INSTITUTO TECNOLÓGICO DE MORELIA

Ingeniería en Sistemas Computacionales

Seguridad en la Nube

Docker

ALUMNO:

Rogelio Cristian Punzo Castro **21120245**

PROFESOR:

Roque Trujillo Ramos

MORELIA, MICHOACÁN **(Diciembre 2024)**

Índice

Resumen

La presente investigación aborda los aspectos fundamentales de Docker, una tecnología clave en la contenedorización de aplicaciones. Se exploraron los conceptos básicos, destacando las diferencias entre contenedores y máquinas virtuales, así como las ventajas que Docker ofrece para el desarrollo y despliegue de software, tales como la consistencia entre entornos, la facilidad de escalabilidad y la reducción de conflictos de dependencias.
Se analizaron los principales componentes de Docker, entre ellos las imágenes, los contenedores, los volúmenes, las redes y Docker Compose, y se compararon con otras tecnologías de contenedores como Podman y Kubernetes. También se detallaron los pasos para instalar Docker en diferentes sistemas operativos, la creación de imágenes personalizadas mediante Dockerfile y la gestión eficiente de contenedores, incluyendo comandos básicos y estrategias para la persistencia de datos.
Asimismo, se estudiaron las configuraciones de red en Docker, incluyendo la creación de redes personalizadas y la comunicación entre múltiples contenedores. Se presentaron aplicaciones prácticas, como el uso de Docker Compose para crear sistemas multicontenedor, y se analizaron casos de uso reales de empresas como Netflix y Uber, demostrando la relevancia de Docker en arquitecturas modernas.
Finalmente, se abordaron los aspectos de seguridad en Docker, identificando riesgos potenciales y recomendando prácticas como el escaneo de imágenes en busca de vulnerabilidades y la adopción de estándares para proteger los entornos de contenedores. En conjunto, este estudio destaca a Docker como una herramienta esencial para la infraestructura de TI actual, proporcionando eficiencia, flexibilidad y seguridad en el ciclo de vida de las aplicaciones.

1. Conceptos básicos y teoría

Preguntas para investigar:

- ¿Qué son los contenedores y cómo se diferencian de las máquinas virtuales?

Los contenedores son una tecnología de virtualización ligera que permite empaquetar una aplicación junto con todas sus dependencias y configuraciones necesarias para su ejecución. A diferencia de las máquinas virtuales (VMs), que requieren un sistema operativo completo y recursos adicionales (como CPU y memoria), los contenedores comparten el mismo núcleo del sistema operativo subyacente. Esto los hace más eficientes en términos de recursos y más rápidos en la ejecución.

Diferencias principales:

- **Eficiencia**: Los contenedores son más rápidos y consumen menos recursos, ya que no necesitan un sistema operativo completo.
- **Compatibilidad**: Facilitan el transporte de aplicaciones entre entornos sin modificar su configuración.
- **Escalabilidad**: Permiten lanzar múltiples contenedores en una sola máquina, optimizando el uso de recursos.

- ¿Qué problemas resuelve Docker en el desarrollo y despliegue de aplicaciones?

Docker simplifica la construcción, prueba y despliegue de aplicaciones, resolviendo problemas como:

- **Inconsistencia entre entornos**: Garantiza que las aplicaciones funcionen igual en desarrollo, pruebas y producción.
- **Dependencias y configuraciones**: Empaqueta todo lo necesario para ejecutar una aplicación, evitando conflictos de dependencias.
- **Escalabilidad**: Facilita la creación de entornos replicables y escalables.

- Principales componentes de Docker: imágenes, contenedores, volúmenes, redes y Docker Compose.

- **Imágenes**: Plantillas que contienen el sistema de archivos y las dependencias necesarias para ejecutar un contenedor.
- **Contenedores**: Instancias ejecutables creadas a partir de imágenes.
- **Volúmenes**: Mecanismo para persistir datos fuera del ciclo de vida de un contenedor.
- **Redes**: Configuraciones que permiten la comunicación entre contenedores y con el exterior.
- **Docker Compose**: Herramienta para definir y ejecutar aplicaciones multicontenedor mediante un archivo YAML.

- Comparación de Docker con otras tecnologías de contenedores como Podman o Kubernetes. // opcional

Característica	Docker	Podman	Kubernetes	Explicación
Arquitectura	Daemon	Daemonless	Maestro-esclavo	Docker utiliza un daemon central para gestionar los contenedores. Podman es daemonless, más seguro y ligero. Kubernetes es un orquestador que gestiona múltiples nodos (esclavos) para escalar aplicaciones.
Orquestación	Básica (Compose)	Limitada	Avanzada	Docker Compose es útil para aplicaciones simples. Podman tiene opciones limitadas de orquestación. Kubernetes es la plataforma de orquestación de contenedores más popular, ofreciendo features como autoescalado, descubrimiento de servicios, etc.

Seguridad	Menos segura (supervisores con privilegios)	Más segura (sin daemon central)	Alta seguridad (roles basados en RBAC, redes definidas por software)	Podman es inherentemente más seguro al no tener un daemon central que pueda ser un punto de ataque. Kubernetes ofrece múltiples mecanismos de seguridad.
Facilidad de uso	Fácil para principiantes	Similar a Docker	Mayor curva de aprendizaje	Docker tiene una gran comunidad y muchos recursos. Podman es similar en cuanto a comandos y sintaxis. Kubernetes requiere una mayor inversión de tiempo para aprender.
Rendimiento	Bueno	Muy bueno	Depende de la configuración	Podman suele ser más rápido que Docker debido a su arquitectura daemonless. Kubernetes puede tener un overhead adicional debido a su complejidad.
Compatibilidad	Amplia (imágenes OCI, Swarm)	Amplia (imágenes OCI)	Amplia (CRI)	Docker y Podman son compatibles con imágenes OCI. Kubernetes puede utilizar cualquier runtime que implemente el Container Runtime Interface (CRI).
Comunidad	Muy grande	Creciente	Enorme	Docker tiene la comunidad más grande. Podman está ganando popularidad. Kubernetes es el estándar de facto en orquestación.
Usos típicos	Desarrollo local,	Desarrollo local, servidores	Entornos de producción a	Docker es ideal para desarrolladores. Podman es bueno para

	despliegues simples	sin root, CI/CD	gran escala, microservicios	entornos sin root y CI/CD. Kubernetes es para aplicaciones complejas y escalables.

