Experiment-2: Volumes

Volume is simply a directory inside our container. Firstly, We have to declare this as a volume and then share Volume.

Even if we stop container, still we can access volume. Volume will be created in one container. We can declare a directory as a volume only while creating Container. You can’t Create Volume from existing container. You can share one volume across any number of container. Volume wil not be included when you update an image and make a new container of that image. Volume can be mapped container to container and Host to container and vice-versa.

**Benefits:**

* Decoupling Cintainers from storage.
* Share Volume among different Containers.
* Attach Volume to Containers.
* On deleting containers volume does not delete.

**Lets create a volume using Dockerfile.**

Step-1

Create dockerfile and type below command

Vi Dockerfile //to open dockerfile

In dockerfile use below command:

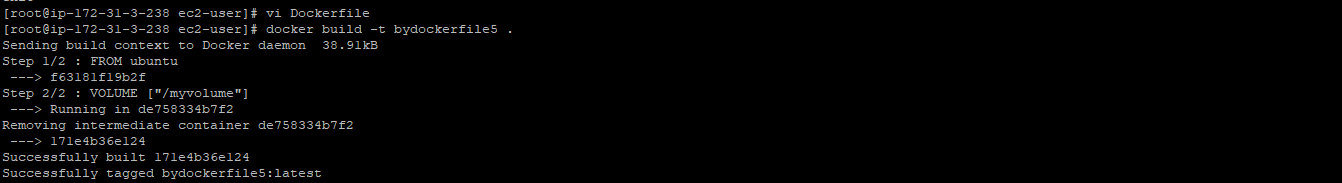
FROM ubuntu

VOLUME ["/myvolume"]

Now to execute this dockerfile to make an image of your own use command.

docker build –t <image\_name>

docker build –t bydockerfile5



Step-2:

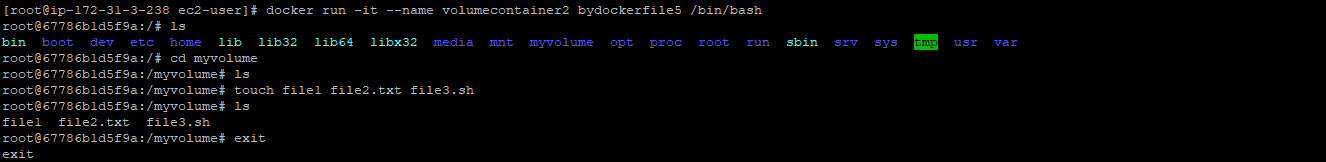
Create a container of the above image we created.

Command:

docker run -it –name <container\_name> <image\_name> /bin/bash

docker run -it –name volumecontainer2 bydockerfile5 /bin/bash

After this you will be redirect to your bash of the container and you will see your volume directory with a name myvolume. Make some files like file.txt, file.sh, file inside myvolume container and do exit from that container.

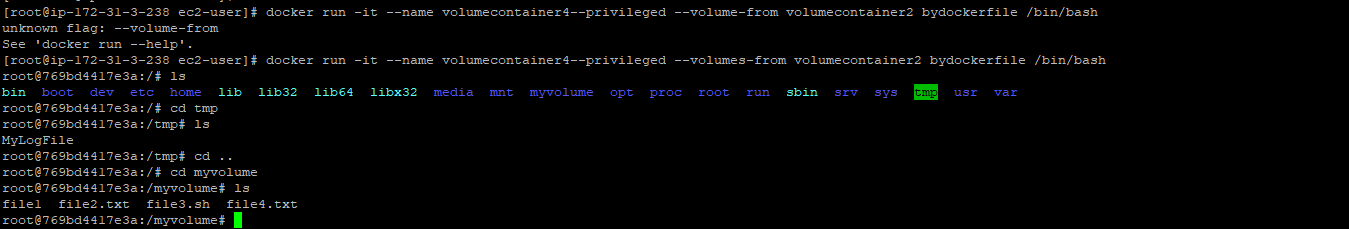


Step-3: Now we have to create a new container which we can have of any image while giving the access of the volume directory.

