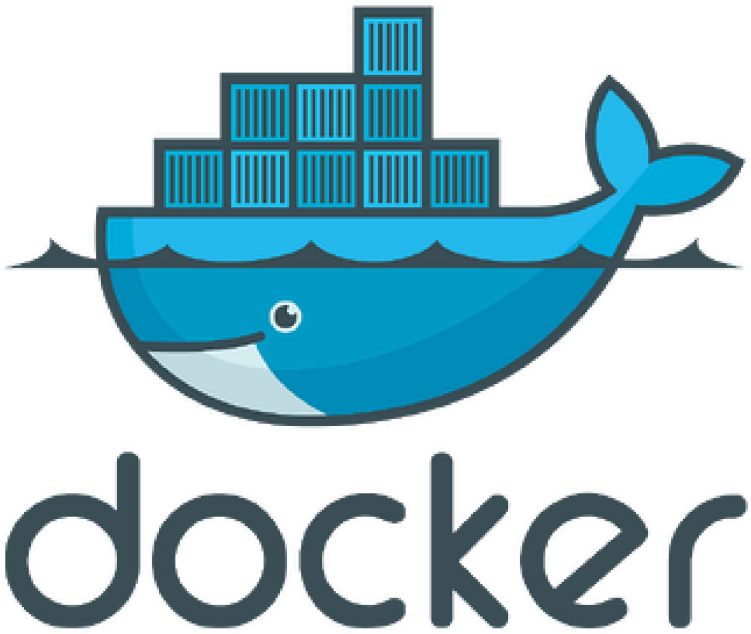
Proyecto 1



Autor: Pau Oliver López
Administración de Sistemas Informáticos en red (Asix de 2º año)
Profesorado: Sergio, Oriol, Victor, Jaume i Michael
4/11/2022

Índice

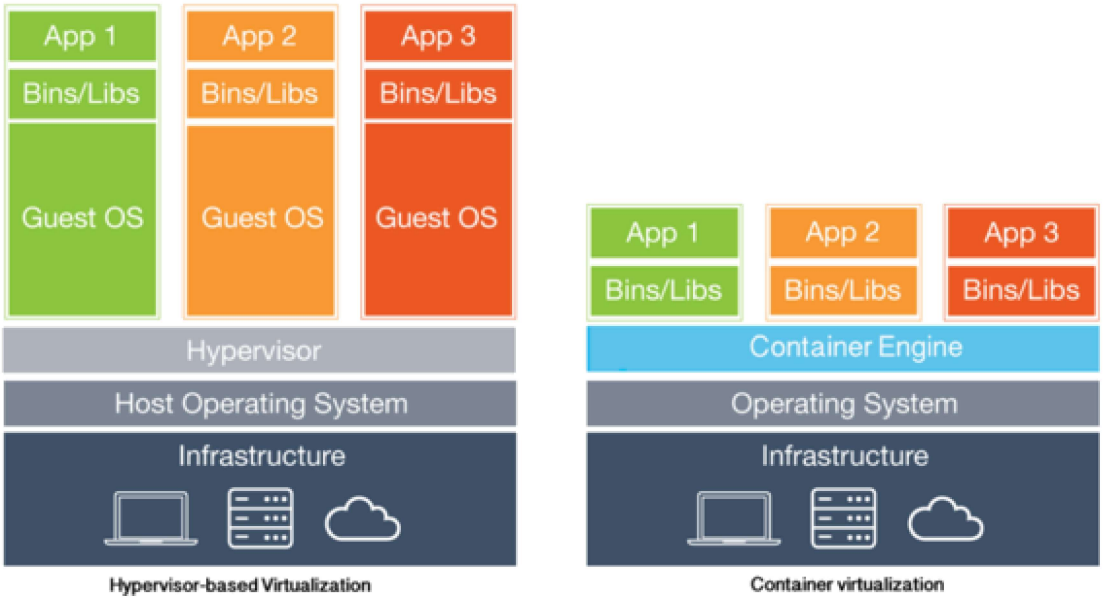
Introducción

Que es Docker?

Docker es una plataforma de software que permite crear, probar e implementar aplicaciones rápidamente. Docker empaqueta software en unidades estandarizadas llamadas contenedores que incluyen todo lo necesario para que el software se ejecute, incluidas bibliotecas, herramientas de sistema, código y tiempo de ejecución,etc. Los contenedores son fáciles de utilizar gracias a su interfaz de líneas de comandos y permite a los usuarios poder utilizar y controlar varios contenedores a la misma vez gracias a la herramienta Docker Compose.

