Laboratorio de administración y gestión de redes y sistemas
Examen teorico-práctico parcial. 25 de noviembre de 2020.
Grado en Ingeniería Telemática, Universidad Rey Juan Carlos

- Entra en tu cuenta del laboratorio y ejecuta

  `~mortuno/prepara`

- Esto creará el fichero `~/parcial.2020.TULOGIN.txt` (donde TULOGIN es tu nombre de usuario). Contesta las dos preguntas en este fichero, solo aquí, en ningún otro sitio.

Ejercicio 1 (3 puntos)

Supongamos que tienes que tienes que diseñar el sistema informático basado en Linux para un colegio con unos pocos cientos de ordenadores. La solución podría basarse tanto en virtualización como en clonación. Explica brevemente en qué consiste cada enfoque, cuáles son sus ventajas y cuáles sus inconvenientes. Sin duda en tu respuesta habrá *dependes*. En esos casos, considera las opciones posibles. Ejemplo: Si los puestos son relativamente potentes entonces ..., en otro caso .... Si son muy heterogéneos, ..., y si no, ...

Ejercicio 2 (7 puntos)

Deseamos un par de contenedores con la siguiente especificación:

- Estarán basados en la misma imagen.
- Tendrán ambos el servicio *iperf* listo para atender peticiones TCP en el puerto 9999.
- Montarán ambos por **sshfs** en el directorio */media/(lab)* tu *home* de la maquina *f-l2108-pc02*. Supodremos que está funcionando bien, en un sistema más robusto habría que verificarlo, pero aquí lo omitimos). Observa que **no** se te pide un montaje *bind*.

  La autenticación será manual, esto es, al arrancarse el contenedor, te pedirá tu contraseña en el laboratorio.

  Esto no lo has hecho en prácticas (usar sshfs dentro de un contenedor). Pero sí lo vimos en clase de teoría. Recuerda que la orden `docker run` necesitará ciertos parámetros, documentados en las transparencias sobre Docker.

Especifica los ficheros necesarios para conseguir esto en Docker (una versión moderna, la incluida Ubuntu 20.04):

1. Indica qué ficheros habrá en el directorio *context* y con qué contenido.
2. Al igual que en prácticas, queremos un script para preparar la imagen. Indica qué nombre tendría y qué contenido.
3. Al igual que en prácticas, queremos un script para lanzar cada contenedor. Indica qué nombre tendrían y qué contenido.
4. Al lanzar los contenedores, el usuario escribirá su contraseña, a continuación deberá averiguar la IP del *otro* contenedor y lanzar el cliente de iperf. Indica cómo hacer estos pasos.

Solución

- Fichero `context/Dockerfile`

```
FROM ubuntu:20.04

RUN apt update && apt upgrade -y
RUN apt-get install -y net-tools iputils-ping sshfs iperf

COPY entrypoint.sh /

EXPOSE 9999
CMD ["/entrypoint.sh"]
```

- Fichero `context/entrypoint.sh`.

```
#!/bin/bash
iperf -s -p 9999 &
mkdir /media/lab
sshfs mortuno@f-l2108-pc02.aulas.etsit.urjc.es: /media/lab
/bin/bash
```

- Script para crear la imagen, `construye.sh`

```
#!/bin/bash
docker build -t jperez/exa context
```

- Script para lazar el primer contenedor, `lanza_jperexa01.sh`

```
#!/bin/bash
PREFIJO=jper
IMAGEN=exa
USUARIO=jperez
CONTENEDOR=$PREFIJO${IMAGEN}01
docker run --rm -it -h $CONTENEDOR --name $CONTENEDOR   \
    --cap-add SYS_ADMIN --device /dev/fuse  \
    --security-opt apparmor:unconfined \
    $USUARIO/$IMAGEN
```

  El script para lanzar el segundo contenedor, `lanza_jperexa02.sh` es idéntico, excepto la línea `CONTENEDOR=$PREFIJO${IMAGEN}02`

- Todos los scripts necesitan permiso de ejecución

```
\verb|chmod ugo+x context/entrypoint.sh|
\verb|chmod ugo+x *.sh|
```

- Una vez lanzados los contendores, introducimos la contraseña, averiguamos la dirección IP de cada uno con la orden *ifconfig* y ejecutamos

```
iperf -c <IP_SERVIDOR> -p 9999
```

  Donde `<IP_SERVIDOR>` es la dirección IP del contenedor donde está el servidor iperf.