Laboratorio de Administración y Gestión de Redes y Sistemas
Grado en Ingeniería Telemática, 2024-2025

Prácticas bloque 1

Escuela de Ingeniería de Fuenlabrada
Universidad Rey Juan Carlos

2024-2025

Cambios en el documento

- 13 de noviembre. Práctica 1.1. Añadida indicación sobre `~/lagrs/docs`

Formato de la memoria de prácticas

La memoria de estas prácticas debes escribirla en un fichero en formato markdown. El fichero estará en tu cuenta del laboratorio. El dia del examen, recogeremos automáticamente tus prácticas.

1. Es imprescindible que respetes al pie de la letra los nombres de los ficheros especificados en el guión. Una letra mal puesta equivale a una práctica no presentada (o un examen no presentado). Si bien dispondrás de un script que verificará que has usado los nombres correctos

2. Redacta la memoria describiendo escuetamente todo lo que haces. Esta memoria te será muy útil para preparar el examen práctico (recuerda que podrás llevarla al examen).

   No merece la pena que te preocupes de tener una redacción muy cuidada: copia y pega órdenes y resultados, describe telegráficamente lo que haces. Vete preparando la memoria a la vez que trabajas, no lo dejes para el final.

   a) Cuando un paso del guión te pida una orden de shell o similar, es recomendable que primero ejecutes la orden y luego copies y pegues en la memoria. Si resulta algún texto, pégalo también.
   b) Cuando el guión pida que escribas o edites un fichero, basta con que lo modifiques en su sitio. No es necesario que copies y pegues en la memoria.

Práctica 1.1. Directorios de las prácticas

Ahora prepararás el directorio de la práctica 1 y el fichero de texto para la memoria de la esta práctica.

1. Entra en tu cuenta del laboratorio Linux de la ETSIT. Crea el directorio `~/lagrs`. Aquí guardarás la mayoría de tu trabajo de prácticas en la asignatura

2. Ponle permisos `rwx------`

   Debes mantenerlo así todo el curso. Recuerda que debes ser autor del 100% de tus prácticas y no permitir que ningún compañero tuyo tenga acceso a ningún fragmento.

3. Crea el directorio

   `~/lagrs/practica01`

4. Crea el fichero

   `~/lagrs/practica01.md`

   Observa que el nombre de este fichero **no** es

   ```
   ~/lagrs/practica01/practica01.md       # ¡Este no es el nombre correcto!
   ```

   Por el momento déjalo vacío.

5. Crea un directorio

   `~/lagrs/images`

   para las imágenes de tus memorias de prácticas (aunque tal vez tus ficheros no tengan imágenes)

6. Crea un directorio

   `~/lagrs/docs`

   Guarda aquí cualquier documento o *chuleta* relacionado con la asignatura que te parezca útil y que quieras usar el día del examen.

7. Seguiremos este convenio en todas las prácticas, un directorio para cada práctica y un fichero de texto plano para cada práctica, fichero que cuelga de `~/lagrs`

Práctica 1.2. Uso básico de vi

El objetivo de esta práctica es que sepas usar al menos las órdenes elementales de vi. Si no te gusta este editor, podrás usar algún otro editor sencillo en modo texto. Pero para poder instalar un editor nuevo, necesitas usar vi.

1. Usando vi, crea el fichero `~/lagrs/practica01/ejemplo.md` y escribe en él 4 titulares de cualquier periódico de hoy. Escribe 3 o 4 faltas de ortografía, intencionadamente. Guárdalo.

2. Vuelva a abrirlo y corrige las faltas.

Práctica 1.3. Uso de un editor sin gráficos

Ahora empezarás a usar el fichero de la memoria de prácticas. Escribe la memoria de este apartado y de todos los apartados siguientes de esta práctica en el fichero que creaste en el apartado 1.1. `~/lagrs/practica01.md`

Es necesario que manejes con soltura funciones al menos intermedias de algún editor de texto sin gráficos. La recomendación es vi/vim, pero también puedes elegir uno más sencillo.

1. Lee las transparencias sobre editores de texto. Arranca al menos una vez los editores en modo texto presentados (vim, mcedit, joe y nano). Elige uno, para usar en esta asignatura. ¿Cuál has elegido? ¿por qué?. Puedes cambiar de opinión mas adelante, pero en tal caso, indícalo (recuerda que estas prácticas las recogeremos el día del examen).