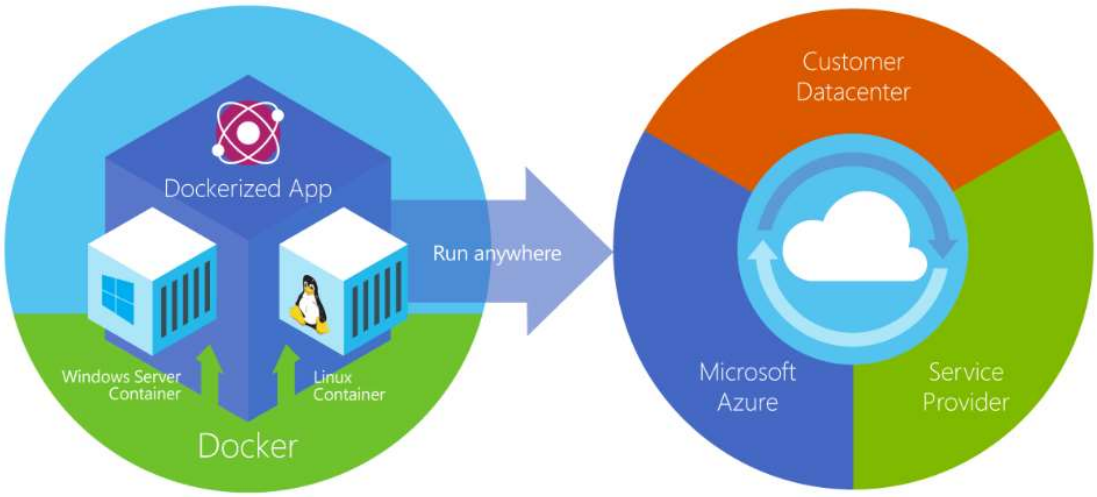
Funcionamiento de Docker?

Gracias a estos contenedores que hemos comentado anteriormente podemos desplegar la misma versión en distintos sistemas, siempre que tengan soporte de Docker, esto nos permite desarrollarlo por ejemplo en tu portátil Apple y ponerlo a funcionar en un servidor Cloud.



Como podemos observar en la imagen los contenedores adaptan las aplicaciones a nivel de despliegue y homogeneizan su ejecución en diferentes entornos, ya sea en la nube o en las instalaciones. Estas pequeñas piezas de software que componen el equipo de desarrollo

son fácilmente ejecutables por el equipo de despliegue y consumen muchos menos recursos que otras opciones como podrían ser las máquinas virtuales.



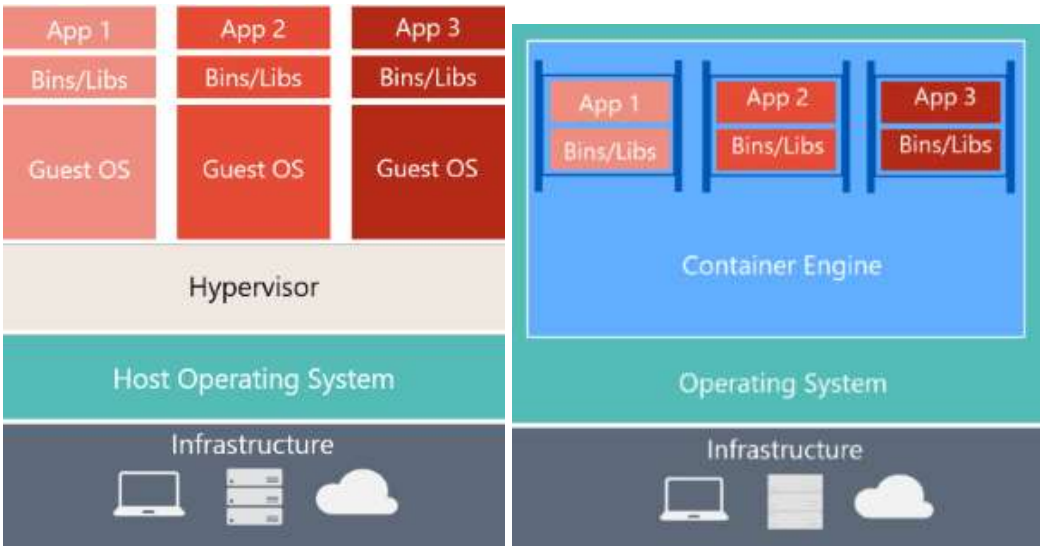
Cómo comenzó?

