Lab 5 – Data Persistence with Volumes

June 2019

Document Control

Version	1.0
Created by	Priyanka Gawada
Updated by	
Reviewed by	Chand Ali
Date	June 2019
Earlier versions	

Table of Contents

1. Overview

In this lab, we will cover volumes that are preferred mechanism for persisting data generated by and used by Docker containers. Docker provides three options - bind mounts, volumes and in memory option called tmpfs, as in temporary file system

1.1. Lab Environment

- Environment will look as follows:

1.2. Lab Objectives

- Create new volumes
- Create a Container without Volume and examine it
- Attach Volume to a Container
- Other Volume Commands
- Sharing Data Between Containers

2. Create new Volumes

- Step 1: Creating an independent volume.

```
$ docker create volume --name DataVolume1
```

```
[root@ip-172-31-23-19 ec2-user]# docker volume create --name DataVolume1
DataVolume1
[root@ip-172-31-23-19 ec2-user]#
```

- Step 2: To make use of volumes, we will create a new container from the Ubuntu image, using the –-rm flag to automatically delete it when we exit.

```
$ docker run –-name container1 –t1 –rm –v DataVolume1:/datavolume1 ubuntu
```

```
[root@ip-172-31-23-19 ec2-user]# docker run --name container1 -ti --rm -v DataVolume1:/datavolume1 ubuntu
Unable to find image 'ubuntu:latest' locally
latest: Pulling from library/ubuntu
124c757242f8: Pull complete
2ebc019eb4e2: Pull complete
dac0825f7ffb: Pull complete
82b0bb65d1bf: Pull complete
ef3b655c7f88: Pull complete
Digest: sha256:72f832c6184b55569be1cd9043e4a80055d55873417ea792d989441f207dd2c7
Status: Downloaded newer image for ubuntu:latest
root@4a4907ebb5bc:/#
```

Here, -v is used to mount the volume

-v requires the name of the volume, a colon, and then the absolute path to where the volume should appear inside the container.

Note: If the directories in the path don't exist as part of the image, they'll be created when the command runs. If they do exist, the mounted volume will hide the existing content.

- Step 3: Create a file and write something while being in the container.

```
$ docker  run –name container1 –t1 –name –v  DataVolume1:/datavolume1 ubuntu
```

```
[root@ip-172-31-23-19 ec2-user]# docker run --name container1 -ti --rm -v DataVolume1:/datavolume1 ubuntu
Unable to find image 'ubuntu:latest' locally
latest: Pulling from library/ubuntu
124c757242f8: Pull complete
2ebc019eb4e2: Pull complete
dac0825f7ffb: Pull complete
82b0bb65d1bf: Pull complete
ef3b655c7f88: Pull complete
Digest: sha256:72f832c6184b55569be1cd9043e4a80055d55873417ea792d989441f207dd2c7
Status: Downloaded newer image for ubuntu:latest
root@4a4907ebb5bc:/# echo "Data written from Container 1" > /datavolume1/example1.txt
root@4a4907ebb5bc:/# cat /datavolume1/example.txt
cat: /datavolume1/example.txt: No such file or directory
root@4a4907ebb5bc:/# cat example.txt
cat: example.txt: No such file or directory
root@4a4907ebb5bc:/# ls -l
total 68
drwxr-xr-x  2 root root 4096 Aug 21 21:14 bin
drwxr-xr-x  2 root root 4096 Apr 24 08:34 boot
drwxr-xr-x  2 root root 4096 Sep  3 21:23 datavolume1
drwxr-xr-x  5 root root  360 Sep  3 21:17 dev
drwxr-xr-x  1 root root 4096 Sep  3 21:17 etc
drwxr-xr-x  2 root root 4096 Apr 24 08:34 home
drwxr-xr-x  8 root root 4096 Aug 21 21:12 lib
drwxr-xr-x  2 root root 4096 Aug 21 21:13 lib64
drwxr-xr-x  2 root root 4096 Aug 21 21:12 media
drwxr-xr-x  2 root root 4096 Aug 21 21:12 mnt
drwxr-xr-x  2 root root 4096 Aug 21 21:12 opt
dr-xr-xr-x 98 root root    0 Sep  3 21:17 proc
drwx------  2 root root 4096 Aug 21 21:14 root
drwxr-xr-x  1 root root 4096 Aug 22 17:28 run
drwxr-xr-x  1 root root 4096 Aug 22 17:28 sbin
drwxr-xr-x  2 root root 4096 Aug 21 21:12 srv
dr-xr-xr-x 13 root root    0 Sep  3 21:17 sys
drwxrwxrwt  2 root root 4096 Aug 21 21:14 tmp
```

```
drwxr-xr-x  1 root root 4096 Aug 21 21:12 usr
drwxr-xr-x  1 root root 4096 Aug 21 21:14 var
root@4a4907ebb5bc:/# vmi datavolume1/example.txt
bash: vmi: command not found
root@4a4907ebb5bc:/# cat  datavolume1/example.txt
cat: datavolume1/example.txt: No such file or directory
root@4a4907ebb5bc:/# cat  datavolume1/example1.txt
Data written from Container 1
root@4a4907ebb5bc:/# ls -l datacolume1
ls: cannot access 'datacolume1': No such file or directory
root@4a4907ebb5bc:/# ls -l datavolume1
total 4
-rw-r--r-- 1 root root 30 Sep  3 21:23 example1.txt
root@4a4907ebb5bc:/# exit
exit
[root@ip-172-31-23-19 ec2-user]#
```

