Crear un contenedor con Docker y meter una imagen de nuestro servidor.

Docker Avanzado

Servidor

Rubén Morante González

Tabla de contenido

Tabla de ilustraciones

1. INTRODUCCIÓN

Docker es un proyecto de código abierto que automatiza el despliegue de aplicaciones dentro de contenedores software, proporcionando una capa adicional de abstracción y automatización de Virtualización a nivel de sistema operativo en Linux. Docker utiliza características de aislamiento de recursos del kernel Linux, tales como cgroups y espacios de nombres (namespaces) para permitir que "contenedores" independientes se ejecuten dentro de una sola instancia de Linux, evitando la sobrecarga de iniciar y mantener máquinas virtuales. [1]

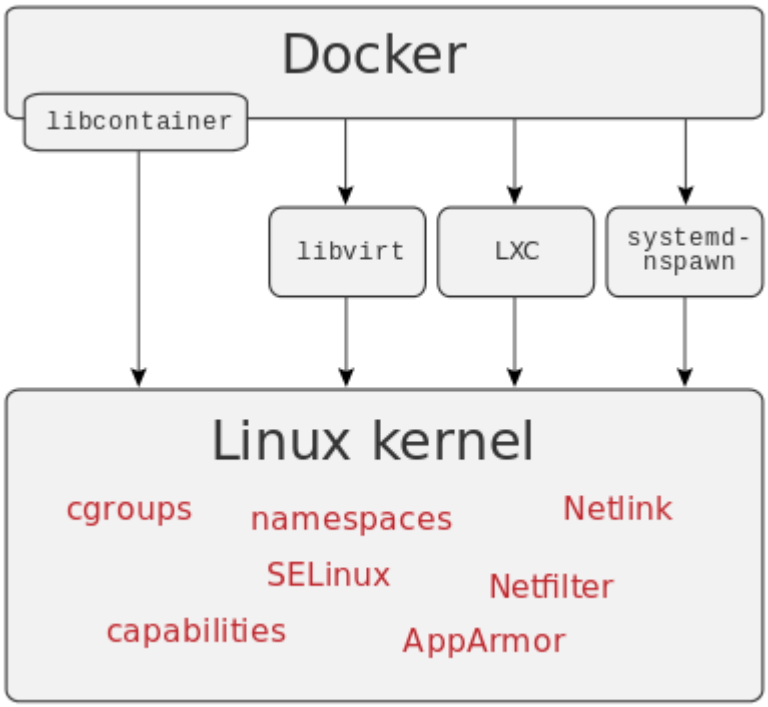

Ilustración 11 Docker, Wikipedia

Docker nos ayuda a no malgastar nuestro tiempo configurando el entorno, y las dependencias del sistema, porque lo vamos a poder desplegar fácilmente, algo muy útil tanto para grandes empresas como para las pequeñas startups.

Docker Hub es un repositorio de contenedores de Docker, que permite a los usuarios compartir las imágenes construidas, se podría decir que es el GitHub de los contenedores Docker y quizá por ello el paralelismo en el nombre entre ambos. Docker Hub permite subir imágenes o usar las imágenes oficiales de postgresql, redis, mysql, ubuntu, rabbitmq, … y otra multitud de proyectos.

El archivo Dockerfile con el que construimos una imagen podemos hospedarlo en un repositorio de GitHub y que Docker Hub lo obtenga para construir la imagen. Docker Hub ofrece repositorios públicos en los que colocar las imágenes que cualquier otro usuario puede acceder y usar o repositorios privados con cierto coste según el número de repositorios privados, el primer repositorio privado es gratuito.[2]

2. DESARROLLO

En este apartado se explica cómo se ha llevado a cabo el desarrollo de la práctica de Docker Avanzado.

2.1. Entorno utilizado

- Asus con Windows 10, 8 Gb de RAM
- Asus con Kali Linux, 4 Gb de RAM
- Spring Tool Suite para el desarrollo del código Java
- Docker Hub

2.2. Desarrollo de la práctica

2.2.1. Punto de partida

Partiendo de la práctica de Docker, en la cual creamos un contenedor con una imagen de MySql, vamos a crear otro contenedor con la imagen de nuestro servidor para posteriormente subirla a Docker Hub. Para ello se realizarán una serie de cambios en el proyecto servidor, cambiando el pom.xml, el archivo application.yml y creando el archivo Dockerfile.

