Práctica 2: Persistencia de dades Volums i xarxes amb Docker

1. Crear 2 xarxes (xarxa_frontend i xarxa_backend). Comprova que s'han creat i quin és el rang d'adreces que tenen.

Per crear les 2 xarxes frontend i backend hem d'escriure les següents comandes

sudo docker network create xarxa_frontend

sudo docker network create xarxa_backend

```
root@enti:/home/enti# docker network  create xarxa_frontend
bd82b7938c23dbe3654d9061be23fdb6fc5452473f2c8bba82714b4eb5b004da
root@enti:/home/enti# docker network  create xarxa_backend
a6362cba1ba7d23985091df0ef24ebe2b0619ebfd9a33903d082851b51f21d4c
```

La comanda **"docker network inspect"** proporciona una descripció completa de la configuració i la informació relacionada amb una xarxa Docker en particular, el que és útil per resoldre problemes de xarxa, entendre com està configurada la xarxa i com es relaciona amb altres contenidors o xarxes al sistema.

Així que ho farem en les dues màquines: **sudo docker network inspect xarxa_frontend**

```
enti@enti:~$ sudo docker network inspect xarxa_frontend
[
    {
        "Name": "xarxa_frontend",
        "Id": "bd82b7938c23dbe3654d9061be23fdb6fc5452473f2c8bba82714b4eb5b004da",
        "Created": "2023-05-05T12:19:26.563306121+02:00",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": {},
            "Config": [
                {
                    "Subnet": "172.18.0.0/16",
                    "Gateway": "172.18.0.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": {},
        "Options": {},
        "Labels": {}
    }
]
```

El rang de xarxes d'aquest Docker és 172.18.0.0/16. Això significa que la xarxa té un total de 2^16 (65.536) adreces IP disponibles, començant des de 172.18.0.1 fins a 172.18.255.255. L'adreça 172.18.0.0 és l'adreça de xarxa i l'adreça 172.18.255.255 és l'adreça de difusió. L'adreça 172.18.0.1 es reserva per a la porta d'enllaç de la xarxa.

D’altra banda, el docker backend es: **sudo docker network inspect xarxa_backend**

```
enti@enti:~$ sudo docker network inspect xarxa_backend
[
    {
        "Name": "xarxa_backend",
        "Id": "a6362cba1ba7d23985091df0ef24ebe2b0619ebfd9a33903d082851b51f21d4c",
        "Created": "2023-05-05T12:19:36.506485109+02:00",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": {},
            "Config": [
                {
                    "Subnet": "172.19.0.0/16",
                    "Gateway": "172.19.0.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": {},
        "Options": {},
        "Labels": {}
    }
]
```

El rang d'adreces IP de la xarxa "xarxa_backend" és 172.19.0.0/16. Això significa que la xarxa té un total de 2^16 (65.536) adreces IP disponibles, que van des de 172.19.0.1 fins a 172.19.255.255. L'adreça 172.19.0.0 és l'adreça de xarxa i l'adreça 172.19.255.255 és l'adreça de difusió. L'adreça 172.19.0.1 està reservada per a la porta d'enllaç de la xarxa.

2. Crear un contenidor anomenat c_bd amb la imatge "alpine" connectat a la xarxa_backend. Esbrina la seva IP.

```
enti@enti:~$ sudo docker run -it --network xarxa_backend --name c_bd alpine
[sudo] contraseña para enti:
/ #
```

Per esbrinar la seva IP, ens anem a una altre terminal, en el meu cas utilitzo byobu i faig la combinación ctrl+a+c per crear una de nova y utilitzo la següent comanda: **sudo docker container inspect c_bd | grep**

IPAddress

```
enti@enti:~$ sudo docker container inspect c_bd | grep IPAddress
            "SecondaryIPAddresses": null,
            "IPAddress": "",
                    "IPAddress": "172.19.0.2",
```

La seva IP és “172.19.0.2”.

