

Prácticas de Ingeniería de Servidores

Alonso Bueno Herrero

Curso 2019-20. Última modificación: 2021-03-25

Contents

1	Instalación de servidores: <i>Ubuntu Server</i> y <i>centOS</i>	5
1.1	Objetivos de la práctica	5
1.2	Sesión 1: Instalación de Ubuntu Server y CentOS	5
1.3	Sesión 2: Ampliación de capacidad para un usuario en el servidor <i>centOS</i>	8
1.4	Sesión 3: Ampliación cifrada y redundante de la capacidad para un usuario.	15
2	Literature	17
3	Methods	19
4	Applications	21
4.1	Example one	21
4.2	Example two	21
5	Final Words	23

Chapter 1

Instalación de servidores: *Ubuntu Server y centOS*

1.1 Objetivos de la práctica

Objetivos mínimos:

1. Familiarizarse con distintos Sistemas Operativos (SOs) usados en servidores.
2. Conocer alternativas comerciales para tener un servidor.
3. Configurar unidades de disco RAID y LVM.
4. Configurar una red local de máquinas virtuales.

1.2 Sesión 1: Instalación de Ubuntu Server y centOS

1.2.1 Conceptos básicos

Repasamos los siguientes conceptos básicos que conviene conocer antes de iniciar las tareas de esta sesión:

1. Servidor, como un sistema informático que recibe peticiones y da respuestas a las mismas.
2. Tecnología de servidores, distinguiendo:

- a. **Hosting** dedicado: es una configuración de alojamiento en el que se dedica un servidor a una sola organización o para un solo propósito, como una página web. No escalable.
 - b. **VPS** (Virtual Private Server), donde varias MV comparten una serie de recursos hardware (Disco Duro, RAM, CPU, ...).
 - c. **Serverless** (servicios en la nube), las aplicaciones montan ahí sus recursos y todas las tareas de administración de sistema están abstraídas para el usuario hasta el punto que él no tiene que preocuparse de las mismas. Es el modelo más actual por su alta capacidad de escalabilidad.
3. Una tecnología de servidores basada en contenedores se caracteriza porque cada uno de estos contenedores se interpreta como una MV, sólo que cada una de esas MV son aplicaciones sobre el SO anfitrión, en tanto en cuanto:
- Los contenedores están (pueden estar) interconectados entre sí,
 - Puedo añadir nuevos contenedores dinámicamente e ir conectándolos a otros que estén en activo (arrancados, funcionando).

Un ejemplo es un servidor web donde un contenedor C1 contiene el motor web, otro C2 contiene la base de datos y el C3, que contendría el intérprete de peticiones web.

4. La tecnología *RAID* surge como solución a la problemática del precio del almacenamiento. Las siglas se refieren a **Redundant Array of Independent Disks** (Almacén Redundante de Discos Independientes). Existen diversos tipos de tecnologías RAID, las más importantes son RAID-0 y RAID-1:
- a. **RAID-0**. Un RAID-0 distribuye los datos equitativamente entre dos o más discos (usualmente se ocupa el mismo espacio en dos o más discos) sin información de paridad que proporcione redundancia. Es importante señalar que el RAID 0 no es redundante. Un RAID 0 puede ser creado con discos de diferentes tamaños, pero el espacio de almacenamiento añadido al conjunto estará limitado por el tamaño del disco más pequeño (por ejemplo, si se hace un conjunto dividido con un disco de 450 GB y otro de 100 GB, el tamaño del conjunto resultante será sólo de 200 GB, ya que cada disco aporta 100 GB).
 - b. Un RAID 1 crea una copia exacta (o espejo) de un conjunto de datos en dos o más discos. Esto resulta útil cuando queremos tener más seguridad desaprovechando capacidad, ya que si perdemos un disco, tenemos el otro con la misma información. Un conjunto RAID 1 sólo puede ser tan grande como el más pequeño de sus discos. Un RAID 1 clásico consiste en dos discos en espejo, lo que incrementa exponencialmente la fiabilidad respecto a un solo disco; es decir, que para que el conjunto falle es necesario que lo hagan todos sus discos.

- c. **Ejercicio** Busca información sobre los tipos de RAID siguientes: RAID-10, RAID-5 y RAID-6.

