¿Qué es Docker?

Docker es un proyecto de código abierto que automatiza el despliegue de aplicaciones dentro de contenedores de software, proporcionando una capa adicional de abstracción y automatización de *virtualización* de aplicaciones en múltiples sistemas operativos.

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*.

¿Qué NO es Docker?

Docker no es como otros sistemas de virtualización completa de un sistema operativo, no emula el hardware necesario para arrancarlo, utiliza el de la máquina anfitriona compartiendo los recursos de la misma entre los *contenedores.*

En resumen Docker es una herramienta que nos da la capacidad de trabajar con *contenedores* (máquinas virtuales) que comparten un mismo *Kernel*.

Vamos a ver un poco las partes básicas de Docker para hacer funcionar un sistema, y luego explicaremos como podemos automatizarlo con *docker-compose*.

Contenedores

Es el espacio donde se almacenan los datos de la *máquina virtual*, contiene todos los archivos necesarios para funcionar independientemente del sistema anfitrión. Un contenedor es una instancia de una imagen.

Imagen

Es un prototipo de un contenedor, así como una clase es el prototipo de un objeto y este una instancia de la misma, un *contenedor* es una instancia de una imagen. Varios *contenedores* pueden compartir una imagen, pero una imagen solo se corresponde con un contenedor.

Volumen

Es un *espacio de almacenamiento* que comparte el contenedor con la máquina anfitriona para tener persistencia de los datos, los volúmenes pueden ser de varios tipos como los internos (que los gestiona Docker) o pueden usarse mayormente carpetas compartidas.

Red

Las *interfaces de red* pueden configurarse según se necesite, tenemos varias como host, que comparte con la máquina anfitriona, el bridge que genera un puente de red o null que no tiene red.

Automatizar Docker con docker-compose

Docker compose es una herramienta creada por Docker, que permite crear una pila de contenedores intercomunicados, partiendo de distintas imágenes.

Docker compose se basa en un fichero con extensión yml donde vamos a indicar que imagen queremos desplegar, cómo se va a configurar y de qué depende.

```
version: '3'
services:
  web:
    build: .
    ports:
    - "80:80"
    volumes:
    - ./html:/var/www/html
    networks:
      net:

networks:
  net:
```

En el anterior ejemplo vemos como se crea un contenedor que compila un Dockerfile de la carpeta local, que comparte el volumen de la carpeta local *html* y también crea la configuración de red.

Conclusión

Las principales ventajas de la virtualización basada en contenedores son:

- Menos recursos, ahorro en costes. En una misma máquina pueden desplegarse más contenedores que máquinas virtuales tradicionales. Las exigencias en el proceso de inicio y espacio en disco son menores y más rápidas.
- Gestión TI más fácil, aumento de la productividad TI. La creación de contenedores permite estandarizar los despliegues ya que son entornos repetibles para tareas de desarrollo, prueba y producción. La compatibilidad con todos los sistemas de implantación elimina un valioso tiempo de configuración. Son elementos totalmente portables. Con Docker la implementación se realiza en segundos.
- Múltiples aplicaciones independientes en un mismo host. Cada aplicación se ejecuta en su contenedor o clúster de contenedores de forma independiente, sin entrar en conflicto con el resto de aplicaciones que aloje el host que ejecutarán mediante sus propios *contenedores*. Esto garantiza un entorno seguro y eficiente.

Cuando una aplicación ya no es necesaria, simplemente se elimina su contenedor sin dejar huella en el sistema donde se ejecutaba.