- Step 4: After the running container is exited, where the data is physically stored.

```
$ ls –l  /var/lib/docker/volumes
```

```
drwxr-xr-x 3 root root  4096 Sep  3 21:11 DataVolume1
drwxr-xr-x 3 root root  4096 Aug 24 12:18 e6976db7474cff59a7f6e18092d6dda7792ba646f5d0e83b7762842fdab4f
```

Now, you can go inside this physical folder to view the created file **example1.txt.**

```
$ ls –l /var/lib/docker/volumes/DataVolume1
```

```
[root@ip-172-31-23-19 ec2-user]# ls -l /var/lib/docker/volumes/DataVolume1
total 4
drwxr-xr-x 2 root root 4096 Sep  3 21:23 _data
[root@ip-172-31-23-19 ec2-user]# ls -l /var/lib/docker/volumes/DataVolume1/_data
total 4
-rw-r--r-- 1 root root 30 Sep  3 21:23 example1.txt
[root@ip-172-31-23-19 ec2-user]# ls -l /var/lib/docker/volumes/DataVolume1
```

Note: On deleting container, the volume is not deleted, and you can inspect it.

- Step 5: Inspect the volume.

```
$ docker volume inspect DataVolume1
```

```
[root@ip-172-31-23-19 ec2-user]# docker volume inspect DataVolume1
[
    {
        "CreatedAt": "2018-09-03T21:23:07Z",
        "Driver": "local",
        "Labels": {},
        "Mountpoint": "/var/lib/docker/volumes/DataVolume1/_data",
        "Name": "DataVolume1",
        "Options": {},
        "Scope": "local"
    }
]
[root@ip-172-31-23-19 ec2-user]#
```

3. Create Container without Volume and Examine it

- Create a container without volume and add a file in it. Check for its persistence on physical file system.

```
$ docker run –-name withoutvolumecontainer1 –ti –-rm ubuntu
```

```
[root@ip-172-31-23-19 ec2-user]# docker run --name withoutvolumecontainer1 -ti --rm ubuntu
root@80178465be27:/#
root@80178465be27:/# echo "Hello from non-volume container" > example2.txt
root@80178465be27:/# ls -l
total 68
drwxr-xr-x  2 root root 4096 Aug 21 21:14 bin
drwxr-xr-x  2 root root 4096 Apr 24 08:34 boot
drwxr-xr-x  5 root root  360 Sep  3 21:42 dev
-rw-r--r--  1 root root   32 Sep  3 21:42 example2.txt
drwxr-xr-x  8 root root 4096 Aug 21 21:12 lib
drwxr-xr-x  2 root root 4096 Aug 21 21:13 lib64
drwxr-xr-x  2 root root 4096 Aug 21 21:12 media
drwxr-xr-x  2 root root 4096 Aug 21 21:12 mnt
drwxr-xr-x  2 root root 4096 Aug 21 21:12 opt
dr-xr-xr-x 98 root root    0 Sep  3 21:42 proc
drwx------  2 root root 4096 Aug 21 21:14 root
drwxr-xr-x  1 root root 4096 Aug 22 17:28 run
drwxr-xr-x  1 root root 4096 Aug 22 17:28 sbin
drwxr-xr-x  2 root root 4096 Aug 21 21:12 srv
dr-xr-xr-x 13 root root    0 Sep  3 21:26 sys
drwxrwxrwt  2 root root 4096 Aug 21 21:14 tmp
drwxr-xr-x  1 root root 4096 Aug 21 21:12 usr
drwxr-xr-x  1 root root 4096 Aug 21 21:14 var
root@80178465be27:/#
```