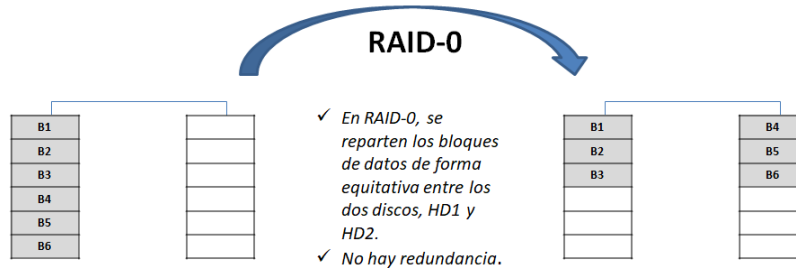


Figure 1.1: Configuración de RAID-0. Elaboración propia.

5. Sistema de archivos

- a. Sistema de archivos *ext2*: principales características:

- No admite Journaling
- Adecuado para tarjetas SD y unidades USB, ya que tiene un alto rendimiento y escritura baja (ya que el registro en diario no está disponible). USB y almacenamiento SD están limitados con ciclos de escritura por lo tanto su mejor ajuste para ellos.
- Límites: Tamaño de archivo individual de 16 GB a 2 TB.
- Tamaño del sistema de archivos de 2TB a 32TB.

- b. Sistema de archivos *ext4*

- Soporta Journaling
- Muchas de las nuevas características introducidas. Extents, Compatibilidad con versiones anteriores, Pre-asignación persistente, Asignación diferida, Número ilimitado de subdirectorios, Suma de comprobación del diario, Comprobación FS más rápida, Encriptación transparente.
- Límites: Tamaño de archivo individual de 16GB a 16TB. Tamaño del sistema de archivos hasta 1EB.
- No es necesario actualizar el sistema de archivos. Debido a la compatibilidad hacia atrás, *ext2*, *ext3* se puede montar directamente como *ext4*.

1.2.2 Proceso de instalación de Ubuntu Server 16.04

1.3 Sesión 2: Ampliación de capacidad para 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 a la “carpeta” de ese 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...

1.3.1 Contextualización del escenario profesional

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.

1.3.2 Pasos previos: instalación de *centOS*

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 todo lo siguiente.

1.3.3 Procedimiento

Pasos a realizar:

1. Una vez hecha la instalación por defecto, y al iniciar sesión en el servidor, se nos proporciona el siguiente *sistema de particiones* sobre el único disco que tenemos, de 8 GB, que le habíamos proporcionado (ver figura 1.2).

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

2. Añadamos el nuevo disco y configurémoslo para que `/var` tenga ese espacio extra (todo esto tiene detrás un proceso relativamente corto que iremos viendo).

1.3. SESIÓN 2: AMPLIACIÓN DE CAPACIDAD PARA UN USUARIO EN EL SERVIDOR CENTOS.9

```
[root@localhost abh]# lvsdisplay root
Volume group "root" not found
Cannot process volume group root
[root@localhost abh]# lsblk
NAME        MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda          8:0    0   8G  0 disk
├─sda1       8:1    0    1G  0 part /boot
├─sda2       8:2    0    7G  0 part
│   └─c1-root 253:0    0  6.2G  0 lvm /
│       └─c1-swap 253:1    0  820M  0 lvm [SWAP]
└─sr0        11:0    1 1024M  0 rom
[root@localhost abh]# _
```

Figure 1.2: Situación de particionado del primer disco tras instalar centOS con las opciones por defecto.

Por ahora, apagamos el ordenador (comando `poweroff`), añadimos el nuevo disco a nuestra Máquina Virtual y arrancamos. La situación tras añadir el nuevo disco “físico” y hacer `lsblk` es se ve en la figura 1.3.

```
[abh@localhost ~]$ lsblk
NAME        MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda          8:0    0   8G  0 disk
├─sda1       8:1    0    1G  0 part /boot
├─sda2       8:2    0    7G  0 part
│   └─c1-root 253:0    0  6.2G  0 lvm /
│       └─c1-swap 253:1    0  820M  0 lvm [SWAP]
sdb          8:16    0   8G  0 disk
sr0        11:0    1 1024M  0 rom
[abh@localhost ~]$ _
```

Figure 1.3: Situación tras aña

3. Como vemos, aparece el disco `sdb`, que 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 al completo. Comprobémoslo con la(s) orden(es) de LVM que nos informa(n) de los volúmenes físicos existentes en nuestro sistema: `pvdisplay` o de forma resumida, `pvs` (ver figura 1.4).

