WHERE ARE DOCKER VOLUMES STORED?

Innovative Team
Leading cloud innovation

Digital Solutions
Advanced cloud tech

24/7 Expert Support
Continuous assistance

Introduction

About Cloud Control

Docker image is a stack of read-only layers wherein each layer a Docker command is recorded. When we start a Docker container from Docker images, the Docker engine takes the read-only stack of layers and adds a read-write stack on top of it. The changes which we make within a running container will be stored on this read-write layer. This file system of Docker is known as the Union File System. Docker also ensures that the changes on read-write will not affect the original files in the read-only layer.

The Requirement Of Docker Volumes

Since the files created or modified inside a container will be stored on a writable container layer, the data doesn't persist when the container no longer exists. Also, it isn't easy to get the data out of a container if another process needs it. The Union File system is provided by a storage driver using Linux kernels. When we use the writable container layer, we have an extra abstraction with a storage driver, which reduces the performance.

Different Types Of Storage Options

Docker has multiple options for containers to store files in the host machine.

1. Volumes
2. Bind mounts
3. Tmpfs (If you are running Docker on Linux, you can also use *tmpfs* mount.)

Docker Volumes

Volumes are also stored as part of the host file system, which is managed by Docker. On Linux, volumes are stored in "/var/lib/docker/volume". Non-Docker processes should not be allowed to modify this part of the file system. One of the best ways to persist data in Docker is Volumes.

```
#create a volume
sudo docker volume create AppZ_vol
```

```
# List volumes
```

Cloud Control simplifies cloud management with AppZ, DataZ, and ManageZ, optimizing operations, enhancing security, and accelerating time-to-market. We help businesses achieve cloud goals efficiently and reliably.

Read More ➞

2025
Convergence
India Expo

```
sudo docker volume ls
```

```
# Inspect volumes
sudo docker volume inspect  AppZ_vol
```

```
user@workstation:~$ sudo docker volume create  AppZ_vol

AppZ_vol

user@workstation:~$ sudo docker volume ls|grep -i AppZ_vol

local                AppZ_vol

user@workstation:~$ sudo docker volume inspect  AppZ_vol
[

{

        "CreatedAt": "2020-08-10T19:56:01+05:30",

        "Driver": "local",

        "Labels": {},

        "Mountpoint": "/var/lib/docker/volumes/AppZ_vol/_data",

        "Name": "AppZ_vol",

        "Options": {},

        "Scope": "local"
```

Join Us ➞

19th – 21st March

New Delhi, India

2025

Tech Talk | AI In Action

}

Now let us run the Nginx container with the welcome page mounted to volume **AppZ_vol**

```
sudo docker run -d --name=AppZwebApp  -v  AppZ_vol:/usr/share/nginx/html -p 80:80 nginx
```

Bind Mounts

"Bind mounts" can be stored anywhere on the host system, which may even contain important system files or directories. In this case, Non-Docker processes on the Docker host or a Docker container can modify them at any time. If you bind-mount into a non-empty directory on the container, the directory's existing contents are obscured by the bind mount.

```
#Create a folder named target at the present working directory
mkdir target
```

```
#create index.html inside target folder
echo "Testing bind mount" >target/index.html
```

Now let's run a container with bind mount

```
#Mount the folder target to the AppZwebApp_bindmount container using bind mount
sudo docker run -d --name=AppZwebApp_bindmount  -v  "$(pwd)"/target:/usr/share/nginx/html  -p 80:
```

If you execute an interactive bash shell on the container and check the */usr/share/nginx/html/* path you will see the index.html

Difference Between Bind Mount And Volume Mount

Join Us ➞

May 29

Travancore Hall, Technopark
Phase 1
Trivandrum

Bind mounts are basically just binding a certain directory or file from the host inside the container. The file or directory is referenced by its absolute path on the host machine.

When you use a volume, a new directory is created within Docker's storage directory on the host machine, and Docker manages that directory's contents.

If you use binds and want to transfer your containers/applications to another host, you have to rebuild your directory-structure, whereas volumes are more uniform on every host.

Tmpfs Mounts

tmpfs mounts are temporary. They are stored in the host system's memory only and are never written to the host system's file system. If your container generates non-persistent state data, consider using a tmpfs mount to avoid storing the data anywhere permanently and increase the container's performance by avoiding writing into the container's writable layer. This option can be used for keeping sensitive data during execution.

Comparison Of Volumes, BindMounts & Tempfs

	Volumes	**Bind Mounts**	**Tmpfs Mounts**
Storage	In a host file system, which is managed by Docker. (var/lib/docker/volumes in linux)	Anywhere in the host system.	In system memory (RAM). The container creates files outside its writable layer.

	Volumes	**Bind Mounts**	**Tmpfs Mounts**
Access Control	By docker only	Docker and other processes.	Data is not written to the host filesystem.
Persistence	Persistent	Persistent	Non-persistent
Security	More secure as volumes are managed by docker itself and other processes can't modify the concerned file system	Not secure enough as the host file system can be modified by other processes	We can use this mount option for keeping sensitive data during execution, which should not persist
Support for volume drivers	Supports volume drivers. You can store data in remote hosts or cloud providers	Doesn't support volume drivers	Doesn't support volume drivers
Volume sharing	Multiple containers can mount the same volume simultaneously.	Sharing data across multiple containers is not possible.	Data can't be shared between containers.
Use Cases	• Easy to store your data on remote hosts or cloud providers using volumes.	• To share configuration files from host to container.	• To protect the performance of containers when your

	Volumes	Bind Mounts	Tmpfs Mounts
	• If Docker host is not having a definite file structure, volumes are preferred.	• Sharing source code or building artifacts between development environments in docker host and container. • When file or directory structure is guaranteed to be consistent,	application needs to write a large volume of nonpersistent data. • Tmpfs can be used in Linux machines only. • To mount secrets to containers or to store sensitive info that no one

About The Author

Pratheesh P

Cloud DevOps Engineer | CloudControl

Cloud DevOps Engineer with 2+ years of experience in supporting, automating, and optimizing deployments to hybrid cloud platforms using DevOps processes, CI/CD, containers and Kuberneties in both Production and Development environments

DOCKER

On Linked-In

1600 Providence Hwy, Walpole, MA 02081,
+1 (617) 970-8563

ACE-12, 4th Floor, C-DAC Building,
Technopark, Trivandrum, Kerala, India, +91
9946356641

LowTouch Cloud, DMCC Business Centre,
Dubai, UAE, sales@ecloudcontrol.com

info@ecloudcontrol.com