**INGENIERÍA DE SERVIDORES (2019-20)**
Grado en Ingeniería Informática
Universidad de Granada

# Práctica 1: Instalación de servidores (Ubuntu Server y centOS)

Alonso Bueno Herrero

11 de agosto de 2020

# Índice



# Referencias

[1] https://blog.inittab.org/administracion-sistemas/lvm-para-torpes-i/

# 1. Introducción y objetivos planteados

# 2. Sesión 1ª: Instalación de Ubuntu Server y centOS

## 2.1. Conceptos necesarios

## 2.2. Instalación de Ubuntu Server

## 2.3. Instalación de centOS.

Esta tarea es muy sencilla, y únicamente habrá que crear la máquina virtual, iniciarla y seguir los pasos básicos para la instalación por defecto.

## 2.4. Estado final de las unidades de almacenamiento tras la Sesión 1.

Lo vemos en la tabla 1.

| Punto de Montaje | /boot | SWAP | / (raíz) | /home |
|---|---|---|---|---|
| LV | arranque | swap | raiz | home |
| VG | Servidor | | | |
| PV | md0 | | | |
| Disco | sda | | | |

Cuadro 1: Estructuras de almacenamiento del servidor tras acabar la Sesión 1ª.

# 3. Sesión 2ª: Ampliación de la capacidad de un usuario en el servidor centOS.

En este caso, la idea es que, **una vez instalado el servidor de centOS en el disco por defecto** que nos crea VirtualBox, añadamos más capacidad al directorio de un usuario concreto.

Para estas operaciones habrá que tener en cuenta algunas consideraciones y delicadezas de centOS con respecto a Ubuntu, pero ya las iremos viendo.

## 3.1. Gestión *software* del almacenamiento en Linux: LVM

El *solftware* de gestión de las unidades de almacenamiento de Linux es el denominado **Gestor de Volúmenes Lógicos** o Logical Volume Manager (LVM).

Aquí presentaremos los conceptus estrictamente necesarios para desarrollar la práctica, pero puede encontrar una buena referencia para aclarar posibles dudas sobre LVM en (1)

## 3.2. Contextualización del escenario profesional

Tal y como se indica en el guión, el escenario profesional es el siguiente: en esta ocasión, en la empresa en la que le acaban de contratar tenían adquirido un servidor y su predecesor había realizado la instalación del S.O. CentOS, según le han comentado los compañeros, él solía hacer instalaciones por defecto y luego aplicar *scripts* de configuración. Sin más información, nuestro jefe nos informa que esa máquina va a alojar unos cursos con vídeos de alta calidad y relativamente largos. Por tanto, viendo la configuración del sistema, prevemos que `/var` necesitará más espacio, incluso es conveniente asignarle un LV exclusivamente. Para ello, incluiremos un nuevo disco y configuraremos LVM para que `/var` se monte en el nuevo VL que crearemos para él.

## 3.3. Requisitos para la realización de estas tareas

**Recuerda que al instalar centOS hay que crear el usuario con su clave y darle permisos de administrador. Además tienes que establecer contraseña de *root* para poder hacer el resto de tareas.**

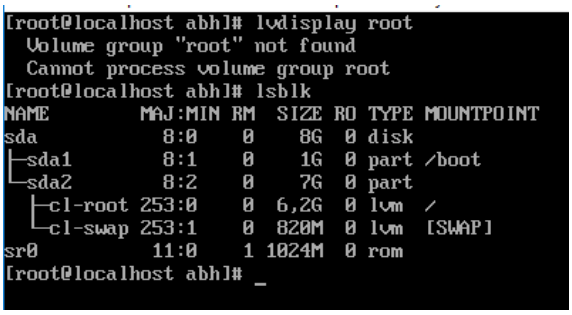


Figura 1: Situación de particionado del primer disco (`sda`) tras instalar centOS con las opciones por defecto.

Vamos a describir en los siguientes subapartados el proceso para realizar la tarea solicitada: ampliar la capacidad de `/var`.

### 3.4. Procedimiento práctico

Al instalar centOS con las opciones predeterminadas y *logearnos*, se nos proporciona el siguiente sistema de particiones sobre el disco único de 8 GB que le habíamos proporcionado al principio de esta práctica (ver figura 1):

Es decir, que el volumen físico `sda2` tiene 7GB y en él se ubica el VG[1] llamado `cl`, donde están los LV[2] `root` (de 6,2 GB) y `swap` (de 820 MB).

El procedimiento a seguir a partir de ahora es el siguiente:

1. Añadamos el nuevo disco y vamos a configurarlo para que /var tenga ese espacio extra (todo esto tiene detrás un proceso relativamente corto que iremos viendo).

