

Prácticas Ingeniería de Servidores

Ismael Sallami Moreno

`ism350zsallami@correo.ugr.es`

`https://ismael-sallami.github.io/`

`https://elblogdeismael.github.io/`

Universidad de Granada

Licencia

Este trabajo está licenciado bajo una [Licencia Creative Commons Reconocimiento-NoComercial-SinObraDerivada 4.0 Internacional](https://creativecommons.org/licenses/by-nc-nd/4.0/). <https://creativecommons.org/licenses/by-nc-nd/4.0/>

Usted es libre de:

- Compartir — copiar y redistribuir el material en cualquier medio o formato.

Bajo los siguientes términos:

- **Reconocimiento** — Debe otorgar el crédito adecuado, proporcionar un enlace a la licencia e indicar si se han realizado cambios. Puede hacerlo de cualquier manera razonable, pero no de una manera que sugiera que tiene el apoyo del licenciante o lo recibe por el uso que hace.
- **NoComercial** — No puede utilizar el material para fines comerciales.
- **SinObraDerivada** — Si remezcla, transforma o crea a partir del material, no puede distribuir el material modificado.



Índice general

I	Apuntes de Clase	5
1.	Bloque 1	7
1.1.	Ping y SSH	7
1.2.	LVM y RAID	7
1.2.1.	LVM (Logical Volume Manager)	7
1.2.2.	Directorios base en Linux	7
1.2.3.	RAID (Redundant Array of Independent Disks)	8
1.2.4.	Ventajas	8
1.2.5.	Niveles de RAID	8
1.2.6.	Tipos de RAID	8
1.2.7.	Ejercicio Opcional: Configuración de RAID1 para /var	9
1.2.8.	Objetivo	9
1.2.9.	Pasos a seguir	9
1.2.10.	LVM (Logical Volume Manager)	9
1.2.11.	Conceptos Clave	9
1.2.12.	Visualización de LVM	10
1.2.13.	Consideraciones sobre Volumen Groups	10
1.2.14.	Creación del LVM sobre el RAID	10
1.2.15.	Movimiento de /var al RAID	10
1.2.16.	Resolución de Problemas	10
1.2.17.	Problema 1: Configurar el sistema de archivos en rvar	10
1.2.18.	Problema 2: Montar y trasladar /var al nuevo volumen	11
1.2.19.	Interpretación de lsblk	12
1.3.	Firewall + SSHD	12
1.3.1.	Ejercicio Opcional	13
1.3.2.	SSH	13
1.3.3.	Ejercicio Opcional	16
II	Prácticas	17
2.	Bloque 1	19
2.1.	Ejercicio 1 Opcional	19
2.1.1.	Solución	19
2.2.	Servidor con LVM + RAID	23
2.2.1.	Aspectos clave de LVM	23
2.2.2.	Niveles de RAID: 0, 1 y 5	24
2.2.3.	Aspectos clave para la administración de servidores Linux . .	26

2.2.4. Ejercicio Opcional 27

Parte I

Apuntes de Clase

Capítulo 1

Bloque 1

1.1. Ping y SSH

Sobre esta parte no decidí tomar apuntes en clase debido a que son conceptos vistos en la Asignatura de Fundamentos de Redes, por lo que consideré que no era necesario. De todas formas en la parte de Prácticas (Resolución) se comenta paso a paso lo que se hizo en esta parte.

1.2. LVM y RAID

1.2.1. LVM (Logical Volume Manager)

- **Discos y particiones:**
 - **sda**: Primer disco del sistema.
 - Luego pueden existir **sdb**, **sdc**, etc.
- **Consideraciones importantes:**
 - Si el directorio **/boot** se llena, el sistema podría fallar al arrancar debido a la falta de espacio disponible.
 - Se recomienda evitar el uso de **swap** en sistemas virtualizados, ya que reduce el rendimiento.

1.2.2. Directorios base en Linux

- **/boot**: Contiene los archivos de arranque del sistema.
- **/etc**: Almacena los archivos de configuración del sistema operativo.
- **/dev**: Contiene archivos especiales que representan dispositivos del sistema.
- **/mnt**: Punto de montaje temporal para sistemas de archivos.
- **/var**: Contiene datos variables del sistema, como logs y archivos temporales. Puede crecer mucho, por lo que se recomienda monitorearlo.

1.2.3. RAID (Redundant Array of Independent Disks)

RAID es una tecnología que permite combinar múltiples discos para mejorar la redundancia y/o el rendimiento del sistema de almacenamiento.

1.2.4. Ventajas

- Permite unir volúmenes de almacenamiento.
- Mejora el rendimiento si los discos están en buses distintos, ya que permite acceso en paralelo.

1.2.5. Niveles de RAID

- **RAID 0 (Striping):**
 - Divide los datos en bloques y los distribuye entre varios discos.
 - **Problema:** Si un disco falla, se pierde toda la información.
 - Se usa poco en la práctica debido a su baja robustez.
- **RAID 1 (Mirroring):**
 - Duplica los datos en dos o más discos.
 - Si un disco falla, el otro sigue funcionando con los mismos datos.
 - Aporta robustez al sistema.
 - **Problema:** Se paga el doble en almacenamiento.
 - Se usa frecuentemente en `/boot` para garantizar que el sistema pueda arrancar en caso de fallo de un disco.
- **RAID 5 (Paridad distribuida):**
 - Se distribuyen los datos y la paridad entre todos los discos.
 - Si un disco falla, se pueden recuperar los datos utilizando la paridad.
 - **Problema:** Puede reducir el rendimiento debido al tiempo necesario para calcular la paridad.
 - **Ventaja:** Equilibra costos, robustez y capacidad de recuperación.
 - Se puede usar un disco de repuesto que entra en acción si uno falla.

1.2.6. Tipos de RAID

- **RAID por Hardware:**
 - Utiliza un controlador RAID físico.
 - Es más eficiente y transparente para el sistema operativo.
- **RAID por Software:**
 - Administrado por el sistema operativo.
 - Puede ser modificado por el administrador, lo que representa un riesgo.
 - Requiere más recursos del sistema.

1.2.7. Ejercicio Opcional: Configuración de RAID1 para /var

1.2.8. Objetivo

El objetivo es proporcionar a /var un respaldo frente a fallos mediante la creación de un RAID1. Para ello, debemos montar un RAID1 y mover /var dentro de este.

1.2.9. Pasos a seguir

1. Creación de discos virtuales en la máquina virtual (MV)

- Se crean dos discos: `raid1` y `raid2`.
- Usamos `lsblk` para verificar la existencia de `sdb` y `sdc`.

2. Instalación de mdadm

- `sudo dnf provides mdadm` (para verificar qué paquete lo proporciona).
- `sudo dnf install mdadm` (para instalarlo).

3. Creación del RAID1

- `sudo mdadm -create /dev/md0 -level=1 -raid-devices=2 /dev/sdb /dev/sdc`
- Aparecerá un *warning* sobre la creación de metadatos, confirmamos con "sí".
- Para monitorear la sincronización del RAID: `watch -n 1 more /proc/mdstat`
- Luego, verificamos con `lsblk`.

4. Prueba de fallo de disco

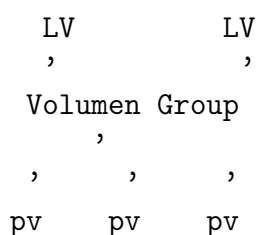
- Podemos simular la falla de un disco con: `echo 1 >/dev/sdb`
- Se observa que `md0` sigue funcionando al ser un *mirror*.

