

Systems Advanced Docker Containers

Container images en layers



Elfde-Liniestraat 24, 3500 Hasselt, www.pxl.be

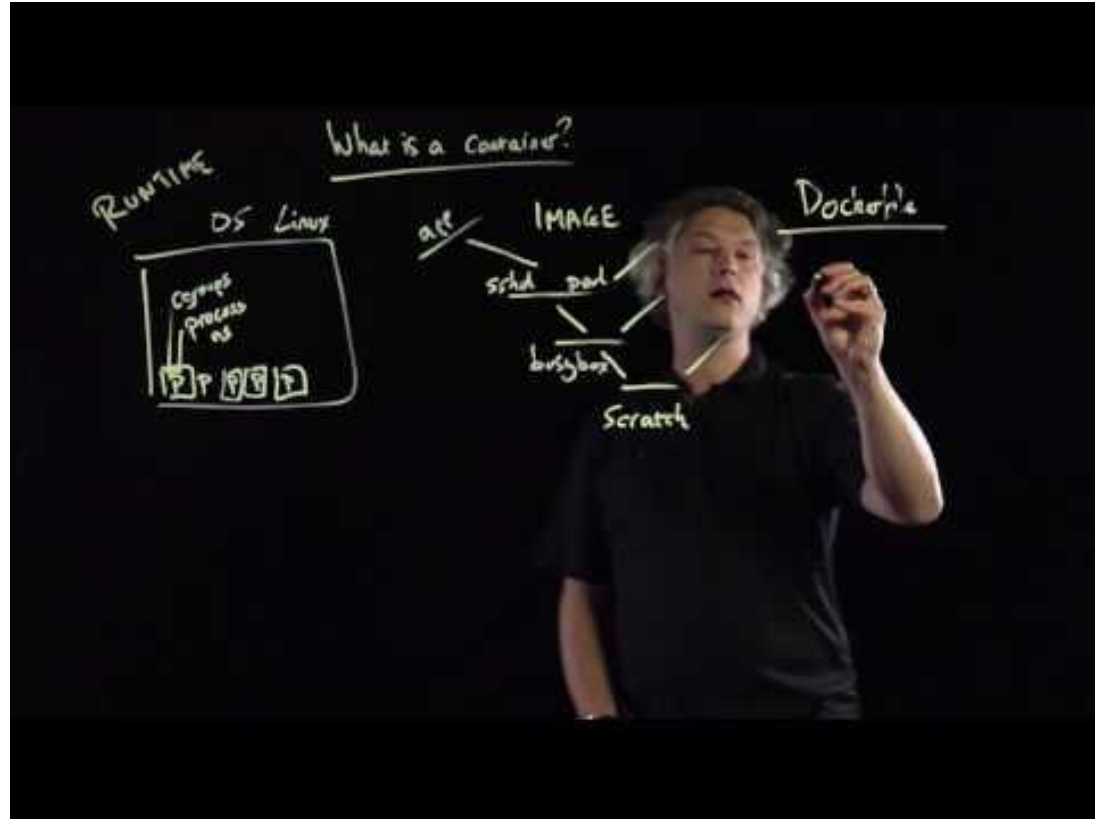


Hoe werken containers in linux?

Ben Corrie

"VMWare Senior Staff Engineer, lead architect behind the Kubernetes integration in vSphere"

<https://youtu.be/EnJ7qX9fkU?t=253> (4:15 tot 8:22)