   Por ahora, apagamos el servidor (comando `poweroff`), añadimos el nuevo disco a nuestra máquina virtual[3] y arrancamos. La situación tras crear el disco y hacer lsblk es la que se muestra en la figura 2.

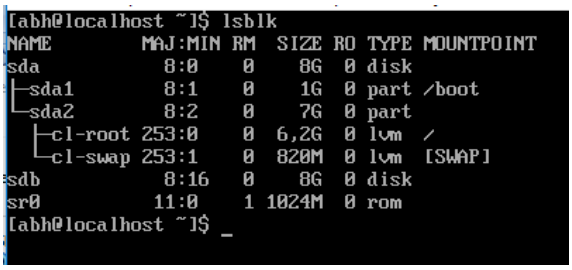


Figura 2: Situación tras añadir el nuevo disco, que el sistema lo ha llamado `sdb`.

2. Como vemos, aparece *sdb* con la etiqueta `disk` (penúltima columna, figura 2), porque es el nuevo disco que hemos añadido, pero vemos que no tiene ningún PV (volumen físico) asociado, con lo cual es como si ese espacio lo tuviéramos **inútil**, inválido para su uso por parte del sistema. Comprobémoslo con la(s) orden(es) LVM que nos informa(n) de los **volúmenes físicos existentes** en nuestro sistema: `pvdisplay` o su versión resumida, `pvs`, en la figura 3.

3. Y vemos que en efecto no habla nada del segundo disco creado. Lo primero que vamos a hacer (persiguiendo nuestro objetivo) es definir el volumen físico asociado a ese nuevo disco, para ello usaremos el comando `pvcreate` (ver figura 4).

[1] Grupo de Volúmenes
[2] Volúmenes Lógicos
[3] Como hemos apagado la máquina virtual antes de añadir el disco, decimos que hemos añadido el disco «en frío».

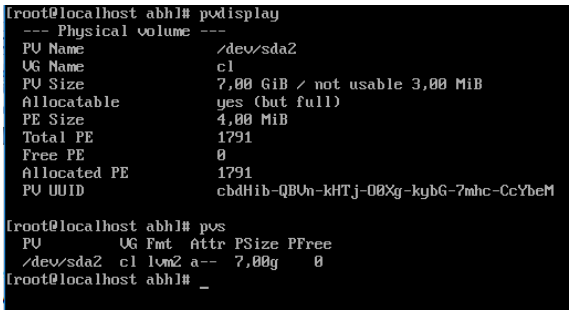


Figura 3: Resultado de la orden `pvdisplay`.

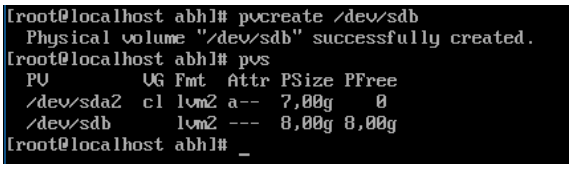


Figura 4: Creación del nuevo volumen físico (PV) `sdb` y comprobación de que se ha creado con la orden `pvs`.

4. Ahora hay que extender el Grupo de Volúmenes (VG) llamado `cl` con este nuevo volumen físico (recordemos la jerarquía de particiones y nomenclatura que establecía LVM) mediante la orden `vgextend`, y comprobamos que lo hemos hecho bien (ver figura 5):

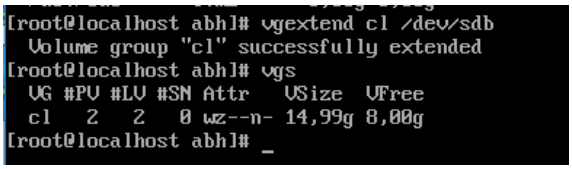


Figura 5: Extendiendo el grupo de volúmenes llamado `cl` y comprobando que lo hemos hecho bien con el comando `vgs`.

   Y vemos que ya el campo `#PV` aparece a `2`.

5. A continuación **creamos el nuevo volumen lógico** donde almacenar los datos que queremos, y que montaremos en /var. Lo haremos con la orden de la figura 6.

6. Y queda comprobado que se ha creado bien. Vamos ahora a **asignarle un sistema de ficheros** mediante el comando `mkfs` y con la opción `-t` para indicarle el tipo de Sistema de Archivos (SA) que le vamos a dar (en este caso, `ext4`). El comando y su resultado se muestra en la figura 7.