1.2.10. LVM (Logical Volume Manager)

1.2.11. Conceptos Clave

- **LV (Logical Volume)**
- **VG (Volume Group)**
- **PV (Physical Volume)**

La estructura básica de LVM es la siguiente:



Un **Logical Volume (LV)** puede expandirse tomando espacio libre de otros discos o estructuras.

1.2.12. Visualización de LVM

- Para ver los discos que son de tipo LVM, usamos `lsblk` en la columna de *type*.
- Para ver opciones de `pv`, `vg` y `lv`, usamos `tab-completion`.

1.2.13. Consideraciones sobre Volumen Groups

Si un **Volume Group** contiene un disco magnético, un SSD y un RAID, el acceso a los datos puede no ser homogéneo. Por ello, el volumen lógico debe montarse sobre el RAID.

1.2.14. Creación del LVM sobre el RAID

1. Convertir el RAID en un Physical Volume: `pvcreate /dev/md0`
2. Crear un Volume Group llamado `raid1`: `vgcreate raid1 /dev/md0`
3. Crear un Logical Volume de 10GB dentro del Volume Group: `lvcreate -L 10G -n rvar raid1`

1.2.15. Movimiento de `/var` al RAID

Después de crear el volumen lógico, debemos mover `/var` dentro del RAID:

- El nombre del volumen dentro del RAID es `/dev/raid1/rvar`.
- También puede aparecer como `/dev/mapper/raid1-rvar`, ya que son sinónimos creados mediante enlaces simbólicos.

1.2.16. Resolución de Problemas

1.2.17. Problema 1: Configurar el sistema de archivos en `rvar`

Actualmente, el volumen lógico `rvar` está vacío y sin un sistema de archivos. Para solucionarlo, seguimos estos pasos:

1. **Seleccionar un sistema de archivos:** Los sistemas de archivos recomendados son `ext4` y `xfs`, ya que son transaccionales y previenen la corrupción de datos en caso de fallos.
2. **Verificar los sistemas de archivos soportados:** Ejecutamos el siguiente comando para listar los sistemas disponibles en el kernel:

```
ls /lib/modules/$(uname -r)/kernel/fs
```

3. **Formatear el volumen lógico con `ext4`:**

```
mkfs.ext4 /dev/mapper/raid1-rvar
```

1.2.18. Problema 2: Montar y trasladar /var al nuevo volumen

El volumen lógico **rvar** debe ser montado en el sistema y debemos trasladar **/var** sin perder datos. Para ello:

1. **Montar el volumen lógico:** Primero, montamos **rvar** en un directorio temporal:

```
mount /dev/mapper/raid1-rvar /mnt/
```

Podemos verificar con:

```
mount
```

Al revisar **/mnt/**, aparecerá el directorio **lost+found**, indicando que el sistema de archivos está activo.

2. **Cambiar a modo mantenimiento:** Para evitar la pérdida de datos al copiar **/var**, debemos entrar en el **runlevel1**, también conocido como **modo mantenimiento**. Esto se hace con:

```
systemctl isolate runlevel1.target
```

Podemos confirmar el estado con:

```
systemctl status
```

3. **Copiar los datos de /var a /mnt/:** Copiamos todo el contenido de **/var** manteniendo atributos con:

```
cp -a /var/* /mnt/
```

4. **Crear un respaldo de /var:** Antes de reemplazar **/var**, hacemos una copia de seguridad por si algo falla:

```
mv /var /var_old
```

5. **Desmontar /mnt y montar el nuevo /var:**

```
umount /mnt
mkdir /var
mount /dev/mapper/raid1-rvar /var
```

Podemos verificar con:

```
df -h
```

6. **Hacer el montaje permanente en /etc/fstab:** Si reiniciamos ahora, la configuración se perdería. Para evitarlo, editamos el archivo `/etc/fstab` y agregamos la siguiente línea:

```
/dev/mapper/raid1-rvar    /var    ext4    defaults    0 0
```

7. **Probar la configuración antes de reiniciar:**

```
mount -a
systemctl daemon-reload
mount
```

Si todo está correcto, reiniciamos el sistema.

1.2.19. Interpretación de lsblk

El comando `lsblk` nos permite visualizar la estructura de almacenamiento del sistema. Es importante identificar:

- **Discos físicos y particiones:** Aparecen como `sda`, `sdb`, `sdc`, etc.
- **Volúmenes lógicos:** Se muestran bajo `/dev/mapper/`.
- **SR0:** Indica la unidad de CD-ROM.

1.3. Firewall + SSHD

Ya sabemos de otras Asignaturas que un **firewall** es un sistema que controla el tráfico de red, permitiendo o bloqueando ciertas conexiones. En Linux, el firewall más común es **iptables**.

El comando que tenemos que aprender a gestionar es `firewall-cmd`, tiene diversas opciones como `status`, `state`. Otra forma para ver si está arrancado es `systemctl status firewall`.

Otras opciones: `firewall-cmd --list-all`, en este comando debemos de ver los filtros que están definidos, en `services` podemos ver los servicios que están habilitados.

Con `sudo firewall-cmd --add-service=http`, cuando lo listo otra vez, se añade el puerto `http` en la opción `services`. Cuando se hace con `firewall-cmd`, se hace dinámica, por lo que si reiniciamos esto se pierde. Para que se quede debemos de ejecutar el comando `sudo fire firewall-cmd --runtime-to-permanent`. De forma alternativa, podemos ejecutar primero la opción `--permanent` y luego `--add-port=443/http`, pero si lo listamos no lo lleva a memoria, no hay ninguna opción que lo deje permanente y que lo lleve a memoria. Tenemos dos opciones o trabajar con servicios o llevarlo a memoria.

Si queremos conocer los servicios, podemos ejecutar `firewall-cmd --get-services`.

Los firewalls en Linux son muy eficientes debido a que en este SO se trabaja con servicios, por lo que se puede controlar el tráfico de forma muy precisa. Al implementarse a nivel de Kernel ayuda a que sea muy eficiente. *No se va a preguntar nada sobre iptables.*

Debemos de usar el programa `nmap` ya que nos dice que puertos están abiertos en un servidor. Para instalarlo, usamos `sudo dnf install nmap`. En el guión debemos de mirar la referencia número 31.

1.3.1. Ejercicio Opcional

Debemos de instalar un servicio HTTP, se recomienda Apache. Podemos escanear los 100 puertos mas usados con el comando `nmap -F localhost`. Para instalar Apache, usamos `sudo dnf install httpd`. Para arrancar el servicio, usamos `sudo systemctl start httpd`. Para comprobar que está arrancado, usamos `sudo systemctl status httpd`. Para que arranque en el inicio, usamos `sudo systemctl enable httpd`. Para comprobar que está escuchando en el puerto 80, usamos `sudo ss -tulnp | grep 80`.

Debemos de poder acceder desde el navegador de la máquina anfitrión.

1.3.2. SSH

Es un programa de terminal remoto, ya lo hemos usado en Asignaturas anteriormente. Antes se usaba Telnet, en este se especifica la dirección IP y el puerto, pero no era seguro, por lo que cualquier man in the middle podía interceptarlo y suplantar la identidad. Además, todas las respuestas que se producían eran en abierto, los cat,... En cambio, SSH es seguro, ya que se cifra la información. Este hace lo mismo que Telnet, pero de forma segura. Se recomienda hacerlo `ssh usuario@ip`. O bien sustituir la IP por el nombre del dominio. Para cerrar la conexión, usamos `exit`.