If we do cat **example2.txt**, then we can view its contents.

```
root@80178465be27:/# cat example2.txt
Hello from non-volume container
root@80178465be27:/#
```

- Now, exit from container, type # **exit**
- Now, check the running container docker **ps –a**
- Here, you will not view the container running as it is deleted.
- Now, once the container is deleted, the data persisted in it will also be deleted. Thus, on doing

```
$ ls -l /var/lib/docker/volumes, it will not show example2.txt
```

Note, if that container was stopped and restarted, the files will exists in it. However, on removing container, the files are deleted.

```
[root@ip-172-31-23-19 ec2-user]# docker run --name withoutvolumecontainer1 -ti  ubuntu
root@0eaf40a30e3b:/# echo "hello" > test.txt
root@0eaf40a30e3b:/# ls -l
total 68
drwxr-xr-x  2 root root 4096 Aug 21 21:14 bin
drwxr-xr-x  2 root root 4096 Apr 24 08:34 boot
drwxr-xr-x  5 root root  360 Sep  3 21:57 dev
drwxr-xr-x  1 root root 4096 Sep  3 21:57 etc
drwxr-xr-x  2 root root 4096 Apr 24 08:34 home
drwxr-xr-x  8 root root 4096 Aug 21 21:12 lib
drwxr-xr-x  2 root root 4096 Aug 21 21:13 lib64
drwxr-xr-x  2 root root 4096 Aug 21 21:12 media
drwxr-xr-x  2 root root 4096 Aug 21 21:12 mnt
drwxr-xr-x  2 root root 4096 Aug 21 21:12 opt
dr-xr-xr-x 98 root root    0 Sep  3 21:57 proc
drwx------  2 root root 4096 Aug 21 21:14 root
drwxr-xr-x  1 root root 4096 Aug 22 17:28 run
drwxr-xr-x  1 root root 4096 Aug 22 17:28 sbin
drwxr-xr-x  2 root root 4096 Aug 21 21:12 srv
dr-xr-xr-x 13 root root    0 Sep  3 21:26 sys
-rw-r--r--  1 root root    6 Sep  3 21:57 test.txt
drwxrwxrwt  2 root root 4096 Aug 21 21:14 tmp
drwxr-xr-x  1 root root 4096 Aug 21 21:12 usr
drwxr-xr-x  1 root root 4096 Aug 21 21:14 var
root@0eaf40a30e3b:/# cat test.txt
hello
```