2. Es muy conveniente conocer atajos de teclado. Es mucho más eficaz memorizar, por ejemplo, 2 al día durante 5 días que intentar retener 10 atajos 1 día. Aprende unos cuantos de esta forma, al menos media docena, e indícalo aquí. Ejemplo:

   ```
   2022.09.24
   Elijo vi porque es el más potente. Practico el copy-paste con yy p
   2022.09.28
   Me coloco en una palabra y pulso asterisco para buscar esa misma palabra
   en el texto. n minúscula para repetir la búsqueda hacia adelante y N mayúscula
   para buscar hacia atrás.
   2022.09.30
   Practico ma, mb, mc para poner las marcas a, b y c en un texto. Vuelvo
   a las marcas con 'a 'b 'c
   ```

Práctica 1.4. Markdown

Prepara un documento llamado
`~/lagrs/practica01/test_markdown.md`

1. Escribe en él un texto de prueba con diferentes secciones, negritas, cursivas, enlaces, imágenes, listas de varios tipos, código fuente y alguna tabla.
2. Usa `pandoc` para *limpiar* el fichero.
3. Usa `pandoc` para generar una versión en html usando alguna de las plantillas css que vimos en clase.

Práctica 1.5. Gestión de contraseñas

Guarda dos o tres contraseñas, de prueba o reales, usando

1. gpg
2. LibreOffice
3. KeePassx. O alguna herramienta similar como Bitwarden.

Usa recordatorios de contraseña en todos los casos, en un fichero de texto aparte. Guarda los ficheros donde quieras, describe brevemente en la memoria los pasos que has seguido

Práctica 1.6. Secret Sharing

Usa ssss para descomponer una contraseña en 6 trozos, de forma que baste con 4 para restaurarlos. Recomponla.

Práctica 1.7. Vagrant

En este ejercicio probarás Vagrant, lanzando una máquina virtual de VirtualBox. Si quieres hacer esta práctica en tu ordenador de casa, tendrás que instalar Vagrant y VirtualBox si no lo tenías ya.

1. Si estás en el laboratorio, configura VirtualBox para usar `/var/tmp/tulogin` como *carpeta predeterminada de máquinas*
2. Crea un *project directory* de Vagrant en el directorio `~/lagrs/vbox01` de tu pc (de casa o del laboratorio)
3. Prepara una máquina Ubuntu 24.04 LTS Noble Numbat.
4. Lanza la máquina y entra por ssh, usando vagrant.
5. Comprueba que el *project directory* del host está montado dentro de la máquina virtual en `/vagrant`

   Para ello, edita un fichero en este directorio desde el *guest* o el *host*, guárdalo y vuelve a editarlo desde la otra máquina (el *host* si antes usaste el *guest* y viceversa).
6. Apaga la máquina desde vagrant, sin destruirla.
7. Apaga la máquina (sin destruirla) y haz que su nombre sea `vbox`, tanto visto desde VirtualBox como desde dentro de la propia máquina. Vuelve a lanzarla y comprueba que lo has hecho bien. Observa la máquina virtual desde el GUI de VirtualBox.
8. Si hiciste la práctica en casa o en tu portátil, copia el directorio `~/lagrs/vbox01` de tu pc de al directorio `~/lagrs` de tu cuenta del laboratorio. De esta forma, el directorio `~/lagrs/vbox01` de ambas máquinas será idéntico.

Práctica 1.8. Usuarios y grupos

En este ejercicio practicarás con los usuarios y grupos de linux.

1. Vuelve a entrar en la máquina virtual del apartado anterior.
2. Crea un nuevo usuario, con el mismo nombre que tu usuario en el laboratorio.
3. Abre una sesión de este usuario (con la orden `su`).
4. Este usuario no tendrá privilegios para ejecutar sudo, pero intenta lanzar alguna orden y observa qué pasa.
5. Haz que este usuario sí pueda ejecutar sudo. Pruébalo.
6. Con este usuario, crea dos nuevos grupos con el nombre que quieras. Mete al usuario en estos grupos. Comprueba que están incluidos.
7. Prueba la orden `newgrp`, observa su efecto.