Ya hemos estudiado en otras Asignaturas el tema de criptografía en cuanto al cifrado simétrico y asimétrico. En especial, en Fundamentos de Redes. Por si acaso, vamos a hacer una pequeña introducción. Tenemos dos tipos de cifrado:

- **Cifrado simétrico:** Se usa una clave para cifrar y descifrar. El problema es que si alguien intercepta la clave, puede descifrar todo. Por ello, se usa poco.
- **Cifrado asimétrico:** Se usa una clave pública para cifrar y una privada para descifrar. La clave pública se puede compartir, pero la privada no. Se usa para firmar documentos, ya que si se cifra con la clave privada, solo se puede descifrar con la clave pública.

Debemos de destacar los conceptos de **autenticación** y **autorización**. La autenticación es el proceso de verificar la identidad de un usuario, mientras que la autorización es el proceso de verificar si un usuario tiene permiso para acceder a un recurso. Además, del uso de certificados digitales, que son un tipo de credencial que se usa para autenticar la identidad de un usuario o un servidor. Este se consigue a través de una entidad certificadora, usando una clave pública y privada (Big Brother).

Algoritmos :

- Llave simétrica: DES(tenemos dos llaves, una que es privada y otra que es pública, lo que se cifra con la privada solo se puede descifrar con la pública, por dentro se gestiona en base a números primos muy grandes).
- Llave asimétrica: RSA (Se guarda la privada, mientras que la pública se comparte, estando incluso compartida en la propia web, de manera que la persona que quiera compartir contigo usa la pública para codificar el texto y te lo manda codificado, de esta manera si hay un man in the middle lo ve cifrado, solo la persona con la llave privada puede ver su contenido).

Otro uso de la llave privada es la *firma*, de esta manera se garantiza la confidencialidad y la autenticación del mensaje debido a que es confidencial debido a que solo se puede descifrar con la llave pública, y autenticidad debido a que solo se puede cifrar con la llave privada, la cual solo conoce esa entidad.

Firma Digital: Usando Hash SHA256, se cifra con la llave privada y se envía el mensaje cifrado y el mensaje en abierto. La persona que recibe el mensaje cifrado con la llave privada, lo descifra y lo compara con el mensaje en abierto, si son iguales, se garantiza la autenticidad del mensaje. Ejemplo de ello puede ser el minado de monedas como es el Bitcoin. Además, otro ejemplo de ello es cuando descargamos algo y podemos verificar que es el del propio distribuidor usando el Hash que me proporciona. *Por ende, podemos considerar que un HASH + llave privada es el certificado digital.* Para asegurarnos de que la llave pública es de quien dice ser debemos de ver el certificado digital, como hemos mencionado anteriormente. Este proceso para cuando encontramos un certificado digital que es de confianza, ya que si no lo es, no podemos asegurar que la llave pública sea de quien dice ser. Para ver los certificados, vemos que en Firefox, accedemos a settings y buscamos *certificados*, podemos ver las entidades que nuestro browser reconoce.

Advertencia

El tema de certificados y demás al ser materia de Fundamentos de Redes, no se va a preguntar en el examen.

Nos conectamos a un ordenador remoto mediante ssh, de esta manera le decimos que nos queremos conectar, el te envía la llave pública, es decir, se va a su configuración, recupera la llave y te la envía. A continuación, le enviamos nuestras credenciales cifradas con la llave pública, y cuando le llega, las descifra con la llave privada y comprueba si son correctas. Si lo son, nos deja entrar. Si no lo son, nos dice que no podemos entrar. Además, de esta manera se prevee que un *man-in-the-middle* no pueda interceptar las credenciales.

Podemos meterle una entrada en el fichero de host, de esta manera cuando nos conectamos a un servidor, nos conectamos a una IP, pero si le metemos una entrada en el fichero de host, le decimos que esa IP es un nombre, de esta manera cuando nos conectamos a ese nombre, nos conectamos a esa IP. Para ello, debemos de editar el fichero `/etc/hosts` y añadir la IP y el nombre del servidor, así es más cómodo, para ello en Linux es `ssh usuario@nombre`.

Cuando ejecutamos el comando para conectarnos, nos imprime un mensaje con el HASH, y nos pregunta que si nos lo creemos, si en vez de una llave pública esta reconocida en nuestras llaves y ve que tenemos el certificado no nos preguntaría. En todos los equipos se crea el directorio ssh en home, dentro de este tenemos un archivo

que se llama `known_hosts`, el cual contiene las llaves públicas de los ordenadores que ya has confiado, de manera que no nos va a preguntar cuando nos conectemos de nuevo, si queremos que en la primera vez tampoco pregunta la copiamos y pegamos, entre otras opciones. *Nota: Por defecto se crea ese directorio cuando se conecta por ssh.*

Este proceso es muy costoso en términos de cómputo, por lo que se usa lo más mínimo posible. Por esto se usa un proceso de handshaking, de esta manera el equipo le manda una propuesta de llave simétrica y de esta manera ya se usa en adelante, ya que el uso de clave privada y pública es muy costoso.

¿Cómo puedo acceder sin contraseña? Para ello nos autenticamos con llave pública y llave privada. En el caso de que decidamos hacerlo así cuando le mandamos que soy 'nombre', recupera la llave de el directorio `.ssh` y le manda un mensaje de firmame esto con la clave privada de dicho usuario, y de esta manera se asegura de que efectivamente es el usuario que dice ser. Además, en el directorio `home` solo puede escribir el usuario y el `root`.

En la MV:

- `ssh-keygen`: crea la llave privada dentro de `home`, luego nos pregunta si queremos protegerla con contraseña, si no queremos, le damos a enter, *pero siempre debe de hacerse*. Luego si nos vamos al directorio vemos dos documentos la llave privada y la pública. Si queremos que no nos pregunte siempre la contraseña, usamos el comando `ssh-agent`, de esta manera el la escribe por ti, hasta que usamos el comando `exit`, en este caso se olvida.

Para usar la pública:

- Accedemos al directorio `/home/.ssh`
- usamos el comando `ssh-copy-id usuario@ip/hostname` (copia mi identidad, se puede hacer con `ctrl+c/v`), y luego nos hace una prueba para que le demostremos que de verdad somos el usuario al que nos queremos conectar, para ello le mandamos la contraseña.
- Luego comprobamos que podemos entrar sin contraseña.
- En la máquina donde nos hemos conectado, vemos un fichero llamado `id_rsa.pub`, y si lo listamos debe de ser el mismo que la clave pública de la máquina inicial, desde la que nos conectamos a esta.

Nota: El directorio `.ssh` se crea en el `home` cuando es el cliente, mientras que en el servidor se crea en el `/etc/ssh`, también podemos crearlo nosotros a priori.

Ahora podemos ejecutar un comando remoto, por ejemplo: `ssh usuario@ip comando`, de esta manera se ejecuta el comando en la máquina remota. En este punto comienza la automatización de configuración remota. *Ansible son scripts pero ejecutados de manera remota*. Además, podemos copiar archivos usando el comando `scp` (secure copy), de esta manera se copia el archivo de una máquina a otra. *Nota: `scp` es un comando de ssh*. El comando es `scp origen destino`, por ejemplo: `scp /home/usuario/archivo usuario@ip:/home/usuario/`, de esta manera se copia el archivo de la máquina local a la remota. Si queremos copiar un directorio, usamos el comando `scp -r origen destino`. Además podemos conectarnos mediante `sftp`

(secure file transfer protocol), de esta manera se conecta a la máquina remota y podemos copiar archivos de manera interactiva. *Nota: sftp es un comando de ssh.* Para conectarnos, usamos el comando `sftp usuario@ip`, y luego podemos usar los comandos `put` y `get` para copiar archivos de una máquina a otra.

