Ejercicios

Trabajar 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.

Creamos las 2 redes con los datos del ejercicio:

```
docker network create -d bridge --subnet 172.28.0.0/16 --gateway 172.28.0.1 red1
```

```
docker network create -d bridge red2
```

Hacemos un docker `network inspect` para ver la información de las redes:

```
daw@daw-docker:~$ docker inspect red1
[
    {
        "Name": "red1",
        "Id": "598cd24ab868a21462eed97c2bcb930339c0cab36fabbdb2be8983becd093684",
        "Created": "2023-01-27T09:16:18.376890801+01:00",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": {},
            "Config": [
                {
                    "Subnet": "172.28.0.0/16",
                    "Gateway": "172.28.0.1"
                }
            ]
        },
```

```
daw@daw-docker:~$ docker inspect red2
[
    {
        "Name": "red2",
        "Id": "e3eed7b1a01aaaf9c42a606e173fee928d93d0064cd7b5123ab73263a53d8cbf",
        "Created": "2023-01-27T09:16:46.389724949+01:00",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": {},
            "Config": [
                {
                    "Subnet": "172.18.0.0/16",
                    "Gateway": "172.18.0.1"
                }
            ]
        },
```

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`.

Arrancamos el contenedor con los datos solicitados:

```
docker run -it --name u1 --network red1 --hostname host1 --ip 172.28.0.10
ubuntu:20.04
```

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

Instalamos las inetutils para poder usar el comando ping:

```
root@host1:/# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8): 56 data bytes
64 bytes from 8.8.8.8: icmp_seq=0 ttl=103 time=20.129 ms
64 bytes from 8.8.8.8: icmp_seq=1 ttl=103 time=17.639 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=103 time=18.103 ms
--- 8.8.8.8 ping statistics ---
```

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` .

Arrancamos 2º contenedor:

```
docker run -it --name u2 --network red2 --hostname host2 ubuntu:20.04
```

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

Instalamos inetutils en el 2º contenedor y comprobamos que funciona:

```
root@host2:/# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8): 56 data bytes
64 bytes from 8.8.8.8: icmp_seq=0 ttl=103 time=18.188 ms
64 bytes from 8.8.8.8: icmp_seq=1 ttl=103 time=18.464 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=103 time=17.585 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=103 time=15.980 ms
```

El documento debe contener, además, los siguientes pantallazos:

- Pantallazo donde se vea la configuración de red del contenedor u1.

```
root@host1:/# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.28.0.10  netmask 255.255.0.0  broadcast 172.28.255.255
        ether 02:42:ac:1c:00:0a  txqueuelen 0  (Ethernet)
        RX packets 116  bytes 205400 (205.4 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 84  bytes 4701 (4.7 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 4  bytes 447 (447.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 4  bytes 447 (447.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

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

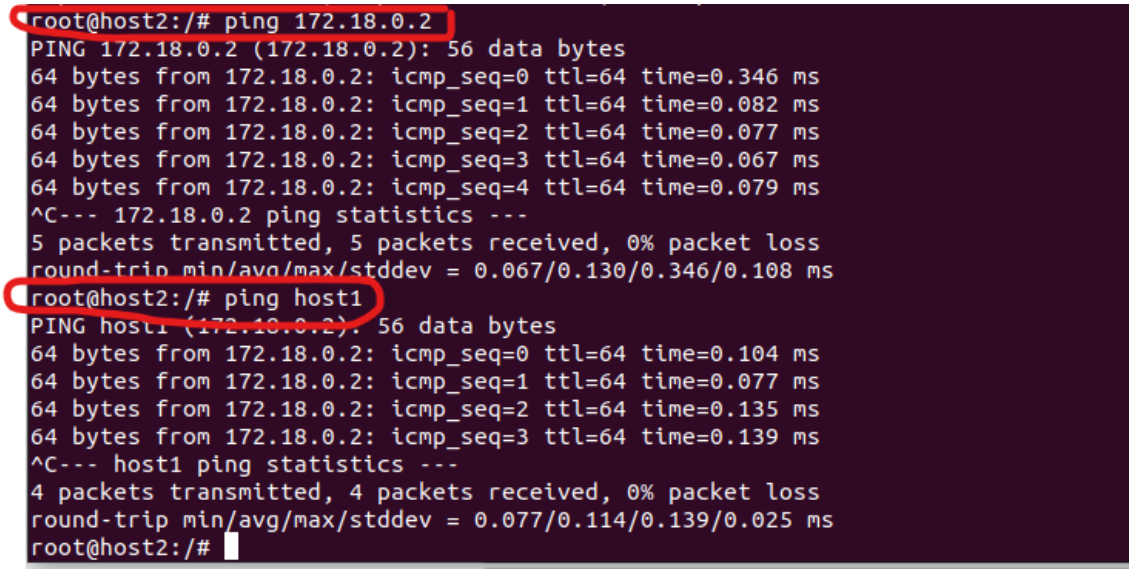
```
root@host2:/# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.18.0.2  netmask 255.255.0.0  broadcast 172.18.255.255
        ether 02:42:ac:12:00:02  txqueuelen 0  (Ethernet)
        RX packets 112  bytes 205184 (205.1 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 75  bytes 4215 (4.2 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 4  bytes 447 (447.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 4  bytes 447 (447.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