7. Ahora vamos a montar el volumen lógico `newvar` **para que esté accesible**[4] sobre un punto de montaje que vamos a crearnos y que se va a llamar `/mnt/newvar`:

```
mkdir /mnt/newvar                          # crear carpeta aux.
mount /dev/cl/newvar /mnt/newvar           # montar VL en 'aux'.
```

8. Vamos a comprobar que el montaje ha sido satisfactorio. Para ello volvemos a ejecutar la orden `mount` pero sin nada más (o bien con `mount | grep var`, para que marque dónde está `var`), y en la última línea debería aparecer nuestro dispositivo `newvar` montado sobre `/mnt/newvar`: lo vemos en la figura 8:

[4]Una analogía para recordar por qué montar un volumen: cuando conectamos una unidad de almacenamiento o un lector de DVD, el sistema tiene que crearle lo que se denomina un punto de montaje para poderlo utilizar. Este punto de montaje en casos de discos duros y *pendrive*, suele ser una carpeta que creamos manualmente en el sistema o nos lo crea él mismo automáticamente en una partición denominada `/media` en distribuciones basadas en Ubuntu. En nuestro caso, directamente las montamos sobre `/mnt/<dir_pto_montaje>`.

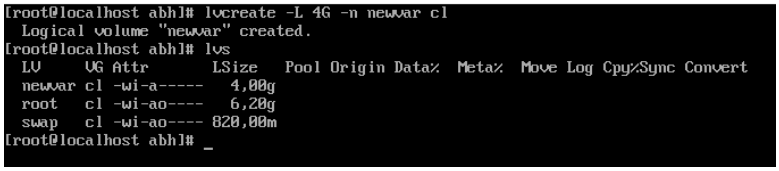


Figura 6: Creación del volumen lógico con la orden `lvcreate`.

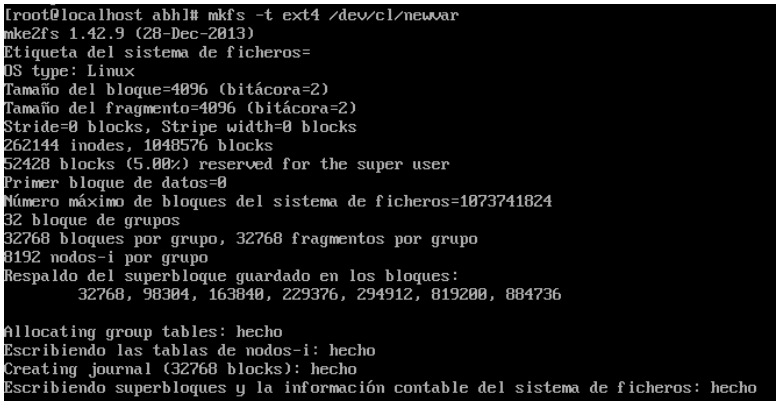


Figura 7: Asignando un sistema de archivos al nuevo volumen lógico.

9. Ahora habría que copiar el contenido de /`var`, que es lo que queremos ampliar, en `/mnt/newvar`. Pero para evitar que otros usuarios hagan operaciones de lectura/escritura mientras se hace esto, vamos a hacerlo de manera atómica, usando el *modo aislado (isolate)*. Para ello, ejecutamos la siguiente orden y nos autenticamos con la orden:

```
systemctl isolate runlevel1.target
```

   Nos saldrá esto, tras meter la clave de root (ver figura 9):

10. Ahora vamos a lo que queríamos hacer: copiar de `/var` a `/mnt/newvar`. Lo haremos con el comando `cp` y con la opción `-a` (para corregir errores de contexto de SELinux) y seleccionando `/var/.` (el detalle es el punto tras el `/var`, que indica que copie todo lo que haya, de manera recursiva incluyendo los ficheros ocultos). El resultado se puede ver en la figura 10

    Y lo comparamos con lo que hay en /var ejecutando:

```
ls -lhaZ /var
```

    y vemos que hay exactamente lo mismo, con lo cual verificamos que lo hemos hecho bien.

11. Ahora falta decirle al sistema lo más importante: que **cada vez que arranque, monte el volumen lógico llamado `newvar` en `/var`, para que dicho cliente pueda seguir accediendo a /var de manera transparente a dónde estén sus archivos**.

    Este es un paso delicado donde hay que **modificar un archivo del sistema, el** `/etc/fstab`, que da información en el arranque del sistema sobre montaje de dispositivos y particiones. Lo haremos con extremo cuidado, usando el editor `vi` y en modo *root*.

    Lo que haremos será abrir con `vi` el fichero anterior en modo escritura, situar el cursor en la *línea siguiente a la última línea escrita*, y escribir una nueva con el mismo contenido que el de la figura 11.