1.3.3. Ejercicio Opcional

En el ejercicio debemos de dar acceso mediante ssh usando la llave pública y privada, debemos de modificar el puerto para conectarnos en vez del 22 para el ssh usar otro que sea mayor que el 1024, para ello podemos asegurarnos de que no estamos colisionando con un puerto conocido buscando en Internet.

1. Accedemos al contenido del demonio del ssh que se encuentra en `/etc/ssh`, una vez dentro nos pone el comando que debemos de ejecutar `semanage...` (aparece en el fichero de ssh de la máquina Rocky), para que se apliquen los cambios debemos de reiniciar el servicio, para ello usamos el comando `sudo systemctl restart sshd` o bien reiniciar la máquina.
2. Además debemos de modificar el firewall para permitir ese puerto, para ello usamos el comando `sudo firewall-cmd -add-port=puerto/tcp -permanent`, y luego reiniciamos el servicio con `sudo systemctl restart firewallld`.
3. Luego debemos de copiar la llave pública a la máquina remota, para ello usamos el comando `ssh-copy-id usuario@ip`, y luego nos pide la contraseña, una vez introducida, ya podemos conectarnos sin contraseña.
4. Para comprobar que todo está correcto, usamos el comando `ssh usuario@ip -p puerto`, de esta manera nos conectamos a la máquina remota.

Con el parámetro `-p` le especificamos el puerto al que nos queremos conectar, de esta manera nos conectamos al puerto que hemos especificado, ya que de manera predeterminada se conecta al puerto 22.

Parte II

Prácticas

Capítulo 2

Bloque 1

2.1. Ejercicio 1 Opcional

Cuando se hace referencia a la imagen X, se refiere a la imagen 2.X

El alumno/a debe ser capaz de presentar un MV con la configuración descrita en este apartado. La configuración debe ser permanente, es decir, en todo caso, tras reiniciar el equipo, la configuración será la esperada. Para validar la configuración de red, el alumno/a debe ser capaz de:

- Hacer ping desde el equipo anfitrión a la MV y viceversa.
- Hacer ping desde la MV a cualquier equipo accesible públicamente en Internet por FQHN o IP.
- Conectar por ssh desde el equipo anfitrión a la MV .

2.1.1. Solución

Una vez que hayamos instalado el SO que se nos pide correctamente. Debemos de realizar una serie de ajustes previos:

- Añadir nuestro usuario, para ello debemos de ejecutar lo siguientes comandos (iniciando como usuario root):
 - `sudo useradd nombre_de_usuario`
 - `sudo passwd nombre_de_usuario`
 - `sudo usermod -aG wheel nombre_de_usuario` para que pueda usar el comando sudo.
- Configurar la red NAT y una de tipo Host-Only, para ello en Herramientas en la VM debemos seleccionar la opción de Red y añadir una nueva interfaz de red de tipo Host-Only, y paso seguido configurar la red NAT (ver Figura 1 y 2)
- Comprobar que el servicio SSH está instalado, por defecto se suele instalar, para asegurarnos debemos de ejecutar el comando `sudo systemctl status`

`ssh`. En el caso de que no venga instalado debemos de ejecutar el comando `sudo dnf install -y openssh-server openssh-clients`¹ (Ver Figura 5).

- Cambiar la variable `PS1` como se nos pedía, para ello debemos de editar el fichero de `bashrc` y exportar la variable `PS1` con el valor que se nos pedía:
 - `PS1='\u@\h:\t:\w\$ '` (Ver Figura 5), donde:
 - `\u`: Nombre del usuario actual.
 - `\h`: Nombre del hostname (nombre del sistema).
 - `\t`: Hora actual en formato de 24 horas (HH:MM:SS).
 - `\w`: Directorio de trabajo actual.
 - `$`: Símbolo del prompt, que será `$` para un usuario normal y `#` para root.

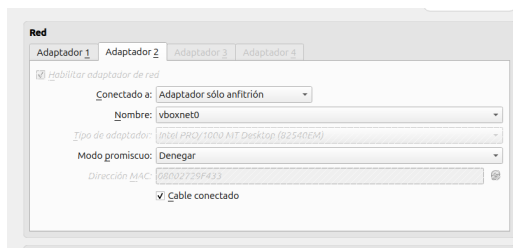


Figura 2.1: Configuración de la red NAT y Host-Only

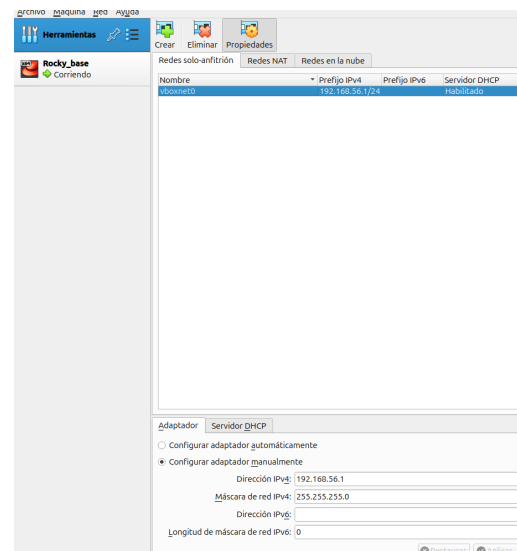


Figura 2.2: Configuración de Virtual-Box para la red de Host-Only

Además se nos pide que la Ip de Host Only sea estática, para ello vamos a asegurarnos usando la herramienta `nmtui`, en la que vamos a ver si es estática o no la ip. Como podemos ver en la siguiente imagen esta configurada como ip automática, que viene siendo lo mismo que dinámica por lo que debemos de cambiarlo a manual para poder configurar la ip estática. (Ver Figura 3 y 4). Para ver que efectivamente la ip cambió, podemos verlo en la Figura 5.

Una vez hayamos cambiado la ip estática, debemos de verificar que efectivamente se ha cambiado y para ello usamos el comando `ip a` y vemos que efectivamente se ha cambiado la ip a la que hemos asignado. (Ver Figura 7).

Llegado a este punto vamos a realizar un ping a la máquina anfitriona y viceversa, para ello usamos el comando `ping -c <número de pings> ip_de_la_maquina` y vemos que efectivamente hay conexión entre ambas máquinas. (Ver Figura 8, 9 y 10). Además, vemos que gracias al *NAT* podemos hacer ping a cualquier máquina accesible en internet². (Ver Figura 11).

