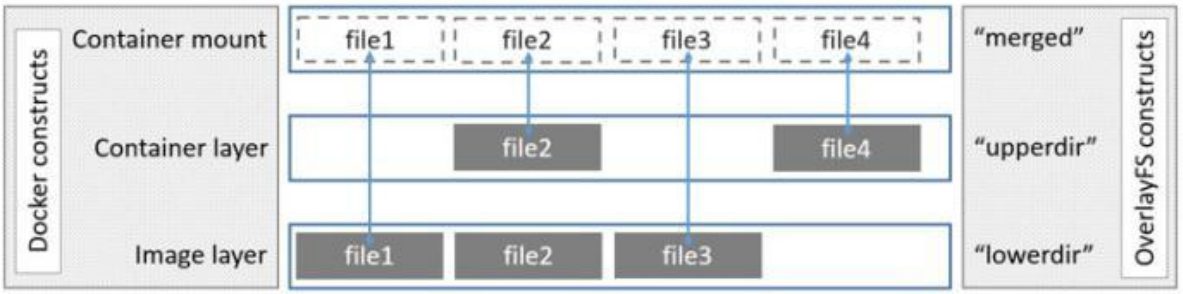
OverlayFS and Overlay2FS

While AUFS is only supported in some distributions (debian, gentoo, etc.), OverlayFS is included in the Linux Kernel.

The schema of OverlayFS is shown in the next image (obtained from Docker docs):



The underlying idea is the same than AUFS, and the concepts are almost the same: layers that combine together to build a unioned filesystem. The *lowerdir* are the readonly layers in AUFS, while the *upperdir* correspond to the read/write layer.

OverlayFS needs an extra folder named workdir that substitutes the .wh. hidden files in AUFS, but also is used to support the atomic operations on the filesystem.*

The main difference between OverlayFS and Overlay2 is that, OverlayFS only supported merging 1 single readonly layer with 1 read/write layer (although overlaying could be nested by overlaying overlayed layers). Now Overlay2 supports 128 lower layers

Working with OverlayFS

We are preparing an equivalent test example to see how OverlayFS works and that it is very easy to understand and to work with.

```
root:root# cd /tmp
root:tmp# mkdir overlay-test
root:tmp# cd overlay-test/
root:overlay-test# mkdir layer1 layer2 upperlayer workdir mountedfs
root:overlay-test# echo "content for file1.txt in layer1" >
layer1/file1.txt
root:overlay-test# echo "content for file1.txt in layer2" >
layer2/file1.txt
root:overlay-test# echo "content for file2.txt in layer1" >
layer1/file2.txt
root:overlay-test# echo "content for file3.txt in layer2" >

layer2/file3.txt
```

With respect to the AUFS example, we had to include an extra folder (workdir) that is needed for OverlayFS to work.

```
root:tmp# mkdir overlay-test
root:tmp# cd overlay-test/
root:overlay-test# mkdir layer1 layer2 upperlayer workdir mountedfs
root:overlay-test# echo "content for file1.txt in layer1" > layer1/file1.txt
root:overlay-test# echo "content for file1.txt in layer2" > layer2/file1.txt
root:overlay-test# echo "content for file2.txt in layer1" > layer1/file2.txt
root:overlay-test# echo "content for file3.txt in layer2" > layer2/file3.txt
root:overlay-test# ls -l *
layer1:
total 8
-rw-r--r-- 1 root root 32 ene 15 10:09 file1.txt
-rw-r--r-- 1 root root 32 ene 15 10:09 file2.txt

layer2:
total 8
-rw-r--r-- 1 root root 32 ene 15 10:09 file1.txt
-rw-r--r-- 1 root root 32 ene 15 10:09 file3.txt

mountedfs:
total 0

upperlayer:
total 0

workdir:
total 0
root:overlay-test#
```

Now we'll mount the filesystem using the basic syntax:

```
root:overlay-test# mount -t overlay -o
lowerdir=layer1:layer2,upperdir=upperlayer,workdir=workdir overlay
mountedfs
```

The result is that folder *mountedfs* contains the union of the files that we have created:

```
root:overlay-test# mount -t overlay -o lowerdir=layer1:layer2,upperdir=upperlayer,workdir=workdir overlay mountedfs
root:overlay-test# cd mountedfs/
root:mountedfs# ls -l
total 12
-rw-r--r-- 1 root root 32 ene 15 10:09 file1.txt
-rw-r--r-- 1 root root 32 ene 15 10:09 file2.txt
-rw-r--r-- 1 root root 32 ene 15 10:09 file3.txt
root:mountedfs# cat file1.txt
content for file1.txt in layer1
root:mountedfs# cat file2.txt
content for file2.txt in layer1
root:mountedfs# cat file3.txt
content for file3.txt in layer2
```

As expected, we have the vision of the union of the 2 existing layers, and the contents of these files are the expected. The *lowerdir* folders are interpreted from left to right for the case of precendece. So if one file exists in different *lowerdirs* the unioned filesystem will show the file in the leftmost *lowerdir*.

Now if we create a new file in the directory mountedfs, the new contents will be created in the *upperdir* folder, while the contents in the other folders will be kept.

```
root:mountedfs# echo "new content for file4.txt" > file4.txt
root:mountedfs# ls -l
total 16
-rw-r--r-- 1 root root 32 ene 15 10:09 file1.txt
-rw-r--r-- 1 root root 32 ene 15 10:09 file2.txt
-rw-r--r-- 1 root root 32 ene 15 10:09 file3.txt
-rw-r--r-- 1 root root 26 ene 15 10:11 file4.txt
root:mountedfs# ls -l ../upperlayer/
total 4
-rw-r--r-- 1 root root 26 ene 15 10:11 file4.txt
root:mountedfs# cat ../upperlayer/file4.txt
new content for file4.txt
root:mountedfs# echo "new content for file1.txt" > file1.txt
root:mountedfs# cat ../upperlayer/file1.txt
new content for file1.txt
root:mountedfs# cat ../layer1/file1.txt
content for file1.txt in layer1
root:mountedfs# cat ../layer2/file1.txt
content for file1.txt in layer2
root:mountedfs#
```