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

```
root@host2:/# ping 172.28.0.10
PING 172.28.0.10 (172.28.0.10): 56 data bytes
^C--- 172.28.0.10 ping statistics ---
15 packets transmitted, 0 packets received, 100% packet loss
root@host2:/# ping u1
ping: unknown host
```

- 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.



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:

1.Crea una red de tipo bridge.

Creamos la red:

```
docker network create -d bridge red1
```

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.

```
docker run -d --name myMariadb --network red1 -v /opt/mysql_javi:/var/lib/mysql -e MYSQL_DATABASE=cosas -e MYSQL_USER=javi -e MYSQL_PASSWORD=temporal -e MYSQL_ROOT_PASSWORD=temporal mariadb
```

```
daw@daw-docker:~$ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS      NAMES
05d33fee306f   mariadb   "docker-entrypoint.s…"   22 seconds ago   Up 21 seconds   3306/tcp   myMariadb
daw@daw-docker:~$
```

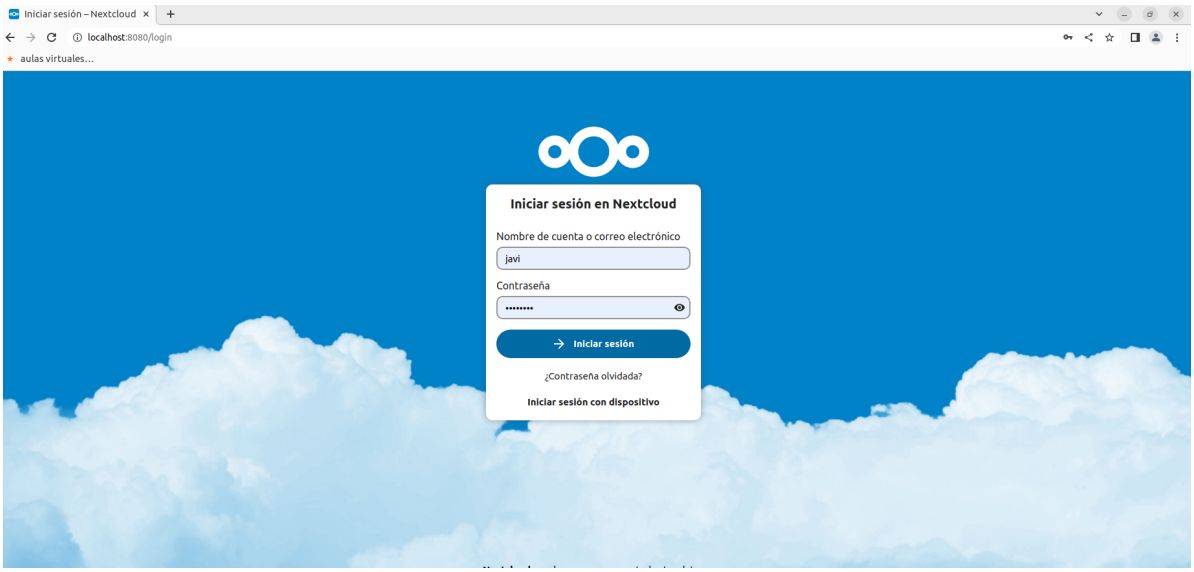
3.A continuación, siguiendo la documentación de la imagen nextcloud, crea un contenedor conectado a la misma red, e indica las variables adecuadas para que se configure de forma adecuada y realice la conexión a la base de datos. El contenedor también debe ser persistente usando almacenamiento.