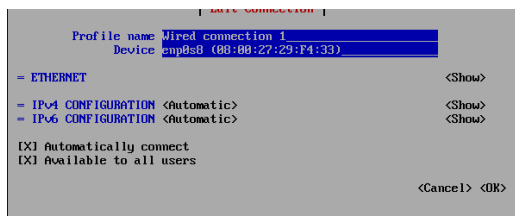


Figura 2.3: Con nmtui vemos que es dinámica

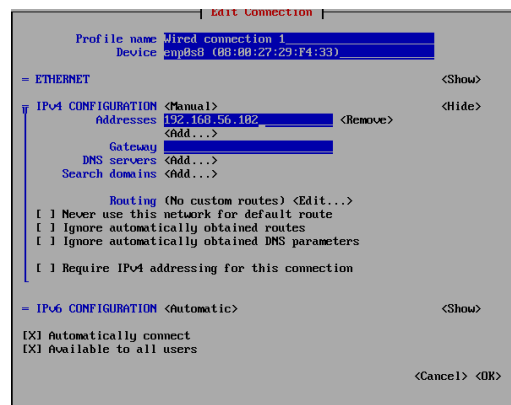


Figura 2.4: Cambiamos a manual y asignamos una ip estática válida

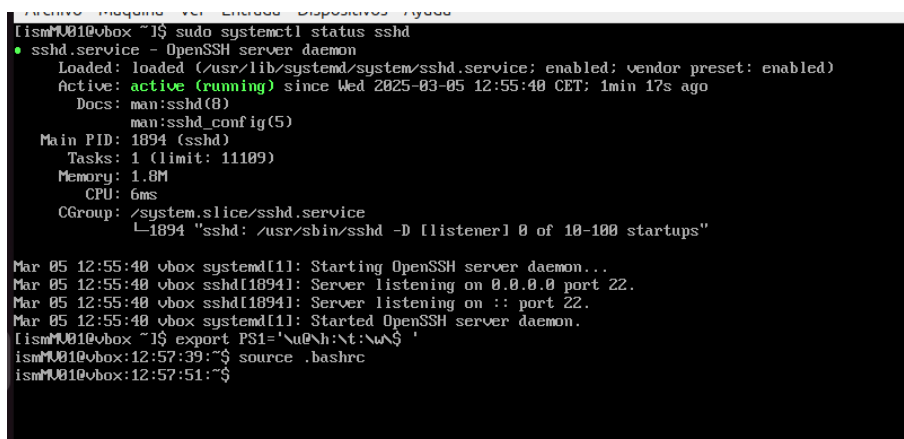


Figura 2.5: Sshd y variable PS1

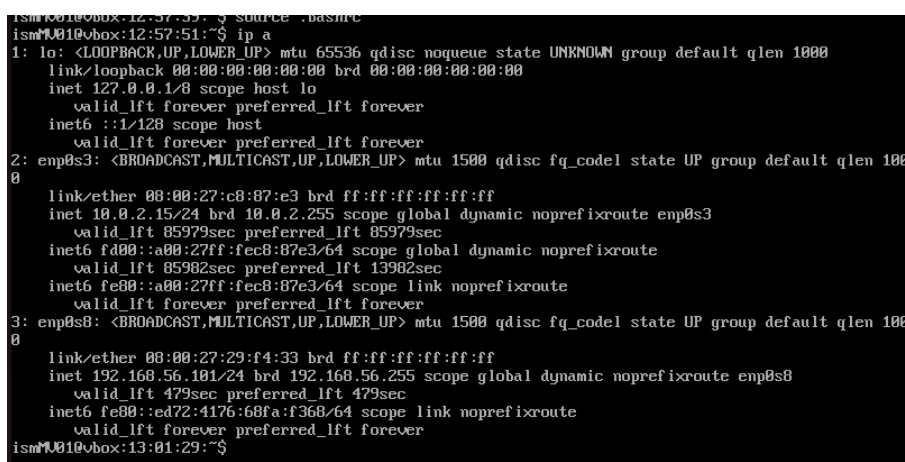


Figura 2.6: Resultado de ip a

```

[ism@010vbox ~]$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:c8:87:e3 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic noprefixroute enp0s3
        valid_lft 86142sec preferred_lft 86142sec
    inet6 fd00:a00:27ff:fec8:87e3/64 scope global dynamic noprefixroute
        valid_lft 86143sec preferred_lft 14143sec
    inet6 fe80::a00:27ff:fec8:87e3/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:29:f4:33 brd ff:ff:ff:ff:ff:ff
    inet 192.168.56.102/24 brd 192.168.56.255 scope global noprefixroute enp0s8
        valid_lft forever preferred_lft forever
    inet6 fe80::ed72:4176:68fa:f368/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
[ism@010vbox ~]$

```

Figura 2.7: Resultado de `ip a` para verificar el cambio de ip

```

> ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: eno1: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc fq_codel state DOWN group default qlen 1000
    link/ether 08:9e:25:bf:41:3e brd ff:ff:ff:ff:ff:ff
    altname enp250
3: wlp3s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
    link/ether 1c:ce:51:46:df:70 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.138/24 brd 192.168.1.255 scope global dynamic noprefixroute wlp3s8
        valid_lft 84512sec preferred_lft 84512sec
    inet6 2a0c:5a82:250b:5f00:7c22:bee5:a6ff:9f6b/64 scope global temporary dynamic
        valid_lft 602913sec preferred_lft 84257sec
    inet6 2a0c:5a82:250b:5f00:f4ae:9dc2:6471:93b/64 scope global mngtmpaddr noprefixroute
        valid_lft forever preferred_lft forever
    inet6 fe80::e202:77d0:65d9:9c71/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
4: vboxnet0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 0a:00:27:00:00:00 brd ff:ff:ff:ff:ff:ff
    inet 192.168.56.1/24 brd 192.168.56.255 scope global vboxnet0
        valid_lft forever preferred_lft forever
    inet6 fe80::800:27ff:fe00:0/64 scope link
        valid_lft forever preferred_lft forever

```

Figura 2.8: Resultado del comando de `ip a` en la máquina anfitriona para ver la ip

```

[ism@010vbox ~]$ ping -c 3 192.168.1.138
PING 192.168.1.138 (192.168.1.138) 56(84) bytes of data:
64 bytes from 192.168.1.138: icmp_seq=1 ttl=255 time=0.356 ms
64 bytes from 192.168.1.138: icmp_seq=2 ttl=255 time=0.362 ms
64 bytes from 192.168.1.138: icmp_seq=3 ttl=255 time=0.466 ms

--- 192.168.1.138 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2042ms
rtt min/avg/max/mdev = 0.356/0.394/0.466/0.050 ms
[ism@010vbox ~]$

```

Figura 2.9: Ping a la máquina anfitriona

```

> ping -c 3 192.168.56.102
PING 192.168.56.102 (192.168.56.102) 56(84) bytes of data:
64 bytes from 192.168.56.102: icmp_seq=1 ttl=64 time=0.505 ms
64 bytes from 192.168.56.102: icmp_seq=2 ttl=64 time=0.146 ms
64 bytes from 192.168.56.102: icmp_seq=3 ttl=64 time=0.336 ms

--- 192.168.56.102 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2071ms
rtt min/avg/max/mdev = 0.146/0.329/0.505/0.146 ms

```

Figura 2.10: Ping de la máquina anfitriona a la máquina virtual

```

[ism@010vbox ~]$ ping -c 3 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data:
64 bytes from 8.8.8.8: icmp_seq=1 ttl=255 time=77.9 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=255 time=101 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=255 time=122 ms

--- 8.8.8.8 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2083ms
rtt min/avg/max/mdev = 77.912/100.358/122.369/18.151 ms
[ism@010vbox ~]$

```

Figura 2.11: Ping a un servidor público (Google)

En cuanto al servicio ssh, debemos de ver el estado del servicio sshd con el comando `sudo systemctl status sshd` y vemos que esta corriendo. En este punto desde el anfitrión podemos introducir la línea de comando `ssh ismMV01@192.168.56.102` y vemos que efectivamente todo funciona correctamente. (Ver Figura 12 y 13).

```

$ ssh ismMV01@192.168.56.102
The authenticity of host '192.168.56.102 (192.168.56.102)' can't be established.
ED25519 key fingerprint is SHA256:v4KdPCXSCNa6e/95NN14fkq/MYFcvacXnh9pLQC35o.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.56.102' (ED25519) to the list of known hosts.
ismMV01@192.168.56.102's password:
Last login: Wed Mar  3 22:50:26 2025
[ismMV01@vbox-22:54:09 ~]$ ls
[ismMV01@vbox-22:54:12 ~]$ echo "el anfitrión ha entrado" >> anfitrión.txt
[ismMV01@vbox-22:54:44 ~]$ ls
anfitrión.txt
[ismMV01@vbox-22:54:46 ~]$ exit
logout
Connection to 192.168.56.102 closed.

