Volume In Docker

Docker volume is to share data between the container and the host so that if something happens to the container, we can launch another container and attach the same volume to it. It is used to share data between multiple containers at the same time.

For example: let's say you're a file to copy between 200 containers, you can just create a volume and share between those 200 containers

NB: We can mount volume only on new launch containers and not running containers

Docker volume = mount point in Linux = share file or network file in windows

The home directory of docker where docker stored everything

```
find / -name docker
/var/lib/docker
cd /var/lib/docker
ll
```

Type of volume in Docker

We have two types of volume in docker

- **Volume mount:** This mounts the volume from the volume directory in docker host in /var/lib/docker/volumes/.
- **Bind mount:** mount a directory from any location on docker host such as HDD or in the directory on the HDD or mount a directory from any location such as HDD or in the directory on the HDD

Difference between volumes mount and bind mount

Volumes mount are managed by docker daemon because they are mounted from the default docker home `/var/lib/docker` while bind mount is managed by the file system of the server, and it can be mounted anywhere on the docker host.

Difference between docker volumes docker copy?

Docker volume will share data from host to container and container to host between multiplier containers while **docker copy** is to copy a file from host to container and container to host and it can't be shared between multiple containers.

Pros of docker volumes

- Data will be consistent on the host even if the container is deleter
- The same volume can be attached to multiple containers at the same time
- Data will be in sync
- Data will be data sharing between the host and container

Create a volume

```
docker volume create <volume name>

docker volume create httpd_volume
docker volume create nginx_volume
docker volume create tomcat_volume
```

List the volume

```
docker volume ls
```

Delete a volume

```
docker volume rm <volume name>
docker volume rm httpd
```

Check the volumes created on a docker host

NB: _data directory is where we put the content that we want to share between multiple containers

```
ll /var/lib/docker/volumes/
/var/lib/docker/volumes/httpd_volume/_data
/var/lib/docker/volumes/nginx_volume/_data
/var/lib/docker/volumes/tomcat_volume/_data
```

Web content location in the container for HTTPD, NGINX, AND TOMCAT

```
HTTPD: /usr/local/apache2/htdocs/
NGINX: /usr/share/nginx/html/
TOMCAT: /usr/local/tomcat/webapps/
```

Create web content for NGINX and mount the container with a volume mount

1. Create web content for Nginx

```
vim /var/lib/docker/volumes/nginx_volume/_data/index.html
ls /var/lib/docker/volumes/nginx_volume/_data/
cat /var/lib/docker/volumes/nginx_volume/_data/index.html

<!DOCTYPE html>
<html>
<body style="background-color:rgb(217, 250, 210);">

<h1>Welcome to DevOps Easy Learning</h1>
<h3>This is to test Nginx Web server at devopseasylearning.com</h3>

</body>
</html>
```

1. Run the container and mount the volume for NGINX

v = volume

```
docker run -d --name my-nginx -p 8015:80 -v nginx_volume:/usr/share
/nginx/html nginx:latest
```

1. Test it

```
http://<IP>:8015/
http://10.0.0.94:8015/
```

1. Modify the content and refresh the browser

```
vim /var/lib/docker/volumes/nginx_volume/_data/index.html

<!DOCTYPE html>
<html>
<body style="background-color:rgb(217, 250, 210);">

<h1>Welcome to DevOps Easy Learning</h1>
<h3>This is to test Nginx Web server at devopseasylearning.com with TIA
and some awesome students</h3>

</body>
</html>
```

Create a web content for HTTPD OR APACHE and mount the container with volume mount

1. Create a web content for for httpd

```
cd /var/lib/docker/volumes/httpd_volume/_data/
git clone https://github.com/tia12/static-website-example.git
cp -R static-website-example/* .
rm -rf static-website-example
```

1. Run the container and mount the volume for httpd

v = volume

```
docker run -d --name my-httpd -p 8020:80 -v httpd_volume:/usr/local
/apache2/htdocs/ httpd:latest
```