Práctica 1.9. ssh sin contraseñas (1)

Configura tu cuenta del laboratorio para entrar desde cualquier máquina hasta cualquier máquina, sin escribir contraseña.

Práctica 1.10. ssh sin contraseñas (2)

Configura la máquina virtual vbox01 para poder abrir una sesión del usuario que creaste en el apartado 1.8, desde tu cuenta del laboratorio, sin escribir contraseña.

Práctica 1.10bis. Provisionamiento del box

Editando el fichero `~/lagrs/vbox01/Vagrantfile`, provisiona el *box* para:

1. Crear automáticamente el usuario con el mismo nombre que tu cuenta en el laboratorio. Con sus grupos y privilegios.
2. Que este usuario pueda abrir sesión en el *box* desde el puesto del laboratorio con criptografía asimétrica, sin necesida de teclear contraseñas. Para ello, prepara un fichero `~/lagrs/vbox01/authorized_keys` y haz que en el provisionamiento se copie en el lugar adecuado de la máquina virtual.

De esta forma, cada vez que trabajes en un puesto nuevo del laboratorio, podrás crear automáticamente un nuevo *box*. Antes de usar uno nuevo, tendrás que borrar el viejo. Normalmente podría usarse el comando `vagrant destroy`, pero esta orden no *sabe* que la máquina ya no está en `/var/tmp/tulogin`, así que tendrás que eliminar la máquina desde el GUI de VirtualBox.

Práctica 1.11. scp

Mediante scp:

1. Copia un fichero cualquiera desde tu puesto del laboratorio al directorio `/tmp` de un puesto vecino
2. Entra en un puesto vecino, crea el directorio `/tmp/tulogin`. (donde *tulogin* es tu nombre de usuario en el laboratorio). En este directorio, crea unos cuantos ficheros. Copia este directorio al directorio `/tmp/` de tu puesto.
3. Copia un fichero cualquiera desde tu puesto hasta el directorio `/tmp` de `vbox`. Recuerda, mediante scp, sin usar el directorio `/vagrant` de la máquina `vbox`
4. Copia un directorio cualquiera desde tu puesto hasta el directorio `/tmp` de `vbox`.

Práctica 1.12. split

Ahora practicarás el troceado de ficheros.

1. Prepara en tu *host* un directorio cualquiera con unos cuantos ficheros de cierto tamaño, unos cuantos megas. Por ejemplo un puñado de fotos.
2. Calcula y guarda los *hash* de todos estos ficheros.
3. Compríme el directorio (nos los ficheros por separado) en un .tgz.
4. Usando *split*, trocea este fichero .tgz en unos cuantos ficheros más pequeños.
5. Mediante *scp*, copia los trozos en el directorio `/tmp` de `vbox`.
6. En `vbox`, recompón los trozos y extrae los ficheros.
7. Calcula los hash de los ficheros, comprueba que sean idénticos a los orioginales

Práctica 1.13. rsync

Elige los directorios que quieras en las máquinas que quieras (una local, otra remota). Debes poder acceder desde la local a la remota sin escribir contraseña. Usando rsync:

1. Clona un directorio local en uno remoto.
2. Clona un directorio remoto en uno local.
3. Consulta la página de manual, elige tres opciones y comprueba que se comportan como deben.

Práctica 1.14. FreeFileSync

Ahora practicarás el uso de FreeFileSync para simular que sincronizas tu pc de casa con el laboratorio. Empieza creando en tu pc las carpetas `~/simula_labo/lagrs` y `~/simula_casa/lagrs`. Vamos a imaginar que

1. El puesto físico del laboratorio es tu ordenador de casa.
2. La carpeta `~/simula_casa/lagrs` es la carpeta `lagrs` de tu ordenador de casa.
3. La carpeta `~/simula_labo/lagrs` es la carpeta `lagrs` de tu cuenta en el laboratorio.

Observaciones:

- Hay una diferencia entre esta simulación y el trabajo normal en casa: como el puesto físico y el puesto virtual comparten la cuenta, desde cualquiera de los dos podrás escribir en cualquiera de las dos carpetas. Pero para hacer bien este ejercicio, tienes que usar la carpeta `~/simula_casa/lagrs` solamente desde tu puesto físico y la carpeta `~/simula_labo/lagrs` solamente desde el puerto virtual.