Command:

docker run -it --name <new\_container\_name> --privileged=true --volume-from <container\_consist\_volume\_Directory> <image\_name\_any> /bin/bash

docker run -it --name volumecontainer4 –privileged=true --volume-from volumecontainer2 bydockerfile /bin/bash



Now, since we can create n number of containers giving access of volume directory.

If any container create a file in “myvolume” directory will reflect changes in all the containers having volume directory privileged.

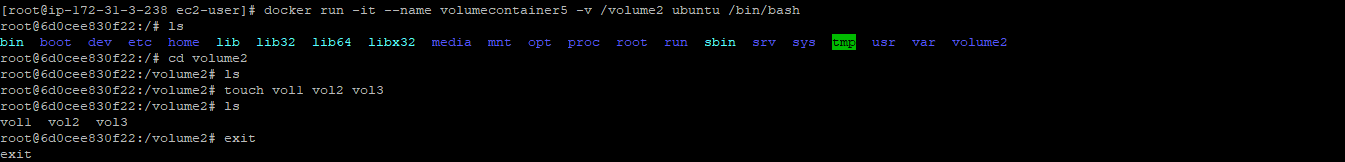
**Let’s create a volume using command line**

Step-1:

Commad:

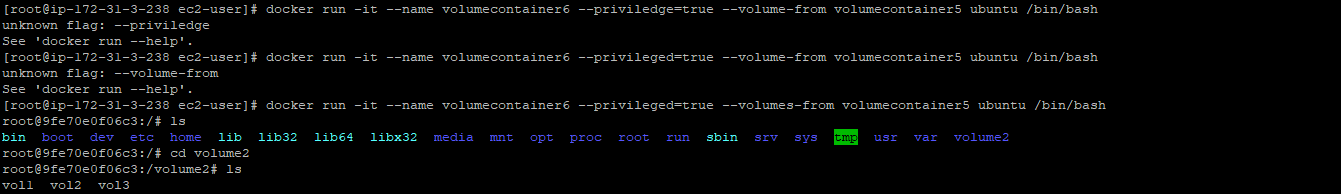
docker run -it --name volumecontainer5 -v /volume2 ubuntu /bin/bash

-v is used to declare a volume directory.



Step-2: Create another container on any image and give volume privileged.

Command: docker run -it --name volumecontainer6 --privileged=true --volumes-from volumecontainer5 ubuntu /bin/bash



**How to share Volume Between host and container**

Step-1:

Check your present directory using ‘pwd’ command. Create some file, if already present no need to make.

/home/ec2-user

Command to create file: touch file1 file2.txt

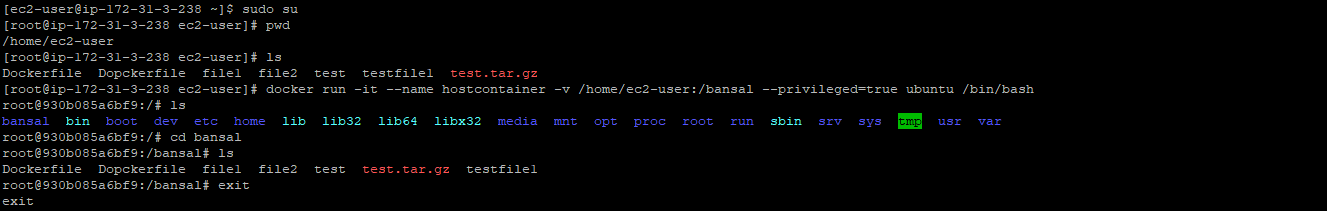
We can map our container and host volume while creating a container using ‘:’ symbol and ‘-v’ is used to define a volume we will create.

Command:

docker run -it --name <container\_name> -v /home/ec2-user:/<volume name> --privileged=true <image\_name> /bin/bash

docker run -it --name hostcontainer -v /home/ec2-user:/bansal --privileged=true ubuntu /bin/bash

We will see all the file present in our host is mapped to a container volume named ‘bansal’.



Step-2:

To verify exit the container again start the container and add some files in a volume directory and exit the container. Check the update pwd of the host.