1. Test it

```
http://<IP>:8020/
http://10.0.0.94:8020/
```

1. Modify the content and refresh the browser

```
/var/lib/docker/volumes/httpd_volume/_data/index.html
```

Create a web content for HTTPD OR APACHE and mount the container with volume mount

1. Create a web content for Tomcat

```
cd /var/lib/docker/volumes/tomcat_volume/_data/
wget https://linux-devops-course.s3.amazonaws.com/warfiles
/happy_new_year.war
```

1. Run the container and mount the volume for httpd

v = volume

```
docker run -d --name my-tomcat -p 8030:8080 -v tomcat_volume:/usr/local
/tomcat/webapps/ tomcat:latest
```

1. Test it

```
firewall-cmd --permanent --add-port=8080/tcp
firewall-cmd --reload

http://<IP>:8030/happy_new_year
http://10.0.0.94:8030/happy_new_year
```

Volume mount using the mount instead of -v:

```
docker volume create tomcat2_volume
cd /var/lib/docker/volumes/tomcat2_volume/_data/
wget https://linux-devops-course.s3.amazonaws.com/warfiles/addressbook.
war

docker run -d --name my-tomcat2 -p 8050:8080 --mount type=volume,
source=tomcat2_volume,target=/usr/local/tomcat/webapps/ tomcat:latest

http://<IP>:8050/addressbook
http://10.0.0.94:8050/addressbook
```

Create a web content for HTTPD OR APACHE and mount the container with bind mount

1. Create a web content for for httpd

```
mkdir /covid-19

cd /covid-19
git clone https://github.com/tia12/covid19.git
cp -R covid19/* .
rm -rf covid19
```

1. Run the container and mount the volume for httpd

v = volume

```
docker run -d --name covid19 -p 8040:80 --mount type=bind,source=/covid-
19,target=/usr/local/apache2/htdocs/ httpd:latest
```

1. Test it

```
http://<IP>:8040/
http://10.0.0.94:8040/
```

1. Modify the content and refresh the browser

```
cd /covid-19
vim index.html
```

```
wget https://linux-devops-course.s3.amazonaws.com/WEB+SIDE+HTML/covid19.
zip
unzip covid19.zip
cp -R covid19/* .
rm -rf covid19.zip
rm -rf covid19
```

```
HTTPD: /usr/local/apache2/htdocs/
NGINX: /usr/share/nginx/html/
TOMCAT: /usr/local/tomcat/webapps/
```

```
docker run -it -d -p 8084:80 --name httpd_tia -v "$PWD":"/usr/local
/apache2/htdocs/" --workdir "/usr/local/apache2/htdocs/" httpd


docker run -it -d \
       -p 8083:80 \
       --name httpd_tia \
       -v "$PWD":"/usr/local/apache2/htdocs/" \
       --workdir "/usr/local/apache2/htdocs/" \
       httpd

docker run -it --rm -v "$PWD":"/code" --workdir "/code" ubuntu

docker run -it --rm \
  -v "$PWD":"/code" \
  --workdir "/code" \
  ubuntu


docker run --rm \
       -v "$PWD":"/code" \
```

```
       --workdir "/code" \
       ubuntu \
          bash script.sh

docker run --rm \
  -v "$PWD":"/code" \
  --workdir "/code" \
  ubuntu bash script.sh


docker run --rm \
       -v "$PWD":"/code" \
       --workdir "/code" \
       ubuntu bash script.sh


docker run --rm \
       -v "$PWD":"/code" \
       --workdir "/code" \
       ubuntu cat /etc/*release


docker run --rm \
       -v "$PWD":"/python" \
       --workdir "/python" \
       python \
           python python.py


docker run -it --rm \
       -v "$PWD":"/awscli" \
       --workdir "/awscli" \
       organs/awscli


## We will have a project on this for interview
docker run -it --rm \
       -v "$PWD":"/code" \
       --workdir "/code" \
       container-here

docker run -it --rm \
       --name some-postgres2 \
       -e POSTGRES_PASSWORD=12345 \
       -d postgres \
       postgres bash
```