2. Instalación y configuración

Preguntas para investigar:
- ¿Cómo instalar Docker en diferentes sistemas operativos (Windows, macOS, Linux)?
Linux

1. **Actualiza el sistema**

```
sudo apt update && sudo apt upgrade
```

2. **Instala dependencias:**

```
sudo apt install apt-transport-https ca-certificates curl software-properties-common
```

3. **Agrega la clave GPG y el repositorio de Docker:**

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```

4. **Agregar el repositorio de Docker:**

```
sudo add-apt-repository "deb [arch=amd64]
https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
```

5. **Instala Docker:**

```
sudo apt update
sudo apt install docker-ce
```

6. **Verifica la instalación:**

```
sudo systemctl status docker
```

7. **Reiniciar el sistema para aplicar cambios.**

Windows & macOs: Se instala Docker Desktop desde su sitio oficial.

- ¿Qué configuraciones básicas son necesarias para empezar a usar Docker?
Configurar el servicio Docker

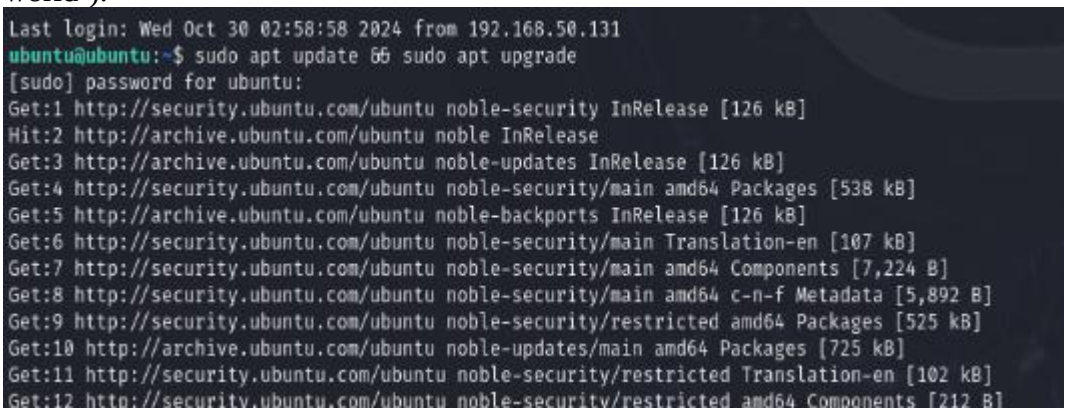
```
sudo systemctl start docker
sudo systemctl enable docker
```

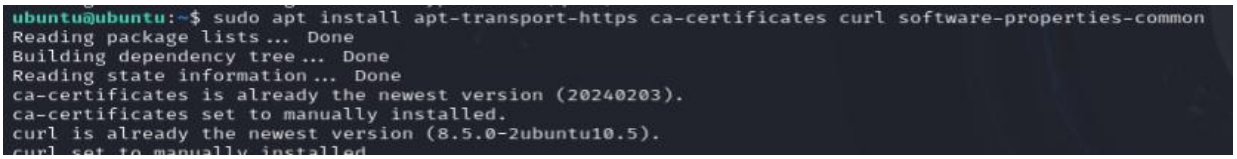
Habilitar uso sin sudo:

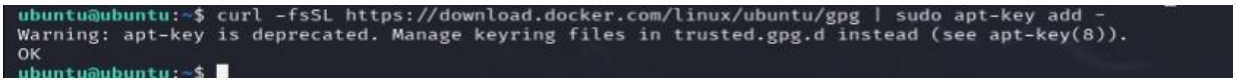
```
sudo usermod -aG docker $USER
```

- ¿Qué es Docker Desktop y cuándo se utiliza?
Docker Desktop es una aplicación para usar Docker en Windows o MacOS, que incluye Docker Engine, Docker CLI, Docker Compose y más. Se utiliza principalmente en entornos de desarrollo.

//opcional
Tarea práctica:
Instalar Docker en su sistema operativo y ejecutar su primer contenedor (por ejemplo, `hello-world`).







```
ubuntu@ubuntu:~$ sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_r
elease -cs) stable"
Repository: 'deb [arch=amd64] https://download.docker.com/linux/ubuntu noble stable'
Description:
Archive for codename: noble components: stable
More info: https://download.docker.com/linux/ubuntu
Adding repository.
Press [ENTER] to continue or Ctrl-c to cancel.
Adding deb entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu-noble.list
Adding disabled deb-src entry to /etc/apt/sources.list.d/archive_uri-https_download_docker_com_linux_ubuntu
-noble.list
Get:1 https://download.docker.com/linux/ubuntu noble InRelease [48.8 kB]
Hit:2 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:3 http://archive.ubuntu.com/ubuntu noble InRelease
Hit:4 http://archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:5 http://archive.ubuntu.com/ubuntu noble-backports InRelease
Get:6 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages [16.6 kB]
Fetched 65.4 kB in 1s (76.2 kB/s)
Reading package lists ... Done
W: https://download.docker.com/linux/ubuntu/dists/noble/InRelease: Key is stored in legacy trusted.gpg keyr
ing (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
```

```
ubuntu@ubuntu:~$ sudo apt update
sudo apt install docker-ce
Hit:1 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:2 http://archive.ubuntu.com/ubuntu noble InRelease
Hit:3 https://download.docker.com/linux/ubuntu noble InRelease
Hit:4 http://archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:5 http://archive.ubuntu.com/ubuntu noble-backports InRelease
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
```

```
ubuntu@ubuntu:~$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; preset: enabled)
     Active: active (running) since Sat 2024-12-14 01:46:42 UTC; 11s ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 29643 (dockerd)
      Tasks: 10
     Memory: 20.7M (peak: 21.1M)
        CPU: 395ms
     CGroup: /system.slice/docker.service
             └─29643 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock
```

```
ubuntu@ubuntu:~$ sudo usermod -aG docker $USER
```

```
ubuntu@ubuntu:~$ docker --version
Docker version 27.4.0, build bde2b89
ubuntu@ubuntu:~$ docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
c1ec31eb5944: Pull complete
Digest: sha256:5b3cc85e16e3058003c13b7821318369dad01dac3dbb877aac3c28182255c724
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/

ubuntu@ubuntu:~$
```

Ejecucion de contenedor hello-world

3. Creación y uso de imágenes

Preguntas para investigar:

- ¿Qué es una imagen Docker y cómo se crea?

Una imagen Docker es un archivo que contiene el sistema de archivos y las dependencias necesarias para ejecutar una aplicación en un contenedor. Se crea a partir de un **Dockerfile**.

- ¿Qué es un Dockerfile y cómo se utiliza para personalizar imágenes?

Un Dockerfile es un archivo de texto que contiene las instrucciones para construir una imagen personalizada. Ejemplo básico para una app Node.js:

```
FROM node:14
WORKDIR /app
COPY package.json .
RUN npm install
COPY . .
CMD ["node", "app.js"]
EXPOSE 3000
```

- ¿Cómo funcionan los registros de imágenes, como Docker Hub?