Práctica 1.15. Conflictos con FreeFileSync

1. Operando de forma análoga al ejercicio anterior, y con los mismos ficheros, simula un conflicto de sincronización. Observa los mensajes de error de FreeFileSync.
2. Resuelve el conflicto. Comprueba que FreeFileSync ya no muestra errores.

Práctica 1.16. Sincronización real de tu cuenta (práctica imprescindible)

Prepara en tu PC de casa un directorio con nombre *lagrs*, en el directorio que prefieras (mis documentos, escritorio...) y sincroniza allí tu cuenta *lagrs* del laboratorio. **Mantenlo sincronizado todo el curso**. Te será util como copia de seguridad y para trabajar indistintamente en ambos entornos. Recuerda que eres responsable de custodiar tus prácticas: tu directorio del laboratorio podría perderse. (Sería raro pero no imposible, el servicio ofrecido es de tipo *best effort*).

En otras palabras: si haces mal esta práctica, si no mantienes durante todo el curso una copia de seguridad de tu trabajo, podrías suspender la asignatura.

Recuerda: esta no es una práctica puntual. Esta práctica consiste en mantener sincronizados estos directorio, de forma continua, **todo el curso**.

Práctica 1.17 Localización de procesos

Desde el interface gráfico de Gnome podemos lanzar programas. Pero ¿cómo saber el nombre de esos programas si queremos lanzarlos desde la shell o desde un script?. Una forma es guardar en un fichero el resultado de la orden `ps -ef`, lanzar el programa, volver a capturar la salida de `ps` y comparar ambos ficheros con diff.

Práctica 1.18 Invocación de la shell

1. Observa los ficheros de inicio de la shell de tu cuenta del laboratorio. Créalos si no los tienes. ¿Cuándo se ejecuta `.bashrc`? ¿Solamente en las shell de login? ¿Solamente en las shell que no son de login? ¿O en ambos tipos de shell? ¿Por qué?
2. Comprueba que los diferentes tipos de ficheros que se tienen que ejecutar en la invocación de la shell (interactivo y de login, interactivo no de login, no interactivo) se comportan como cabe esperar. Usa trazas como

   ```
   echo prueba blabla
   ```

   o

   ```
   echo prueba blabla >> /tmp/mi_traza.txt
   ```

3. Prepara tu cuenta del laboratorio para que el `.bash_profile` invoque al `.bashrc` (si es que no está así ya). Es recomendable que también lo hagas en tu pc de casa.
4. Usando la máquina virtual, comprueba qué sucede cuando, desde la shell, cambiamos de usuario mediante la orden `su`. ¿Qué ficheros se ejecutan? ¿Pasamos a tener una shell de login o no?

Práctica 1.19. Instalación de Docker

En este apartado instalarás Docker en una máquina Linux

1. Instala el paquete *docker.io* en la máquina virtual *vbox01*
2. Comprueba que la instalación de docker es correcta, lanzando un contenedor de tipo *holamundo* basada en debian.
3. Haz una prueba de tipo *holamundo* basada en ubuntu.

Práctica 1.20. Uso básico de imágenes

En este apartado empezarás a usar los contenedores en la máquina virtual *vbox01*. Mientras el enunciado no especifique lo contrario, usa imágenes basadas en Ubuntu 24.04.

1. Crea un contenedor interactivo. No le pongas nombre. Ejecuta alguna orden básica de la shell. Indica alguna orden básica que esté disponible y alguna otra que no.
2. No detengas el contenedor. Crea otro contenedor interactivo, poniéndole el nombre xxxxc01, donde xxxx son las primeras 4 letras de tu nombre de usuario en el laboratorio. Para jperez, sería jperc01.
3. En un nuevo terminal, usa las ordenes de docker listar contenedores e imágenes. Explica lo que estás viendo.
4. Termina la ejecución de los contedores y observa el estado de las imágenes y de los contenedores detenidos.
5. Comprueba que el sistema de ficheros dentro del contenedor no es persistente. Esto es: escribe algún fichero, apaga el contenedor, vuelva a entrar y observa que ha desaparecido.