Now, stop and start the container and go into the shell.

```
[root@ip-172-31-23-19 ec2-user]# docker container stop  withoutvolumecontainer1
withoutvolumecontainer1
```

```
[root@ip-172-31-23-19 ec2-user]# docker container start  withoutvolumecontainer1
withoutvolumecontainer1
```

```
[root@ip-172-31-23-19 ec2-user]# docker exec -it  withoutvolumecontainer1 /bin/bash
root@0eaf40a30e3b:/# ls -l
total 68
drwxr-xr-x   2 root root 4096 Aug 21 21:14 bin
drwxr-xr-x   2 root root 4096 Apr 24 08:34 boot
drwxr-xr-x   5 root root  360 Sep  3 21:59 dev
drwxr-xr-x   1 root root 4096 Sep  3 21:57 etc
drwxr-xr-x   2 root root 4096 Apr 24 08:34 home
drwxr-xr-x   8 root root 4096 Aug 21 21:12 lib
drwxr-xr-x   2 root root 4096 Aug 21 21:13 lib64
drwxr-xr-x   2 root root 4096 Aug 21 21:12 media
drwxr-xr-x   2 root root 4096 Aug 21 21:12 mnt
drwxr-xr-x   2 root root 4096 Aug 21 21:12 opt
dr-xr-xr-x 101 root root    0 Sep  3 21:59 proc
drwx------   1 root root 4096 Sep  3 21:58 root
drwxr-xr-x   1 root root 4096 Aug 22 17:28 run
drwxr-xr-x   1 root root 4096 Aug 22 17:28 sbin
drwxr-xr-x   2 root root 4096 Aug 21 21:12 srv
dr-xr-xr-x  13 root root    0 Sep  3 21:26 sys
-rw-r--r--   1 root root    6 Sep  3 21:57 test.txt
drwxrwxrwt   2 root root 4096 Aug 21 21:14 tmp
drwxr-xr-x   1 root root 4096 Aug 21 21:12 usr
drwxr-xr-x   1 root root 4096 Aug 21 21:14 var
root@0eaf40a30e3b:/#
```

4. Attach Volume to a Container

- Exercise 3: Create a new container and attach DataVolume1 to it.

```
[root@ip-172-31-23-19 ec2-user]#
[root@ip-172-31-23-19 ec2-user]# docker run --name container2 --rm -ti -v DataVolume1:/datavolume1 ubuntu
root@0ceb9723d51a:/# ls -l
total 68
drwxr-xr-x  2 root root 4096 Aug 21 21:14 bin
drwxr-xr-x  2 root root 4096 Apr 24 08:34 boot
drwxr-xr-x  2 root root 4096 Sep  3 21:23 datavolume1
drwxr-xr-x  5 root root  360 Sep  3 22:10 dev
drwxr-xr-x  1 root root 4096 Sep  3 22:10 etc
drwxr-xr-x  2 root root 4096 Apr 24 08:34 home
drwxr-xr-x  8 root root 4096 Aug 21 21:12 lib
drwxr-xr-x  2 root root 4096 Aug 21 21:13 lib64
drwxr-xr-x  2 root root 4096 Aug 21 21:12 media
drwxr-xr-x  2 root root 4096 Aug 21 21:12 mnt
drwxr-xr-x  2 root root 4096 Aug 21 21:12 opt
dr-xr-xr-x 98 root root    0 Sep  3 22:10 proc
drwx------  2 root root 4096 Aug 21 21:14 root
drwxr-xr-x  1 root root 4096 Aug 22 17:28 run
drwxr-xr-x  1 root root 4096 Aug 22 17:28 sbin
drwxr-xr-x  2 root root 4096 Aug 21 21:12 srv
dr-xr-xr-x 13 root root    0 Sep  3 21:26 sys
drwxrwxrwt  2 root root 4096 Aug 21 21:14 tmp
drwxr-xr-x  1 root root 4096 Aug 21 21:12 usr
drwxr-xr-x  1 root root 4096 Aug 21 21:14 var
root@0ceb9723d51a:/# ls -l datavolume1
total 4
-rw-r--r-- 1 root root 30 Sep  3 21:23 example1.txt
root@0ceb9723d51a:/# cat datavolume1/example1.txt
Data written from Container 1
root@0ceb9723d51a:/# echo "Data written from Container 2" > datavolume1/example2.txt
root@0ceb9723d51a:/# cat datavolume1/example2.txt
Data written from Container 2
root@0ceb9723d51a:/# exit
exit
[root@ip-172-31-23-19 ec2-user]#
```

View the created files in the volume after the container is deleted.

```
drwxr-xr-x 2 root root 4096 Sep  3 22:11 _data
[root@ip-172-31-23-19 ec2-user]# ls -l /var/lib/docker/volumes/DataVolume1/_data
total 8
-rw-r--r-- 1 root root 30 Sep  3 21:23 example1.txt
-rw-r--r-- 1 root root 30 Sep  3 22:11 example2.txt
[root@ip-172-31-23-19 ec2-user]#
```

Note: If no container is attached with the volume, then it can be deleted. However, if a single container is connected to the volume, then deleting it is not allowed.