    Y el resultado de esta modificación sobre el fichero `/etc/fstab` es el de la figura 12:

12. Ahora vamos a desmontar cosas para evitar errores:

    *a*) desmontar `/mnt/newvar`

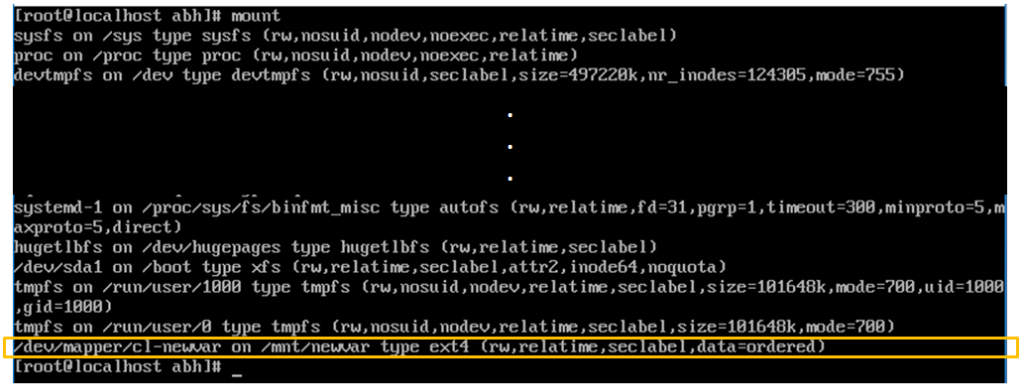


Figura 8: Fragmento (inicio y fin) de la orden `mount` tras montar `newvar` sobre el directorio auxiliar `/mnt/newvar` recién creado.

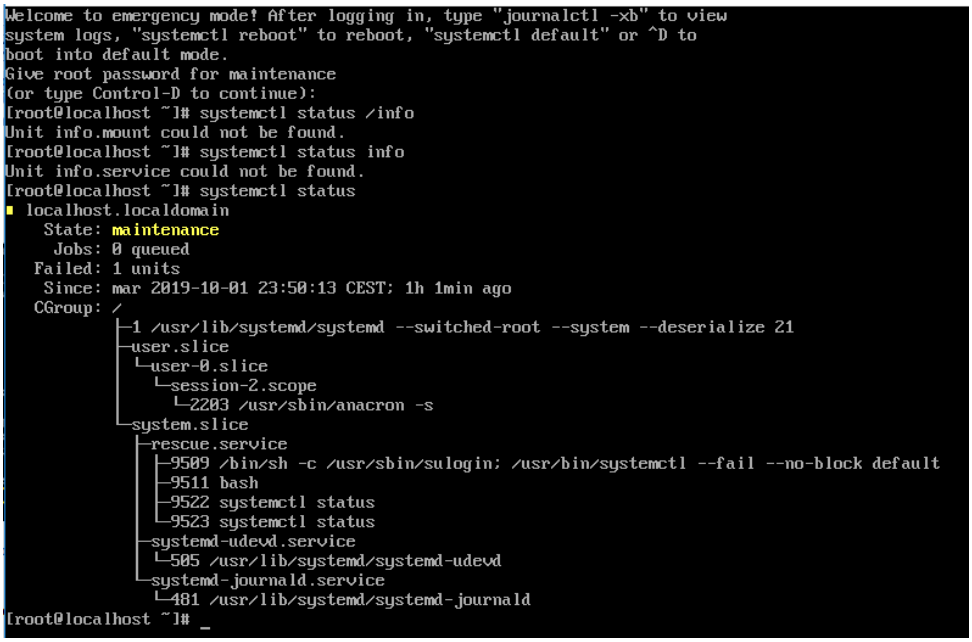


Figura 9: Aspecto inicial del modo de mantenimiento.

*b*) y es posible que haga falta: desmontar `/dev/cl/newvar`.

13. Para finalizar, nos queda mover `/var` a un directorio auxiliar (que llamaremos `/varold`, por ejemplo) para evitar posibles errores. Ahora vamos a crear un nuevo `/var` mediante `mkdir /var`, y le asignaremos un contexto a esa nueva carpeta con el comando que se muestra en la figura 13.

14. Ya está todo listo, sólo falta reiniciar la máquina y comprobar con `ls -lhaZ` que tanto /var como /varold tienen lo mismo.

### 3.5. Estado de las estructuras de almacenamiento tras la Sesión 2.

Lo vemos en la tabla 2. Se partía de la instalación **por defecto** de centOS (Sesión 1ª) y ahora en esta sesión añadíamos un nuevo disco para ampliar el espacio de la carpeta /var de forma *manual*.