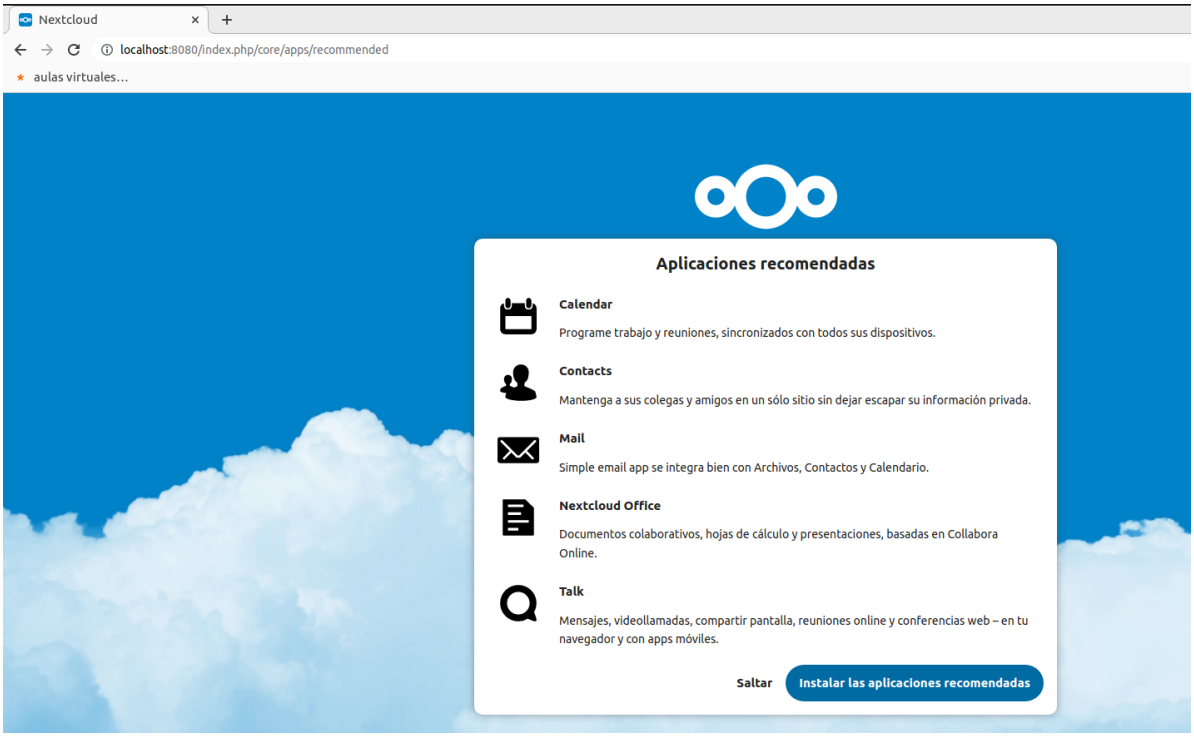
```
docker run -d --name myNextCloud -p 8080:80 --network red1 -v nextcloud:/var/www/html nextcloud
```

```
daw@daw-docker:~$ docker ps
CONTAINER ID   IMAGE       COMMAND                  CREATED          STATUS          PORTS                                   NAMES
957e5ea1d145   nextcloud   "/entrypoint.sh apac…"   5 minutes ago    Up 5 minutes    0.0.0.0:8080->80/tcp, :::8080->80/tcp   myNextCloud
05d33fee306f   mariadb     "docker-entrypoint.s…"   10 minutes ago   Up 10 minutes   3306/tcp                                myMariadb
daw@daw-docker:~$
```

4.Accede a la aplicación usando un navegador web.

El documento debe contener, además, los siguientes pantallazos:




Nextcloud
localhost:8080/index.php/core/apps/recommended
aulas virtuales...
Aplicaciones recomendadas
Calendar
Programe trabajo y reuniones, sincronizados con todos sus dispositivos.
Contacts
Mantenga a sus colegas y amigos en un sólo sitio sin dejar escapar su información privada.
Mail
Simple email app se integra bien con Archivos, Contactos y Calendario.
Nextcloud Office
Documentos colaborativos, hojas de cálculo y presentaciones, basadas en Collabora Online.
Talk
Mensajes, videollamadas, compartir pantalla, reuniones online y conferencias web – en tu navegador y con apps móviles.
Saltar
Instalar las aplicaciones recomendadas