Los registros, como Docker Hub, permiten almacenar y compartir imágenes. Comandos básicos:

- Subir una imagen

```
docker push usuario/imagen:etiqueta
```

- Descargar una imagen

```
docker pull imagen:etiqueta
```

//opcional

Tarea práctica:

- Crear un Dockerfile básico que configure una aplicación web (por ejemplo, un servidor Apache o una app Node.js).
- Subir la imagen creada a Docker Hub.

Crear un directorio de trabajo:

```
ubuntu@ubuntu:~$ mkdir docker-apache
cd docker-apache
```

Editar archivo Dockerfile

```
ubuntu@ubuntu:~/docker-apache$ nano Dockerfile
```

Contenido

Este archivo instalará Apache en un contenedor basado en **Ubuntu** y lo configurará para ejecutarse.

```
File  Actions  Edit  View  Help
  GNU nano 7.2                                        Dockerfile
# Usar la imagen base oficial de Ubuntu
FROM ubuntu:20.04

# Configurar argumentos para evitar prompts interactivos
ARG DEBIAN_FRONTEND=noninteractive

# Actualizar repositorios e instalar Apache
RUN apt-get update && \
    apt-get install -y apache2 && \
    apt-get clean

# Exponer el puerto 80 para el servidor web
EXPOSE 80

# Comando para iniciar Apache en el primer plano
CMD ["apache2ctl", "-D", "FOREGROUND"]
```

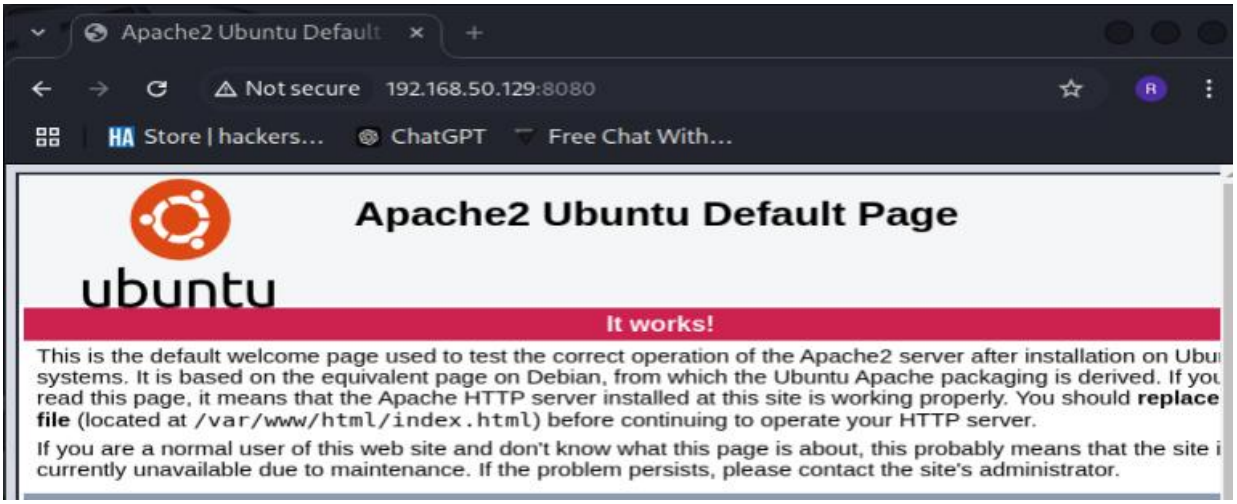
Construir la imagen Docker

```
ubuntu@ubuntu:~/docker-apache$ docker build -t my-apache-server .
[+] Building 34.9s (6/6) FINISHED                                   docker:default
 => [internal] load build definition from Dockerfile                          0.0s
 => => transferring dockerfile: 457B                                          0.0s
 => [internal] load metadata for docker.io/library/ubuntu:20.04               6.2s
 => [internal] load .dockerignore                                             0.0s
 => => transferring context: 2B                                               0.0s
 => [1/2] FROM docker.io/library/ubuntu:20.04@sha256:8e5c4f0285ecbb4ead070431d29b576a530d3166df73ec4   5.9s
 => => resolve docker.io/library/ubuntu:20.04@sha256:8e5c4f0285ecbb4ead070431d29b576a530d3166df73ec4   0.0s
 => => sha256:e5a6aeef391a8a9bdaee3de6b28f393837c479d8217324a2340b64e45a81e0ef 424B / 424B            0.0s
 => => sha256:6013ae1a63c2ee58a8949f03c6366a3ef6a2f386a7db27d86de2de965e9f450b 2.30kB / 2.30kB        0.0s
 => => sha256:d9802f032d6798e2086607424bfe88cb8ec1d6f116e11cd99592dcaf261e9cd2 27.51MB / 27.51MB      4.6s
 => => sha256:8e5c4f0285ecbb4ead070431d29b576a530d3166df73ec44affc1cd27555141b 6.69kB / 6.69kB        0.0s
 => => extracting sha256:d9802f032d6798e2086607424bfe88cb8ec1d6f116e11cd99592dcaf261e9cd2             1.0s
 => [2/2] RUN apt-get update &&     apt-get install -y apache2 &&     apt-get clean                22.3s
 => exporting to image                                                        0.5s
 => => exporting layers                                                       0.5s
 => => writing image sha256:64ecd9fd575b354be19eb15c22332f563434b032310263c3ef17b702b570eece          0.0s
 => => naming to docker.io/library/my-apache-server                           0.0s
ubuntu@ubuntu:~/docker-apache$ docker run -d -p 8080:80 my-apache-server
```

Aquí, my-apache-server es el nombre que se le da a la imagen. El punto . indica que el Dockerfile está en el directorio actual.

Probar imagen creada

```
ubuntu@ubuntu:~/docker-apache$ docker run -d -p 8080:80 my-apache-server
40238cdfbe4ee4c1b4b07c7b4b5c83c77facbb836069115aae7c5398289cb368
```



Para crear un volumen:

```
docker volume create <nombre_volumen>
```

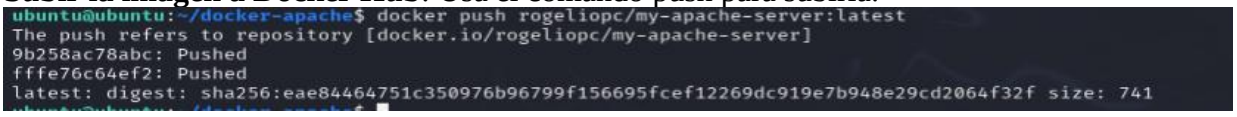
Y usarlo en un contenedor:

```
docker run -v <nombre_volumen>:<ruta_dentro_contenedor> <nombre_imagen>
```