Figura 10: Paso importante: copiando todo el contenido de /var a /mnt/newvar y comprobación de que lo hemos hecho bien con la orden `ls` sobre /mnt/newvar.

```
/dev/cl/newvar<tabulador>/var<tabulador>ext4<tabulador>defaults<tabulador>0<tabulador>0
```

Figura 11: Nueva línea del fichero `/etc/fstab`.

# 4. Sesión 3ª: Ampliación cifrada y redundante de la capacidad para un usuario en centOS

Este caso es una versión "mejorada" de la Sesión 2, pero hay que realizarla partiendo desde 0, es decir, tras haber instalado centOS únicamente.

> **¡IMPORTANTE!** Esta sesión se realizará sobre una máquina virtual sobre la que NO se haya implementado lo realizado en la Sesión 2.

## 4.1. Contextualización del escenario provisional

Este apartado ha sido extraído del guión de prácticas oficial de la asignatura.

Tras ver el éxito de los vídeos alojados en el servidor configurado en la práctica anterior, un amigo de su cliente quiere proceder del mismo modo pero va a necesitar alojar información sensible así que le pide explícitamente que cifre la información y que ésta esté siempre disponible. Por tanto, la decisión que toma es configurar un RAID1 por software y cifrar el VL en el que /var estará alojado.

## 4.2. ¿Qué necesitamos?

Aparte de lo mencionado antes sobre el Sistema Operativo y su estado, vamos a tener que crear dos discos extras, que serán los que conformen el RAID, uno de ellos tendrá el contenido exacto del otro (política de RAID1).

## 4.3. Pasos completos para la realización práctica de esta sesión

Como es preceptivo, presentamos ahora los pasos a seguir necesarios para completar satisfactoriamente esta sesión práctica:

```
[root@localhost abh]# cat /etc/fstab

#
# /etc/fstab
# Created by anaconda on Tue Oct  1 23:24:42 2019
#
# Accessible filesystems, by reference, are maintained under '/dev/disk'
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info
#
/dev/mapper/cl-root     /                       xfs     defaults        0 0
UUID=9e409233-f558-43e3-8c43-eff0db6bc0c8 /boot                   xfs     defaults        0 0
/dev/mapper/cl-swap     swap                    swap    defaults        0 0
/dev/cl/newvar  /var    ext4    defaults        0       0

[root@localhost abh]#
```

Figura 12: Contenido actualizado del fichero /etc/fstab.

```
[root@localhost ~]# restorecon -Rv /var
restorecon reset /var context unconfined_u:object_r:default_t:s0->unconfined_u:object_r:var_t:s0
[root@localhost ~]# _
```

Figura 13: Restaurar el contexto del nuevo directorio `/var`, donde montaremos nuestro nuevo disco `/dev/cl/newvar`.

1. Partimos inicialmente del estado original de centOS (puede estar la red configurada, no hay problema con eso) de la Figure 14.

2. Entonces, vamos a apagar el sistema, añadimos los dos discos y reiniciamos. Lo que nos muestra ahora lsblk es lo siguiente (ver Figure 15).

3. Ahora vamos a crear directamente el dispositivo ***RAID-1*** a partir de los dos discos *sdb* y *sdc* mediante un comando que se llama `mdadm` (*multimedia-admin*). Para instalarlo, ejecutamos:

   ```
   yum install mdadm
   ```

   Es posible que al ejecutar este comando nos **dé un error** como el mostrado en la Figure 16. Este fallo es por un error de conexión a la red. Lo comprobamos con ipaddr, y vemos que la interfaz enp0s3 (la que tenemos que mirar en nuestro caso) no tiene IP asignada. Hay que levantar la red mediante el comando:

   ```
   ifup enp0s3  # levantar la red
   ```

   Ahora ya podremos instalar el programa con el comando anterior. Durante la instalación, «decimos que sí» a todo lo que nos pregunte hasta que acabe. Ahora ya podemos crear el RAID-1. Para crearlo correctamente y con las opciones que queramos habría que mirar el manual, pero aquí ponemos directamente las opciones (indicadas por el profesor), lo vemos en la Figure 17.

   Y a modo de comprobación, usamos el comando `lsblk`, el cual nos muestra si hemos creado correctamente el RAID-1 (ver Figura 18).

