

Ejercicios docker 4 Enol Aláez

Trabajando con redes docker

1. Vamos a crear dos redes de ese tipo (BRIDGE) con los siguientes datos: Red1 Nombre: red1 Dirección de red: 172.28.0.0 Máscara de red: 255.255.0.0 Gateway: 172.28.0.1 Red2 Nombre: red2 Es resto de los datos será proporcionados automáticamente por Docker.

```
daw@daw-docker:~$ docker network create red1 --subnet 172.28.0.0/16 --gateway 172.28.0.1
6c34d16ce7936fec6fde72dd147f2ce394a71955b86fa5b80e1640841533b69c
daw@daw-docker:~$ docker network ls
NETWORK ID      NAME      DRIVER      SCOPE
467da3dec607    bridge    bridge      local
36e69cdf563f    host      host        local
756c0e0f1eb5    none      null        local
6c34d16ce793    red1      bridge      local
daw@daw-docker:~$
```

```
docker network create "nombre red" --subnet "subred" --gateway "puerta de enlace"
```

```
daw@daw-docker:~$ docker network create red2
8ea3d1d984187a87b6b5c2020f041e7966d37ee55a49c44504a4fa40b1e29273
daw@daw-docker:~$ docker network ls
NETWORK ID      NAME      DRIVER      SCOPE
467da3dec607    bridge    bridge      local
36e69cdf563f    host      host        local
756c0e0f1eb5    none      null        local
6c34d16ce793    red1      bridge      local
8ea3d1d98418    red2      bridge      local
daw@daw-docker:~$
```

```
docker network create "nombre red"
```

2. Poner en ejecución un contenedor de la imagen ubuntu:20.04 que tenga como hostname host1 , como IP 172.28.0.10 y que esté conectado a la red1. Lo llamaremos u1 .

```
daw@daw-docker:~$ docker run --name contenedor4 --network red1 --ip 172.28.0.10 --hostname host1 ubuntu:20.04
daw@daw-docker:~$ docker ps -a
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS
bb8bd1883b16   ubuntu:20.04  "bash"                  7 seconds ago   Exited (0) 6 seconds ago
```

```
docker run --name contenedor4 --network "nombr red" --ip "direccion ip" --hostname "nombre contenedor host" ubuntu:20.04
```

3. Entrar en ese contenedor e instalar la aplicación ping (apt update && apt install inetutils-ping).

```
daw@daw-docker:~$ docker exec -it contenedor4 bash
root@host1:/# apt update
Get:1 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Get:2 http://archive.ubuntu.com/ubuntu focal InRelease [265 kB]
```

```
"despues de crear "contenedor4""
docker exec -it contenedor4 bash
```

Ejercicios Docker 4

```
root@host1:/# apt install inetutils-ping
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libidn11 netbase
```

"despues de hacer apt-update y apt-upgrade"
apt install inetutils-ping

4. Poner en ejecución un contenedor de la imagen ubuntu:20.04 que tenga como hostname host2 y que esté conectado a la red2. En este caso será docker el que le de una IP correspondiente a esa red. Lo llamaremos u2 .

```
daw@daw-docker:~$ docker run -it --name u2 --network red2 ubuntu:20.04 bash
root@e61b5e3a1dc4:/# exit
exit
daw@daw-docker:~$ doekr ps -a
doekr: orden no encontrada
daw@daw-docker:~$ docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	NAMES	CREATED	STATUS
e61b5e3a1dc4	ubuntu:20.04	"bash"	u2	21 seconds ago	Exited (0) 11 seconds ago

5. Entrar en ese contenedor e instalar la aplicación ping (apt update && apt install inetutils-ping). El documento debe contener, además, los siguientes pantallazos:
 - Pantallazo donde se vea la configuración de red del contenedor u1(red1).

