UNIVERSITY OF PETROLEUM & ENERGY STUDIES

College of Engineering Studies

Dehradun

Lab Exercise 9

Programme: B. Tech in Computer Science and Engineering with Specialization in DevOps

Course: Build and Release Management Lab

Semester: IV

Session: Jan-May 2022

Batch: 2020-2024

Faculty: Dr Hitesh Kumar Sharma

Submitted By:

Name	SAPID	ROLL NO
Rohit Kumar	500082652	R214220968

Cybernetics Cluster
School of Computer Science
University of Petroleum and Energy Studies
Dehradun-248007

Aim: Create a Docker Volume and Complete the following Scenario on Katacoda.

https://www.katacoda.com/courses/docker/persisting-data-using-volumes

In this scenario, you'll learn how to use Docker Volumes to persistent data within Containers. Docker Volumes allow directories to be shared between containers and container versions.

Docker Volumes allows you to upgrade containers, restart machines and share data without data loss. This is essential when updating database or application versions.

Step 1 - Data Volumes

Docker Volumes are created and assigned when containers are started. Data Volumes allow you to map a host directory to a container for sharing data.

This mapping is bi-directional. It allows data stored on the host to be accessed from within the container. It also means data saved by the process inside the container is persisted on the host.

Task

This example will use Redis as a way to persist data. Start a Redis container below, and create a data volume using the -v parameter. This specifies that any data saved inside the container to the /data directory should be persisted on the host in the directory /docker/redis-data.

```
docker run -v /docker/redis-data:/data \ --name r1 -d redis \ redis-server --
```

We can pipe data into the Redis instance using the following command.

```
cat data | docker exec -i r1 redis-cli --pipe
```

Redis will save this data to disk. On the host we can investigate the mapped direct which should contain the Redis data file.

```
ls /docker/redis-data
```

This same directory can be mounted to a second container. One usage is to have a Docker Container performing backup operations on your data.

```
docker run -v /docker/redis-data:/backup ubuntu ls /backup
```

Step 2 - Shared Volumes

Data Volumes mapped to the host are great for persisting data. However, to gain access to them from another container you need to know the exact path which can make it error-prone.

An alternate approach is to use -volumes-from. The parameter maps the mapped volumes from the source container to the container being launched.

In this case, we're mapping our Redis container's volume to an Ubuntu container. The /data directory only exists within our Redis container, however, because of -volumes-from our Ubuntu container can access the data.

```
docker run --volumes-from r1 -it ubuntu ls /data
```

This allows us to access volumes from other containers without having to be concerned how they're persisted on the host.

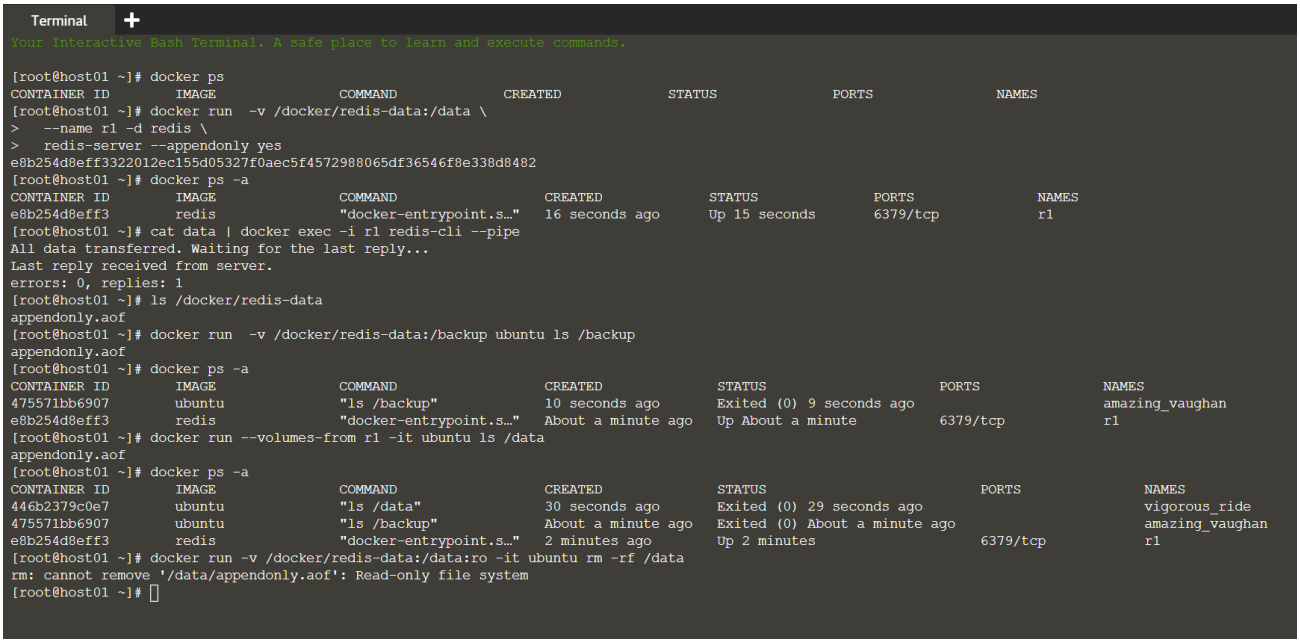
Step 3 - Read-only Volumes

Mounting Volumes gives the container full read and write access to the directory. You can specify read-only permissions on the directory by adding the permissions :ro to the mount.

If the container attempts to modify data within the directory it will error.

```
docker run -v /docker/redis-data:/data:ro -it ubuntu rm -rf /data
```

Solution :


Terminal
Your Interactive Bash Terminal. A safe place to learn and execute commands.
[root@host01 ~]# docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
[root@host01 ~]# docker run -v /docker/redis-data:/data \
> --name r1 -d redis \
> redis-server --appendonly yes
e8b254d8eff3322012ec155d05327f0aec5f4572988065df36546f8e338d8482
[root@host01 ~]# docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
e8b254d8eff3 redis "docker-entrypoint.s…" 16 seconds ago Up 15 seconds 6379/tcp r1
[root@host01 ~]# cat data | docker exec -i r1 redis-cli --pipe
All data transferred. Waiting for the last reply...
Last reply received from server.
errors: 0, replies: 1
[root@host01 ~]# ls /docker/redis-data
appendonly.aof
[root@host01 ~]# docker run -v /docker/redis-data:/backup ubuntu ls /backup
appendonly.aof
[root@host01 ~]# docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
475571bb6907 ubuntu "ls /backup" 10 seconds ago Exited (0) 9 seconds ago amazing_vaughan
e8b254d8eff3 redis "docker-entrypoint.s…" About a minute ago Up About a minute 6379/tcp r1
[root@host01 ~]# docker run --volumes-from r1 -it ubuntu ls /data
appendonly.aof
[root@host01 ~]# docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
446b2379c0e7 ubuntu "ls /data" 30 seconds ago Exited (0) 29 seconds ago vigorous_ride
475571bb6907 ubuntu "ls /backup" About a minute ago Exited (0) About a minute ago amazing_vaughan
e8b254d8eff3 redis "docker-entrypoint.s…" 2 minutes ago Up 2 minutes 6379/tcp r1
[root@host01 ~]# docker run -v /docker/redis-data:/data:ro -it ubuntu rm -rf /data
rm: cannot remove '/data/appendonly.aof': Read-only file system
[root@host01 ~]#