3. Crear dos contenidors més amb la imatge "alpine" (anomenats c_web, c_control). Esbrina les IP's.

Per crear els 2 contenidors:

sudo docker run -it --name c_web alpine

```
enti@enti:~$ sudo docker run -it --name c_web alpine
[sudo] contraseña para enti:
/ #
```

sudo docker run -it --name c_control alpine

```
enti@enti:~$ sudo docker run -it --name c_control alpine
[sudo] contraseña para enti:
/ #
```

Per esbrinar les Ip’s: **sudo docker container inspect c_web | grep IPAddress**

sudo docker container inspect c_control | grep IPAddress

```
enti@enti:~$ sudo docker container inspect c_web | grep IpAddress
[sudo] contraseña para enti:
1 enti@enti:~$ sudo docker container inspect c_web | grep IPAddress
            "SecondaryIPAddresses": null,
            "IPAddress": "172.17.0.2",
                    "IPAddress": "172.17.0.2",
enti@enti:~$ sudo docker container inspect c_control | grep IPAddress
            "SecondaryIPAddresses": null,
            "IPAddress": "172.17.0.3",
                    "IPAddress": "172.17.0.3",
```

4. Connecta el contenidor c_web a les dues xarxes que hem creat.

Per conectar el contenidor c_web a les xarxes que hem creat:

sudo docker network connect xarxa_backend c_web

sudo docker network connect xarxa_frontend c_web

```
enti@enti:~$ sudo docker network connect xarxa_backend c_web
enti@enti:~$ sudo docker network connect xarxa_frontend c_web
```

5. Connecta el contenidor c_bd només a la xarxa_backend.

Per connectar el contenidor c_bd només a la xarxa_backend: **sudo docker network connect xarxa_backend c_bd**

Després per comprovar-ho fem: **sudo docker network inspect xarxa_backend**

```
enti@enti:~$ sudo docker network inspect xarxa_backend
```


```
enti@enti:~$ sudo docker network inspect xarxa_backend
[
    {
        "Name": "xarxa_backend",
        "Id": "a6362cba1ba7d23985091df0ef24ebe2b0619ebfd9a33903d082851b51f21d4c",
        "Created": "2023-05-05T12:19:36.506485109+02:00",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": {},
            "Config": [
                {
                    "Subnet": "172.19.0.0/16",
                    "Gateway": "172.19.0.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": {
            "dbcbdaf818480ca7cdc4730ea9a9ed55a0ea59a11bc94b0d89be5defdc68edc1": {
                "Name": "c_bd",
                "EndpointID": "3c060ac983c9e2bbe915a0904066079aaaa42be5d4ac0522e5fef951fc70b000",
                "MacAddress": "02:42:ac:13:00:02",
                "IPv4Address": "172.19.0.2/16",
                "IPv6Address": ""
            }
        },
        "Options": {},
        "Labels": {}
    }
]
```


Podem veure que està connectat.

6. Comprova a les xarxes (bridge, xarxa_frontend, xarxa_backend) que tenen connectats els contenidors corresponents. Esbrina les adreces IP dels contenidors i observa si han canviat.

Primer de tot hem de fer un start al dockers per assegurar-nos de que se estàn executant. Per això fem

sudo docker start c_web

sudo docker start c_bd

Amb **sudo docker ps**, podem comprovar que estàn connectats.

```
enti@enti:~$ sudo docker ps
CONTAINER ID   IMAGE     COMMAND     CREATED         STATUS         PORTS     NAMES
dbcbdaf81848   alpine    "/bin/sh"   19 hours ago    Up 5 minutes             c_bd
4536a3eacf5c   alpine    "/bin/sh"   19 hours ago    Up 4 minutes             c_web
```