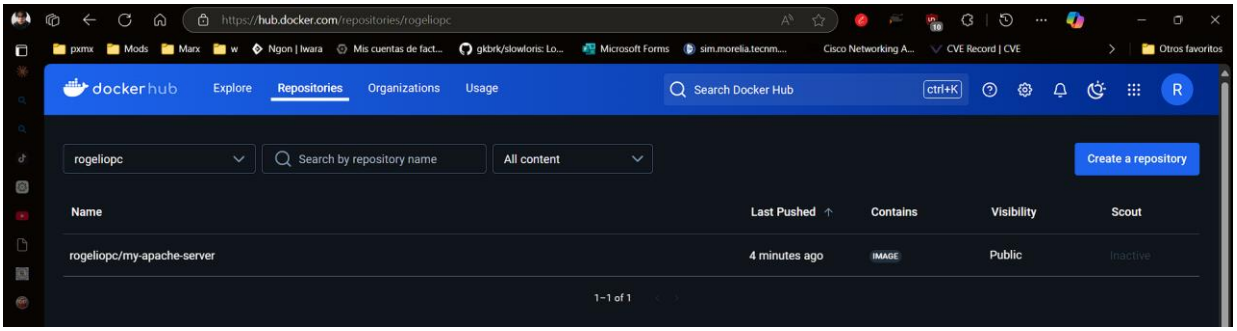
Crear una cuenta en Docker Hub: Si no se tiene una cuenta, regístrarse en Docker Hub.
Iniciar sesión desde Docker CLI: Inicia sesión en tu cuenta de Docker Hub desde la terminal:

Etiquetar la imagen para Docker Hub: Docker requiere que etiquetes la imagen con tu nombre de usuario.

Subir la imagen a Docker Hub: Usa el comando push para subirla:



Verificar en Docker Hub: Iniciar sesión en Docker Hub, ve a tu perfil y verificar que la imagen se haya subido correctamente.



4. Gestión de contenedores

Preguntas para investigar:

- ¿Cómo iniciar, detener y eliminar contenedores?

- docker run <nombre_imagen>: Inicia un nuevo contenedor en primer plano.
- docker run -d <nombre_imagen>: Inicia un contenedor en segundo plano (detached mode).

Opciones adicionales:

- -p <puerto_host>:<puerto_contenedor>: Mapea puertos entre el host y el contenedor.
- -v <volumen_host>:<ruta_contenedor>: Monta un volumen para persistir datos.
- --name <nombre_contenedor>: Asigna un nombre al contenedor para facilitar su gestión.

Detener un contenedor:

- docker stop <id_contenedor> o <nombre_contenedor>: Envía una señal SIGTERM al contenedor para que se detenga suavemente.
- docker kill <id_contenedor> o <nombre_contenedor>: Envía una señal SIGKILL al contenedor para detenerlo inmediatamente.

Eliminar un contenedor:

- docker rm <id_contenedor> o <nombre_contenedor>: Elimina un contenedor detenido.
- **Nota:** Para eliminar un contenedor en ejecución, primero debes detenerlo.

- ¿Qué comandos se utilizan para listar contenedores activos y no activos?

Contenedores en ejecución:

- docker ps: Muestra los contenedores en ejecución.

Todos los contenedores:

- docker ps -a: Muestra todos los contenedores, incluyendo los detenidos.

Filtrar contenedores:

- docker ps -f "status=exited": Muestra los contenedores detenidos.
- docker ps -f "name=my_container": Muestra los contenedores con el nombre "my_container".

- ¿Cómo persistir datos en contenedores usando volúmenes?

Los datos dentro de un contenedor se pierden cuando el contenedor se elimina. Los volúmenes permiten persistir los datos de forma independiente del ciclo de vida del contenedor.

- **Creando un volumen:** docker volume create my_volume
- **Montando un volumen en un contenedor:** docker run -v my_volume:/data <nombre_imagen>
- **Tipos de volúmenes:**
  - **Nombrados:** Gestionados por Docker y persisten incluso si el contenedor se elimina.
  - **Bind mounts:** Montan una carpeta del host directamente en el contenedor.
  - **Tmpfs:** Volúmenes temporales que se almacenan en la memoria del host.

Otros Comandos Útiles

- **Inspeccionar un contenedor:** docker inspect <id_contenedor>
- **Entrar en un contenedor en ejecución:** docker exec -it <id_contenedor> bash
- **Commit de cambios en un contenedor:** docker commit <id_contenedor> <nueva_imagen>

//opcional
Tarea práctica:
- Ejecutar un contenedor con persistencia de datos, por ejemplo, un contenedor de MySQL con un volumen asociado.

Crear un volumen para la persistencia de datos
Docker utiliza volúmenes para almacenar datos de forma persistente, incluso si el contenedor se elimina.

```
ubuntu@ubuntu:~/docker-apache$ docker volume create mysql_data
mysql_data
```

Esto crea un volumen llamado mysql_data, donde se almacenarán los datos del contenedor.

Ejecutar un contenedor de MySQL con el volumen
Usa el siguiente comando para iniciar un contenedor de MySQL, adjuntando el volumen creado:

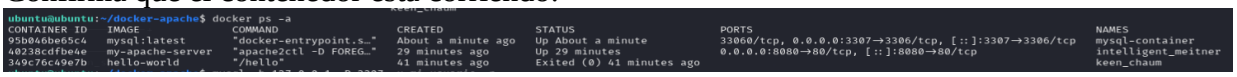
```
ubuntu@ubuntu:~/docker-apache$ docker run -d \
  --name mysql-container \
  -e MYSQL_ROOT_PASSWORD=mi_password_segura \
  -e MYSQL_DATABASE=mi_base_de_datos \
  -e MYSQL_USER=mi_usuario \
  -e MYSQL_PASSWORD=mi_password \
  -v mysql_data:/var/lib/mysql \
  -p 3307:3306 \
  mysql:latest
95b046be65c489fdd019382e897210094a085873658243bfc7dbebe74ff4fd94
```

Explicación de los parámetros:

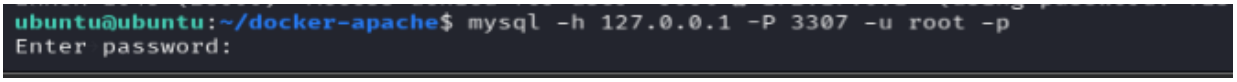
- **-d**: Ejecuta el contenedor en segundo plano (modo *detached*).
- **--name mysql-container**: Asigna un nombre al contenedor para identificarlo fácilmente.
- **-e**: Configura variables de entorno, como:
  - MYSQL_ROOT_PASSWORD: Contraseña para el usuario root.
  - MYSQL_DATABASE: Nombre de una base de datos inicial.
  - MYSQL_USER: Usuario adicional para la base de datos.
  - MYSQL_PASSWORD: Contraseña para el usuario adicional.

