Docker y despliegue de apps con tomcat

Bernat Costa

Docker como herramienta de trabajo

Docker nos permitirá "virtualizar" varias bbdd en un mismo equipo

Un contenedor por servicio

Podremos tener varios mysql, sqlservers, apaches... en una misma máquina funcionando en puertos distintos

¿Porque docker y no VirtualBox?

- Más ágil de levantar máquinas
- No replica todo el SO, el kernel se aprovecha el del anfitrión
- Contenedores más libianos que Maquinas virtuales
- Contenedores preconfigurados.

¿Que debo saber de docker?

- Imagenes y contenedores
- Puertos
- Variables de entorno
- Volumenes

Imagenes y contenedores

Con docker vamos a construir contenedores a partir de una imagen.

La imagen es la base de nuestro contentenedor.

Docker nos garantiza que una imagen, se comportará igual en cualquier host.

Los contenedores son volatiles. Cuando se paran y destruyen, los datos se pierden.

Un contenedor corre un servicio (una bbdd, un servidor web, un servidor de correo....)

Parámetros de configuración

¿De donde sacamos las imagenes?

Las podemos crear con un fichero Dockerfile y un comando de docker (docker build).

A diferencia del año pasado, vamos a centrarnos en la creación de imagenes y publicarlas

Hub.docker.com

El Fichero Dockerfile

- Es un fichero de texto que pondremos en nuestro proyecto web, en el raiz.
- Cuando lancemos el comando docker build . Estaremos ejecutando los comandos de ese fichero.
- Deberemos seleccionar una imagen de partida, con el comando FROM, luego con el comando COPY podremos añadir ficheros de nuestro ordenador a nuestra imagen, para luego poder exponer un puerto con el EXPOSE

Exemplo de Dockerfile

- FROM ubuntu/apache2
- COPY ./index.html /var/www/html/
- EXPOSE 80
-

Dockerfile para compilar y desplegar un proyecto con java con maven.

- FROM maven:3-openjdk-18 as builder
- RUN mkdir -p /build
- WORKDIR /build
- COPY pom.xml /build
- RUN mvn -B dependency:resolve dependency:resolve-plugins
- COPY src /build/src
- RUN mvn package

- FROM tomcat:8.5-jdk11-openjdk-slim
- COPY --from=builder /build/target/daw.war /usr/local/tomcat/webapps/
- EXPOSE 80
- CMD ["catalina.sh", "run"]

El orden en el Dockerfile

- Docker estructura sus contenedores en capas. Cada línea del Dockerfile es una capa.
- Si cambiamos algo en el código de un fichero que se añade a nuestra imagen al principio, volverá a realizar todas las líneas siguientes del Dockerfile.
- Si cambiamos algo en el código de un fichero que se añade al final, solo rehará aquellas líneas posteriores al cambio de ese fichero.
- Conclusión, hacer los apt installs o resolver las dependencias mvn antes de pasar nuestro código. Eso va a tardar y mejor hacerlo una vez y no cada vez que tocamos una línea de código.

Docker push

- Con docker push, podremos publicar nuestra imagen en docker hub.
- Antes tendremos que crear el repositorio en docker hub y ponerle el mismo nombre a la imagen.
- Ejemplo
  - Mi repositorio en docker hub se llama minombre/webestatica
  - Tendré que lanzar:
    - Docker build . -t minombre/webestatica
    - Docker push
  - Con docker image ls podemos ver las imágenes en nuestro ordenador. Si no le pasamos el –t y un nombre, docker nos asigna un nombre al azar.

¿Cómo corremos un contenedor?