Y vemos que en efecto no refleja nada sobre el segundo disco creado.

4. Lo primero que vamos a hacer es definir el volumen físico asociado a ese nuevo disco, para ello usaremos el comando `pvccreate` (ver figura 1.5).
5. Ahora hay que extender el Grupo de Volúmenes (VG) llamado `c1` 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 en la figura 1.6.

```

[root@localhost abhl# podisplay
--- Physical volume ---
PU Name      /dev/sda2
VG Name      cl
PU Size      7,00 GiB / not usable 3,00 MiB
Allocatable  yes (but full)
PE Size      4,00 MiB
Total PE     1791
Free PE      0
Allocated PE 1791
PU UUID      cbdHib-QBUn-kHTj-00Xg-kybG-7mhc-CcYbeM

[root@localhost abhl# pvs
PU          VG Fmt Attr PSize PFree
/dev/sda2   cl  lvm2 a--  7,00g    0
[root@localhost abhl# _

```

Figure 1.4: Mostrando los volúmenes físicos actuales.

```

[root@localhost abhl# pvcreate /dev/sdb
Physical volume "/dev/sdb" successfully created.
[root@localhost abhl# pvs
PU          VG Fmt Attr PSize PFree
/dev/sda2   cl  lvm2 a--  7,00g    0
/dev/sdb     cl  lvm2 ---  8,00g  8,00g
[root@localhost abhl# _

```

Figure 1.5: Creación del nuevo volumen físico 'sdb' y comprobación de que se ha creado con la orden 'pvs'.

```

[root@localhost abhl# vgextend cl /dev/sdb
Volume group "cl" successfully extended
[root@localhost abhl# vgs
VG #PU #LV #SN Attr   VSize VFree
cl   2   2   0 wz--n- 14,99g 8,00g
[root@localhost abhl# _

```

Figure 1.6: Extendiendo el grupo de volúmenes 'cl' y comprobando que lo hemos hecho bien con el comando 'vgs'.

1.3. SESIÓN 2: AMPLIACIÓN DE CAPACIDAD PARA UN USUARIO EN EL SERVIDOR CENTOS.11

Y vemos que ya el campo #PV aparece con el valor 2.

6. Ahora vamos a **crear el nuevo volumen lógico** donde almacenar los datos que queremos, y que montaremos en `/var` (ver figura 1.7):

```
[root@localhost abhl# lvcreate -L 4G -n newvar cl
Logical volume "newvar" created.
[root@localhost abhl# lvs
LU      VG Attr      LSize   Pool Origin Data%  Meta%  Move Log Cpy%Sync Convert
newvar  cl  -wi-a-----  4,00g
root    cl  -wi-ao-----  6,20g
swap    cl  -wi-ao-----  820,00m
[root@localhost abhl# _
```

Figure 1.7: Creación del volumen lógico con la orden ‘lvcreate’.

Y queda comprobado que se ha creado bien.

7. Ahora vamos a asignarle un **sistema de ficheros** mediante el comando `mkfs` con la opción `-t` para indicarle el tipo de Sistema de Archivos que le vamos a dar (en este caso, *ext4*), tal y como se muestra en la figura 1.8.

```
[root@localhost abhl# mkfs -t ext4 /dev/cl/newvar
mke2fs 1.42.9 (28-Dec-2013)
Etiqueta del sistema de ficheros=
OS type: Linux
Tamaño del bloque=4096 (bitácora=2)
Tamaño del fragmento=4096 (bitácora=2)
Stride=0 blocks, Stripe width=0 blocks
262144 inodes, 1048576 blocks
52428 blocks (5.00%) reserved for the super user
Primer bloque de datos=0
Número máximo de bloques del sistema de ficheros=1073741824
32 bloque de grupos
32768 bloques por grupo, 32768 fragmentos por grupo
8192 nodos-i por grupo
Respaldo del superbloque guardado en los bloques:
      32768, 98304, 163840, 229376, 294912, 819200, 884736
Allocating group tables: hecho
Escribiendo las tablas de nodos-i: hecho
Creating journal (32768 blocks): hecho
Escribiendo superbloques y la información contable del sistema de ficheros: hecho
```