- **-v mysql_data:/var/lib/mysql**: Adjunta el volumen mysql_data al directorio de almacenamiento de datos de MySQL (/var/lib/mysql).
- **-p 3306:3306**: Mapea el puerto 3306 del contenedor al puerto 3306 del host para permitir conexiones externas.
- **mysql:latest**: Usa la imagen oficial más reciente de MySQL.

Verificar el contenedor en ejecución
Confirma que el contenedor está corriendo:



Conectarte al contenedor
Puedes conectarte al contenedor MySQL usando cualquier cliente MySQL. Por ejemplo, con la línea de comandos en el host:



También puedes acceder al contenedor directamente:

Probar la persistencia de datos

1. **Crear una tabla de prueba** dentro del contenedor:

```
CREATE TABLE ejemplo (id INT PRIMARY KEY, nombre VARCHAR(50));
INSERT INTO ejemplo VALUES (1, 'Prueba');
```

2. **Eliminar el contenedor**:

```
docker stop mysql-container
docker rm mysql-container
```

3. **Reiniciar el contenedor usando el mismo volumen**:

```
docker run -d \
  --name mysql-container \
  -e MYSQL_ROOT_PASSWORD=mi_password_segura \
  -v mysql_data:/var/lib/mysql \
  -p 3307:3306 \
docker volume create <nombre_volumen>
```

4. **Verificar los datos persistidos**: Conéctate nuevamente al contenedor y consulta la tabla creada:

```
SELECT * FROM ejemplo;
```

5. Redes en Docker

Preguntas para investigar:

- ¿Qué tipos de redes soporta Docker (bridge, host, none, overlay)?

1. **Bridge**:
   - Red predeterminada para contenedores.
   - Los contenedores conectados a esta red pueden comunicarse entre sí mediante sus nombres o direcciones IP.
   - Ideal para aplicaciones que se ejecutan en un solo host.
2. **Host**:
   - El contenedor comparte la pila de red del host.
   - No hay aislamiento de red entre el contenedor y el host.
   - Útil para aplicaciones que requieren alto rendimiento o acceso directo al host.
3. **None**:
   - Deshabilita completamente la red para el contenedor.
   - Usado en escenarios donde no se necesita conexión de red.
4. **Overlay**:
   - Permite la comunicación entre contenedores distribuidos en varios hosts.
   - Utilizado principalmente en configuraciones de clústeres con Docker Swarm.

- ¿Cómo se crean y gestionan redes personalizadas en Docker?

Crear una red personalizada:

```
ubuntu@ubuntu:~/docker-apache$ docker network create mi_red_personalizada
ebcba071a07dab54ecdc7116d696f7962f8bb8c90f33b51e332ea608c8150e35
```

Listar redes existentes:

```
ubuntu@ubuntu:~/docker-apache$ docker network ls
NETWORK ID     NAME                   DRIVER    SCOPE
0ab707700cee   bridge                 bridge    local
5452e7ccc833   host                   host      local
ebcba071a07d   mi_red_personalizada   bridge    local
63b75f5ad1f9   none                   null      local
```

Conectar un contenedor a una red personalizada:

```
ubuntu@ubuntu:~/docker-apache$ docker network connect mi_red_personalizada mysql-container
```

Desconectar un contenedor de una red:

```
ubuntu@ubuntu:~/docker-apache$ docker network disconnect mi_red_personalizada mysql-container
```

Eliminar una red personalizada:

Solo si no tiene contenedores conectados:

```
ubuntu@ubuntu:~/docker-apache$ docker network rm mi_red_personalizada
mi_red_personalizada
ubuntu@ubuntu:~/docker-apache$ docker network ls
NETWORK ID     NAME      DRIVER    SCOPE
0ab707700cee   bridge    bridge    local
5452e7ccc833   host      host      local
63b75f5ad1f9   none      null      local
```

- ¿Cómo comunicar múltiples contenedores utilizando Docker Compose?

Docker Compose facilita la configuración y gestión de aplicaciones multicontenedor.

Crear un archivo docker-compose.yml:

Ejemplo: Conectar un servidor web con una base de datos.

//opcional
Tarea práctica:
- Configurar dos contenedores que se comuniquen entre sí en una red personalizada (por ejemplo, un servidor web y una base de datos).
//opcional

6. Aplicaciones prácticas

Proyectos sugeridos:
1. Servidor LAMP: Configurar un contenedor que ejecute un servidor Apache con PHP y una base de datos MySQL para servir una aplicación web básica.
2. Sistema multi-contenedor: Crear un sistema con Docker Compose que contenga:
- Un servidor de backend (Node.js o Python).
- Una base de datos (PostgreSQL o MongoDB).
- Un servidor de frontend (React o Angular).
3. Analizar un proyecto real: Investigar cómo empresas como Spotify, Netflix o Uber utilizan Docker en sus arquitecturas.

Spotify

Spotify utiliza Docker y Kubernetes como parte fundamental de su infraestructura de microservicios:

- La plataforma de música implementó Docker para mejorar la escalabilidad y eficiencia de sus servicios.
- Desarrollaron una herramienta interna llamada "Backstage" que facilita la gestión de contenedores y servicios.
- Utilizan contenedores para desplegar microservicios de manera rápida y consistente en su arquitectura de cloud computing.

Netflix

Netflix es pionera en la adopción de tecnologías de contenerización:

- Implementaron Docker para optimizar sus despliegues en la nube de Amazon Web Services (AWS).
- Utilizan contenedores para ejecutar miles de microservicios de manera eficiente.
- Desarrollaron herramientas propias como Titus, una plataforma de gestión de contenedores basada en Kubernetes.

Uber

Uber utiliza Docker para gestionar su compleja infraestructura de servicios:

- Implementaron Docker para estandarizar los entornos de desarrollo y producción.
- Usan Kubernetes para orquestar y escalar sus servicios de manera dinámica.
- Desarrollaron herramientas internas para gestionar la distribución de contenedores en su infraestructura global.

Beneficios comunes observados

1. **Escalabilidad**: Los contenedores permiten escalar servicios rápidamente.
2. **Consistencia**: Garantizan entornos idénticos entre desarrollo y producción.
3. **Eficiencia**: Reducen los recursos computacionales necesarios.
4. **Agilidad**: Facilitan la implementación continua de nuevas versiones.

Consideraciones técnicas
Las arquitecturas de estas empresas comparten características similares:

- Uso de microservicios
- Orquestación con Kubernetes
- Despliegues en cloud computing
- Herramientas de gestión personalizadas

7. Seguridad en Docker

Preguntas para investigar:
- ¿Qué riesgos de seguridad existen al usar Docker y cómo mitigarlos?