Una vegada veiem que ja estàn funcionant. Mirem si estàn connectats a la xarxa_backend amb la següent comanda: **sudo docker network inspect xarxa_backend**

```
enti@enti:~$ sudo docker network inspect xarxa_backend
[
    {
        "Name": "xarxa_backend",
        "Id": "a6362cba1ba7d23985091df0ef24ebe2b0619ebfd9a33903d082851b51f21d4c",
        "Created": "2023-05-05T12:19:36.506485109+02:00",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": {},
            "Config": [
                {
                    "Subnet": "172.19.0.0/16",
                    "Gateway": "172.19.0.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": {
            "4536a3eacf5c90f35e67ad2788400c161f616c1c4b184c3eb58a9c10ec1899d3": {
                "Name": "c_web",
                "EndpointID": "04bdac77cc39d7db5f2d1357009a0cc9733d41e81674313b42555ae1d033bbb9",
                "MacAddress": "02:42:ac:13:00:03",
                "IPv4Address": "172.19.0.3/16",
                "IPv6Address": ""
            },
            "dbcbdaf818480ca7cdc4730ea9a9ed55a0ea59a11bc94b0d89be5defdc68edc1": {
                "Name": "c_bd",
                "EndpointID": "b8b21d464772e4afb050dd47960b4b61be66238281fdbde57f41c97950d1fd5e",
                "MacAddress": "02:42:ac:13:00:02",
                "IPv4Address": "172.19.0.2/16",
                "IPv6Address": ""
            }
        },
        "Options": {},
        "Labels": {}
    }
]
```

Aquí podem veure que estàn connectats els 2 dockers a la xarxa backend, desde aquí ja podem veure la seva Ip pero per filtrar-la podem utilitzar les següents comandes:

sudo docker container inspect c_bd | grep IPAddress

sudo docker container inspect c_web | grep IPAddress

```
enti@enti:~$ sudo docker container inspect c_bd | grep IPAddress
            "SecondaryIPAddresses": null,
            "IPAddress": "",
                    "IPAddress": "172.19.0.2",
enti@enti:~$ sudo docker container inspect c_web | grep IPAddress
            "SecondaryIPAddresses": null,
            "IPAddress": "172.17.0.2",
                    "IPAddress": "172.17.0.2",
                    "IPAddress": "172.19.0.3",
                    "IPAddress": "172.18.0.2",
```

Cal dir que s'han afegit adreces extra a c_web i c_bd. En el cas de c_bd, no hi ha connexió a la xarxa Bridge perquè hem connectat directe a la xarxa_backend.

7. Fes ping (amb adreça IP i usant el nom dels contenidors) des de c_control a les altres. Anota quines veiem.

```
enti@enti:~$ sudo docker start c_control
[sudo] contraseña para enti:
c_control
enti@enti:~$ sudo docker attach c_control
/ # ping c_bd
ping: bad address 'c_bd'
/ # ping c_web
ping: bad address 'c_web'
/ # ping 172.17.0.2
PING 172.17.0.2 (172.17.0.2): 56 data bytes
64 bytes from 172.17.0.2: seq=0 ttl=64 time=0.184 ms
64 bytes from 172.17.0.2: seq=1 ttl=64 time=0.067 ms
64 bytes from 172.17.0.2: seq=2 ttl=64 time=0.069 ms
^C
--- 172.17.0.2 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 0.067/0.106/0.184 ms
/ # ping 172.18.0.2
PING 172.18.0.2 (172.18.0.2): 56 data bytes
^C
--- 172.18.0.2 ping statistics ---
10 packets transmitted, 0 packets received, 100% packet loss
/ # ping 172.19.0.2
PING 172.19.0.2 (172.19.0.2): 56 data bytes
^C
--- 172.19.0.2 ping statistics ---
8 packets transmitted, 0 packets received, 100% packet loss
```