4. **Creamos el volumen físico**, que podemos llamar `newvar` y que servirá de base para el grupo de volúmenes del RAID. Lo creamos con la orden:

```
[root@localhost abh]# lsblk
NAME        MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda           8:0    0    4G  0 disk
├─sda1        8:1    0    1G  0 part /boot
└─sda2        8:2    0    3G  0 part
  ├─cl-root 253:0    0  2,6G  0 lvm  /
  └─cl-swap 253:1    0  412M  0 lvm  [SWAP]
sr0          11:0    1 1024M  0 rom
[root@localhost abh]#
```

Figura 14: Estado inicial antes de crear el RAID-1.

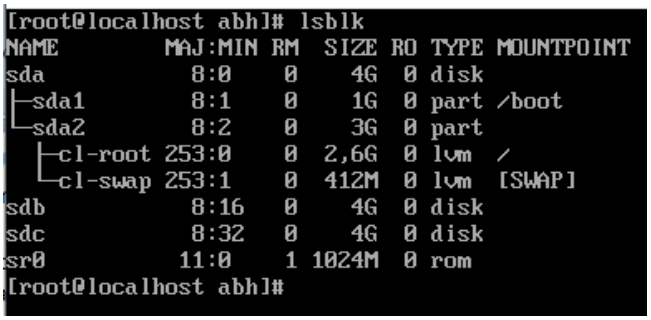


Figura 15: Comando `lsblk` tras crear dos discos vírgenes.

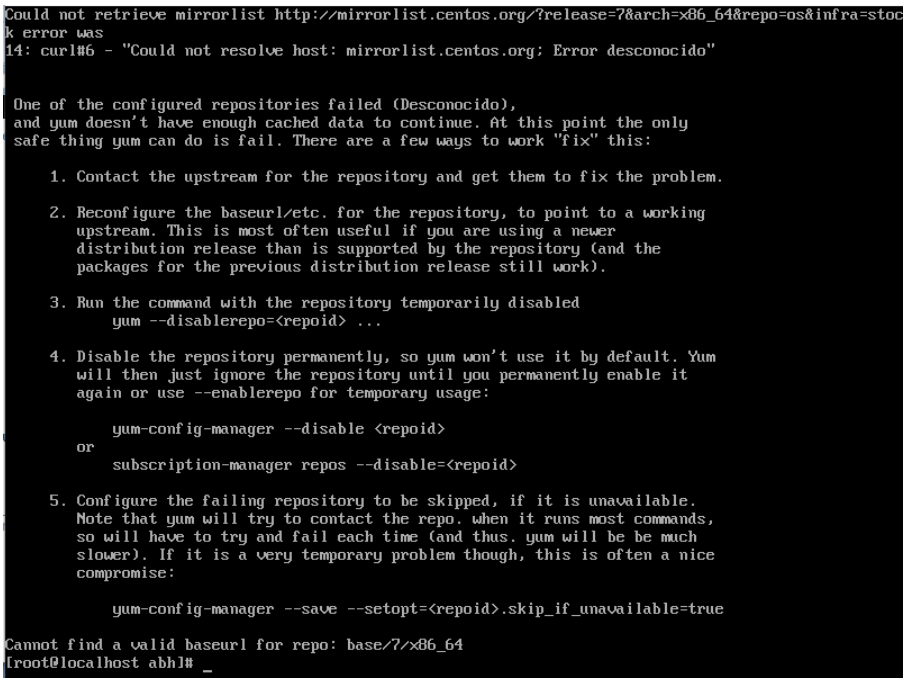


Figura 16: Error al instalar `mdadm`.

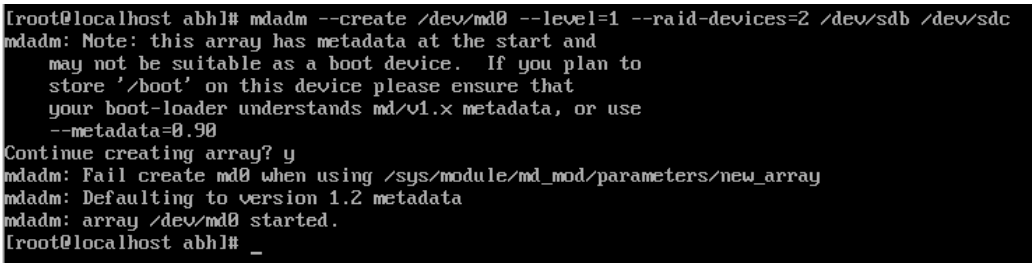


Figura 17: Creando el RAID-1 a partir de los dos nuevos discos, `sdb` y `sdc`.