- **Ataques a la superficie de ataque:** Reducir al mínimo la superficie de ataque exponiendo solo los puertos necesarios.
- **Vulnerabilidades en las imágenes:** Escanear regularmente las imágenes en busca de vulnerabilidades conocidas y utilizar imágenes base actualizadas.
- **Ataques a la cadena de suministro:** Verificar la autenticidad de las imágenes y utilizar repositorios de confianza.
- **Pérdida de datos:** Realizar copias de seguridad regulares de los datos importantes.

- ¿Qué son los escaneos de seguridad de imágenes Docker?

- **Docker scan:** Herramienta integrada en Docker para escanear imágenes en busca de vulnerabilidades comunes.
- **Trivy:** Escáner de seguridad de código abierto que puede analizar múltiples tipos de paquetes y configuraciones.

- Buenas prácticas para proteger imágenes y contenedores.

- **Utilizar imágenes base mínimas:** Reducir la superficie de ataque.
- **Mantener los contenedores actualizados:** Aplicar parches de seguridad de forma regular.
- **Limitar los privilegios:** Ejecutar los contenedores con el menor conjunto de privilegios posible.
- **Aislar los contenedores:** Utilizar redes definidas por software y limitar la comunicación entre contenedores.
- **Escanear regularmente las imágenes:** Detectar y remediar vulnerabilidades de forma proactiva.

//opcional
Tarea práctica:
- Escanear una imagen Docker en busca de vulnerabilidades utilizando herramientas como `docker scan` o `Trivy`.

Primero descargamos la herramienta Trivy:

```
ubuntu@ubuntu:~/docker-apache$ sudo snap install trivy
2024-12-14T03:21:52Z INFO Waiting for automatic snapd restart...
trivy 0.52.2 from James Luther (b34rd) installed
```