 Arrancar un apache:	`docker run  httpd`
 En background	`docker run –d httpd`
 Ponerle un nombre al contenedor	`docker run -d --name apachealpine  httpd`
 Tags	`docker run -d --name apachealpine  httpd:alpine`

Que podemos hacer con un contenedor

- Ver la lista de contenedores corriendo
  - Docker ps
- Ver el log de un contenedor
  - Docker logs nombredelcontenedor (o el hash)
- Parar/arrancar/borrar un contenedor:
  - Docker stop/start/rm

Puertos

- Para que el contenedor sea útil, debemos exponer puertos del contenedor en la máquina anfitrión.
- ¿De que nos sirve un apache si no nos da una web?
- Con el parámetro -p "enlazamos" el puerto de nuestra máquina con el del contenedor.
  - `docker run -d --name apachealpine -p 81:80 httpd:alpine`
- IMPORTANTE: cada contenedor, tendrá sus puertos abiertos. Mirar la documentación en hub.docker.com

Variables de entorno del contenedor

- Hay imagenes que te permiten pasarle variables de entorno, por ejemplo, el pass de root de un sgbbdd.
  - `docker run -d --name mysql -e MYSQL_ROOT_PASSWORD=root -p 3306:3306 mysql:5.7.32`
- Ahora, con un cliente de mysql, podriamos conectarnos al localhost:3306 con password root.

Volúmenes

Los contenedores son volatiles. No son persistentes

Las bbdd son persistentes.

¿Como podemos montar una bbdd en un contenedor volatil?

Parámetro v

```
docker run -d  --name mysql -e
MYSQL_ROOT_PASSWORD=root -p 3306:3306  -v
mysql_data:/var/lib/mysql mysql:5.7.32
```

Docker-compose

- Es una herramienta para levantar y parar contenedores de forma amigable
- Son scripts, donde pondremos toda la configuración, y así nos ahorramos de poner todos los comandos.
- Se basa en un fichero docker-compose.yml
- Se arranca/para con
  - `Docker-compose up -d`
  - Docker-compose down

Ejemplos docker compose

Un wordpress con su bbdd

Una bbdd Microsoft SQLServer persistente.

Docker compose con línea de backup

docker-compose.yml

```yaml
version: "3.2"
services:

  sql-server-db:
    container_name: sql-server-db
    image: mcr.microsoft.com/mssql/server:2017-latest
    ports:
      - "1533:1433"
    environment:
      SA_PASSWORD: "12345Ab##"
      ACCEPT_EULA: "Y"
    volumes:
        - ./backup:/var/opt/mssql/backup
        - ./data:/var/opt/mssql/data

```

Ejercicio

- Crea un contenedor propio con una web tuya estática.
- Pruébalo en tu máquina
- Publícala en docker hub.

Tomcat con docker

- docker run -p 8080:8080 --name tomcat tomcat:7-alpine
- Alpine son buenas versiones ya que llevan un gnu/linux muy ligero.
- Algunas versiones de Tomcat, vienen con una pag de inicio con un login para poder gestionar nuestras apps java cargadas previamente.
    - 7-alpine Si lo trae
    - Latest no lo trae.
- Para poder acceder al panel interno de tomcat, necesitamos un usuario y contraseña que se configura desde un fichero XML interno del contenedor. Podremos tunearlo copiando ese fichero en un Dockerfile.

Ejemplo Dockerfile con carga de usuarios propia

- Pondremos un fichero XML personalizado en nuestro proyecto
- Dockerfile
  - FROM tomcat:7-alpine
  - COPY tomcat/tomcat-users.xml /usr/local/tomcat/conf/
  - EXPOSE 8080
  - CMD ["/usr/local/tomcat/bin/catalina.sh", "run"]
- Carpeta tomcat y fichero tomcat-users.xml

```
<tomcat-users>
  <role rolename="manager-gui"/>
  <role rolename="manager-script"/>
  <user username="tomcat" password="s3cret" roles="manager-gui,manager-script"/
</tomcat-users>
```

Copiar ficheros de configuración en contenedores Docker

- Esta "técnica" de copiar ficheros de configuración a nuestro contenedor, puede ser muy útil cuando queramos cambiar algún parametro del mismo y no encontremos una versión ya preconfigurada a nuestro gusto.
- Siempre podremos añadir esos ficheros de conf a nuestro proyecto y en el DockerFile copiarlos donde deban estar dentro del contenedor.