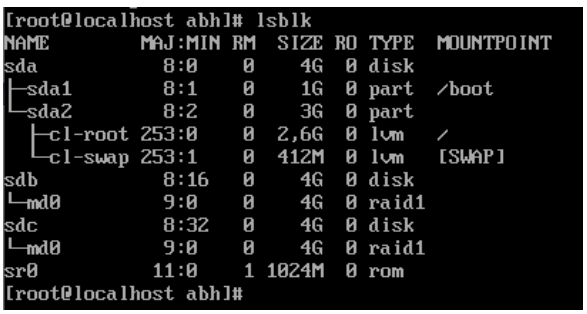


Figura 18: Ejecución de `lsblk` tras crear el RAID-1.

| Punto de Montaje | /boot | / (raíz) | SWAP | /home |
|---|---|---|---|---|
| LV | boot | root | swap | newvar |
| VG | – | cl | | |
| PV | – | sda | sda | sdb |
| Disco | sda | sda | sda | sdb |

Cuadro 2: Estructuras de almacenamiento del servidor tras acabar la Sesión 2ª.

```
[root@localhost abhl]# pvs
  PV         VG      Fmt  Attr PSize PFree
  /dev/md0   pmraid1 lvm2 a--  3,99g 2,99g
  /dev/sda2  cl      lvm2 a--  3,00g    0
[root@localhost abhl]# vgs
  VG      #PV #LV #SN Attr   VSize VFree
  cl        1   2   0 wz--n- 3,00g    0
  pmraid1   1   1   0 wz--n- 3,99g 2,99g
[root@localhost abhl]# lvs
  LV     VG      Attr       LSize   Pool Origin Data%  Meta%  Move Log Cpy%Sync Convert
  root   cl      -wi-ao----   2,59g
  swap   cl      -wi-ao---- 412,00m
  newvar pmraid1 -wi-a-----   1,00g
[root@localhost abhl]# _
```

Figura 19: Estado tras crear el volumen lógico newvar (y lo que lo soporta).

```
pvcreate /dev/md0
```

y lo comprobamos con las opciones habituales correctas.

Ahora **creamos el grupo de volúmenes** con

```
vgcreate pmraid1 /dev/md0
```

Después **creamos el volumen lógico** (que se duplicará mediante la técnica RAID) de 1 G de tamaño, con:

```
lvcreate -L 1G -n newvar pmraid1
```

Veamos cómo va todo con pvs, vgs y lvs. Lo vemos en la figura 19.

5. En este punto, recordemos cuál era nuestro **objetivo**: **crear una nueva unidad lógica redundante y encriptada**. Solo nos falta la **encriptación**. Esto lo haremos con el comando `cryptsetup` y algunas opciones del mismo. Lo primero, vamos a instalarlo con la orden:

```
yum install cryptsetup
```

Y no debería de haber errores. Ahora **entraremos en modo aislado** con

```
systemctl isolate runlevel1.target
```

y ahí ejecutamos los comandos indicados en la figura 20, en el mismo orden en que aparecen.

¿Qué ha pasado? Que hemos encriptado satisfactoriamente el volumen *newvar*, pero la zona encriptada dentro de ese volumen no está abierto, y entonces `lsblk` no lo reconoce. Abrámoslo, asignémosle un nombre que nos ayude a identificarlo (`pmraid1-newvar_crypt`), y después `lsblk` ya lo conocerá. Lo el comando necesario para esta tarea se indica en la figura 21, así como la comprobación de su resultado satisfactorio con `lsblk`.

6. Ahora quedan tareas que ya conocemos (salvo una que veremos después), en primer lugar, vamos a **asignar un sistema de archivos** a ese volumen encriptado (y **no** al *newvar* a secas) con el comando:

```
mkfs -t ext4 /dev/mapper/pmraid1-newvar_crypt
```


Figura 20: Resultado de la encriptación.