Quan vaig executar la comanda "sudo docker attach c_control", vaig accedir a la terminal del contenidor "c_control". Des d'allà, vaig intentar fer ping als altres contenidors a la xarxa, com "c_bd" i "c_web", però tots dos van donar un error de "bad address". No obstant, vaig poder fer ping al contenidor "c_bd" utilitzant la seva adreça IP "172.17.0.2", que pertany a la xarxa "bridge". A més, vaig intentar fer ping al contenidor "c_web" utilitzant la seva adreça IP "172.18.0.2", però no vaig rebre cap resposta, el que indica que no està a la mateixa xarxa que "c_control". Finalment, vaig intentar fer ping al contenidor "c_bd" utilitzant la seva adreça IP a la xarxa "xarxa_backend" ("172.19.0.2"), però també vaig rebre una resposta de "100% packet loss", el que indica que no està disponible a la xarxa

8. Connecta el contenidor c_control a la xarxa_frontend.

Per connectar el contenidor c_control a la xarxa_frontend utilitzem la comanda: **sudo docker network connect xarxa_frontend c_control**

```
enti@enti:~$ sudo docker network connect xarxa_frontend c_control
```

Per comprovar-ho: **sudo docker inspect xarxa_frontend**

```
enti@enti:~$ sudo docker inspect xarxa_frontend
[
    {
        "Name": "xarxa_frontend",
        "Id": "bd82b7938c23dbe3654d9061be23fdb6fc5452473f2c8bba82714b4eb5b004da",
        "Created": "2023-05-05T12:19:26.563306121+02:00",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": {},
            "Config": [
                {
                    "Subnet": "172.18.0.0/16",
                    "Gateway": "172.18.0.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": {
            "bf5f259c709ecb960285cfc20cc294382b0a711fb0f61e2f541f3c2ebc9a1673": {
                "Name": "c_control",
                "EndpointID": "49071b355a9661b8488b428e3221616cc02033fa964a6992b5b4e884859949b0",
                "MacAddress": "02:42:ac:12:00:02",
                "IPv4Address": "172.18.0.2/16",
                "IPv6Address": ""
            }
        },
        "Options": {},
        "Labels": {}
    }
]
```

9. Fes ping (amb adreça IP i usant el nom dels contenidors) des de c_control a les altres. Anota quines veiem.

Ping c_bd

En fer ping al contenidor c_bd utilitzant el seu nom, em dóna un error. Això és perquè hem afegit el contenidor c_control a la xarxa "front_end", i no a la "back_end". Per tant, no és possible fer ping a c_bd des de c_control

```
/ # ping c_bd
ping: bad address 'c_bd'
```

Ping c_web

Fent ping al contenidor c_web utilitzant el seu nom, sí que obtinc una resposta. Això és perquè el contenidor c_control està ara dins de la mateixa xarxa que el contenidor c_web, ambdós estan a la xarxa front_end.

```
/ # ping c_web
PING c_web (172.18.0.3): 56 data bytes
64 bytes from 172.18.0.3: seq=0 ttl=64 time=0.165 ms
64 bytes from 172.18.0.3: seq=1 ttl=64 time=0.062 ms
64 bytes from 172.18.0.3: seq=2 ttl=64 time=0.071 ms
^C
--- c_web ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 0.062/0.099/0.165 ms
```

Ping a les Ip's 172.19.0.2 i 172.19.0.3.

Fent ping als contenidors c_web i c_bd que pertanyen a la xarxa back_end amb les seves respectives adreces IP (172.19.0.3 i 172.19.0.2), no obtinc cap resposta, ja que no pertanyen a la mateixa xarxa. Per tant, no puc connectar amb ells ni amb els seus noms ni amb les seves adreces IP.

```
/ # ping 172.19.0.2
PING 172.19.0.2 (172.19.0.2): 56 data bytes
^C
--- 172.19.0.2 ping statistics ---
7 packets transmitted, 0 packets received, 100% packet loss
/ # ping 172.19.0.3
PING 172.19.0.3 (172.19.0.3): 56 data bytes
^C
--- 172.19.0.3 ping statistics ---
7 packets transmitted, 0 packets received, 100% packet loss
```