5. Other Volume Commands

- To remove volume: **$ docker volume rm DataVolume1**
- To list volumes: **$ docker volume ls**
- Exercise 4: Creating a volume from an existing directory with Data

```
$ docker run --name container3 -ti --rm -v DataVolume3:/var Ubuntu
```

```
[root@ip-172-31-23-19 ec2-user]# docker run --name container3 -ti --rm -v DataVolume3:/var ubuntu
root@cbcabfa56b78:/# ls -l
total 64
drwxr-xr-x  2 root root 4096 Aug 21 21:14 bin
drwxr-xr-x  2 root root 4096 Apr 24 08:34 boot
drwxr-xr-x  5 root root  360 Sep  3 22:53 dev
drwxr-xr-x  1 root root 4096 Sep  3 22:53 etc
drwxr-xr-x  2 root root 4096 Apr 24 08:34 home
drwxr-xr-x  8 root root 4096 Aug 21 21:12 lib
drwxr-xr-x  2 root root 4096 Aug 21 21:13 lib64
drwxr-xr-x  2 root root 4096 Aug 21 21:12 media
drwxr-xr-x  2 root root 4096 Aug 21 21:12 mnt
drwxr-xr-x  2 root root 4096 Aug 21 21:12 opt
dr-xr-xr-x 95 root root    0 Sep  3 22:53 proc
drwx------  2 root root 4096 Aug 21 21:14 root
drwxr-xr-x  1 root root 4096 Aug 22 17:28 run
drwxr-xr-x  1 root root 4096 Aug 22 17:28 sbin
drwxr-xr-x  2 root root 4096 Aug 21 21:12 srv
dr-xr-xr-x 13 root root    0 Sep  3 21:26 sys
drwxrwxrwt  2 root root 4096 Aug 21 21:14 tmp
drwxr-xr-x  1 root root 4096 Aug 21 21:12 usr
drwxr-xr-x 11 root root 4096 Sep  3 22:53 var
root@cbcabfa56b78:/# ls -l var
total 36
drwxr-xr-x 2 root root  4096 Apr 24 08:34 backups
drwxr-xr-x 5 root root  4096 Aug 21 21:14 cache
drwxr-xr-x 7 root root  4096 Aug 21 21:12 lib
drwxrwsr-x 2 root staff 4096 Apr 24 08:34 local
lrwxrwxrwx 1 root root     9 Aug 21 21:12 lock -> /run/lock
drwxr-xr-x 3 root root  4096 Aug 21 21:12 log
drwxrwsr-x 2 root mail  4096 Aug 21 21:12 mail
drwxr-xr-x 2 root root  4096 Aug 21 21:12 opt
lrwxrwxrwx 1 root root     4 Aug 21 21:12 run -> /run
drwxr-xr-x 2 root root  4096 Aug 21 21:12 spool
drwxrwxrwt 2 root root  4096 Aug 21 21:14 tmp
root@cbcabfa56b78:/# exit
```

Create another container and bind with the same volume

```
[root@ip-172-31-23-19 ec2-user]# docker run --name container4  --rm -v DataVolume3:/datavolume3 ubuntu ls
datavolume3
backups
cache
lib
local
lock
log
mail
opt
run
spool
tmp
[root@ip-172-31-23-19 ec2-user]#
```

Here, the directory datavolume3 now has a copy of the contents of the base image's /var directory.

6. Sharing Data Between Containers

Often, we'll want multiple containers to attach to the same data volume. Docker doesn't handle file locking. If you need multiple containers writing to the volume, the applications running in those containers must be designed to write to shared data stores in order to prevent data corruption.