Servidores web seguros: HTTPS

HTTP vs HTTPS

- El protocolo http funciona en el puerto 80, el https en el 443.
  - (Tendremos que tener eso en cuenta en el firewall)
- Los dos son protocolos para servir webs. El navegador nos los reconoce como puerto por defecto y no es necesario especificar los puertos.
- La principal diferencia es que con http, el tráfico no va cifrado. Por lo tanto, toda la información que mandemos por formularios, viaja por la red SIN CIFRAR
  - Es relativamente fácil ver contraseñas, envios de formularios... de todo lo que vaya sin cifrar por una misma red.
  - Si instagram usara http, ahora mismo podriamos ver el usuario y contraseña de TODAS las personas que se conecten a instagram en CESUR CARTUJA.

HTTPS, S de Seguro

- Con HTTPS, los datos enviados al servidor se cifran. Ya no pueden ser leidos por un tercero.
- Para ello, necesitamos de unas claves de cifrado que deben ser configuradas en el servidor.
- Vamos a ver dos formas:
  - Desde una instalación de Apache
  - Desde Docker con otro servidor web, nginx. (una alternativa a apache)

Instalación de certificados en Apache

- Tenemos que tener instalado apache2 y openssl
  - Sudo apt install apache2 openssl
- Apache disponie de varios módulos que se pueden habilitar. Uno de ellos para el https
  - A2enmod ssl
- Hay que editar el fichero de configuración de apache
  - sudo nano/etc/apache2/apache2.conf
- Y añadir al final:
- <Directory /var/www/html>
- AllowOverride All
- </Directory>

Crear una clave y un certificado

- Creamos una carpeta para mis certificados
  - mkdir /etc/apache2/certificate
  - cd /etc/apache2/certificate
- Y lanzamos el comando siguiente para crear los certificados para nuestro servidor.
  - openssl req -new -newkey rsa:4096 -x509 -sha256 -days 365 -nodes -out apache-certificate.crt -keyout apache.key
- Debes rellenar la información que te solicita a continuación, en la parte de COMMON_NAME, deberemos introducir la ip de nuestro servidor o el dominio que estemos usando.

Configurar mis virtual host

- Editando el fichero de mi virtualhost
- Nano /etc/apache2/sites-available/000-default.conf

- Deberia ver algo parecido a esto
- <VirtualHost *:80>
- ServerAdmin webmaster@localhost
- DocumentRoot /var/www/html
- ErrorLog ${APACHE_LOG_DIR}/error.log
- CustomLog ${APACHE_LOG_DIR}/access.log combined
- </VirtualHost>

Cambiar el virtual host

- Deberé modificarlo por algo como esto:

- <VirtualHost *:443>
- ServerAdmin webmaster@localhost
- DocumentRoot /var/www/html
- ErrorLog ${APACHE_LOG_DIR}/error.log
- CustomLog ${APACHE_LOG_DIR}/access.log combined
- SSLEngine on
- SSLCertificateFile /etc/apache2/certificate/apache-certificate.crt
- SSLCertificateKeyFile /etc/apache2/certificate/apache.key
- </VirtualHost>

Reiniciar apache...

- service apache2 restart
- Y ya puedes entrar en tu servidor con https
- Abre el navegador y pruebalo con https://ipdemiservidor.

Pero... por que el navegador no me lo reconoce como a google?

- El navegador, a parte de encriptar la info que manda, también nos verifica la autenticidad de la página web.
- Eso lo hace por que hay entidades certificadoras que te dan el fichero CRT. De esta forma, se que nadie, se está haciendo pasar por instagram.
- Para certificar un dominio, antes habia que pagar a una entidad certificadora (CA se les llama), pero ahora existen proyectos como LetsEncrypt que certifican dominios gratis.
- No solo lo certifican, sino que te lo configuran todo.

¿Podemos probar letsencrypt?

- Para probar letsEncrypt, necesitamos que la CA, acceda a nuestra web. Por lo tanto, nuestro servidor tiene que tener una ip pública.
- Además, necesitamos un dominio, no podemos certificar una IP. Y eso cuesta dinero. (15€ al año)