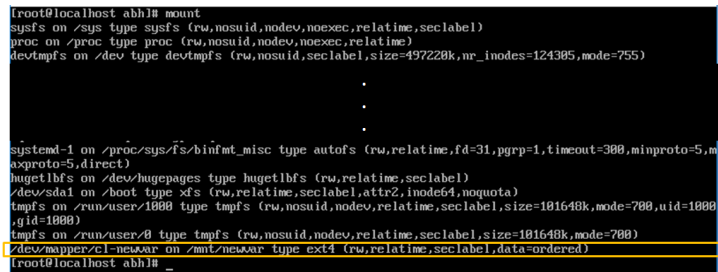
Figure 1.8: Asignando un sistema de archivos al nuevo volumen lógico.

8. A continuación, tenemos que **montar el volumen lógico newvar para que esté accesible**¹ sobre un punto de montaje que vamos a crearnos y que se va a llamar `/mnt/newvar`:

¹Una 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's, 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>`.

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

9. Vamos a comprobar que el montaje ha sido satisfactorio. Para ello volvemos a ejecutar `mount` pero sin ninguna opción (o bien con `$ mount | grep var`, para que marque en color donde está `/var`), y en la última línea debería aparecer nuestro dispositivo `newvar` montado sobre `/mnt/newvar`, tal y como se muestra en la figura 1.9.



```
[root@localhost abh]# mount
sysfs on /sys type sysfs (rw,nosuid,nodev,noexec,relatime,seclabel)
proc on /proc type proc (rw,nosuid,nodev,noexec,relatime)
devtmpfs on /dev type devtmpfs (rw,nosuid,seclabel,size=497228k,nr_inodes=124385,mode=755)
.
.
.
systemd-1 on /proc/sys/fs/binfmt_misc type autofs (rw,relatime,fd=31,prp=1,timeout=300,minproto=5,maxproto=5,direct)
hugetlbfs on /dev/hugepages type hugetlbfs (rw,relatime,seclabel)
/dev/sda1 on /boot type xfs (rw,relatime,seclabel,attr2,inode64,noquota)
tmpfs on /run/user/1000 type tmpfs (rw,nosuid,nodev,relatime,seclabel,size=101648k,mode=700,uid=1000,gid=1000)
tmpfs on /run/user/0 type tmpfs (rw,nosuid,nodev,relatime,seclabel,size=101648k,mode=700)
2dev/mapper/zel-newvar on /mnt/newvar type ext4 (rw,relatime,seclabel,data=ordered)
[root@localhost abh]#
```

Figure 1.9: Fragmento (inicio y fin) de la orden ‘`mount`’ tras montar `newvar` sobre el directorio auxiliar ‘`/mnt/newvar`’ recién creado.

10. 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 `r/w` mientras se hace esto, vamos a hacerlo de manera atómica, usando el modo **AISLADO**. Para ello, ejecutamos el comando `systemctl isolate runlevel1.target`, y tras autenticarnos (con usuario `root`) debe de aparecer lo mismo que en la figura 1.10.
11. Ahora vamos a lo que queríamos hacer: copiar de `/var` a `/mnt/newvar`. Lo haremos (ver figura 1.11) con `cp`, obviamente, 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 y los ficheros ocultos incluidos*). Y lo comparamos con lo que hay en `/var` ejecutando: `> ls -lhaZ /var` y vemos que hay exactamente lo mismo, lo cual indica que hemos procedido correctamente.
12. Ahora falta decirle al sistema lo más importante: que cada vez que arranque, monte el volumen lógico `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