Data blijft als container stopt

Indien een container stopt, blijft de data bestaan, totdat de container wordt verwijderd.

```
docker run -it ubuntu bash
```

```
# de file /tmp/testfile maken met de tekst "Hallo" erin
echo "Hallo" > /tmp/testfile
# om de container te verlaten
exit
```

```
# de container is gestopt, dus niet zichtbaar
docker container ls
```

```
# we zien de container met een short-id en een funkyname
docker container ls --all
```

```
# toont aanpassingen aan het filesystem van de container
docker diff <CONTAINER>
```

```
# start de container opnieuw, -i geeft ook ineens de prompt
docker start -i <CONTAINER>
```

```
# we zien dat de file nog bestaat
ls /tmp
# om de container te stoppen en te verlaten
exit
```

```
# thraa @ DESKTOP-TOMC in ~ [22:12:12]
$ docker run -it ubuntu bash
root@0188ab5866af:/# echo "Hallo" > /tmp/testfile
root@0188ab5866af:/# exit
exit

# thraa @ DESKTOP-TOMC in ~ [22:12:40]
$ docker container ls
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES

# thraa @ DESKTOP-TOMC in ~ [22:12:47]
$ docker container ls --all
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES
0188ab5866af   ubuntu   "bash"    31 seconds ago   Exited (0) 10 seconds ago           tender_spence

# thraa @ DESKTOP-TOMC in ~ [22:12:50]
$ docker diff tender_spence
C /root
A /root/.bash_history
C /tmp
A /tmp/testfile

# thraa @ DESKTOP-TOMC in ~ [22:12:56]
$ docker start -i tender_spence
root@0188ab5866af:/# ls /tmp
testfile
root@0188ab5866af:/# exit
exit

# thraa @ DESKTOP-TOMC in ~ [22:13:12]
$ _
```



Container data in het host file systeem

in `/var/lib/docker/image/overlay2/layerdb/mounts/<containerid>/mount-id` staat de koppeling met de disk die zich bevindt in `/var/lib/docker/overlay2/<container-(writable)layerid>/diff/...`

`docker container rm <CONTAINER>` verwijdert alle layers van de container dus ook de writable layer die de files bevat!

```
student@ubuntu-server-2:~$ docker container ls --all
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
114d6dfdadf0   ubuntu   "bash"    3 minutes ago    Exited (0) 2 minutes ago         magical_fermi

student@ubuntu-server-2:~$ sudo bash
root@ubuntu-server-2:/home/student# cat /var/lib/docker/image/overlay2/layerdb/mounts/114d6dfdadf06cf6cdabfc4eef4b720649ae85e06b73b4cf76a115b0038b9fbd//mount-id
83f9c637c17ba75e53f213898796860b9b673c91c0df48fdd4df508a233c8bc8root@ubuntu-server-2:/home/student#
root@ubuntu-server-2:/home/student# ls /var/lib/docker/overlay2/83f9c637c17ba75e53f213898796860b9b673c91c0df48fdd4df508a233c8bc8/diff/
root  tmp
root@ubuntu-server-2:/home/student# ls /var/lib/docker/overlay2/83f9c637c17ba75e53f213898796860b9b673c91c0df48fdd4df508a233c8bc8/diff/tmp/
testfile
root@ubuntu-server-2:/home/student# cat /var/lib/docker/overlay2/83f9c637c17ba75e53f213898796860b9b673c91c0df48fdd4df508a233c8bc8/diff/tmp/
testfile
root@ubuntu-server-2:/home/student#
```

Container data in het host file systeem (Windows)

Op Windows draait Docker in een Hyper-V VM via WSL 2, waar het pad `/var/lib/docker/` wordt gemapt op `\\wsl$\\docker-desktop-data\\version-pack-data\\community\\docker\\` in het Windows file system.

```
# thraa @ DESKTOP-TOMC in ~ [22:18:21]
$ docker container ls --all
CONTAINER ID   IMAGE      COMMAND                  CREATED          STATUS          PORTS          NAMES
0188ab5866af   ..ubuntu  "bash"                  6 minutes ago   Exited (0) 5 minutes ago          tender_spence

# thraa @ DESKTOP-TOMC in ~ [22:18:31]
$ cat "\\wsl$\\docker-desktop-data\\version-pack-data\\community\\docker\\image\\overlay2\\layerdb\\mounts\\0188ab5866af\\e3bb3440c4b2c2206ea80e87fd16d15531c0314d2d4ee4bcea5\\mount-id"
d54aef42a11ca11edcee0601c062a6e9c351b3ef041818245cc69d9718012f57

# thraa @ DESKTOP-TOMC in ~ [22:24:28]
$ ls \\wsl$\\docker-desktop-data\\version-pack-data\\community\\docker\\overlay2\\d54aef42a11ca11edcee0601c062a6e9c351b3ef041818245cc69d9718012f57\\diff
root tmp

# thraa @ DESKTOP-TOMC in ~ [22:25:36]
$ ls \\wsl$\\docker-desktop-data\\version-pack-data\\community\\docker\\overlay2\\d54aef42a11ca11edcee0601c062a6e9c351b3ef041818245cc69d9718012f57\\diff\\tmp
testfile

# thraa @ DESKTOP-TOMC in ~ [22:25:48]
$ cat '\\wsl$\\docker-desktop-data\\version-pack-data\\community\\docker\\overlay2\\d54aef42a11ca11edcee0601c062a6e9c351b3ef041818245cc69d9718012f57\\diff\\tmp\\testfile'
Hallo

# thraa @ DESKTOP-TOMC in ~ [22:25:56]
$
```