- Step 1: Create container and attach it with volume.

```
$ docker run -ti --name container1 -v  DataVolume4:/datavolume4  ubuntu
```

```
[root@ip-172-31-23-19 ec2-user]# docker run -ti --name container1 -v DataVolume4:/datavolume4 ubuntu
root@f1f90ee443e8:/# echo "This file is shared between two containers - 1" > /datavolume4/fileIO.txt
root@f1f90ee443e8:/# ls -l
total 68
drwxr-xr-x  2 root root 4096 Aug 21 21:14 bin
drwxr-xr-x  2 root root 4096 Apr 24 08:34 boot
drwxr-xr-x  2 root root 4096 Sep  3 23:14 datavolume4
drwxr-xr-x  5 root root  360 Sep  3 23:13 dev
drwxr-xr-x  1 root root 4096 Sep  3 23:13 etc
drwxr-xr-x  2 root root 4096 Apr 24 08:34 home
drwxr-xr-x  8 root root 4096 Aug 21 21:12 lib
drwxr-xr-x  2 root root 4096 Aug 21 21:13 lib64
drwxr-xr-x  2 root root 4096 Aug 21 21:12 media
drwxr-xr-x  2 root root 4096 Aug 21 21:12 mnt
drwxr-xr-x  2 root root 4096 Aug 21 21:12 opt
dr-xr-xr-x 95 root root    0 Sep  3 23:13 proc
drwx------  2 root root 4096 Aug 21 21:14 root
drwxr-xr-x  1 root root 4096 Aug 22 17:28 run
drwxr-xr-x  1 root root 4096 Aug 22 17:28 sbin
drwxr-xr-x  2 root root 4096 Aug 21 21:12 srv
dr-xr-xr-x 13 root root    0 Sep  3 21:26 sys
drwxrwxrwt  2 root root 4096 Aug 21 21:14 tmp
drwxr-xr-x  1 root root 4096 Aug 21 21:12 usr
drwxr-xr-x  1 root root 4096 Aug 21 21:14 var
root@f1f90ee443e8:/# ls -l datavolume4
total 4
-rw-r--r-- 1 root root 47 Sep  3 23:14 fileIO.txt
root@f1f90ee443e8:/# cat datavolume4/fileIO.txt
This file is shared between two containers - 1
root@f1f90ee443e8:/# exit
exit
```

- Step 2: Now, create container2 and mount volumes from container1.

```
$ docker run –ti –-name container2 –-volumes-from  container1 ubuntu
```

```
[root@ip-172-31-23-19 ec2-user]#
[root@ip-172-31-23-19 ec2-user]# docker run -ti --name=container2 --volumes-from container1 ubuntu
root@ed795927b2b0:/# ls -l
total 68
drwxr-xr-x  2 root root 4096 Aug 21 21:14 bin
drwxr-xr-x  2 root root 4096 Apr 24 08:34 boot
drwxr-xr-x  2 root root 4096 Sep  3 23:14 datavolume4
drwxr-xr-x  5 root root  360 Sep  3 23:21 dev
drwxr-xr-x  1 root root 4096 Sep  3 23:21 etc
drwxr-xr-x  2 root root 4096 Apr 24 08:34 home
drwxr-xr-x  8 root root 4096 Aug 21 21:12 lib
drwxr-xr-x  2 root root 4096 Aug 21 21:13 lib64
drwxr-xr-x  2 root root 4096 Aug 21 21:12 media
drwxr-xr-x  2 root root 4096 Aug 21 21:12 mnt
drwxr-xr-x  2 root root 4096 Aug 21 21:12 opt
dr-xr-xr-x 95 root root    0 Sep  3 23:21 proc
drwx------  2 root root 4096 Aug 21 21:14 root
drwxr-xr-x  1 root root 4096 Aug 22 17:28 run
drwxr-xr-x  1 root root 4096 Aug 22 17:28 sbin
drwxr-xr-x  2 root root 4096 Aug 21 21:12 srv
dr-xr-xr-x 13 root root    0 Sep  3 21:26 sys
drwxrwxrwt  2 root root 4096 Aug 21 21:14 tmp
drwxr-xr-x  1 root root 4096 Aug 21 21:12 usr
drwxr-xr-x  1 root root 4096 Aug 21 21:14 var
root@ed795927b2b0:/# cat datavolume4/fileIO.txt
This file is shared between two containers - 1
root@ed795927b2b0:/#
```

