Docker Lab 3 — OverlayFS & Docker Storage Internals

Author: Dr. Sandeep Kumar Sharma

Lab Description

This lab focuses on one of the most essential foundations of Docker — **OverlayFS storage internals**. Understanding how Docker stores images, layers, and container filesystems is mandatory for advanced troubleshooting, optimization, and debugging.

In this lab, we will: - Explore how Docker uses OverlayFS - Inspect image layers and storage directories - Understand `lowerdir`, `upperdir`, `workdir`, and `merged` layers - Run controlled experiments using **nginx** and **ubuntu** containers - Analyze how changes inside the container are stored

All custom containers and images created in this lab will follow naming conventions like **sandeep-overlay**, **sandeep-nginx**, etc.

Topics Covered in This Lab

- What is OverlayFS?
- How Docker uses Overlay2 driver
- The anatomy of layers: lowerdir, upperdir, merged
- Inspecting Docker storage folder (`/var/lib/docker/overlay2`)
- Understanding copy-on-write behavior
- Determining when layers change and how writes are stored
- Hands-on experiments with image layering

Learning Objectives

By the end of this lab, you will be able to: - Understand Docker's storage mechanism at OS level - Inspect image layers and trace modifications - Differentiate between read-only and writable layers - Demonstrate how container changes are stored - Troubleshoot Docker storage-related issues

---

Learning Outcomes

After completing this lab, you will: - Be capable of analyzing container layer corruption issues - Understand exactly how Docker stores data internally - Be able to optimize images and storage usage - Have experience digging into Docker's filesystem with precision

---

Section 1 — Understanding OverlayFS in Docker

Docker uses **Overlay2** as the default storage driver on modern Linux.

OverlayFS uses three main directories:

1. lowerdir (Read-Only Layers)

Contains all the read-only image layers stacked together.

2. upperdir (Writable Layer)

This is where all container changes are stored. Examples of changes: - Edited files - New files created - Deleted files (represented as whiteout files)

3. merged (Final View)

This is what you actually see when you `exec` into a container.

Section 2 — Hands-On: Inspecting Docker Storage Internals

Step 1: Run an Nginx Container

```
docker run -d --name sandeep-nginx nginx:latest
```

Step 2: Find Container ID

```
docker ps
```

Copy the container ID.

Step 3: Inspect the Container

```
docker inspect sandeep-nginx | grep -i dir -A 3
```

Look for: - `UpperDir` - `LowerDir` - `MergedDir` - `WorkDir`

These paths point to:

```
/var/lib/docker/overlay2/<layer-id>/diff
```

Section 3 — Explore the Layer Directories

Navigate to the storage directory:

```
cd /var/lib/docker/overlay2/
```

Find the directory listed under `UpperDir`:

```
sudo ls <upperdir_path>
```

Initially, it should be almost empty.

Section 4 — Modify Files & Observe Copy-on-Write

Enter the container:

```
docker exec -it sandeep-nginx /bin/bash
```

Modify the default nginx HTML file:

```
echo "Hello from Sandeep OverlayFS Demo" > /usr/share/nginx/html/index.html
exit
```

Now check `upperdir` again:

```
sudo ls <upperdir_path>
```

You should now see new files created inside the upperdir — proving copy-on-write behavior.

Section 5 — Create a Custom Image and Observe Layers

Run Ubuntu and modify it:

```
docker run -it --name sandeep-overlay ubuntu:latest
```

Inside container:

```
apt update && apt install -y nginx
mkdir /data-demo
echo "Sandeep overlay test" > /data-demo/test.txt
exit
```

Now commit the container:

```
docker commit sandeep-overlay sandeep/overlay-image:v1
```

Inspect layers:

```
docker history sandeep/overlay-image:v1
```

Each RUN/COPY creates a new layer.

Section 6 — Explore Lowerdir (Base Image Layers)

Inspect:

```
docker inspect sandeep/overlay-image:v1
```

Note the `LowerDir` entries — these are compressed base layers.

You can `cd` into them and explore read-only content.

Section 7 — Delete a File Inside Container & Observe Whiteouts

Re-run container:

```
docker run -it --name sandeep-overlay2 sandeep/overlay-image:v1
```

Inside:

```
rm /data-demo/test.txt
exit
```

Check upperdir again:

```
sudo ls <upperdir_path>
```

Look for **whiteout markers**, named like:

```
.wh.test.txt
```

These markers hide files from lowerdir.

Section 8 — Cleanup

```
docker rm -f sandeep-nginx sandeep-overlay sandeep-overlay2
```

```
docker rmi sandeep/overlay-image:v1
```

Section 9 — Practice Tasks

1. Modify multiple files and track changes in upperdir.
2. Delete directories and observe whiteout behavior.
3. Compare OverlayFS behavior between Ubuntu and Nginx containers.
4. Write a script to automatically detect space used by upper layers.

Summary

- Docker stores image layers using OverlayFS.
- lowerdir = read-only layers from images.
- upperdir = writable layer for container changes.
- merged = final mount seen by the user.
- Whiteouts hide files from lowerdir.
- Understanding these concepts is critical for debugging and optimization.

Lab 3 is complete. You may now say: **“Create Lab 4”**.