Container image Layers

Docker images zijn opgebouwd uit meerdere image layers via het overlay2-filesysteem dat werkt met union layers (**union file system**)

Image layers

- onderste layer: het rootfs met het OS (redhat, debian, ubuntu, centos, ...)
- middenste layer(s): de applicatie (apache, nginx, mysql, ...)
- bovenste layer(s): updates en bug-fixes

Transparent

- deze layers worden op elkaar gelegd door “union mounts” en zijn transparant voor de eindgebruiker

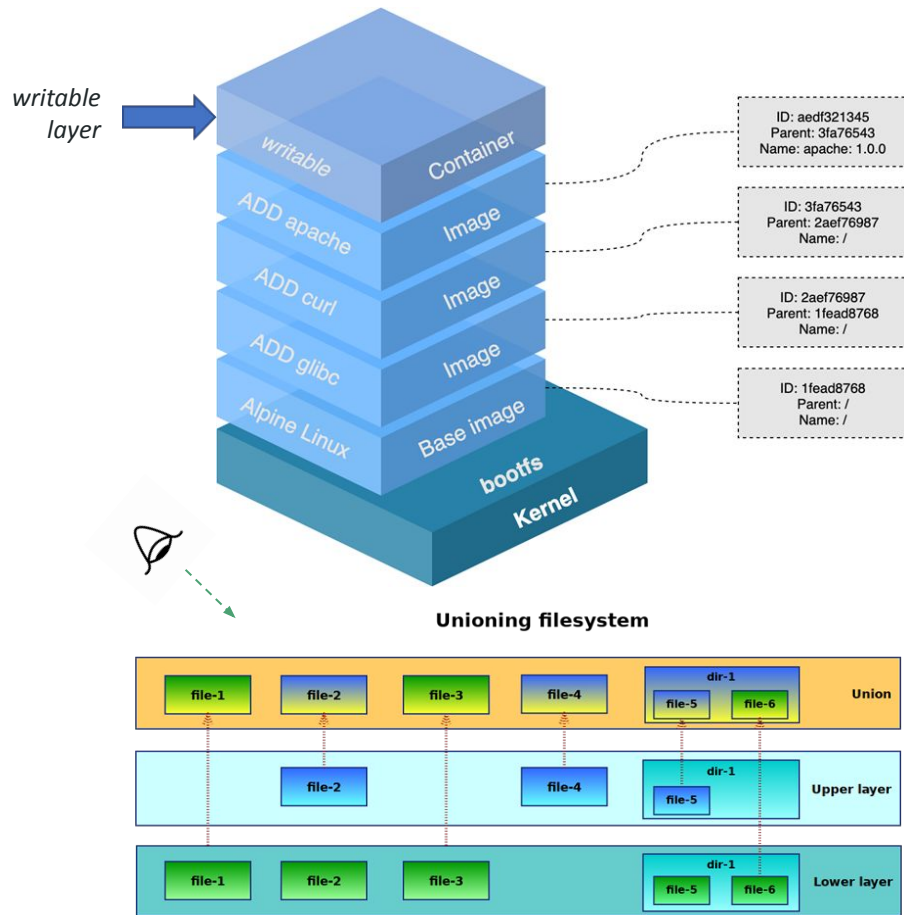
At runtime

- Eens een container wordt gestart wordt hier bovenop nog een **container layer** geplaatst, die alle toevoegingen en aanpassingen aan het filesystem bijhoudt.
 - Dit is de enige beschrijfbare layer in de container!!
- Er is ook nog kortstondig een bootfs-layer (helemaal onderaan de stack) die er enkel is tijdens het opstarten (booten) van de container
- De kernel is reeds geboot, want de kernel van de host wordt gedeeld