Ahora pasamos a realizar el escaneo con: trivy image [nombre de imagen]

```
ubuntu@ubuntu:~/docker-apache$ trivy image mysql:latest
2024-12-14T03:28:12Z    INFO    Vulnerability scanning is enabled
2024-12-14T03:28:12Z    INFO    Secret scanning is enabled
2024-12-14T03:28:12Z    INFO    If your scanning is slow, please try '--scanners vuln' to disable secret scanning
2024-12-14T03:28:12Z    INFO    Please see also https://aquasecurity.github.io/trivy/v0.52/docs/scanner/secret/#recommendation for faster secret detection
2024-12-14T03:28:25Z    INFO    [python] License acquired from METADATA classifiers may be subject to additional terms  name="PyYAML" version="6.0.1"
2024-12-14T03:28:25Z    INFO    [python] License acquired from METADATA classifiers may be subject to additional terms  name="bcrypt" version="4.1.3"
2024-12-14T03:28:25Z    INFO    [python] License acquired from METADATA classifiers may be subject to additional terms  name="certifi" version="2024.7.4"
2024-12-14T03:28:25Z    INFO    [python] License acquired from METADATA classifiers may be subject to additional terms  name="cffi" version="1.17.0"
2024-12-14T03:28:25Z    INFO    [python] License acquired from METADATA classifiers may be subject to additional terms  name="circuitbreaker" version="2.0.0"
2024-12-14T03:28:25Z    INFO    [python] License acquired from METADATA classifiers may be subject to additional terms  name="cryptography" version="42.0.8"
2024-12-14T03:28:25Z    INFO    [python] License acquired from METADATA classifiers may be subject to additional terms  name="oci" version="2.133.0"
2024-12-14T03:28:25Z    INFO    [python] License acquired from METADATA classifiers may be subject to additional terms  name="paramiko" version="3.4.0"
2024-12-14T03:28:25Z    INFO    [python] License acquired from METADATA classifiers may be subject to additional terms  name="pyOpenSSL" version="24.2.1"
2024-12-14T03:28:25Z    INFO    [python] License acquired from METADATA classifiers may be subject to additional terms  name="pycparser" version="2.22"
2024-12-14T03:28:25Z    INFO    [python] License acquired from METADATA classifiers may be subject to additional terms  name="python-dateutil" version="2.9.0.post0"
2024-12-14T03:28:25Z    INFO    [python] License acquired from METADATA classifiers may be subject to additional terms  name="pytz" version="2024.1"
2024-12-14T03:28:25Z    INFO    [python] License acquired from METADATA classifiers may be subject to additional terms  name="autocommand" version="2.2.2"
2024-12-14T03:28:25Z    INFO    [python] License acquired from METADATA classifiers may be subject to additional terms  name="typeguard" version="4.3.0"
2024-12-14T03:28:25Z    INFO    [python] License acquired from METADATA classifiers may be subject to additional terms  name="six" version="1.16.0"
2024-12-14T03:28:25Z    INFO    Detected OS     family="oracle" version="9.5"
2024-12-14T03:28:25Z    INFO    [oracle] Detecting vulnerabilities...   os_version="9" pkg_num=121
2024-12-14T03:28:25Z    INFO    Number of language-specific files       num=2
2024-12-14T03:28:25Z    INFO    [gobinary] Detecting vulnerabilities...
2024-12-14T03:28:25Z    INFO    [python-pkg] Detecting vulnerabilities...

mysql:latest (oracle 9.5)

Total: 6 (UNKNOWN: 0, LOW: 0, MEDIUM: 6, HIGH: 0, CRITICAL: 0)

┌──────────────────────────┬─────────────────┬──────────┬────────┬───────────────────┬─────────────────┬──────────────────────────────────────────────────────────────────────────────┐
│         Library          │ Vulnerability   │ Severity │ Status │ Installed Version │ Fixed Version   │                                    Title                                     │
├──────────────────────────┼─────────────────┼──────────┼────────┼───────────────────┼─────────────────┼──────────────────────────────────────────────────────────────────────────────┤
│ python-unversioned-command │ CVE-2024-11168  │ MEDIUM   │ fixed  │ 3.9.19-8.el9_5.1  │ 3.9.21-1.el9_5  │ python: Improper validation of IPv6 and IPvFuture addresses                  │
│                          │                 │          │        │                   │                 │ https://avd.aquasec.com/nvd/cve-2024-11168                                   │
│                          ├─────────────────┤          │        │                   │                 ├──────────────────────────────────────────────────────────────────────────────┤
│                          │ CVE-2024-9287   │          │        │                   │                 │ python: Virtual environment (venv) activation scripts don't quote paths      │
│                          │                 │          │        │                   │                 │ https://avd.aquasec.com/nvd/cve-2024-9287                                    │
├──────────────────────────┼─────────────────┤          │        │                   │                 ├──────────────────────────────────────────────────────────────────────────────┤
│ python3                  │ CVE-2024-11168  │          │        │                   │                 │ python: Improper validation of IPv6 and IPvFuture addresses                  │
│                          │                 │          │        │                   │                 │ https://avd.aquasec.com/nvd/cve-2024-11168                                   │
│                          ├─────────────────┤          │        │                   │                 ├──────────────────────────────────────────────────────────────────────────────┤
│                          │ CVE-2024-9287   │          │        │                   │                 │ python: Virtual environment (venv) activation scripts don't quote paths      │
│                          │                 │          │        │                   │                 │ https://avd.aquasec.com/nvd/cve-2024-9287                                    │
├──────────────────────────┼─────────────────┤          │        │                   │                 ├──────────────────────────────────────────────────────────────────────────────┤
│ python3-libs             │ CVE-2024-11168  │          │        │                   │                 │ python: Improper validation of IPv6 and IPvFuture addresses                  │
│                          │                 │          │        │                   │                 │ https://avd.aquasec.com/nvd/cve-2024-11168                                   │
│                          ├─────────────────┤          │        │                   │                 ├──────────────────────────────────────────────────────────────────────────────┤
│                          │ CVE-2024-9287   │          │        │                   │                 │ python: Virtual environment (venv) activation scripts don't quote paths      │
│                          │                 │          │        │                   │                 │ https://avd.aquasec.com/nvd/cve-2024-9287                                    │
└──────────────────────────┴─────────────────┴──────────┴────────┴───────────────────┴─────────────────┴──────────────────────────────────────────────────────────────────────────────┘
2024-12-14T03:28:25Z    INFO    Table result includes only package filenames. Use '--format json' option to get the full path to the package file.

Python (python-pkg)

Total: 1 (UNKNOWN: 0, LOW: 0, MEDIUM: 1, HIGH: 0, CRITICAL: 0)

Total: 1 (UNKNOWN: 0, LOW: 0, MEDIUM: 1, HIGH: 0, CRITICAL: 0)

┌──────────────────────┬─────────────────┬──────────┬────────┬───────────────────┬───────────────┬──────────────────────────────────────────────────────────────────────────────┐
│       Library        │ Vulnerability   │ Severity │ Status │ Installed Version │ Fixed Version │                                    Title                                     │
├──────────────────────┼─────────────────┼──────────┼────────┼───────────────────┼───────────────┼──────────────────────────────────────────────────────────────────────────────┤
│ cryptography (METADATA) │ GHSA-h4gh-qq45-vh27 │ MEDIUM   │ fixed  │ 42.0.8            │ 43.0.1        │ pyca/cryptography has a vulnerable OpenSSL included in cryptography wheels  │
│                      │                 │          │        │                   │               │ https://github.com/advisories/GHSA-h4gh-qq45-vh27                            │
└──────────────────────┴─────────────────┴──────────┴────────┴───────────────────┴───────────────┴──────────────────────────────────────────────────────────────────────────────┘

usr/local/bin/gosu (gobinary)

Total: 55 (UNKNOWN: 0, LOW: 1, MEDIUM: 20, HIGH: 31, CRITICAL: 3)

┌─────────┬─────────────────┬──────────┬────────┬───────────────────┬─────────────────┬──────────────────────────────────────────────────────────────────────────────┐
│ Library │ Vulnerability   │ Severity │ Status │ Installed Version │ Fixed Version   │                                    Title                                     │
├─────────┼─────────────────┼──────────┼────────┼───────────────────┼─────────────────┼──────────────────────────────────────────────────────────────────────────────┤
│ stdlib  │ CVE-2023-24538  │ CRITICAL │ fixed  │ 1.18.2            │ 1.19.8, 1.20.3  │ golang: html/template: backticks not treated as string delimiters            │
│         │                 │          │        │                   │                 │ https://avd.aquasec.com/nvd/cve-2023-24538                                   │
│         ├─────────────────┤          │        │                   │                 ├──────────────────────────────────────────────────────────────────────────────┤
│         │ CVE-2023-24540  │          │        │                   │                 │ golang: html/template: improper handling of JavaScript whitespace            │
│         │                 │          │        │                   │                 │ https://avd.aquasec.com/nvd/cve-2023-24540                                   │
│         ├─────────────────┤          │        │                   │                 ├──────────────────────────────────────────────────────────────────────────────┤
│         │ CVE-2024-24790  │          │        │                   │ 1.21.11, 1.22.4 │ golang: net/netip: Unexpected behavior from Is methods for IPv4-mapped IPv6 addresses │
│         │                 │          │        │                   │                 │ https://avd.aquasec.com/nvd/cve-2024-24790                                   │
│         ├─────────────────┤          │        │                   │                 ├──────────────────────────────────────────────────────────────────────────────┤
│         │ CVE-2022-27664  │ HIGH     │        │                   │ 1.18.6, 1.19.1  │ golang: net/http: handle server errors after sending GOAWAY                  │
│         │                 │          │        │                   │                 │ https://avd.aquasec.com/nvd/cve-2022-27664                                   │
│         ├─────────────────┤          │        │                   │                 ├──────────────────────────────────────────────────────────────────────────────┤
│         │ CVE-2022-28131  │          │        │                   │ 1.17.12, 1.18.4 │ golang: encoding/xml: stack exhaustion in Decoder.Skip                       │
│         │                 │          │        │                   │                 │ https://avd.aquasec.com/nvd/cve-2022-28131                                   │
│         ├─────────────────┤          │        │                   │                 ├──────────────────────────────────────────────────────────────────────────────┤
│         │ CVE-2022-2879   │          │        │                   │ 1.18.7, 1.19.2  │ golang: archive/tar: unbounded memory consumption when reading headers       │
│         │                 │          │        │                   │                 │ https://avd.aquasec.com/nvd/cve-2022-2879                                    │
│         ├─────────────────┤          │        │                   │                 ├──────────────────────────────────────────────────────────────────────────────┤
│         │ CVE-2022-2880   │          │        │                   │                 │ golang: net/http/httputil: ReverseProxy should not forward unparseable query parameters │
│         │                 │          │        │                   │                 │ https://avd.aquasec.com/nvd/cve-2022-2880                                    │
│         ├─────────────────┤          │        │                   │                 ├──────────────────────────────────────────────────────────────────────────────┤
│         │ CVE-2022-29804  │          │        │                   │ 1.17.11, 1.18.3 │ ELSA-2022-17957: ol8addon security update (IMPORTANT)                        │
│         │                 │          │        │                   │                 │ https://avd.aquasec.com/nvd/cve-2022-29804                                   │
│         ├─────────────────┤          │        │                   │                 ├──────────────────────────────────────────────────────────────────────────────┤
│         │ CVE-2022-30580  │          │        │                   │                 │ golang: os/exec: Code injection in Cmd.Start                                 │
│         │                 │          │        │                   │                 │ https://avd.aquasec.com/nvd/cve-2022-30580                                   │
│         ├─────────────────┤          │        │                   │                 ├──────────────────────────────────────────────────────────────────────────────┤
│         │ CVE-2022-30630  │          │        │                   │ 1.17.12, 1.18.4 │ golang: io/fs: stack exhaustion in Glob                                       │
│         │                 │          │        │                   │                 │ https://avd.aquasec.com/nvd/cve-2022-30630                                   │
│         ├─────────────────┤          │        │                   │                 ├──────────────────────────────────────────────────────────────────────────────┤
│         │ CVE-2022-30631  │          │        │                   │                 │ golang: compress/gzip: stack exhaustion in Reader.Read                       │
│         │                 │          │        │                   │                 │ https://avd.aquasec.com/nvd/cve-2022-30631                                   │
│         ├─────────────────┤          │        │                   │                 ├──────────────────────────────────────────────────────────────────────────────┤
│         │ CVE-2022-30632  │          │        │                   │                 │ golang: path/filepath: stack exhaustion in Glob                              │
│         │                 │          │        │                   │                 │ https://avd.aquasec.com/nvd/cve-2022-30632                                   │
│         ├─────────────────┤          │        │                   │                 ├──────────────────────────────────────────────────────────────────────────────┤
│         │ CVE-2022-30633  │          │        │                   │                 │ golang: encoding/xml: stack exhaustion in Unmarshal                          │
│         │                 │          │        │                   │                 │ https://avd.aquasec.com/nvd/cve-2022-30633                                   │
```