Ping 172.18.0.2 que si que em retorna (bridge)

```
/ # ping 172.18.0.2
PING 172.18.0.2 (172.18.0.2): 56 data bytes
64 bytes from 172.18.0.2: seq=0 ttl=64 time=0.099 ms
64 bytes from 172.18.0.2: seq=1 ttl=64 time=0.039 ms
64 bytes from 172.18.0.2: seq=2 ttl=64 time=0.041 ms
^C
--- 172.18.0.2 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 0.039/0.059/0.099 ms
```

Docker Volumes

1. Amb la imatge que havíem creat abans per baixar de Youtube, inicia un contenidor (anomenat mus-1) i fes que el fitxer baixat es guardi al directori /tmp de l'equip amfitrió (volum enllaçat a carpeta local).

Comanda: **sudo docker run -v /tmp:/mp3 --device /dev/snd:/dev/snd --name mus-1 -it reproductor bash -c "yt-dlp -x --audio-format mp3 --audio-quality 0 https://www.youtube.com/watch?v=zcV7RptHqro -o /mp3/music.mp3 && mplayer /mp3/music.mp3"**

-v /tmp:/mp3: Aquest paràmetre especifica que el directori /tmp del sistema host està mapejat amb el directori /mp3 del contenidor.

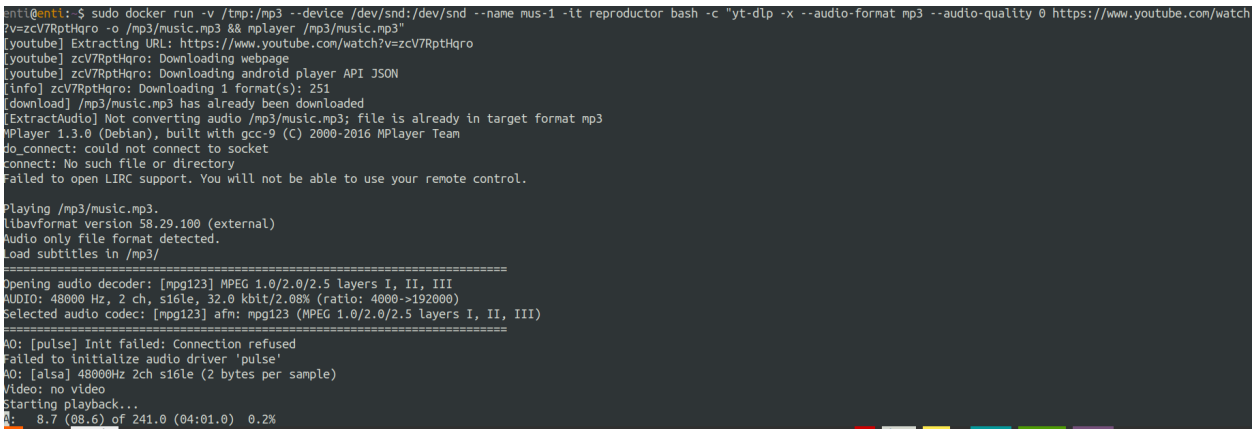
--device /dev/snd:/dev/snd: Aquest paràmetre permet al contenidor accedir al dispositiu de so /dev/snd del sistema host.

--name mus-1: nom del contenidor creat

-it: Aquest paràmetre indica que es vol utilitzar la terminal interactiva del contenidor.

reproductor: Aquest és el nom de la imatge de Docker que es vol utilitzar per a crear el contenidor.

-c "yt-dlp -x --audio-format mp3 --audio-quality 0 https://www.youtube.com/watch?v=zcV7RptHqro -o /mp3/music.mp3 && mplayer /mp3/music.mp3": Aquesta és la comanda que es vol executar dins del contenidor. En aquest cas, s'està utilitzant el programa yt-dlp per a descarregar un vídeo de YouTube en format MP3 i després reproduir-lo amb el reproductor mplayer. La sortida del programa yt-dlp es guarda en el fitxer /mp3/music.mp3 que està ubicat dins del directori mapejat /mp3 del contenidor.



