -h host1 --net mybridge --ip 192.168.10.5 --privileged centos:all /usr/sbin/init
docker run -d --name container2 -h host2 --net mybridge --ip 192.168.10.9 --privileged centos:all /usr/sbin/init
docker run -d --name container3 -h host3 --net mybridge --ip 192.168.10.20 --privileged centos:all /usr/sbin/init

```
[root@nagios ~]# docker run -d --name container1 -h host1 --net mybridge --ip 192.168.10.5 --privileged centos:all /usr/sbin/init
ae98210d8dc2fb34331d47d9a64c9d71551c6792edd486d90abe0cc9022fd345
[root@nagios ~]#
[root@nagios ~]#
[root@nagios ~]# docker run -d --name container2 -h host2 --net mybridge --ip 192.168.10.9 --privileged centos:all /usr/sbin/init
4dfa52493cca33859c53d7cf599707b5acebeb9867058a413c94a1cc6df81cb7
[root@nagios ~]#
[root@nagios ~]#
[root@nagios ~]#
[root@nagios ~]# docker run -d --name container3 -h host3 --net mybridge --ip 192.168.10.20 --privileged centos:all /usr/sbin/init
1f2a2e7639fc3b300812b1452a211221322bf1b1b564b3683eece5e834f1601e
[root@nagios ~]#
```

```
[root@nagios ~]# docker inspect container1|grep IPA
            "SecondaryIPAddresses": null,
            "IPAddress": "",
                    "IPAMConfig": {
                    "IPAddress": "192.168.10.5",
[root@nagios ~]# docker inspect container2|grep IPA
            "SecondaryIPAddresses": null,
            "IPAddress": "",
                    "IPAMConfig": {
                    "IPAddress": "192.168.10.9",
[root@nagios ~]# docker inspect container3|grep IPA
            "SecondaryIPAddresses": null,
            "IPAddress": "",
                    "IPAMConfig": {
                    "IPAddress": "192.168.10.20",
[root@nagios ~]#
```

```
[root@nagios ~]# docker exec -it container3 bash
[jayesh@host3 /]$
[jayesh@host3 /]$
[jayesh@host3 /]$ ping container3
PING container3 (192.168.10.20) 56(84) bytes of data.
64 bytes from host3 (192.168.10.20): icmp_seq=1 ttl=64 time=0.028 ms
^C
--- container3 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.028/0.028/0.028/0.000 ms
[jayesh@host3 /]$ ping container2
PING container2 (192.168.10.9) 56(84) bytes of data.
64 bytes from container2.mybridge (192.168.10.9): icmp_seq=1 ttl=64 time=0.048 ms
64 bytes from container2.mybridge (192.168.10.9): icmp_seq=2 ttl=64 time=0.082 ms
^C
--- container2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1000ms
rtt min/avg/max/mdev = 0.048/0.065/0.082/0.017 ms
[jayesh@host3 /]$
[jayesh@host3 /]$
[jayesh@host3 /]$ ping container1
PING container1 (192.168.10.5) 56(84) bytes of data.
64 bytes from container1.mybridge (192.168.10.5): icmp_seq=1 ttl=64 time=0.145 ms

64 bytes from container1.mybridge (192.168.10.5): icmp_seq=2 ttl=64 time=0.096 ms
64 bytes from container1.mybridge (192.168.10.5): icmp_seq=3 ttl=64 time=0.064 ms
^C
--- container1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2001ms
rtt min/avg/max/mdev = 0.064/0.101/0.145/0.035 ms
[jayesh@host3 /]$
```

```
[root@nagios ~]# docker exec -it container2 bash
[jayesh@host2 /]$
[jayesh@host2 /]$
[jayesh@host2 /]$ ping container2
PING container2 (192.168.10.9) 56(84) bytes of data.
64 bytes from host2 (192.168.10.9): icmp_seq=1 ttl=64 time=0.035 ms
64 bytes from host2 (192.168.10.9): icmp_seq=2 ttl=64 time=0.034 ms
^C
--- container2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 0.034/0.034/0.035/0.005 ms
[jayesh@host2 /]$ ping container1
PING container1 (192.168.10.5) 56(84) bytes of data.
64 bytes from container1.mybridge (192.168.10.5): icmp_seq=1 ttl=64 time=0.109 ms
64 bytes from container1.mybridge (192.168.10.5): icmp_seq=2 ttl=64 time=0.091 ms
^C
--- container1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 0.091/0.100/0.109/0.009 ms
[jayesh@host2 /]$ ping container3
PING container3 (192.168.10.20) 56(84) bytes of data.
64 bytes from container3.mybridge (192.168.10.20): icmp_seq=1 ttl=64 time=0.065 ms
64 bytes from container3.mybridge (192.168.10.20): icmp_seq=2 ttl=64 time=0.071 ms
^C
--- container3 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1000ms
rtt min/avg/max/mdev = 0.065/0.068/0.071/0.003 ms
[jayesh@host2 /]$
```