```

Figura 2.12: Ssh en la máquina anfitrión y creación de un archivo en la MV

```

[ismMV01@vbox-22:51:47 ~]$ ls
anfitrión.txt
[ismMV01@vbox-22:55:10 ~]$ cat anfitrión.txt
el anfitrión ha entrado
[ismMV01@vbox-22:55:12 ~]$

```

Figura 2.13: Ver el contenido del archivo creado en la MV desde el anfitrión

2.2. Servidor con LVM + RAID

2.2.1. Aspectos clave de LVM

Para gestionar eficazmente el *Logical Volume Manager* (LVM), es fundamental comprender los siguientes componentes y conceptos:

Componentes de la arquitectura de almacenamiento

- **Physical Volume (PV):** Representa los dispositivos de almacenamiento físico, como discos duros o particiones, que se incorporan al sistema LVM.
- **Volume Group (VG):** Es una agrupación de uno o más PVs que forman un pool de almacenamiento, del cual se pueden asignar espacios para crear volúmenes lógicos.
- **Logical Volume (LV):** Son volúmenes virtuales creados dentro de un VG. Los LVs se utilizan como si fueran particiones de disco tradicionales, permitiendo la creación de sistemas de archivos o la asignación directa a aplicaciones.

Gestión de almacenamiento con diferentes características físicas

LVM ofrece flexibilidad para manejar dispositivos de almacenamiento con diversas características físicas:

- **HDD y SSD:** Se pueden combinar en un mismo VG, permitiendo equilibrar rendimiento y capacidad según las necesidades.
- **RAID:** LVM puede trabajar sobre dispositivos RAID, proporcionando una capa adicional de gestión y flexibilidad sobre la configuración RAID existente.

¹Incluimos clientes para añadir el servicio de cliente.

²Cabe destacar que durante el desarrollo de la actividad, surgían algunas problemas con Net-workManager, polkiy y DBus, pero se solucionaban al reiniciarlos o bien reinstalarlos

Etiquetado y correspondencia con los archivos de dispositivo

Cada componente en LVM tiene una nomenclatura específica y se asocia a archivos de dispositivo en el sistema:

- **Physical Volumes:** Corresponden a dispositivos físicos, como `/dev/sda1`, `/dev/sdb1`, etc.
- **Volume Groups:** Se nombran según la convención establecida por el administrador, por ejemplo, `vg_datos`.
- **Logical Volumes:** Se nombran dentro de su VG correspondiente, como `/dev/vg_datos/lv_backup`, donde `lv_backup` es el nombre del LV.

Comandos de LVM para la gestión de componentes

LVM proporciona una serie de comandos para administrar sus componentes:

- `pvccreate`: Inicializa un dispositivo físico como PV.
- `vgcreate`: Crea un VG a partir de uno o más PVs.
- `lvcreate`: Crea un LV dentro de un VG.
- `pvs`, `vgs`, `lvs`: Muestran información sobre PVs, VGs y LVs respectivamente.
- `pvremove`, `vgremove`, `lvremove`: Eliminan PVs, VGs y LVs respectivamente.

Estos comandos permiten una gestión eficiente y flexible del almacenamiento en sistemas que utilizan LVM.

2.2.2. Niveles de RAID: 0, 1 y 5

Redundant Array of Independent Disks (RAID) es una tecnología que permite combinar múltiples dispositivos de almacenamiento en una unidad lógica para mejorar el rendimiento, la redundancia o ambos. A continuación, se detallan los niveles de RAID 0, 1 y 5, sus ventajas, desventajas y su administración en sistemas Linux utilizando la herramienta de línea de comandos `mdadm`.

RAID 0

RAID 0, conocido como *striping*, distribuye los datos de manera equitativa entre dos o más discos sin información de paridad ni redundancia.

- **Ventajas:**
 - Mayor rendimiento en lectura y escritura debido a la distribución de datos entre los discos.
 - Uso completo de la capacidad de almacenamiento total, ya que no se reserva espacio para paridad o duplicación.
- **Desventajas:**
 - Ausencia de redundancia; la falla de un solo disco resulta en la pérdida total de los datos.

RAID 1

RAID 1, o *mirroring*, duplica los datos en dos o más discos, creando copias idénticas en cada uno.

■ Ventajas:

- Alta redundancia; los datos permanecen intactos incluso si uno de los discos falla.
- Mejora en la velocidad de lectura, ya que los datos pueden leerse desde cualquiera de los discos.

■ Desventajas:

- Capacidad de almacenamiento efectiva reducida al 50 % del total, ya que los datos se duplican.

RAID 5

RAID 5 combina rendimiento y redundancia distribuyendo los datos y la paridad entre tres o más discos.

■ Ventajas:

- Proporciona tolerancia a fallos; si un disco falla, los datos pueden recuperarse con la información de paridad.
- Mejor aprovechamiento del almacenamiento comparado con RAID 1, ya que solo se utiliza una fracción del espacio para la paridad.

■ Desventajas:

- Rendimiento de escritura inferior al de RAID 0 debido al cálculo de la paridad.
- En caso de falla de un disco, la reconstrucción puede ser lenta y afectar el rendimiento.

Administración de RAID en Linux con mdadm

La herramienta `mdadm` permite gestionar arreglos RAID en Linux. A continuación, se presentan comandos esenciales:

■ Crear un RAID 0 con dos discos:

```
1 mdadm --create --verbose /dev/md0 --level=0 --raid-devices  
=2 /dev/sdX /dev/sdY  
2
```

■ Crear un RAID 1 con dos discos:

```
1 mdadm --create --verbose /dev/md0 --level=1 --raid-devices  
=2 /dev/sdX /dev/sdY  
2
```

- Crear un RAID 5 con tres discos:

```
1      mdadm --create --verbose /dev/md0 --level=5 --raid-devices
      =3 /dev/sdX /dev/sdY /dev/sdZ
2
```

- Verificar el estado del RAID:

```
1      cat /proc/mdstat
2
```

- Detener un RAID:

```
1      mdadm --stop /dev/md0
2
```

2.2.3. Aspectos clave para la administración de servidores Linux

Para implementar eficazmente soluciones en servidores Linux, es esencial comprender y manejar los siguientes aspectos:

Modos de ejecución y modo de mantenimiento en un servidor Linux

Los sistemas Linux operan en diferentes niveles de ejecución o *runlevels*, que determinan los servicios y procesos que se ejecutan. Con la adopción de *systemd*, estos niveles se denominan *targets*. El modo de mantenimiento, conocido como *rescue.target* o *emergency.target*, es crucial para tareas de recuperación y administración del sistema. Para cambiar al modo de mantenimiento, se puede utilizar el siguiente comando:

```
1 sudo systemctl isolate rescue.target
```

Para volver al modo multiusuario estándar:

```
1 sudo systemctl isolate multi-user.target
```

Estructura estándar del sistema de archivos en Linux

La estructura de directorios en Linux sigue el estándar de jerarquía de sistemas de archivos (*Filesystem Hierarchy Standard - FHS*). Algunos directorios principales incluyen:

- **/**: Directorio raíz que contiene todos los demás directorios.
- **/bin**: Ejecutables esenciales para todos los usuarios.
- **/etc**: Archivos de configuración del sistema.
- **/home**: Directorios personales de los usuarios.
- **/var**: Datos variables como registros y colas de impresión.