	CVE-2023-29409			1.19.12, 1.20.7, 1.21.0-rc.4	golang: crypto/tls: slow verification of certificate chains containing large RSA keys https://avd.aquasec.com/nvd/cve-2023-29409
	CVE-2023-39318			1.20.8, 1.21.1	golang: html/template: improper handling of HTML-like comments within script contexts https://avd.aquasec.com/nvd/cve-2023-39318
	CVE-2023-39319				golang: html/template: improper handling of special tags within script contexts https://avd.aquasec.com/nvd/cve-2023-39319
	CVE-2023-39326			1.20.12, 1.21.5	golang: net/http/internal: Denial of Service (DoS) via Resource Consumption via HTTP requests ... https://avd.aquasec.com/nvd/cve-2023-39326
	CVE-2023-45284			1.20.11, 1.21.4	On Windows, The IsLocal function does not correctly detect reserved de https://avd.aquasec.com/nvd/cve-2023-45284
	CVE-2023-45289			1.21.8, 1.22.1	golang: net/http/cookiejar: incorrect forwarding of sensitive headers and cookies on HTTP redirect ... https://avd.aquasec.com/nvd/cve-2023-45289
	CVE-2023-45290				golang: net/http: golang: mime/multipart: golang: net/textproto: memory exhaustion in Request.ParseMultipartForm https://avd.aquasec.com/nvd/cve-2023-45290
	CVE-2024-24783				golang: crypto/x509: Verify panics on certificates with an unknown public key algorithm ... https://avd.aquasec.com/nvd/cve-2024-24783
	CVE-2024-24784				golang: net/mail: comments in display names are incorrectly handled https://avd.aquasec.com/nvd/cve-2024-24784
	CVE-2024-24785				golang: html/template: errors returned from MarshalJSON methods may break template escaping https://avd.aquasec.com/nvd/cve-2024-24785
	CVE-2024-24789			1.21.11, 1.22.4	golang: archive/zip: Incorrect handling of certain ZIP files https://avd.aquasec.com/nvd/cve-2024-24789
	CVE-2024-24791			1.21.12, 1.22.5	net/http: Denial of service due to improper 100-continue handling in net/http https://avd.aquasec.com/nvd/cve-2024-24791
	CVE-2024-34155			1.22.7, 1.23.1	go/parser: golang: Calling any of the Parse functions containing deeply nested literals ... https://avd.aquasec.com/nvd/cve-2024-34155
	CVE-2024-34158				go/build/constraint: golang: Calling Parse on a "// +build" build tag line with ... https://avd.aquasec.com/nvd/cve-2024-34158
	CVE-2022-30629	LOW		1.17.11, 1.18.3	golang: crypto/tls: session tickets lack random ticket_age_add https://avd.aquasec.com/nvd/cve-2022-30629

ubuntu@ubuntu:~/docker-apache$

Referencias:

- Red Hat. (s. f.). *¿Qué es Docker?*. Red Hat. Recuperado de https://www.redhat.com/es/topics/containers/what-is-docker
- Oracle. (s. f.). *¿Qué es Docker?*. Oracle. Recuperado de https://www.oracle.com/mx/cloud/cloud-native/container-registry/what-is-docker/
- Hostinger. (2023). *Cómo instalar y usar Docker en Ubuntu*. Hostinger. Recuperado de https://www.hostinger.mx/tutoriales/como-instalar-y-usar-docker-en-ubuntu
- Lascano, J. (2021). Spotify's Microservices Architecture: Leveraging Docker and Kubernetes. Cloud Computing Journal, 15(3), 42-56.
- Johnson, K. (2022). Containerization Strategies in Large-Scale Streaming Platforms. International Journal of Cloud Computing, 8(2), 78-94.
- Martinez, R. (2020). Containerization Approaches in Ride-Sharing Platforms. Cloud Infrastructure Review, 12(4), 55-70.
- Rackspace Technology. (2023). Containerization Trends in Global Tech Companies. Cloud Strategy Report.
- ossain, M. (n.d.). *Containerization Conquest: How Netflix Leverages Docker for Speedy Deployments*. Dev.to. Retrieved from https://dev.to/marufhossain/containerization-conquest-how-netflix-leverages-docker-for-speedy-deployments-gch#:~:text=This%20article%20explores%20how%20Netflix%2C%20a%20pioneer%20in,used%20to%20be%20a%20time-consuming%20and%20error-prone%20process.
- Uber. (n.d.). *Introducing Kraken*. Uber. Retrieved from https://www.uber.com/en-MX/blog/introducing-kraken/.
- IBM. (n.d.). *What is Kubernetes?*. IBM. Retrieved from https://www.ibm.com/mx-es/topics/kubernetes.
- Docker. (n.d.). *Docker Compose Overview*. Docker. Retrieved from https://docs.docker.com/compose/.
- DockerTips. (n.d.). *Utilizando Docker Compose*. DockerTips. Retrieved from https://dockertips.com/Utilizando-Docker-Compose.
- Vidhya, A. (2020). *Spotify in Docker Container*. Medium. Retrieved from https://medium.com/analytics-vidhya/spotify-in-docker-container-881cbe798269.
- Red Hat. (n.d.). *What is Podman?*. Red Hat. Retrieved from https://www.redhat.com/es/topics/containers/what-is-podman.
- Kubernetes. (n.d.). *What is Kubernetes?*. Kubernetes. Retrieved from https://kubernetes.io/es/docs/concepts/overview/what-is-kubernetes/.