2. Crea un volum anomenat "musica" (sense accent) i llança un contenidor (anomenat mus-2) enllaçat al volum creat per guardar el fitxer baixat.

Per crear el volum anomenat musica: **sudo docker volume create musica**

```
enti@enti:~$ sudo docker volume create musica
musica
```

Després amb el comandament: **sudo docker run -v musica:/mp3 --device /dev/snd:/dev/snd --name mus-2 -it reproductor bash -c "yt-dlp -x --audio-format mp3 --audio-quality 0** https://www.youtube.com/watch?v=zcV7RptHqro -o /mp3/music.mp3 && mplayer /mp3/music.mp3"

Crea un contenidor amb el nom "mus-2", que està enllaçat amb el volum "musica" que acabem de crear. També inclou les opcions per permetre la reproducció de l'àudio amb l'ús de mplayer i per descarregar el fitxer de música de YouTube en format mp3 i guardar-lo en el directori del volum /mp3

```
enti@enti:~$ sudo docker run -v musica:/mp3 --device /dev/snd:/dev/snd --name mus-2 -it reproductor bash -c "yt-dlp -x --audio-format mp3 --audio-quality 0 https://www.youtube.com/watch?v=zcV7RptHqro -o /mp3/music.mp3 && mplayer /mp3/music.mp3"
[youtube] Extracting URL: https://www.youtube.com/watch?v=zcV7RptHqro
[youtube] zcV7RptHqro: Downloading webpage
[youtube] zcV7RptHqro: Downloading android player API JSON
[info] zcV7RptHqro: Downloading 1 format(s): 251
[download] /mp3/music.mp3 has already been downloaded
[ExtractAudio] Not converting audio /mp3/music.mp3; file is already in target format mp3
MPlayer 1.3.0 (Debian), built with gcc-9 (C) 2000-2016 MPlayer Team
do_connect: could not connect to socket
connect: No such file or directory
Failed to open LIRC support. You will not be able to use your remote control.

Playing /mp3/music.mp3.
libavformat version 58.29.100 (external)
Audio only file format detected.
Load subtitles in /mp3/
==========================================================================
Opening audio decoder: [mpg123] MPEG 1.0/2.0/2.5 layers I, II, III
AUDIO: 48000 Hz, 2 ch, s16le, 32.0 kbit/2.08% (ratio: 4000->192000)
Selected audio codec: [mpg123] afm: mpg123 (MPEG 1.0/2.0/2.5 layers I, II, III)
==========================================================================
AO: [pulse] Init failed: Connection refused
Failed to initialize audio driver 'pulse'
AO: [alsa] 48000Hz 2ch s16le (2 bytes per sample)
Video: no video
Starting playback...
A:   8.0 (08.0) of 251.0 (04:11.0)  0.2%
```

3. Esbrina on es guarden els volums al host amfitrió i comprova que els fitxers baixats hi són. (fes servir "docker inspect"

Per esbrinar on es guarden els volums al host amfitrió, podem utilitzar la comanda "docker inspect". Aquesta comanda ens permet obtenir informació detallada del contenidor, incloent la informació dels volums que estan connectats

```
enti@enti:~$ sudo docker volume inspect musica
[
    {
        "CreatedAt": "2023-05-07T20:15:46+02:00",
        "Driver": "local",
        "Labels": {},
        "Mountpoint": "/var/lib/docker/volumes/musica/_data",
        "Name": "musica",
        "Options": {},
        "Scope": "local"
    }
]
```

Per comprovar-ho, només hem d'accedir a la ruta absoluta que en s mostra (amb l'usuari root ja que si no, no deixa anar-hi)

```
root@enti:/home/enti# cd /var/lib/docker/volumes/musica/_data
root@enti:/var/lib/docker/volumes/musica/_data# ls
music.mp3
```

Comprovem que la música està allà.