Sistemas de archivos comunes en Linux

Linux soporta diversos sistemas de archivos. Algunos de los más comunes son:

- **ext4**: Sistema de archivos por defecto en muchas distribuciones, conocido por su estabilidad y rendimiento.
- **XFS**: Adecuado para manejar grandes volúmenes de datos y archivos de gran tamaño.
- **Btrfs**: Ofrece características avanzadas como instantáneas (*snapshots*) y compresión.

Montaje y desmontaje de volúmenes

El montaje de sistemas de archivos permite acceder a dispositivos de almacenamiento. Para montar un dispositivo:

```
1 sudo mount /dev/sdX1 /mnt/punto_de_montaje
```

Para desmontarlo:

```
1 sudo umount /mnt/punto_de_montaje
```

Las configuraciones de montaje persistentes se definen en el archivo `/etc/fstab`.

Comandos básicos para la gestión de archivos

La administración de archivos en Linux se realiza mediante comandos de línea.

Estos se deben de haber estudiado en asignaturas anteriores como Sistemas Operativos.

2.2.4. Ejercicio Opcional

Partiendo de un servidor básico configurado de acuerdo al apartado 2, el alumno/a deberá afrontar el caso práctico descrito a continuación:

Se desea instalar un servicio de gestión documental en el servidor. Se espera que este servicio precise de una cantidad espacio de almacenamiento creciente con el tiempo, pudiendo llegar a ser considerable.

Por otro lado, el contenido será crítico, por lo que se desea proporcionar algún mecanismo de respaldo ante fallos en el dispositivo de almacenamiento.

El alumno/a debe diseñar los cambios en el sistema de almacenamiento e implementarlo empleando prácticas adecuadas de administración que garanticen la conservación de la información en el sistema y procuren la máxima disponibilidad del servicio.

Solución

Para la resolución de este ejercicio vamos a seguir lo realizado en clase. (Ver apuntes de clase correspondientes a la sección de LVM+RAID).

En primer lugar creamos los discos, para ello entramos en la configuración de la máquina virtual y añadimos dos discos duros de 1GB cada uno. Una vez añadidos los discos duros, iniciamos la máquina virtual y ejecutamos el comando `lsblk` para ver los discos duros que tenemos disponibles. (Ver Figura 14).

```

[ism@010vbox-23:44:20 ~]$ lsblk
NAME        MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sda          8:0    0   20G  0 disk
├─sda1       8:1    0    1G  0 part /boot
├─sda2       8:2    0   19G  0 part
└─r1_vbox-root 253:0    0   17G  0 lvm  /
   └─r1_vbox-swap 253:1    0    2G  0 lvm  [SWAP]
sdb          8:16   0    1G  0 disk
sdc          8:32   0    1G  0 disk
sr0         11:0    1 1024M  0 rom

```

Figura 2.14: Resultado del comando `lsblk`

```

[ism@010vbox-23:49:46 ~]$ sudo mdadm --create /dev/md0 --level=1 --raid-devices=2 /dev/sdb /dev/sdc
mdadm: Note: this array has metadata at the start and
may not be suitable as a boot device.  If you plan to
store "/boot" on this device please ensure that
your boot-loader understands md/v1.x metadata, or use
--metadata=0.90
Continue creating array [y/N]? y
mdadm: Defaulting to version 1.2 metadata
[ 465.188620] md/raid1:md0: not clean -- starting background reconstruction
[ 465.188724] md/raid1:md0: active with 2 out of 2 mirrors
[ 465.188749] md0: detected capacity change from 0 to 2853856
mdadm: array /dev/md0 started.
[ism@010vbox-23:58:33 ~]$ [ 465.194578] md: rescue of RAID array md0
[ 478.678044] md: md0: rescue done.

```

Figura 2.15: Comando `mdadm`

A continuación, instalamos `mdadm` con el comando `sudo dnf install -y mdadm` y creamos un RAID 1 con los dos discos duros que hemos añadido. Para ello ejecutamos el comando `sudo mdadm -create -verbose /dev/md0 -level=1 -raid-devices=2 /dev/sdb /dev/sdc` y vemos que se ha creado correctamente. (Ver Figura 15).

Acto seguido debemos de crear un PV, VG y LV. Para ello ejecutamos los siguientes comandos (Ver Figura 16):

- `sudo pvcreate /dev/md0`
- `sudo vgcreate vg_datos /dev/md0`
- `sudo lvcreate -L 900M -n lv_datos vg_datos`

```

[ism@010vbox-23:53:45 ~]$ sudo pvcreate /dev/md0
Physical volume "/dev/md0" successfully created.
[ism@010vbox-23:53:59 ~]$ sudo vgcreate raid1 /dev/mapper/mcolog_md0_mcm_mqueue/
[ism@010vbox-23:53:59 ~]$ sudo vgcreate raid1 /dev/md0
Volume group "raid1" successfully created.
[ism@010vbox-23:54:18 ~]$ sudo lvcreate -L 10G -n rvar raid1
Volume group "raid1" has insufficient free space (255 extents): 2560 required.
[ism@010vbox-23:54:42 ~]$ sudo lvcreate -L 1G -n rvar raid1
Volume group "raid1" has insufficient free space (255 extents): 256 required.
[ism@010vbox-23:55:04 ~]$ sudo lvcreate -L 900M -n rvar raid1
Logical volume "rvar" created.
[ism@010vbox-23:55:13 ~]$

```

Figura 2.16: Resultado de crear los volúmenes

```

[ism@010vbox-23:57:29 ~]$ sudo mkfs.ext4 /dev/mapper/raid1-rvar
mkfs2fs 1.46.5 (30-Dec-2021)
Creating filesystem with 230400 4k blocks and 57600 inodes
Filesystem UUID: 22749b96-b1ce-44b3-acef-5444bb69e307
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376
Allocating group tables: done
Writing inode tables: done
Creating journal (4096 blocks): done
Writing superblocks and filesystem accounting information: done
[ism@010vbox-23:57:38 ~]$

```

Figura 2.17: Comando `mkfs`

Ahora debemos de formatear el volumen lógico en formato `ext4`. (Ver Figura 17).

A continuación, montamos (Ver Figura 18) y trasladamos la carpeta `var` al nuevo volumen lógico. Para ello ejecutamos los comando que figuran en los apuntes de clase.

```

[ism@010vbox-00:01:20 ~]$ sudo mount /dev/mapper/raid1-rvar /mnt/
[ 119.180453] EXT4-fs (dev2): mounted filesystem with ordered data mode. Quota mode: none.
[ism@010vbox-00:01:27 ~]$

```

Figura 2.18: Resultado del comando `mount`

```

[ism@010vbox-00:03:55 ~]$ df -h
Filesystem      Size  Used Avail Use% Mounted on
devtmpfs        869M   0  869M   0% /dev
tmpfs           888M   0  888M   0% /dev/shm
tmpfs           356M  5.0M  351M   2% /run
/dev/mapper/r1_vbox-root 17G  1.1G   16G   7% /
/dev/sda1       1814M 196M  819M  20% /boot
/dev/mapper/raid1-rvar 868M  24K  887M   1% /mnt
tmpfs           178M   0  178M   0% /run/user/1000

```

Figura 2.19: Comando `df -h`

Ejecutamos el comando `df -h` para ver que efectivamente se ha montado correctamente el volumen lógico. (Ver Figura 19).