Práctica 1.21. Creación de una imagen de un contenedor

1. Crea una cuenta en docker hub.
2. Prepara la imagen `test/banner` descrita en las transparencias, pero llámala `tulogin/banner` siendo *tulogin* tu nombre de usuario en docker hub.
3. Lanza un contenedor con esa imagen.
4. Modifica la imagen para que una vez lanzada, no solo muestre el banner sino que abra una shell
5. Lanza un contenedor con la imagen para comprobar que puedes ejecutar la shell.
6. Sube la imagen a docker hub.

Práctica 1.22. Creación de una imagen personalizada

En este apartado prepararás una imagen sencilla de un contenedor, dentro de la máquina virtual vbox01. Se llamará *¡TULOGIN¿/cal*, y lo único que hará será invocar a la orden de shell `cal` para mostrar el calendario del mes actual y concluir.

Vamos a establecer los siguientes convenios, que mantendremos el resto de las prácticas:

- Usaremos ubuntu:24.04 como distribución base
- Si la imagen se llama `cal`, y tu login en el laboratorio es `jperez`, los distintos contenedores que ejecutarás a partir de ella se llamarán `jpercal01`, `jpercal02`, etc

  Esto es: las primeras 4 letras de tu nombre de usuario, el nombre de la imagen y un número de dos dígitos.
- Todo lo necesario para construir y lanzar esta imagen estará en el directorio del laboratorio.

  `~/lagrs/vbox01/cal`

  Naturalmente, cuando ejecutes la máquina virtual, este mismo directorio estará en

  `/vagrant/cal`
- El fichero `~/lagrs/vbox01/cal/construye.sh` será un script para construir la imagen `<TULOGIN>/cal`

- El fichero `~/lagrs/vbox01/cal/lanza_jpercal01.sh` será un script para lanzar el contenedor `jpercal01`.

  El fichero `~/lagrs/vbox01/cal/lanza_jpercal02.sh` será un script para lanzar el contenedor `jpercal02`, y así sucesivamente (si quisiéramos lanzar más contenedores)

  Observa que los scripts `lanza_jpercal01.sh` `lanza_jpercal02.sh` incluye en su nombre los nombres de los contenedores, porque varios contenedores podrán compartir la misma imagen inicial.

  Observa que el script `construye.sh` no incluye el nombre de la imagen, porque dentro del directorio `~/lagrs/cal/` solo habrá ficheros relativos a la imagen `<TULGIN>/cal`

- El directorio `~/lagrs/vbox01/cal/context` contendrá el contexto para esta imagen. Por tanto, los ficheros Dockerfile y entrypoint.sh estarán, respectivamente, `~/lagrs/vbox01/cal/context/Dockerfile` y `~/lagrs/vbox01/cal/context/entrypoint.sh`

La siguiente tabla contiene todos los ficheros que usarás, con el nombre visto desde *los hierros* (el puesto del laboratorio) y el nombre visto desde dentro de la máquina vbox01

`~/lagrs/vbox01/cal`	`/vagrant/cal`
`~/lagrs/vbox01/cal/construye.sh`	`/vagrant/cal/construye.sh`
`~/lagrs/vbox01/cal/lanza_jpercal01.sh`	`/vagrant/cal/lanza_jpercal01.sh`
`~/lagrs/vbox01/cal/lanza_jpercal02.sh`	`/vagrant/cal/lanza_jpercal02.sh`
`~/lagrs/vbox01/cal/context`	`/vagrant/cal/context`
`~/lagrs/vbox01/cal/context/Dockerfile`	`/vagrant/cal/context/Dockerfile`
`~/lagrs/vbox01/cal/context/entrypoint.sh`	`/vagrant/cal/context/entrypoint.sh`

Atendiendo a estos convenios,

1. Prepara la imagen del contenedor solicitado. La utilidad cal está en el paquete bsdmainutils.
2. Prepara el script `lanza_jpercal01.sh` y lánzalo.
3. Prepara el script `lanza_jpercal02.sh` y lánzalo.
4. Ejecuta `docker ps -a` y `docker images`, y observa que, naturalmente, se puede comprobar que ambos contenedores están basados en la misma imagen.