Docker apareció en escena en 2013 con una interfaz fácil de usar y la capacidad de empaquetar, aprovisionar y ejecutar tecnología de contenedores. Dado que Docker permitió que varias aplicaciones con diferentes requisitos de sistemas operativos se ejecutan en el mismo kernel de sistemas operativos en contenedores, los administradores informáticos y las organizaciones vieron la oportunidad de simplificar y ahorrar recursos. Un mes después de su primer lanzamiento de prueba, Docker fue probado por 10,000 desarrolladores. Cuando se lanzó Docker 1.0 en 2014, el software se había descargado 2,75 millones de veces y solo tendría que pasar un año para de eso para que llegara a la escandalosa cifra de 100 millones de descargas.

Principales Ventajas de Docker

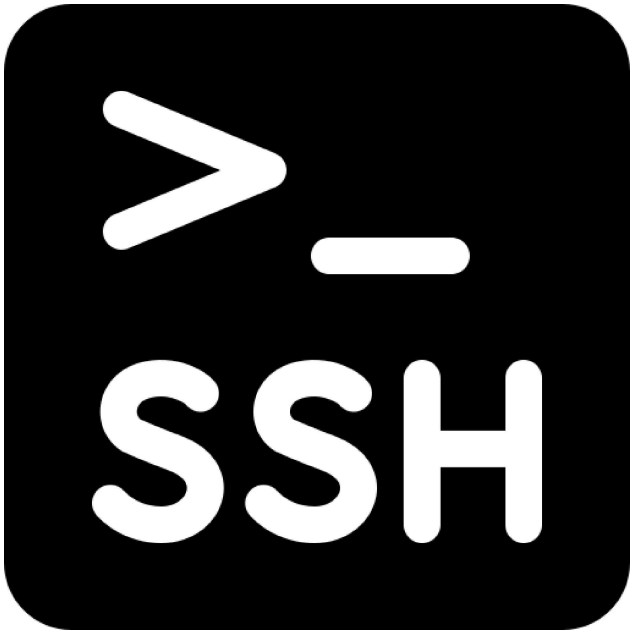
A diferencia de las máquinas virtuales, los contenedores tienen una huella de recursos significativamente menor, requieren menos gastos generales para su administración. Las máquinas virtuales también deben encapsular un sistema operativo completamente independiente y otros recursos, mientras que los contenedores comparten el mismo núcleo del sistema operativo y usan un sistema proxy para conectarse a los recursos que necesitan, dependiendo de dónde se encuentren estos recursos. Como podemos ver en la siguiente imagen las máquinas virtuales incluyen la aplicación, las bibliotecas o los archivos binarios necesarios y un sistema operativo invitado completo. La virtualización completa requiere más recursos que la inclusión en contenedores. En cambio los contenedores incluyen la aplicación y todas sus dependencias. Sin embargo, comparten el kernel del sistema operativo con otros contenedores, que se ejecutan como procesos aislados en el espacio de usuario en el sistema operativo host. Todo esto se traduce en una reducción considerable de costos y un retorno de la inversión mucho más rápido al reducir los recursos materiales necesarios.

Maquina virtual **vs** Docker



¿Por qué me interesa este proyecto?

Este proyecto me ha llamado la atención porque gracias a ello he podido descubrir que es Docker, cómo funciona y sobre todo lo más importante sus salidas profesionales. Durante estos 2 últimos meses he podido emprender un nuevo reto, la verdad es que al principio me costó bastante entender esta herramienta que nos permite a nosotros como desarrolladores o administradores poder hacer simulaciones en entornos de prueba, y sin la necesidad de tener instalada algún tipo de dependencia. Pero al pasar del tiempo fui comprendiendo cómo funciona esta nueva tecnología, y sus virtudes. La siguiente parte que veremos a continuación ya es la parte práctica del proyecto, en la cual enseñó todos los pasos que he llevado a cabo además de todos los servicios y programas utilizados como apache2, python, mySQL y el SSH.



