Introducción. Conceptos básicos

Virtualización vs. Contenedores	La virtualización emula hardware completo; los contenedores comparten el sistema anfitrión, aislando aplicaciones.
Docker	Sistema de contenedores en Linux que facilita el despliegue y ejecución de aplicaciones en entornos controlados y consistentes.
Cliente Docker	Interfaz que se comunica con el servidor.
Servidor Docker (Host)	Gestiona contenedores e imágenes.
Ventajas de Docker	Ahorra espacio y recursos. Compatible con entornos de desarrollo, pruebas y producción. Velocidad de ejecución cercana a la nativa.
Limitaciones	Menor rendimiento que en hardware dedicado. Persistencia de datos más compleja. Preferencia por línea de comandos (puede ser tedioso crear entornos gráficos).

Introducción. Conceptos básicos (cont)

Contenedor	Un contenedor es un entorno aislado que ejecuta aplicaciones con sus dependencias, compartiendo el kernel del sistema operativo anfitrión. Es más ligero y rápido que una máquina virtual.
Imagen	Plantilla inmutable que define qué contiene y cómo funciona un contenedor. Las imágenes incluyen el sistema de archivos y las aplicaciones necesarias y pueden ser almacenadas en repositorios como Docker Hub.
Volumen	Almacenamiento persistente que permite guardar datos de forma independiente al ciclo de vida del contenedor, facilitando el manejo de datos duraderos entre diferentes instancias.

Principales acciones

Crear un grupo docker	sudo groupadd docker
Añadir tu usuario a un grupo docker	sudo usermod -aG docker $USER

Principales acciones (cont)

Crear contenedor apartir de una imagen, modo interactivo. Para poder interactuar con el contenendor creado, asignar terminal interactivo	docker run -it nombreImagen
Crea un contenedor con la imagen "ubuntu"	docker run -it --name=nombreRandom ubuntu /bin/bash
Mostrar informacion contenedores en ejecución	docker ps
Muestra información de todos los contenedores	docker ps -a
Arranca un contenendor concreto	docker start nombreContenedor
Ejecutar comandos dentro de un contenedor que esta en ejecución	docker exec
Crear y inicializar un contenendor	docker run
Detener contenedores activos con nombreContenedor	docker stop nombreContenedor
Conectarse a la consola de un contenedor en ejecución	docker attach IDCONTAINER
Muestra información del proceso de ejecución del contenedor	docker logs IDCONTAINER
Copiar fichero /tmp/nombreFichero del contenedor al directorio actual del anfitrión	docker cp IDCONTAINER:/tmpt/nombreFichero ./

Principales acciones (cont)

Copiar un fichero del directorio actual anfitrión al a la carpeta /tmp del contenedor	docker cp ./nombreFichero IDCONTAINER:/tmp

Volúmenes

Crear un volúmen	docker volume create nombreVolumen
Listar volúmenes	docker volume ls
Listar volúmenes activos	docker volume ls -a
Visualizar volúmenes	docker volume inspect nombreVolumen
Borrar volúmen	docker volume rm nombreVolumen
Crear contenedor y asignar volumen	docker run -d -it --nombreContenedor -v nombreVolumen
Examinar contenido volumen "metadatos"	docker volum inspect nombreVolumen

Gestión de redes

Creamos una red	docker network create nombre-red
Listamos las redes existentes	docker network ls
Borramos una red	docker network rm nombre-red
Conectar el contenedor creado "ubuntu" a una red	docker run -it --network nombre-red ubuntu /bin/bash
Conectar un contenedor a una red	docker network connect IDRED IDCONTENEDOR
Desconectar un contenedor de una red	docker network disconnect IDRED IDCONTENEDOR

Imágenes

Información imagenes disponibles	docker images
Buscar una imagen en concreto	docker search nombreImagen
Descargar localmente una imagen existente en remoto	docker pull nombreImagen
Eliminar localmente una imagen	docker rmi nombreImagen
Borrar un contenedor	docker rm IDCONTENEDOR
Mostrar historial de la creación de una imagen	docker history nombreImagen
Borra toda imagen local que no esté siendo usada por un contenedor	docker rmi $(docker images -q)
Parar todos los contenedores	docker stop $(docker ps -a -q)
Paramos todos los contenedores parados	docker rm $docker ps -a -q)
Borrar imagenes y contenedores parados	docker system prune -a
Commit de un contenedor a una imagen local	docker commit -m “comentario” IDCONTENEDOR usuario/imagen:version
Guarda una copia de seguridad de una imagen en fichero “.tar”.	docker save -o copiaSeguridad.tar imagenA
Restaura una copia de seguridad de una imagen en fichero “.tar”.	docker load -i copiaSeguridad.tar

Docker Hub

Login a Docker Hub en remoto	docker login
Subir al repositorio una imagen	docker push usuario/imagen:version

Docker Compose

Inicia el sistema definido en “docker-compose.yml” en segundo plano. Arranca la aplicación con Compose	docker-compose up -d
Detiene servicios	docker-compose stop
Detiene y elimina los contenedores según la configuración de “docker-compose.yml”	docker-compose down
Detiene y elimina volumenes y contenedores	docker-compose down -v
Muestra información de los servicios que se define en docker-compose.yaml.Muestra información de los contenedores según la configuración de “docker-compose.yml”	docker-compose ps
Construye/descarga las imágenes de contenedores según la configuración de “docker-compose.yml”	docker-compose build/pull
Parámetro docker -link	Parámetro compose depends_on
Parámetro docker -mount	Parámetro compose volumes
Parámetro docker -e	Parámetro compose environment
Parámetro docker -p	Parámetro compose ports

Dockerfile	
Crear imagen a partir de Dockerfile	docker build -t nombre-Imagen
Imagen base de la que partimos	FROM
Comandos que lanzaremos sobre la imagen base, para crear una nueva	RUN
Comando asociado al lanzar un contenedor con la nueva imagen	CMD
Para indicar los puertos por defecto expuestos que tendra el contenedor	EXPOSE
Descomprimir el contendio de un archivo en el directorio destino	ADD
Copiar ficheros de la máquina anfitrión al nuevo contenedor	COPY
Cambiar comandos que se lanzan por defecto al ejecutar un contenedor	ENTRYPOINT
Cambiar el usuario por defecto (root) a la hora de lanzar comandos al crear la imagen	USER
Cambiar el directorio de la imagen donde se ejecutaran los comandos	WORKDIR
Definir variables de entorno en la imagen	ENV
Enviar párametros al propio Dockerfile	ARG
Establecer volúmenes por defecto en la imagen	VOLUME
Establecer metadatos dentro de la imagen mediante etiquetas	LABEL

Dockerfile (cont)	
Definir como se comprobará si ese contenedor está funcio-nando correctamente	HEALTH CHECK