1.3. SESIÓN 2: AMPLIACIÓN DE CAPACIDAD PARA UN USUARIO EN EL SERVIDOR CENTOS.13

```
Welcome to emergency mode! After logging in, type "journalctl -xb" to view
system logs, "systemctl reboot" to reboot, "systemctl default" or "D" to
boot into default mode.
Give root password for maintenance
(or type Control-D to continue):
[root@localhost ~]# systemctl status /info
Unit info.mount could not be found.
[root@localhost ~]# systemctl status info
Unit info.service could not be found.
[root@localhost ~]# systemctl status
[root@localhost localdomain]#
  State: maintenance
    Jobs: 0 queued
  Failed: 1 units
  Since: mar 2019-10-01 23:50:13 CEST; 1h 1min ago
  CGroup: /
          └─1 /usr/lib/systemd/systemd --switched-root --system --deserialize 21
              └─user.slice
                  └─user-0.slice
                      └─session-2.scope
                          └─2203 /usr/sbin/anacron -s
                              └─system.slice
                                  └─rescue.service
                                      └─9509 /bin/sh -c /usr/sbin/sulogin: /usr/bin/systemctl --fail --no-block default
                                          └─9511 bash
                                              └─9522 systemctl status
                                                  └─9523 systemctl status
                                                      └─systemd-udevd.service
                                                          └─505 /usr/lib/systemd/systemd-udevd
                                                              └─systemd-journald.service
                                                                  └─401 /usr/lib/systemd/systemd-journald
[root@localhost ~]# _
```

Figure 1.10: Aspecto inicial del modo de mantenimiento (modo aislado).

```
[root@localhost ~]# cp -a /var/. /mnt/newvar/
[root@localhost ~]# ls -lhaZ /mnt/newvar/
drwxr-xr-x. root root system_u:object_r:var_t:s0 .
drwxr-xr-x. root root system_u:object_r:mnt_t:s0 ..
drwxr-xr-x. root root system_u:object_r:var_t:s0 adm
drwxr-xr-x. root root system_u:object_r:var_t:s0 cache
drwxr-xr-x. root root system_u:object_r:kdump_crash_t:s0 crash
drwxr-xr-x. root root system_u:object_r:system_db_t:s0 db
drwxr-xr-x. root root system_u:object_r:var_t:s0 empty
drwxr-xr-x. root root system_u:object_r:games_data_t:s0 games
drwxr-xr-x. root root system_u:object_r:var_t:s0 gopher
drwxr-xr-x. root root system_u:object_r:var_t:s0 kerberos
drwxr-xr-x. root root system_u:object_r:var_lib_t:s0 lib
drwxr-xr-x. root root system_u:object_r:var_t:s0 local
lrwxrwxrwx. root root system_u:object_r:var_lock_t:s0 lock -> ../run/lock
drwxr-xr-x. root root system_u:object_r:var_log_t:s0 log
drwxr-xr-x. root root system_u:object_r:unlabeled_t:s0 lost+found
lrwxrwxrwx. root root system_u:object_r:mail_spool_t:s0 mail -> spool/mail
drwxr-xr-x. root root system_u:object_r:var_t:s0 nis
drwxr-xr-x. root root system_u:object_r:var_t:s0 opt
drwxr-xr-x. root root system_u:object_r:var_t:s0 preserve
lrwxrwxrwx. root root system_u:object_r:var_run_t:s0 run -> ../run
drwxr-xr-x. root root system_u:object_r:var_spool_t:s0 spool
drwxrwxrwt. root root system_u:object_r:tmp_t:s0 tmp
-rw-r--r--. root root system_u:object_r:etc_runtime_t:s0 .updated
drwxr-xr-x. root root system_u:object_r:var_up_t:s0 yp
[root@localhost ~]# _
```

Figure 1.11: Paso importante: copiando todo el contenido de ‘/var’ a ‘/mnt/newvar’ y comprobación con ‘ls’.

sistema, el `/etc/fstab`, que da información sobre el montaje de dispositivos y particiones al arranque del Sistema Operativo. Lo haremos con extremo cuidado, usando el editor `vi` y en modo **root**.