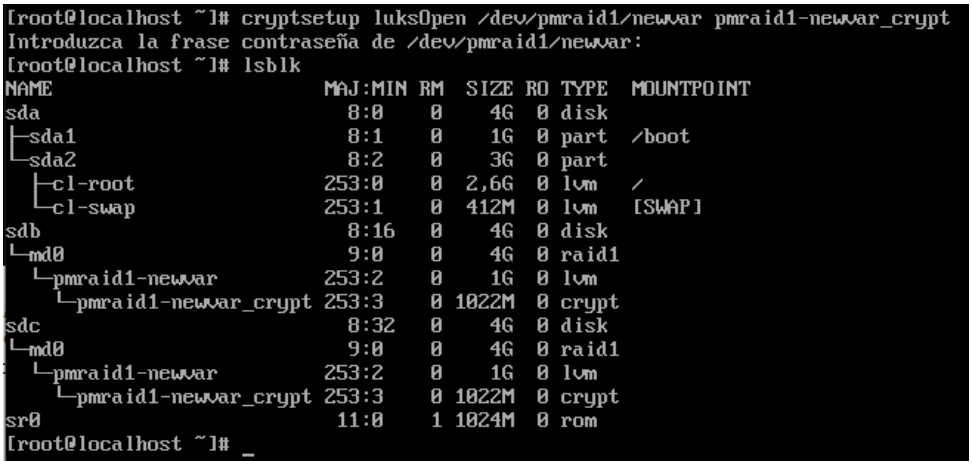


Figura 21: Resultado de abrir la zona encriptada del volumen. Mostramos el resultado con `lsblk`, donde ya aparece ese volumen encriptado con el nombre que le hemos puesto en el comando.


Figura 22: Operaciones de montaje del dispositivo cifrado para copiarle los archivos en un punto de montaje temporal. Todos los pasos se van comprobando.


Figura 23: Modificar el fichero /etc/crypttab.

Ahora lo montamos en un directorio auxiliar de */mnt*, por ejemplo */mnt/varcifr*, y le copiamos todos los archivos. Veamos estas operaciones ejecutadas en la figura 22.

7. Ahora nos quedan unos pasos que son conceptualmente sencillos pero requieren de mucho cuidado:

   *a*) En primer lugar, hay que usar un fichero llamado */etc/crypttab* para poder abrir siempre el volumen encriptado y que el sistema lo sepa al acceder a dicho fichero. Para ello usaremos el nombre (sin camino) del volumen encriptado (en nuestro caso se llama `pmraid1-newvar_crypt`), el UUID del volumen lógico al que pertenece, etc. Vamos a ello:

   *Paso 1.-* En la figura 23 vemos una serie de órdenes para obtener la línea asociada al volumen cifrado y poder pegarla en el */etc/crypttab* mediante la redirección de salida.

   *Paso 2.-* Ahora hay que modificar el fichero `/etc/crypttab` para que quede como en la figura 24.

   *b*) Modificamos el fichero `/etc/fstab` para que quede como en la figura 25.

8. Ya está prácticamente todo hecho. Sólo faltan dos pasos de seguridad:

```
mkdir /varold        # crear carpeta auxiliar para guardar copia de /var
mv /var /varold      # liberar espacio de /var, por seguridad
umount /mnt/varcifr # todo lo que se monta se desmonta
```


Figura 24: Comprobando el contenido de `/etc/crypttab`.

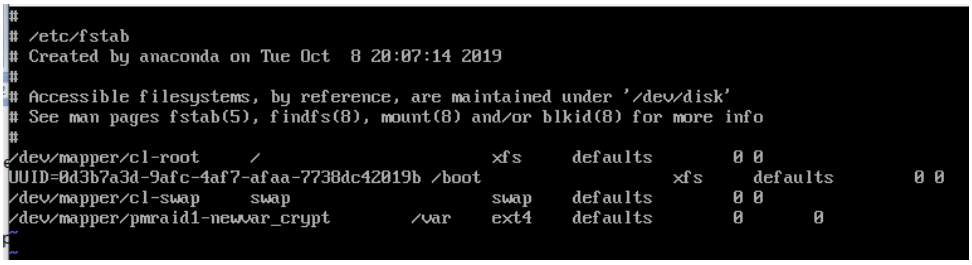


Figura 25: Modificando el fichero `/etc/fstab`.

```
mkdir /var           # nuevo directorio /var (donde accede el user)
restorecon -Rv /var # para restaurar el contexto del nuevo /var
```

9. Ahora podemos reiniciar el servidor, hacer

```
mount | grep crypt
```

y ver que se ha montado satisfactoriamente el volumen encriptado.

## 4.4. Estado final de las estructuras de almacenamiento tras la Sesión 3ª.

Lo vemos en la tabla 3.

| Punto de Montaje | `/boot` | SWAP | `/` (raíz) | `/home` |
|---|---|---|---|---|
| LV | `arranque` | `swap` | `raiz` | `home` |
| VG | `Servidor` | | | |
| PV | `md0` | | | |
| Disco | `sda` | | | |

Cuadro 3: Estructuras de almacenamiento del servidor tras acabar la Sesión 3ª.

# APÉNDICES

## A. Configuración de la red de MVs para las prácticas

Nos ayudaremos del software VirtualBox para poder conectar en red las dos máquinas virtuales y al anfitrión.

Los **pasos** son:

1. Para cada MV, vamos al menú de «Configuración», a la pestaña lateral de «Red»