So we can give container name and hostname same :

docker run -d --name container1 -h container1 --net mybridge --ip 192.168.10.5 --privileged centos:all /usr/sbin/init
docker run -d --name container2 -h container2 --net mybridge --ip 192.168.10.9 --privileged centos:all /usr/sbin/init
docker run -d --name container3 -h container3 --net mybridge --ip 192.168.10.20 --privileged centos:all /usr/sbin/init

```
[root@container2 /]# ssh cotainer3
ssh: Could not resolve hostname cotainer3: Name or service not known
[root@container2 /]# ssh container3
The authenticity of host 'container3 (192.168.10.20)' can't be established.
ECDSA key fingerprint is SHA256:tGM7cRvdu7NLkqC0pE4EFnW8UwIDIJZ0k/VZWHY4pd4.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'container3,192.168.10.20' (ECDSA) to the list of known hosts.
root@container3's password:
[root@container3 ~]#
[root@container3 ~]#
```

Repository or Local/Private Registry Creation

Steps for Insecure Registry

Step1 :

docker pull registry

```
[root@nagios v2]# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
registry            latest              b8604a3fe854        Less than a second ago   26.2MB
```

Step2 :

host entry :

```
[root@nagios v2]# cat /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
192.168.56.101 dockerrepo.com dockerrepo
```

Step 3.

docker insecure entry in /etc/docker/daemon.json

```
[root@nagios v2]# cat /etc/docker/daemon.json
{
  "insecure-registries" : ["dockerrepo.com:5000"]
}
```

Step 4.

mkdir /data/registry/

Step 5.

docker run -d   -p 5000:5000   --restart=always -v /data/registry/:/var/lib/registry --name insecurereg registry

Step 6.

systemctl daemon-reload

systemctl restart docker

Step 7.

http://192.168.56.101:5000/v2/_catalog in browser

Step 8.

docker tag registry:latest dockerrepo.com:5000/registry:latest

Step 9.

docker push dockerrepo.com:5000/registry:latest

Steps for Secure Registry

Step 1 :

docker pull registry

[root@nagios v2]# docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
registry latest b8604a3fe854 Less than a second ago 26.2MB

Step 2:

host entry :

[root@nagios v2]# cat /etc/hosts
127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4
::1 localhost localhost.localdomain localhost6 localhost6.localdomain6
192.168.56.101 dockerrepo.com dockerrepo

Step 3.

Generate certificate :

a. mkdir certs
b. openssl req -newkey rsa:4096 -nodes -sha256 -keyout certs/domain.key -x509 -days 365 -out certs/domain.crt
[root@nagios ~]# openssl req -newkey rsa:4096 -nodes -sha256 -keyout certs/domain.key -x509 -days 365 -out certs/domain.crt

c. Generating a 4096 bit RSA private key
.....++
.....
.....++
writing new private key to 'certs/domain.key'

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]:
State or Province Name (full name) []:
Locality Name (eg, city) [Default City]:
Organization Name (eg, company) [Default Company Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname) []:dockerrepo.com
Email Address []:

[root@nagios certs]# ll
total 8
-rw-r--r-- 1 root root 1980 Oct 22 05:12 domain.crt
-rw-r--r-- 1 root root 3272 Oct 22 05:12 domain.key

Step 4:

cd /etc/docker/

Step 5:
mkdir -p certs.d/dockerrepo.com:5000/
Step 6.
copy the domain.crt to /etc/docker/certs.d/dockerrepo.com:5000/
cp domain.crt /etc/docker/certs.d/dockerrepo.com

Step 7:
mkdir /data/registry/

Step 8:

```
docker run -d \
  --name securereg \
  -v /data/registry/:/var/lib/registry \
 -v /root/certs:/certs \
 -e REGISTRY_HTTP_TLS_CERTIFICATE=/certs/domain.crt \
 -e REGISTRY_HTTP_TLS_KEY=/certs/domain.key  \
 -p 5000:5000 \
 registry
```

```
[root@nagios certs]# docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS                    NAMES
f29b87ac26a4        registry            "/entrypoint.sh /etcâ€¦"   3 seconds ago       Up 2 seconds        0.0.0.0:5000->5000/tcp   securereg
```