Práctica 1.23. Montaje bind

Ahora prepararás una imagen de un contenedor que hará un montaje de tipo *bind*. Este contenedor se ejecutará dentro de la máquina virtual vbox01, como todos los demás que preparemos en esta asignatura, sin necesidad de que el enunciado lo diga explícitamente.
Siguiendo el criterio establecido en el apartado anterior, y suponiendo que tu usuario sea `jperez`

- El directorio del contenedor, visto desde el laboratorio, será `~/lagrs/vbox01/bind/`

  Como sabes, dentro de vbox01 este directorio estará montado en

  `/vagrant/bind/`

- El directorio contexto, `~/lagrs/vbox01/bind/context`
- El contenido del directorio contexto:

  `~/lagrs/vbox01/bind/context/Dockerfile`

  `~/lagrs/vbox01/bind/context/entrypoint.sh`

- El script de creación de la imagen, `~/lagrs/vbox01/bind/construye.sh`
- El script de lanzamiento del contenedor, `~/lagrs/vbox01/bind/lanza_jperbind01.sh`

Funcionamiento del contenedor:

- El contenedor tendrá instalado tu editor de texto preferido.
- El contenedor ejecutará una shell.
- El contenedor montará el directorio `/tmp/test` de vbox01 en el directorio `/tmp/test` del contenedor

Prueba del contenedor:

1. Lanza el contenedor y escribe en el directorio montado un fichero con nombre `hola_jperez`
2. Comprueba que ese directorio es persistente. Esto es, sal del contenedor, vuelve a lanzarlo y observa que el fichero `hola_jperez` sigue en su sitio.
3. Accede a este mismo fichero desde vbox01. Comprueba sus permisos. Edita el fichero con un texto cualquiera. Comprueba que puedes ver el contenido tanto en vbox01 como en el contenedor.

Práctica 1.24. Conectividad entre contenedores

En este apartado instalarás sshd en un contenedor y probarás la red bridge de Docker.

- Los contenedores estarán basados en Ubuntu 24.04.
- El nombre de la imagen que crearás será `<TULOGIN>/remoto`

  Como prácticas anteriores, al nombre de cada contenedor le añadirás como prefijo las primeras 4 letras de tu usuario, y como sufijo, un número de dos dígitos.
- Lanzarás dos contenedores con esta imagen.

  Por tanto, los nombres de los ficheros necesarios serán:

  `~/lagrs/vbox01/remoto/context/Dockerfile`

  `~/lagrs/vbox01/remoto/context/entrypoint.sh`

  `~/lagrs/vbox01/remoto/construye.sh`

  `~/lagrs/vbox01/remoto/lanza_jperremoto01.sh`

  `~/lagrs/vbox01/remoto/lanza_jperremoto02.sh`
- Prepara la imagen de forma que el contenedor que la ejecute tenga lanzado el demonio servidor de ssh.
- Dentro del contenedor debe ser posible ejecutar *ifconfig* y *ping*. Por tanto tendrás que instalar estas aplicaciones en la imagen. Para ello tendrás que averiguar el nombre de los paquetes Ubuntu necesarios. Lo más sencillo es googlear un poco.
- El contenedor tendrá un usuario, con el mismo nombre que tu usuario en el laboratorio y con privilegios para ejecutar sudo.

Una vez que estén listas las imágenes, lanza ambos contenedores y, de forma interactiva, para cada uno de ellos

1. Averigua su dirección IP.
2. Haz ping a tu propio contenedor.
3. Usando *netstat*, comprueba que el servidor *sshd* está funcionando.
4. Comprueba que puedes hacer ping al *otro*.
5. Abre una sesión ssh desde este contenedor al otro.

Práctica 1.25. sshfs

Haz una prueba básica de sshfs. Si tienes aquí tu portátil, monta tu *home* del laboratorio en el directorio que prefieras de tu ordenador. Si no, monta tu *home* del laboratorio en el directorio `/tmp/labo` de la máquina vbox01.

Práctica 1.26. Contenedor con sshfs

En este ejercicio prepararás un contenedor docker, configurado en español, con un usuario del mismo nombre que tu cuenta del laboratorio, capaz de montar un directorio remoto, usando sshfs.