PYTHON

Creación de Directorios

En este caso no es obligatorio crear un directorio donde guardar el Dockerfile pero yo he preferido crear uno para tenerlo más ordenado. Con mkdir creamos el Directorio Docker-Projects y dentro de este creamos un Dockerfile.

```
root@I03-21:~# mkdir Docker-Projects
root@I03-21:~# cd Docker-Projects
root@I03-21:~/Docker-Projects# nano Dockerfile
```

Creación del Dockerfile

Antes de crear un Dokerfile debemos saber que es. Podemos explicar el Dockerfile como un archivo en el cual incluimos una serie de instrucciones necesarias para poder crear una nueva imagen. Para crear un Dockerfile puedes utilizar el comando nano o touch que nos permitirá editar el Dockerfile abriéndolo en forma de archivo de texto.

```
root@I03-21:~/Docker-Projects# nano Dockerfile
```

Instalación de los servicios

Una vez creado el Dockerfile pondremos los comandos siguientes. En estos estamos instalando y actualizando los siguientes servicios: python3.10, ssh server, supervisor, apache2 y el mysql. En la segunda línea de comando podremos ver ENV DEBIAN_FRONTED=noninteractive, esto lo pongo para que se instalen algunos servicios automáticamente sin la necesidad de poner la región, el idioma del teclado, etc. Si nos fijamos también hay una línea de comandos que pone COPY, esto lo he hecho con la finalidad de crear la configuración del supervisor fuera del container y que una vez montemos este mismo (como podremos ver posteriormente) copie directamente la configuración de una manera relativamente más sencilla. Los servicios que hemos de hacer funcionar los arrancaremos con el supervisor, una herramienta que permite monitorear y controlar un número de procesos en sistemas operativos UNIX. Los puertos que pondremos para que funcionen serán: en el ssh el puerto será el 22, el 80 y 443 serán para el apache, en el supervisor el 9001 y para el Mysql el puerto será 3306.

```
FROM ubuntu:22.04

ENV DEBIAN_FRONTED=noninteractive

RUN apt-get update && apt-get -y upgrade

RUN apt-get install -y python3.10

RUN apt-get install -y ssh openssh-server

RUN apt-get install -y supervisor

RUN apt-get install -y apache2

RUN apt-get install -y mysql-server

RUN mkdir -p /var/lock/apache2 /var/run/apache2 /var/run/sshd /var/log/supervisor

COPY supervisord.conf /etc/supervisor/conf.d/supervisord.conf

EXPOSE 22 80 443 9001 3306

CMD ["/usr/bin/supervisord"]
```

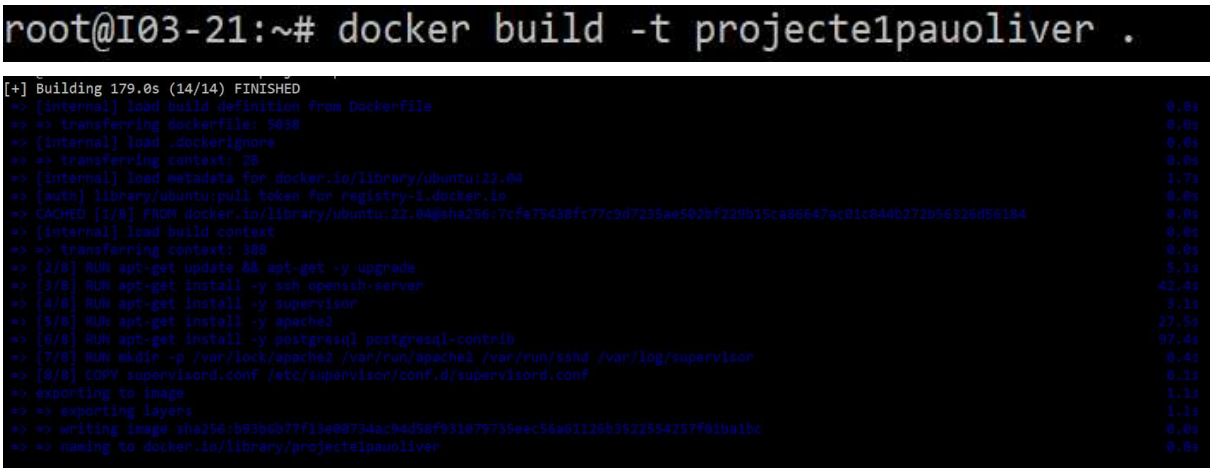
Configuración del supervisor

Como he explicado anteriormente creó el archivo de configuración fuera de la imagen. Lo que podemos ver en la imagen es la configuración que le puse al Supervisor para que funcionara correctamente.

```
[supervisord]
nodaemon=true

[program:sshd]
command=/usr/sbin/sshd -D

[program:apache2]
command=/bin/bash -c "source /etc/apache2/envvars && exec /usr/sbin/apache2 -DFOREGROUND"

[program:mysql]
command=/usr/bin/pidproxy /var/run/mysqld/mysqld.pid /usr/sbin/mysqld
autostart=true
autorestart=false
user=root
```