En el pom.xml se añade 'packaging' jar:

<groupId>com.iwcn</groupId>
<artifactId>TiendaProducto</artifactId>
<version>0.0.1-SNAPSHOT</version>
<packaging>jar</packaging>

En el archivo *application.yml* cambiamos el ddl-auto de *'validate'* a *'create'*:

ddl-auto: validate → *ddl-auto: create*

Creamos en la raíz del proyecto servidor el archivo Dockerfile sin ninguna extensión con el siguiente contenido:

FROM java:8
MAINTAINER Ruben Morante Gonzalez
EXPOSE 8080
VOLUME /tmp
ADD /target/TiendaProducto-0.0.1-SNAPSHOT.jar practicaruben.jar
ENTRYPOINT ["java", "-jar", "practicaruben.jar"]

Una vez realizados todos estos cambios, se debe realizar un Maven Build con los siguientes comandos:

clean install verify package

2.2.2. Crear la imagen Docker

Abrimos la terminal y nos situamos en la carpeta raíz del proyecto servidor, donde hemos situado el archivo Dockerfile. Una vez ahí se ejecuta las siguientes instrucciones:

docker build -t practicaruben .

docker run --name practicaruben -p 8080:8080 --link rubensql -d practicaruben

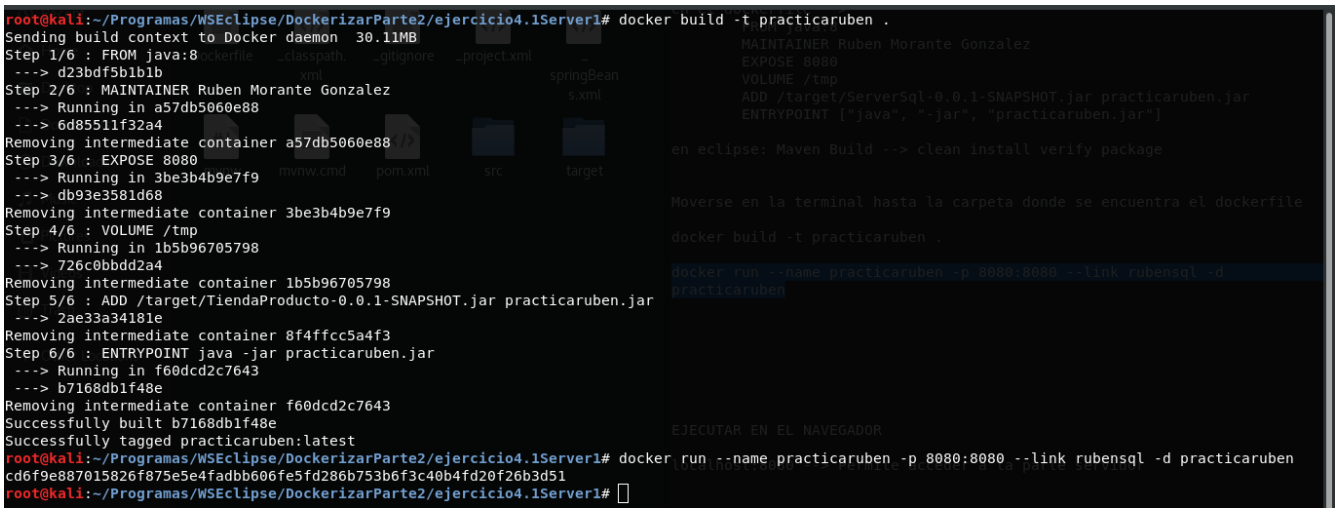

Ilustración 2 Crear la imagen Docker del servidor

Por lo que tenemos un contenedor con el nombre '*practicaruben*' con la imagen de nuestro servidor enlazada a la imagen de la base de datos '*rubensql*'.

Por lo que si escribimos en el navegador *localhost.8080* se tendrá acceso a la parte servidor.

2.2.3. Subir la imagen a Docker Hub

Se debe crear una cuenta de Docker Hub en la página https://hub.docker.com/. Una vez creada se añade un nuevo repositorio, en este caso 'ruben'.

Hecho esto se abre un terminal y ejecutamos el comando '*docker login':*

Docker login

Ahora subiremos la imagen del servidor y de la base de datos al repositorio local:

docker tag practicaruben evaurjc2017/ruben:servidor

docker tag rubensql evaurjc2017/ruben:basedatos

Una vez almacenadas las imágenes en el repositorio local, las subimos a Docker Hub:

docker push evaurjc2017/ruben

3. CONCLUSIÓN

Al utilizar un repositorio de Docker podemos automatizar nuestras pruebas de testing, por lo que reducimos el tiempo. También favorece que podamos encapsular todo el entorno de trabajo de manera que los desarrolladores saben que pueden estar trabajando en su servidor local, con la seguridad de que, al llegar el momento de ponerlo en producción, van a estar ejecutándose con la misma configuración sobre la que se han hecho todas las pruebas.

Al estar en un repositorio, cualquiera puede contribuir a las imágenes, de forma que fomenta la filosofía Open Source.

ANEXO 1: REFERENCIAS

[1] Docker, Wikipedia, Docker Software, https://es.wikipedia.org/wiki/Docker_(software)

[2] Docker Hub, Blog Bitix, Crear y usar un repositorio en Docker Hub, https://picodotdev.github.io/blog-bitix/2015/07/crear-y-usar-un-repositorio-en-docker-hub/