Now, append the content in the shared file.

```
[root@ip-172-31-23-19 ec2-user]# docker run -ti --name=container2 --volumes-from container1 ubuntu
root@ed795927b2b0:/# ls -l
total 68
drwxr-xr-x  2 root root 4096 Aug 21 21:14 bin
drwxr-xr-x  2 root root 4096 Apr 24 08:34 boot
drwxr-xr-x  2 root root 4096 Sep  3 23:14 datavolume4
drwxr-xr-x  5 root root  360 Sep  3 23:21 dev
drwxr-xr-x  1 root root 4096 Sep  3 23:21 etc
drwxr-xr-x  2 root root 4096 Apr 24 08:34 home
drwxr-xr-x  8 root root 4096 Aug 21 21:12 lib
drwxr-xr-x  2 root root 4096 Aug 21 21:13 lib64
drwxr-xr-x  2 root root 4096 Aug 21 21:12 media
drwxr-xr-x  2 root root 4096 Aug 21 21:12 mnt
drwxr-xr-x  2 root root 4096 Aug 21 21:12 opt
dr-xr-xr-x 95 root root    0 Sep  3 23:21 proc
drwx------  2 root root 4096 Aug 21 21:14 root
drwxr-xr-x  1 root root 4096 Aug 22 17:28 run
drwxr-xr-x  1 root root 4096 Aug 22 17:28 sbin
drwxr-xr-x  2 root root 4096 Aug 21 21:12 srv
dr-xr-xr-x 13 root root    0 Sep  3 21:26 sys
drwxrwxrwt  2 root root 4096 Aug 21 21:14 tmp
drwxr-xr-x  1 root root 4096 Aug 21 21:12 usr
drwxr-xr-x  1 root root 4096 Aug 21 21:14 var
root@ed795927b2b0:/# cat datavolume4/fileIO.txt
This file is shared between two containers - 1
root@ed795927b2b0:/# echo "This file is written by another container - 2" >> /datavolume4/fileIO.txt
```

Now, list the contents of the file.

```
root@ed795927b2b0:/# cat /datavolume4/fileIO.txt
This file is shared between two containers - 1
This file is written by another container - 2
root@ed795927b2b0:/#
```

Finally, exit the container.

```
root@ed795927b2b0:/# exit
exit
[root@ip-172-31-23-19 ec2-user]#
[root@ip-172-31-23-19 ec2-user]#
```

- Step 3: Attach the container1 in interactive mode.

```
$ docker start –ai container1
```

```
[root@ip-172-31-23-19 ec2-user]# docker start -ai container1
root@f1f90ee443e8:/# ls -l
total 68
drwxr-xr-x  2 root root 4096 Aug 21 21:14 bin
drwxr-xr-x  2 root root 4096 Apr 24 08:34 boot
drwxr-xr-x  2 root root 4096 Sep  3 23:14 datavolume4
drwxr-xr-x  5 root root  360 Sep  3 23:29 dev
drwxr-xr-x  1 root root 4096 Sep  3 23:13 etc
drwxr-xr-x  2 root root 4096 Apr 24 08:34 home
drwxr-xr-x  8 root root 4096 Aug 21 21:12 lib
drwxr-xr-x  2 root root 4096 Aug 21 21:13 lib64
drwxr-xr-x  2 root root 4096 Aug 21 21:12 media
drwxr-xr-x  2 root root 4096 Aug 21 21:12 mnt
drwxr-xr-x  2 root root 4096 Aug 21 21:12 opt
dr-xr-xr-x 97 root root    0 Sep  3 23:29 proc
drwx------  1 root root 4096 Sep  3 23:15 root
drwxr-xr-x  1 root root 4096 Aug 22 17:28 run
drwxr-xr-x  1 root root 4096 Aug 22 17:28 sbin
drwxr-xr-x  2 root root 4096 Aug 21 21:12 srv
dr-xr-xr-x 13 root root    0 Sep  3 21:26 sys
drwxrwxrwt  2 root root 4096 Aug 21 21:14 tmp
drwxr-xr-x  1 root root 4096 Aug 21 21:12 usr
drwxr-xr-x  1 root root 4096 Aug 21 21:14 var
root@f1f90ee443e8:/# cat /datavolume4/fileIO.txt
This file is shared between two containers - 1
This file is written by another container - 2
root@f1f90ee443e8:/#
```

Note: You can also start a container in READ-ONLY mode and share the data in it

```
$ docker run start –ti --name =container3  --volumes-from container1:ro Ubuntu
```

```
$ docker run -ti --name=Container6 --volumes-from Container4:ro ubuntu
```