Step 9:
systemctl daemon-reload
systemctl restart docker

Step 10:
http://192.168.56.101:5000/v2/_catalog in browser

Step 11:

```
[root@nagios certs]# docker images
REPOSITORY                    TAG                 IMAGE ID            CREATED             SIZE
dockerrepo.com:5000/httpd     v2                  ad17c88403e2        Less than a second ago   143MB
httpd                         2                   ad17c88403e2        Less than a second ago   143MB
nginx                         latest              ea335eea17ab        Less than a second ago   141MB
registry                      2                   b8604a3fe854        Less than a second ago   26.2MB
registry                      latest              b8604a3fe854        Less than a second ago   26.2MB
dockerrepo.com:5000/registry  latest              b8604a3fe854        Less than a second ago   26.2MB
ubuntu                        latest              ba6acccedd29        5 days ago          72.8MB
[root@nagios certs]# docker tag ubuntu:latest dockerrepo.com:5000/ubuntu:latest
[root@nagios certs]# docker push dockerrepo.com:5000/ubuntu:latest

[root@nagios certs]# docker push  dockerrepo.com:5000/ubuntu:latest
The push refers to repository [dockerrepo.com:5000/ubuntu]
9f54eef41275: Pushed
latest: digest: sha256:7cc0576c7c0ec2384de5cbf245f41567e922aab1b075f3e8ad565f508032df17 size: 529
```

Step 11:

```
[root@nagios certs]# cd /data/registry/docker/registry/v2/repositories/
[root@nagios repositories]# ls
httpd  myfirstimage  registry  ubuntu
```

Steps for AuthBased Registry

Step 1.

```
docker pull registry
[root@nagios v2]# docker images
REPOSITORY          TAG             IMAGE ID        CREATED              SIZE
registry            latest          b8604a3fe854    Less than a second ago   26.2MB
```

Step 2.

host entry :

```
[root@nagios v2]# cat /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
192.168.56.101 dockerrepo.com dockerrepo
```

Step 3.

generate user credetials :

```
docker container run --entrypoint htpasswd registry:2.7.0 -bnB jayesh J@yesh2711 > /root/auth/htpasswd
```

Step 4. cd /root/auth

Step 5. mkdir /data/registry/

Step 6.

```
docker run -d \
  --name authreg \
  -v /data/registry/:/var/lib/registry \
  -v /root/auth:/auth \
  -v /root/certs:/certs \
  -e "REGISTRY_AUTH=htpasswd" \
  -e "REGISTRY_AUTH_HTPASSWD_REALM=Registry Realm" \
  -e REGISTRY_AUTH_HTPASSWD_PATH=/auth/htpasswd \
  -e REGISTRY_HTTP_TLS_CERTIFICATE=/certs/domain.crt \
  -e REGISTRY_HTTP_TLS_KEY=/certs/domain.key  \
  -p 5000:5000 \
  registry
```

```
[root@nagios certs]# docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS                    NAMES
f29b87ac26a4        registry            "/entrypoint.sh /etcâ€¦"   3 seconds ago       Up 2 seconds        0.0.0.0:5000->5000/tcp   securereg
```

Step 7. systemctl daemon-reload

```
  systemctl restart docker
```

Step 8. http://192.168.56.101:5000/v2/_catalog in browser

Step 9.

```
[root@nagios auth]# docker images
REPOSITORY                    TAG             IMAGE ID       CREATED              SIZE
httpd                         2               ad17c88403e2   Less than a second ago   143MB
dockerrepo.com:5000/httpd     v2              ad17c88403e2   Less than a second ago   143MB
nginx                         latest          ea335eea17ab   Less than a second ago   141MB
registry                      2               b8604a3fe854   Less than a second ago   26.2MB
registry                      latest          b8604a3fe854   Less than a second ago   26.2MB
dockerrepo.com:5000/registry  latest          b8604a3fe854   Less than a second ago   26.2MB
ubuntu                        latest          ba6acccedd29   6 days ago           72.8MB
dockerrepo.com:5000/ubuntu    latest          ba6acccedd29   6 days ago           72.8MB
registry                      2.7.0           33fbbf4a24e5   2 years ago          24.2MB
```

Step 10.

```
[root@nagios certs]# docker tag registry:2.7.0 dockerrepo.com:5000/registry:2.7.0
```

Step 11.

```
[root@nagios auth]# docker login dockerrepo.com:5000
Authenticating with existing credentials...
Login did not succeed, error: Error response from daemon: login attempt to https://dockerrepo.com:5000/v2/ failed with status: 401 Unauthorized
Username (jayesh): jayesh
Password:
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

Step 12.

```
[root@nagios auth]# docker push dockerrepo.com:5000/registry:2.7.0
The push refers to repository [dockerrepo.com:5000/registry]
e0ce598fab8f: Pushed
85384d42542a: Pushed
d2e29d15f6a6: Pushed
6b6e0aba7201: Pushed
7bff100f35cb: Pushed
2.7.0: digest: sha256:d6df97c93a0b8db7b355e08d54fbf38e46667eaa251cd5a90ddf0e53c35375b1 size: 1363
```

Step 13.

```
[root@nagios certs]# cd /data/registry/docker/registry/v2/repositories/
[root@nagios repositories]# ls
httpd  myfirstimage  registry  ubuntu
```