Layers tonen

- **docker history ubuntu**

Voor meer info over het OverlayFS kan je een kijkje nemen in het document: *OverlayFS explained of*
<https://docs.docker.com/storage/storagedriver/overlayfs-driver/>



docker inspect

We kunnen meer info vragen over een bepaalde container.

Toon de huidige (running) status, network settings, etc...: **docker inspect <CONTAINER>**

```
# thraa @ DESKTOP-TOMC in ~ [22:31:24]
$ docker container ls --all
CONTAINER ID   IMAGE      COMMAND                  CREATED          STATUS          PORTS          NAMES
0188ab5866af   ubuntu    "bash"                  19 minutes ago   Exited (0) 18 minutes ago           tender_spence

# thraa @ DESKTOP-TOMC in ~ [22:31:33]
$ docker inspect tender_spence | grep -A8 GraphDriver
  "GraphDriver": {
    "Data": {
      "LowerDir": "/var/lib/docker/overlay2/d54aef42a11ca11edcee0601c062a6e9c351b3ef041818245cc69d9718012f57-init/diff:/var/lib/docker/overlay2/688c9b721ab4dff703d55a1f59a2d21b21aa9806705222fe096e6545b557c489/diff",
      "MergedDir": "/var/lib/docker/overlay2/d54aef42a11ca11edcee0601c062a6e9c351b3ef041818245cc69d9718012f57/merged",
      "UpperDir": "/var/lib/docker/overlay2/d54aef42a11ca11edcee0601c062a6e9c351b3ef041818245cc69d9718012f57/diff",
      "WorkDir": "/var/lib/docker/overlay2/d54aef42a11ca11edcee0601c062a6e9c351b3ef041818245cc69d9718012f57/work"
    },
    "Name": "overlay2"
  },
},

# thraa @ DESKTOP-TOMC in ~ [22:31:50]
$
```

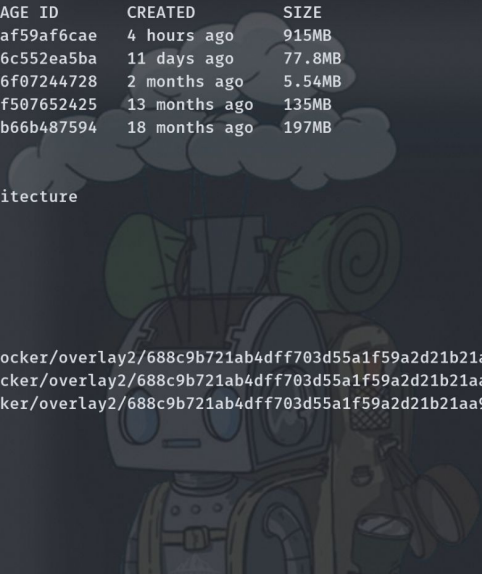
docker inspect

We kunnen meer info vragen over een bepaalde image.

Toon de settings van een image: **docker inspect <IMAGE>**

```
# thraa @ DESKTOP-TOMC in ~ [22:37:58]
$ docker images
REPOSITORY          TAG         IMAGE ID      CREATED       SIZE
tomcoolpxl/python-counter  latest     9daf59af6cae  4 hours ago   915MB
ubuntu              latest     216c552ea5ba  11 days ago   77.8MB
alpine              latest     9c6f07244728  2 months ago  5.54MB
ubuntu              16.04     b6f507652425  13 months ago 135MB
ubuntu              14.04     13b66b487594  18 months ago 197MB

# thraa @ DESKTOP-TOMC in ~ [22:38:12]
$ docker inspect ubuntu | grep -A12 Architecture
  "Architecture": "amd64",
  "Os": "linux",
  "Size": 77837008,
  "VirtualSize": 77837008,
  "GraphDriver": {
    "Data": {
      "MergedDir": "/var/lib/docker/overlay2/688c9b721ab4dff703d55a1f59a2d21b21aa9806705222fe096e6545b557c489/merged",
      "UpperDir": "/var/lib/docker/overlay2/688c9b721ab4dff703d55a1f59a2d21b21aa9806705222fe096e6545b557c489/diff",
      "WorkDir": "/var/lib/docker/overlay2/688c9b721ab4dff703d55a1f59a2d21b21aa9806705222fe096e6545b557c489/work"
    },
    "Name": "overlay2"
  },
  "RootFS": {
```