- La imagen se llamará *jperez/cssh* (reemplazando, como siempre, *jperez* por tu nombre de usuario en el labortorio)
- El contenedor lanzado a partir de esta imagen se llamarán jpercssh01
- Todos los ficheros necesarios estarán en el directorio `~/lagrs/vbox01/cssh` de tu cuenta del laboratario
- Los scritps se llamarán

  `~/lagrs/vbox01/cssh/construye.sh`

  `~/lagrs/vbox01/cssh/lanza_jpercssh01.sh`

  (donde jper representa las primeras 4 letras de tu login)
- El directorio contexto será

  `~/lagrs/vbox01/cssh/context`

  La imagen se creará con los ficheros

  `~/lagrs/vbox01/cssh/context/Dockerfile`

  `~/lagrs/vbox01/cssh/context/entrypoint.sh`

Para ello:

1. Prepara la imagen con los paquetes necesarios, la configuración en español y un usuario.
2. Lanza el contenedor con las opciones necesarias para usar sshfs.
3. Empieza probando sshfs desde el usuario *root* del contenedor:

   Monta el directorios `/tmp` de una máquina cualquiera de las que estén disponibles en el laboratorio (consulta el *parte de guerra*). Por ejemplo `f-l2108-pc05` en el directorios `/tmp/pc05` del contenedor. (Esto es, el directorio local será `/tmp/pcNN`, siendo NN el número del puesto)

   Comprueba que el resultado es el esperado: entra por ssh en esa máquina, edita algún fichero en el directorio `/tmp/`, comprueba que puedes editar el mismo fichero en el directorio montado en tu máquina local.
4. Ahora repite el paso anterior pero con tu usuario del contenedor, no con el usuario *root*.

Práctica 1.27. Contenedor con fichero hosts

En este apartado prepararás un contenedor preparado para hacer ping y ssh a las máquinas del laboratorio, usando solo el nombre de host, no el FQDN (Fully Qualified Domain Name).

Esta imagen se llamará `<TULOGIN>/chosts` (contenedor hosts), estará basada en Ubuntu 24.04 y configurado en español.

Siguiendo el convenio descrito en el apartado anterior:

- Los contenedores lanzados a partir de esta imagen se llamarán jperchosts01, jperchosts02, etc

- Todos los ficheros necesarios estarán en `~/lagrs/chosts`
- Los scritps se llamarán

  `~/lagrs/vbox01/chosts/construye.sh`

  `~/lagrs/vbox01/chosts/lanza_jperchosts01.sh`

  `~/lagrs/vbox01/chosts/lanza_jperchosts02.sh`

  (donde jper representa las primeras 4 letras de tu login)
- El directorio contexto será

  `~/lagrs/vbox01/chosts/context`

  La imagen se creará con los ficheros

  `~/lagrs/vbox01/chosts/context/Dockerfile`

  `~/lagrs/vbox01/chosts/context/entrypoint.sh`

Para conseguir que los contenedores conozcan las direcciones IP de las máquinas del laboratorio, tendrás que añadir al fichero `/etc/hosts` de cada imagen las entradas correspondientes al laboratorio, que encontrarás en el fichero `/etc/hosts` de cualquier puesto.

Para ello

- Prepara, en el directorio contexto, un fichero `delta_hosts` que contenga las entradas necesarias.
- Haz que este fichero aparezca en el directorio `/tmp/` de la imagen.
- Haz que cada vez que se inicie la imagen, se añadan estas entradas al `/etc/hosts` del contenedor.

  (En docker no es posible borrar ni reemplazar el fichero `/etc/hosts` de una imagen, pero sí puedes modificarlo añadiendo entradas)
- Haz que desde los contenedores se pueda hacer ifconfig, ping y ssh. Para ello necesitarás los paquetes ubuntu adecuados, averigua su nombre consultando Google o ChatGPT.

  Puedes instalar algún paquete adicional si quieres, con tal de que no resulte una imagen mucho más pesada.

Comprueba que una vez lanzado el contenedor, puedes hacer ping o entrar por ssh a los puestos del laboratorio que lo soporten, sin necesidad de escribir el FQDN, esto es, sin añadir al nombre el dominio aulas.gsyc.urjc.es.

Ten en cuenta que:

- Solo algunas máquinas del laboratorio acepta conexión por ssh.
- El cortafuegos del laboratorio no permite el tráfico de paquetes ICMP (lo que incluye el ping) desde fuera de su subred. Por tanto, podrás ver que se ejecuta la orden ping y que se envía la petición a la dirección IP adecuada, pero no verás respuesta.