```
"Name": "red1",
"Id": "6c34d16ce7936fec6fde72dd147f2ce394a71955b86fa5b80e1640841533b69c",
"Created": "2023-01-27T08:27:33.722839801+01:00",
"Scope": "local",
"Driver": "bridge",
"EnableIPv6": false,
"IPAM": {
  "Driver": "default",
  "Options": {},
  "Config": [
    {
      "Subnet": "172.28.0.0/16",
      "Gateway": "172.28.0.1"
    }
  ]
},
```

docker network inspect "nombre red"

- Pantallazo donde se vea la configuración de red del contenedor u2(red2).

Ejercicios Docker 4

```
"Name": "red2",
"Id": "8ea3d1d984187a87b6b5c2020f041e7966d37ee55a49c44504a4fa40b1e29273",
"Created": "2023-01-27T08:28:36.400595007+01:00",
"Scope": "local",
"Driver": "bridge",
"EnableIPv6": false,
"IPAM": {
  "Driver": "default",
  "Options": {},
  "Config": [
    {
      "Subnet": "172.18.0.0/16",
      "Gateway": "172.18.0.1"
    }
  ]
},
```

• Pantallazo

donde desde cualquiera de los dos contenedores se pueda ver que no podemos hacer ping al otro ni por ip ni por nombre.

```
"Networks": {
  "red1": {
    "IPAMConfig": {
      "IPv4Address": "172.28.0.10"
    },
    "Links": null,
    "Aliases": [
      "37423f758202",
      "host1"
    ]
  },
}
```

```
daw@daw-docker:~$ docker exec -it u2 bash
root@e61b5e3a1dc4:/# ping 172.28.0.10
PING 172.28.0.10 (172.28.0.10): 56 data bytes
^C--- 172.28.0.10 ping statistics ---
6 packets transmitted, 0 packets received, 100% packet loss
root@e61b5e3a1dc4:/#
```

• Pantallazo donde se pueda comprobar que si conectamos el contenedor u1 a la red2 (con docker network connect), desde el contenedor u1, tenemos acceso al contenedor u2 mediante ping, tanto por nombre como por ip.

```
daw@daw-docker:~$ docker network connect red2 contenedor4
daw@daw-docker:~$
```

```
docker network connect "nombre red" "nombre contenedor"
```

```

"Networks": {
  "red1": {
    "IPAMConfig": {
      "IPv4Address": "172.28.0.10"
    },
    "Links": null,
    "Aliases": [
      "37423f758202",
      "host1"
    ],
    "NetworkID": "6c34d16ce7936fec6fde72dd147f2ce394a71955b86fa5b80e1640841533",
    "EndpointID": "515a0e4efe3374e6dc8011b0d40993ff9d9cd45fd12f2c2a760452ff918",
    "Gateway": "172.28.0.1",
    "IPAddress": "172.28.0.10",
    "IPPrefixLen": 16,
    "IPv6Gateway": "",
    "GlobalIPv6Address": "",
    "GlobalIPv6PrefixLen": 0,
    "MacAddress": "02:42:ac:1c:00:0a",
    "DriverOpts": null
  },
  "red2": {
    "IPAMConfig": {},
    "Links": null,
    "Aliases": [
      "37423f758202",
      "host1"
    ],
    "NetworkID": "8ea3d1d984187a87b6b5c2020f041e7966d37ee55a49c44504a4fa40b1e2",
    "EndpointID": "6f9fadd99592396485346fdfffb3dcc4b7002b6dc3e185e19df56e2cd76",
    "Gateway": "172.18.0.1",
    "IPAddress": "172.18.0.3",
    "IPPrefixLen": 16,
    "IPv6Gateway": "",
    "GlobalIPv6Address": "",
    "GlobalIPv6PrefixLen": 0,
    "MacAddress": "02:42:ac:12:00:03",
    "DriverOpts": {}
  }
}

```

```

daw@daw-docker:~$ docker exec -it u2 bash
root@e61b5e3a1dc4:/# ping host1
PING host1 (172.18.0.3): 56 data bytes
64 bytes from 172.18.0.3: icmp_seq=0 ttl=64 time=0.134 ms
64 bytes from 172.18.0.3: icmp_seq=1 ttl=64 time=0.214 ms
64 bytes from 172.18.0.3: icmp_seq=2 ttl=64 time=0.147 ms
^C--- host1 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.134/0.165/0.214/0.035 ms
root@e61b5e3a1dc4:/# ping 172.18.0.3
PING 172.18.0.3 (172.18.0.3): 56 data bytes
64 bytes from 172.18.0.3: icmp_seq=0 ttl=64 time=0.303 ms
64 bytes from 172.18.0.3: icmp_seq=1 ttl=64 time=0.215 ms
64 bytes from 172.18.0.3: icmp_seq=2 ttl=64 time=0.213 ms
^C--- 172.18.0.3 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.213/0.244/0.303/0.042 ms
root@e61b5e3a1dc4:/#

```

Despliegue de Nextcloud + mariadb/postgreSQL

Vamos a desplegar la aplicación nextcloud con una base de datos (puedes elegir mariadb o PostgreSQL) (NOTA: Para que no te de errores utiliza la imagen mariadb:10.5). Te puede servir el ejercicio que hemos realizado para desplegar Wordpress . Para ello sigue los siguientes pasos:

Ejercicios Docker 4

1. Crea una red de tipo bridge.