docker info

We kunnen bekijken hoeveel gestarte en gestopte containers en images we lokaal hebben.

docker info

Containers: 1

Running: 0

Paused: 0

Stopped: 1

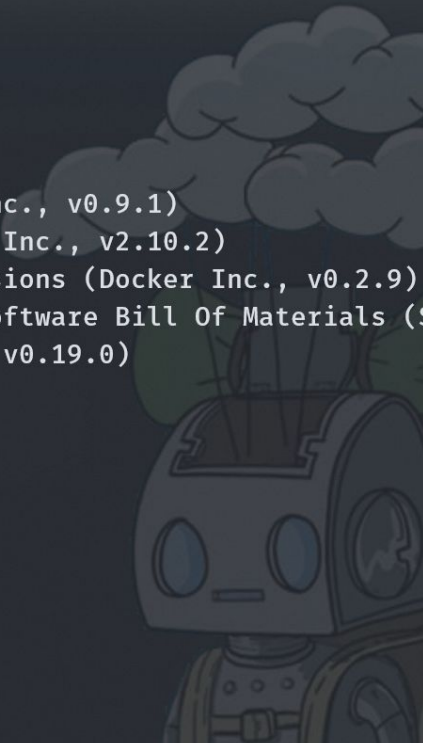
Images: 5

...

Storage Driver: overlay2

```
# thraa @ DESKTOP-TOMC in ~ [22:41:12]
$ docker info
Client:
 Context:      default
 Debug Mode:  false
 Plugins:
  buildx: Docker Buildx (Docker Inc., v0.9.1)
  compose: Docker Compose (Docker Inc., v2.10.2)
  extension: Manages Docker extensions (Docker Inc., v0.2.9)
  sbom: View the packaged-based Software Bill Of Materials (SLSA)
  scan: Docker Scan (Docker Inc., v0.19.0)

Server:
 Containers: 1
  Running: 0
  Paused: 0
  Stopped: 1
 Images: 5
 Server Version: 20.10.17
 Storage Driver: overlay2
  Backing Filesystem: extfs
```



OEFENING: nieuwe image van een container

We installeren iets bij in een container en maken er opnieuw een image van

- `docker run -it --name "ubuntu_ping" ubuntu bash`
 - `apt update && apt -y install iputils-ping`
 - `exit`
- `docker diff ubuntu_ping`
 - toont de toegevoegde en aangepaste bestanden van de container
- `docker commit ubuntu_ping myubuntu` # of via de <container-shortid>
 - commit de gewijzigde container naar een nieuwe image
- `docker image ls`
 - hier zien we de nieuwe image
- `docker history myubuntu` # --no-trunc om cmds volledig te zien
 - toont een historiek van commando's en size van die layer
- `docker run -it myubuntu ping -c 5 8.8.8.8`
 - nu bestaat het ping-commando wel in containers die gebaseerd zijn op de image myubuntu

OEFENING: De writable layer

- Start een ubuntu container en zorg er voor dat je in zijn shell bevindt
 - doe enkel een apt update
 - probeer te achterhalen welke files allemaal zijn toegevoegd door de apt update
 - zoek de files ook in het filesysteem
 - van de Docker container
 - van de Docker host
- Probeer nu zelf uit te zoeken hoe je, door gebruik te maken van de image myubuntu, kan achterhalen welke files er worden toegevoegd met het installeren van sl

OEFENING: OverlayFS explained

Ga door het document OverlayFS explained

- maak alle voorbeelden in je Docker VM
- zorg dat je alles begrijpt

