Segundo laboratorio

1. Trabajar con contenedores

Ejecutar un contenedor con Apache HTTP Server:

```
docker run -dtiP --name web2-lab httpd
```

Este comando ejecuta un contenedor en segundo plano (-d) con una terminal interactiva (-ti), asignando puertos automáticamente (P) y nombrándolo como `web2-lab`.

Ejecutar un contenedor con Nginx en un puerto específico:

```
docker run -dti --name web3 -p 81:80 nginx
```

Este comando ejecuta un contenedor en segundo plano, asigna el puerto local 81 al puerto 80 del contenedor, y lo nombra como `web3`.

Listar contenedores en ejecución:

```
docker ps -a
```

Muestra todos los contenedores, estén activos o detenidos.

Ver reglas de NAT generadas por Docker:

```
iptables -t nat -L DOCKER -v -n
```

Muestra las reglas de red en la tabla NAT gestionadas por Docker.

Consultar la dirección IP del host:

```
192.168.33.10
```

2. Realizar copias de seguridad de imágenes

Filtrar imágenes por nombre:

```
docker images | grep httpd
```

Busca imágenes con el nombre `httpd` en la lista de imágenes locales.

Crear una copia de seguridad de una imagen:

```
docker save -o httpd.tar httpd
```

Guarda la imagen `httpd` en un archivo llamado `httpd.tar`.

Eliminar una imagen local:

```
docker rmi httpd
```

Elimina la imagen `httpd` del sistema local.

Importar una copia de seguridad:

```
docker load -i httpd.tar
```

Carga la imagen desde el archivo `httpd.tar` al sistema local.

Verificar que la imagen fue restaurada:

```
docker images | grep httpd
```

3. Convertir un contenedor en una imagen

Crear un contenedor:

```
docker run -dtiP --name centos6 docker.io/nickistre/centos-lamp
```

Ejecuta un contenedor basado en la imagen `nickistre/centos-lamp`.

Acceder al contenedor:

```
docker exec -ti centos6 /bin/bash
```

Guardar el estado del contenedor como una nueva imagen:

```
docker commit -m "Intranet con joomla" centos6 intranet-martes31
```

Crea una nueva imagen llamada `intranet-martes31` con el mensaje de descripción "Intranet con joomla".

Crear un contenedor desde la nueva imagen:

```
docker run -ditP --name intranet1 intranet-martes31
```

Verificar el contenedor creado:

```
docker ps -l
```

Etiquetar la imagen:

```
docker tag intranet intranet-martes31:v1
```

Crear un contenedor con volúmenes montados:

```
docker run -ditP --name intranet1 -v /bd2:/var/lib/mysql -v /web2:/var/www/html
intranet-martes31
```

También puedes usar volúmenes nombrados:

```
docker run -ditP --name intranet1 -v bd2:/var/lib/mysql -v web2:/var/www/html
intranet-martes31
```

4. Publicar una imagen en Docker Hub

Etiquetar una imagen para Docker Hub:

```
docker tag intranet-lunes19:v1 tuUsuario/intranet
```

Subir la imagen a Docker Hub:

```
docker push tuUsuario/intranet
```

Iniciar sesión en Docker Hub:

```
docker login
```

Ver el archivo de configuración de Docker:

```
cat /root/.docker/config.json
```

Buscar imágenes en Docker Hub:

```
docker search tuUsuario
```

Cerrar sesión en Docker Hub:

```
docker logout
```

5. Trabajar con Nginx y Docker Hub

Etiquetar la imagen de Nginx:

```
docker tag nginx tuUsuario/nginx-lab
```

Subir la imagen de Nginx etiquetada:

```
docker push tuUsuario/nginx-lab
```

Buscar la imagen en Docker Hub:

```
docker search tuUsuario
```

6. Alternativa al registry de docker Hub

Un ejemplo de infraestructura de repositorios privada es harbor.

```
docker login registry.docker-dca.example -u analista -p password-de-nuestra-
instalacion
docker push registry.docker-dca.example/docker-dca/nginx
```