- i. abrir el fichero `/etc/fstab` con `vi` usando la orden:

```
> vi /etc/fstab      # estando en modo root
```

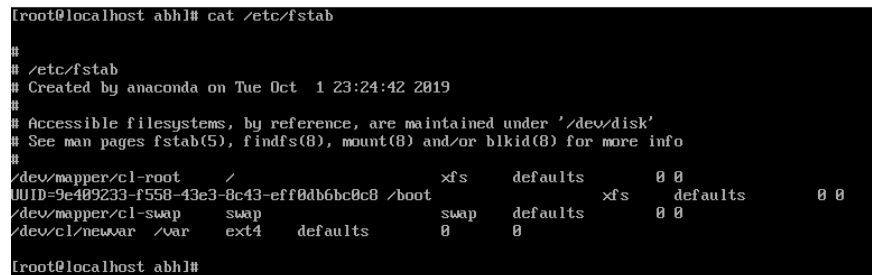
- ii. pulsamos la tecla `i` del teclado para acceder al modo de escritura,
- iii. vamos al final del fichero, insertamos una nueva línea (nos colocamos **al final** de la última línea y pulsamos *Enter*),
- iv. insertamos en la nueva línea el siguiente contenido:

```
/dev/cl/newvarTABULADOR/varTABULADORext4TABULADORdefaultsTABULADOR0TABULADOR0
```

donde la palabra `TABULADOR` indica que en ese lugar pulsemos la tecla de tabulador una sola vez.

- v. tras repasar lo escrito y asegurarnos de que lo hemos escrito correctamente, pulsamos la tecla `Esc` para salir del modo de escritura y tecleamos en la línea de órdenes de la parte inferior de la pantalla que ofrece `vi` el literal `:wq` para guardar los cambios en el fichero (`w`) y salir (`q`).

El aspecto final de `/etc/fstab` será el de la figura 1.12.



```
[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
```

Figure 1.12: Estado final de `/etc/fstab` tras las modificaciones oportunas.

13. Ahora vamos a desmontar cosas para evitar errores:

- i. desmontar `/mnt/newvar`,
- ii. y es posible que haga falta: desmontar `/dev/cl/newvar`

1.4. SESIÓN 3: AMPLIACIÓN CIFRADA Y REDUNDANTE DE LA CAPACIDAD PARA UN USUARIO.15

Y ahora, mover `/var` a un directorio auxiliar llamado `/varold` (o el nombre que queramos) para evitar posibles errores. Ahora vamos a crear un nuevo `/var` mediante la orden `mkdir /var`, y le asignaremos un contexto a esa nueva carpeta con el comando de la figura 1.13.

14. Ahora ya está todo hecho, sólo falta reiniciar la máquina y comprobar con `ls -lhaZ` que tanto `/var` como `/varold` tienen exactamente el mismo contenido.

```
[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 ~]# _
```

Figure 1.13: Restaurar el contexto del nuevo directorio `/var`, donde montaremos nuestro nuevo disco `/dev/cl/newvar`.

1.4 Sesión 3: Ampliación cifrada y redundante de la capacidad para un usuario.

Este caso es una versión “mejorada” de la Sesión 2, pero hay que realizarla partiendo *desde cero*, es decir, tras haber instalado CentOS únicamente.

RECUERDA que esta sesión se realizará sobre una máquina virtual sobre la que NO se haya implementado lo realizado en la Sesión 2.

1.4.1 Contextualización del escenario profesional

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.

1.4.2 ¿Qué necesitamos?

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

1.4.3 Pasos para la realización de la práctica

1. Partimos inicialmente del estado original de CentOS (puede estar la red configurada, no hay problema con eso) de la figura ??.

```
[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
```

Figure 1.14: Estado inicial antes de crear el RAID-1.

2. Entonces, vamos a apagar el sistema, añadimos los dos discos y reiniciamos. Lo que nos muestra ahora `lsblk` es lo siguiente (figura ??):

```
[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]
sdb          8:16      0   4G  0 disk
sdc          8:32      0   4G  0 disk
sr0          11:0      1 1024M  0 rom
```


Chapter 2

Literature

Here is a review of existing methods.

Chapter 3

Methods

We describe our methods in this chapter.

Chapter 4

Applications

Some *significant* applications are demonstrated in this chapter.

4.1 Example one

4.2 Example two

Chapter 5

Final Words

We have finished a nice book.

Bibliography

Xie, Y. (2015). *Dynamic Documents with R and knitr*. Chapman and Hall/CRC, Boca Raton, Florida, 2nd edition. ISBN 978-1498716963.

Xie, Y. (2020). *bookdown: Authoring Books and Technical Documents with R Markdown*. R package version 0.21.