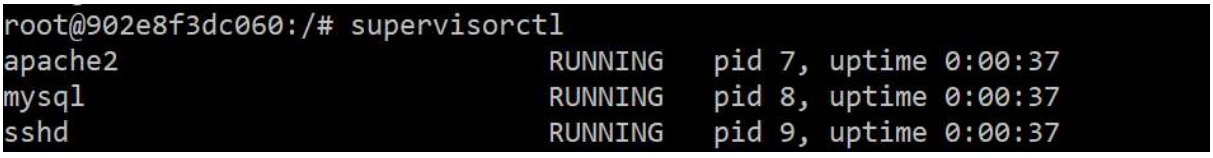
Montamos la Imagen

Una vez hecho lo anterior montamos la imagen. Esto se hace poniendo el comando docker build -t (terminal) más el nombre que le queremos poner, y al final de todo un punto para marcar la ruta. Si todo está correcto debería de ejecutar todas las instalaciones que hemos puesto en el Dockerfile, como podemos ver después de 179 segundos la imagen se ha montado correctamente en un contenedor. Dentro de los contenedores de Docker tenemos las imágenes. Una imagen de Docker, contiene las librerías, junto al código de la aplicación que contiene todo lo necesario para ejecutar nuestra aplicación.



Probamos que los servicios funcionen

Antes de subir la imagen al Docker Hub debemos de asegurarnos que todos los servicios que hemos instalado mediante el Dockerfile están ejecutados y que funcionan correctamente. Con el comando supervisorctl lo podemos contrastar viendo el estado de los servicios y cuanto llevan iniciados.



Otra manera de poder argumentar es poniendo service “servicio” status. Como podemos ver todos los servicios están iniciados y funcionan correctamente.

```
root@902e8f3dc060:/# service ssh status
 * sshd is running
root@902e8f3dc060:/# service apache2 status
 * apache2 is running
root@902e8f3dc060:/# service mysql status
 * /usr/bin/mysqladmin  Ver 8.0.31-0ubuntu0.22.04.1 for Linux on x86_64 ((Ubuntu))
Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Server version          8.0.31-0ubuntu0.22.04.1
Protocol version        10
Connection              Localhost via UNIX socket
UNIX socket             /var/run/mysqld/mysqld.sock
Uptime:                 2 min 35 sec
```

Subimos la imagen

Una vez que hemos montado la imagen y hemos visto que los servicios están iniciados y funcionan ya podemos subirlo a Docker Hub. Para esto utilizaremos el comando docker push agregando el nombre de nuestro usuario de Docker Hub / el nombre que queramos poner.

```
root@I03-21:~# docker push rebeldog/proyecto1
Using default tag: latest
The push refers to repository [docker.io/rebeldog/proyecto1]
c50121e00778: Pushed
0ea1dececd99: Pushed
e16f33853709: Pushed
9de022f5fb19: Pushed
9f64f1200865: Pushed
66bade339cdf: Pushed
e5bc65cbc943: Pushed
fe2ce64df66e: Pushed
7ea4455e747e: Mounted from library/ubuntu
latest: digest: sha256:97a9a4f6c31358b678a82ce37e9efcf4f4541787c177bbd350da184b9afbd40b size: 2214
```

Cómo arrancar una imagen

Por último para poder arrancar la imagen, para poder hacerlo utilizaremos el comando docker run seguido de -it (interactive terminal) y por último la imagen que queremos cargar al contenedor. En este caso ubuntu:20.04 que es el nombre de la imagen.

```
root@I03-21:~# docker run -it ubuntu:20.04
```

Webgrafia

https://techexpert.tips/es/apache-es/apache-instalacion-de-docker/
https://www.itsimplenow.com/supervisord-gestionando-procesos-en-docker/
https://aws.amazon.com/es/docker/
https://www.zdnet.com/article/docker-libcontainer-unifies-linux-container-powers/
https://web.archive.org/web/20150428181821/https://developer.ibm.com/bluemix/2014/12/04/ibm-containers-beta-docker/
https://www.hostinger.es/tutoriales/como-crear-contenedor-docker
https://www.freecodecamp.org/espanol/news/guia-de-docker-para-principiantes-como-crear-tu-primera-aplicacion-docker/
https://barbaraiot.com/es/blog/por-que-todo-el-mundo-habla-de-docker-en-el-iot-industrial
https://blog.sarenet.es/futuro-hosting-2-docker-kubernetes-contenedores/
https://kryptonsolid.com/docker-kubernetes-y-el-futuro/