Ejecutamos el `isolate` del `systemctl` para entrar en modo de mantenimiento y poder trasladar la carpeta `var` al nuevo volumen lógico. (Ver Figura 20).

Ahora debemos de trabajar con la serie de comandos que se especifican para poder mover la carpeta y que se haga de forma correcta. (Ver Figura 21).

Una vez trasladada la carpeta, debemos de hacer permanentes los cambios, para ello debemos de editar el fichero `/etc/fstab` y añadir la siguiente línea (Ver Figura 22):

```

1380.665560 audit: type=1131 audit(1741647077.136:317): pid=1 uid=0 auid=4294967295 ses=4294967295
subsystem=system_r init: t:0 msg=unit=firewalld.com="systemd" exe="/usr/lib/systemd/systemd"
hostname? addr=? terminal=? res=success
1380.663641 audit: type=1131 audit(1741647077.143:318): pid=1 uid=0 auid=4294967295 ses=4294967295
subsystem=system_r init: t:0 msg=unit=polkit.com="systemd" exe="/usr/lib/systemd/systemd" h
hostname? addr=? terminal=? res=success
1380.718221 audit: type=1131 audit(1741647077.190:319): pid=1 uid=0 auid=4294967295 ses=4294967295
subsystem=system_r init: t:0 msg=unit=NetworkManager.com="systemd" exe="/usr/lib/systemd/systemd"
hostname? addr=? terminal=? res=success
1380.818811 audit: type=1131 audit(1741647077.290:320): pid=1 uid=0 auid=4294967295 ses=4294967295
subsystem=system_r init: t:0 msg=unit=rsyslog.com="systemd" exe="/usr/lib/systemd/systemd"
hostname? addr=? terminal=? res=success
1380.825352 audit: type=1131 audit(1741647077.307:321): pid=1 uid=0 auid=4294967295 ses=4294967295
subsystem=system_r init: t:0 msg=unit=dbus-broker.com="systemd" exe="/usr/lib/systemd/systemd"
hostname? addr=? terminal=? res=success
1380.836381 audit: type=1131 audit(1741647077.316:322): prog-id=60 op=1000
You are in rescue mode. After logging in, type "journalctl -xb" to view
system logs, "systemctl reboot" to reboot, "systemctl default" or "exit"
to boot into default mode.
Give root password for maintenance
(or press Control-D to continue): 1310.861307 kauditd_printk_skb: 4 callbacks suppressed
1310.861311 audit: type=1131 audit(1741647087.341:327): pid=1 uid=0 auid=4294967295 ses=4294967295
subsystem=system_r init: t:0 msg=unit=NetworkManager-dispatcher.com="systemd" exe="/usr/lib/
systemd/systemd" hostname? addr=? terminal=? res=success

```

Figura 2.20: Resultado del comando `systemctl isolate`

```

root@vbox:~# cp -a /var/* /mnt/
root@vbox:~# mv /var/* /var.old
root@vbox:~# ls /mnt/
root@vbox:~# mkdir /var
root@vbox:~# mount /dev/mapper/raid1-rvar /var
root@vbox:~# EXT4-fs (dm-2): mounted filesystem with ordered data mode. Quota mode: none.
root@vbox:~# df -h

```

Filesystem	Size	Used	Avail	Usage	Mounted on
devtmpfs	863M	0	863M	0%	/dev
tmpfs	868M	0	868M	0%	/dev/shm
tmpfs	368M	5.0M	363M	2%	/run
dev/mapper/r1_vbox-root	176	1.1G	166	7%	/
dev/sda1	1014M	196M	813M	20%	/boot
tmpfs	176M	0	176M	0%	/run/user/1000
dev/mapper/raid1-rvar	868M	75M	733M	10%	/var

```

root@vbox:~#

```

Figura 2.21: Serie de operaciones para trasladar la carpeta `var`

```
1 /dev/mapper/raid1-rvar /var ext4 defaults 0 0
```

Recargamos la configuraciones con el daemon de `systemd` y vemos que aparece la parte que estabamos montando (Ver Figura 23).

```

/etc/fstab
# Created by anaconda on Wed Mar 5 11:25:01 2025
#
# Accessible filesystems, by reference, are maintained under '/dev/disk/'.
# See man pages fstab(8), findfs(8), mount(8) and/or blkid(8) for more info.
#
# After editing this file, run 'systemctl daemon-reload' to update systemd
# units generated from this file.

```

/dev/mapper/r1_vbox-root /	xfs	defaults	0 0	defaults	0 0
/dev/mapper/raid1-rvar /boot	xfs	defaults	1 0		
/dev/mapper/r1_vbox-swap none	swap	defaults	1 0		
/dev/mapper/raid1-rvar	/var	ext4	defaults	0 0	

Figura 2.22: Resultado de editar `/etc/fstab` como se indica

```

debugfs on /sys/kernel/debug type debugfs (rw,nosuid,nodev,noexec,relatime)
tracefs on /sys/kernel/tracing type tracefs (rw,nosuid,nodev,noexec,relatime)
fusectl on /sys/fs/fuse/connections type fusectl (rw,nosuid,nodev,noexec,relatime)
configfs on /sys/kernel/config type configfs (rw,nosuid,nodev,noexec,relatime)
/dev/sda1 on /boot type xfs (rw,relatime,seclabel,attr2,inode64,logbufs=8)
tmpfs on /run/user/1000 type tmpfs (rw,nosuid,nodev,relatime,seclabel,data=volatile)
/dev/mapper/raid1-rvar on /var type ext4 (rw,relatime,seclabel)

```

Figura 2.23: Resultado del comando `systemctl daemon-reload` y `mount`

Y como podemos comprobar en la Figura 24, hemos conseguido trasladar la carpeta `var` al nuevo volumen lógico y todo funciona correctamente.

```

Rocky Linux 9.0 (Blue Onyx)
Kernel 5.14.0-70.13.1.el9_0.x86_64 on an x86_64

vbox login: ismM001
Password:
Last login: Tue Mar 11 00:03:55 on tty1
[ismM001@vbox-00:26:41 ~]$ lsblk

```

NAME	MAJ:MIN	RM	SIZE	RO	TYPE	MOUNTPOINTS
sda	8:0	0	20G	0	disk	
└─sda1	8:1	0	1G	0	part	/boot
└─sda2	8:2	0	19G	0	part	
└─┬─r1_vbox-root	253:0	0	17G	0	lvm	/
└─┬─r1_vbox-swap	253:1	0	2G	0	lvm	[SWAP]
sdb	8:16	0	1G	0	disk	
└─md127	9:127	0	1022M	0	raid1	
└─┬─raid1-rvar	253:2	0	900M	0	lvm	/var
sdc	8:32	0	1G	0	disk	
└─md127	9:127	0	1022M	0	raid1	
└─┬─raid1-rvar	253:2	0	900M	0	lvm	/var
sr0	11:0	1	1024M	0	rom	

```

[ismM001@vbox-00:26:44 ~]$ _

```

Figura 2.24: Resultado final

Bibliografía

- [1] Ismael Sallami Moreno, **Estudiante del Doble Grado en Ingeniería Informática + ADE**, Universidad de Granada, 2025.