```
daw@daw-docker:~$ docker network create red3
52de1227971ac2a193b7fdb2ce7fee0f50225fffd3b3dae5fad0c8ec37dc184
daw@daw-docker:~$ docker network inspect red3
[
  {
    "Name": "red3",
    "Id": "52de1227971ac2a193b7fdb2ce7fee0f50225fffd3b3dae5fad0c8ec37dc184",
    "Created": "2023-02-01T09:15:37.242393781+01:00",
    "Scope": "local",
    "Driver": "bridge",
    "EnableIPv6": false,
    "IPAM": {
      "Driver": "default",
      "Options": {},
      "Config": [
        {
          "Subnet": "172.19.0.0/16",
          "Gateway": "172.19.0.1"
        }
      ]
    }
  }
],
```

2. Crea el contenedor de la base de datos conectado a la red que has creado. La base de datos se debe configurar para crear una base de dato y un usuario. Además el contenedor debe utilizar almacenamiento (volúmenes o bind mount) para guardar la información. Puedes seguir la documentación de mariadb o la de PostgreSQL .

```
daw@daw-docker:~$ docker run -d --name mariadb2 --env MARIADB_USER=usuario --env MARIADB_PASSWORD=
temporal --env MARIADB_ROOT_PASSWORD=temporal -v volumensql:/var/run/mysql --network red3 -p 8080
:80 mariadb:10.5
de94bd215e484b6c884f5f5008ca054b56c3005fcfb291a21ec37d6f29116879
daw@daw-docker:~$
```

```
"docker run -d --name "nombre contenedor mariadb" --env MARIADB_USER="nombre usuario" --env
MARIADB_PASSWORD="contraseña" --env MARIADB_ROOT_PASSWORD="contraseña del root" -v
volumensql:/var/mysql --network "nombre red" -p "puerto (8080:80)" mariadb:10.5
```

3. A continuación, siguiendo la documentación de la imagen nextcloud , crea un contenedor conectado a la misma red, e indica las variables adecuadas para que se configure de forma adecuada y realice la conexión a la base de datos. El contenedor también debe ser persistente usando almacenamiento.

```
daw@daw-docker:~$ docker volume create nextcloud1
nextcloud1
daw@daw-docker:~$ docker run -d -v nextcloud1:/var/www/html --network red3 --env MYSQL_DATABASE=ma
riadb2 -p 8081:80 nextcloud
Unable to find image 'nextcloud:latest' locally
latest: Pulling from library/nextcloud
8740c948ffd4: Already exists
1873be858264: Pull complete
7ce6a163d8c1: Pull complete
008a172010ba: Pull complete
d15353ae3d77: Pull complete
223eb1888c0f: Pull complete
83374c2a967a: Pull complete
8fdc86711b26: Pull complete
23c0224c39b8: Pull complete
915d82c7f5c5: Pull complete
dc037a9c9035: Pull complete
768542e0b637: Pull complete
d7ade602d94f: Pull complete
7361225e9a5d: Pull complete
fdc75c5d6478: Pull complete
6e598d96642e: Pull complete
2183a95f6531: Pull complete
1b9c17dacd16: Pull complete
ac35d0e862c3: Pull complete
5f2db432cae4: Pull complete
Digest: sha256:aa43a87b0b6acd52c1989dc343d1702f6a9d3ae1be263e62a9d0f99ec10f29a7
Status: Downloaded newer image for nextcloud:latest
42566ad9693d9f746b27fe6f2332afbdeab54de7cdd62bd282a70284c3775dcc
daw@daw-docker:~$
```

```
docker run -d -v nextcloud1:/var/www/html --netowrk "nombre red" --env
MYSQL_DATABASE="nombre contenedro database" -p "puerto" nextcloud:latest
```

4. Accede a la aplicación usando un navegador web. El documento debe contener, además, los siguientes pantallazos:

·Pantallazo donde se ve el acceso a la aplicación desde un navegador web.



