Sistemas de gestión empresarial

Dockers. Prácticas iniciales con contenedores

En esta práctica vamos a poner en marcha un contenedor, con **LAMPP y wordpress** a partir de una imagen de **ubuntu**.

1. LAMP desde un contenedor con Ubuntu

1. Creación del contenedor

A partir de una imagen de ubuntu, versión 22.04, creamos un contenedor que expone el puerto 80 asociándolo al 8080 de nuestro equipo anfitrión. Dejamos una shell abierta para poder instalar todo lo necesario

```
docker run -it -p 8080:80 --name LAMP ubuntu:22.04 /bin/bash
```

2. Instalar LAMP

Seguimos las instrucciones que nos indica este artículo de *itzgeek.com*

Instalación de LAMP y wordpress

(En el shell del contenedor LAMP)

```
# actualización
apt update

# instalación de los paquetes de wordpress, apache y mariadb (tarda un rato)
apt install wordpress php libapache2-mod-php mariadb-server php-mysql

# arrancar el servicio apache
service apache2 start
```

Podemos probar que funciona en el navegador con http://localhost:8080

3. Configurar Apache para usar Wordpress

Hay que instalar un editor como nano para poder trabajar con los ficheros de configuración.

```
apt install nano
```

Hay que crear el fichero de configuración del sitio en Apache **/etc/apache2/sites-available/wordpress.conf** que configura el acceso al sitio Wordpress. Este fichero no existe, se puede crear con *nano*.

```
Alias /blog /usr/share/wordpress
<Directory /usr/share/wordpress>
    Options FollowSymLinks
    AllowOverride Limit Options FileInfo
    DirectoryIndex index.php
    Order allow,deny
    Allow from all
</Directory>
<Directory /usr/share/wordpress/wp-content>
    Options FollowSymLinks
    Order allow,deny
    Allow from all
</Directory>
```

Sin embargo, resulta más cómodo crearlo fuera del contenedor y copiarlo con **docker cp**

```
docker cp wordpress.conf LAMP:/etc/apache2/sites-available
```

Una vez hecho, cargamos el sitio en Apache con las órdenes

```
a2ensite wordpress

a2enmod rewrite

service apache2 restart
```

Al probar con http://localhost:8080/blog vemos un error, pero significa que está respondiendo.

4. Configurar MariaDB Server

Arrancamos el servicio de MariaDB

```
service mariadb start
```

Hay que generar la configuración de seguridad de acceso a MariaDB

```
mysql_secure_installation

(pass) mariadb
(unix_socket_authentication) n
(change pass) n
(remove anonymous) n
(disallow remote) Y
(BD test) n
(Privileges) Y

All done! ....
```

Accedemos a MariaDB para crear la base de datos **wordpress**

```
root@5dd6f29b905d:/# mysql -u root -p
Enter password: (mariadb)

MariaDB [(none)]> CREATE DATABASE wordpress;
MariaDB [(none)]> CREATE USER 'wordpress'@'%' IDENTIFIED BY 'wordpress2024';
MariaDB [(none)]> GRANT ALL PRIVILEGES ON wordpress.* TO 'wordpress'@'%' WITH GRANT OPTION;
MariaDB [(none)]> FLUSH PRIVILEGES;
```

Salimos con *quit;*

5. Configurar Wordpress

Para este punto creamos un fichero **config-localhost.php** que sirve de configuración al servidor de Wordpress para la conexión a la base de datos de MariaDB.

Lo más sencillo es editarlo de forma externa al contenedor y copiarlo en el directorio del contenedor **/etc/wordpress**

El contenido del archivo *config-localhost.php* es

```
<?php
define('DB_NAME', 'wordpress');
define('DB_USER', 'wordpress');
define('DB_PASSWORD', 'wordpress2024');
define('DB_HOST', 'localhost');
define('DB_COLLATE', 'utf8_general_ci');
define('WP_CONTENT_DIR', '/usr/share/wordpress/wp-content');
?>
```

El siguiente paso es conectarse al navegador para configurar el sitio wordpress.

Guarda la configuración para futuras conexiones

6. Arranque de servicios al iniciar el contenedor

Vamos a configurar nuestro contenedor LAMP para que pueda lanzar los servicios al iniciarse. Generalmente, por el bajo coste de poner en marcha un contenedor, los sistemas Docker están pensados para ejecutar mono-servicios.

Los contenedores multiservicio son la excepción y este caso práctico ha sido presentado con un fin didáctico y no de uso real.

Las premisas que indica Docker propone que si necesitamos lanzar varios servicios, lancemos como comando la ejecución de un script que lance los servicios que necesitemos.

En este caso práctico nuestro contenedor está configurado para ejecutar al iniciarse la shell *bash*, vamos a editar el fichero de configuración *.bashrc* con el fin de que al iniciar la shell, se lancen los servicios Apache y MySQL.

En el terminal del contenedor editamos el archivo */root/.bashrc* y añadimos estas líneas al final del archivo

```
nano /root/.bashrc

(al final del archivo)
# Inicio de los servicios de Apache y MariaDB

service apache2 start
service mariadb start
```

7. Comprobación de funcionamiento correcto

Para comprobar que todo ha funcionado correctamente, vamos a parar el contenedor, ponerlo de nuevo en marcha y comprobar que podemos acceder a nuestro Wordpress en http://localhost:8080/blog.

Para parar y lanzar el contenedor utilizaremos los comandos:

```
docker stop LAMP
docker start LAMP
```

Para acceder a la administración del sitio la url es *http://localhost:8080/blog/wp-login.php*

2. Servicio VNC en un navegador

En este ejemplo pondremos en marcha un servicio VNC (servicio de administración remota) junto con un cliente NoVNC (Cliente para VNC en HTML5 y JavaScript) servido vía web.

Creamos un contenedor a partir de la imagen descrita en https://github.com/theasp/docker-novnc, que lo tiene todo listo para funcionar.

Lo lanzamos con la siguiente orden

```
docker run -it -p 8080:8080 theasp/novnc
```

Accedemos a la interfaz a través de la siguiente URL http://localhost:8080/vnc.html y se nos cargará un cliente NoVNC (Cliente VNC en HTML5 y Javascript).

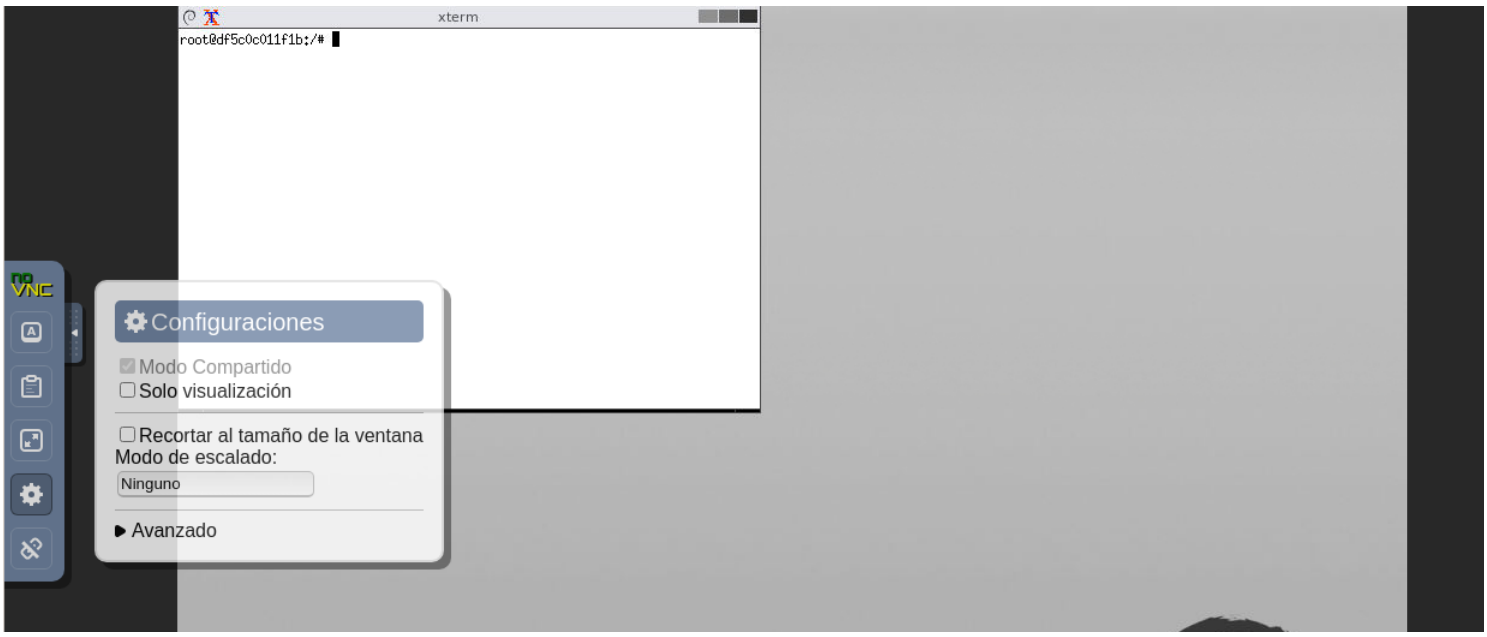

Figure 1: Interfaz gráfica del contenedor NoVNC