9.GESTIÓN DE SISTEMAS DE ARCHIVOS Y SWAP

9.1.- Introducción

En Linux todo es un archivo, incluso los dispositivos que son tratados como archivos dentro del directorio */dev*. Los dispositivos se identifican con dos o tres letras, que indica el tipo de dispositivo, y si el dispositivo admite particiones, seguidas del número de partición. Tenemos *hd*(discos duros IDE y dispositivos IDE como cdroms), *sd* (discos duros SCSI), *scd* (cdrom SCSI), *st* (unidades de cinta SCSI), *ht*(unidades de cinta IDE), *fd*:(unidades de diquetes), *lp(*para puertos paralelos), *tty* (terminales o consolas), *pty* (terminales remotas o de red), *ttyS* (puertos serie), *cua* (puertos de comunicación), *eth* (tarjetas o interfaces de red ethernet), ...

Los dispositivos de almacenamiento se representan por archivos especiales llamados dispositivos de bloque. Para identificar un dispositivo de bloque en el directorio */dev*, al hacer un *ls -l* veremos que el primer carácter, antes de los permisos del archivo, es una letra b.

Estos dispositivos pueden ser particionados para dedicar partes distintas del disco a distintos fines. Otra forma de organizar dispositivos de almacenamiento es usando LVM (Logical Volumen Management).

Los dispositivos que no admiten particiones en sus sistemas de archivos tales como disquetes o cdroms, se numeran secuencialmente a partir de 0 o simplemente se omite.

Una vez hemos particionado un disco, debemos darle formato para crear un sistema de archivos en él y una vez formateado, ya se puede incluir en la estructura de directorios del sistema.

9.2.- Partionar un disco

Particionando un disco, dividimos en varias unidades lógicas, una unidad física para que tengan distintos usos. Los discos en una máquina física se nombran como */dev/sda*, */dev/sdb*, … (la *s* es de *scsi* aunque tengan interfaces IDE, SATA o SAS) y en máquinas virtuales se nombran con */dev/vda*, */dev/vdb*, ...

Las particiones de un disco se nombrarán como el disco seguidas con un número: */dev/sda1*, */dev/sdb1*, */dev/vdb3*, …

Existen dos tipos de esquemas de particionado, ambos utilizan 512 bytes por sector, el esquema MBR y el esquema GPT.

Para particionar discos de hasta 2 TiB podemos utilizar las utilidades *fdisk* o *gdisk* pero para discos mayores, sólo podremos utilizar *gdisk*. Ambas utilidades funcionan de forma muy similar.

Una vez realizado el particionado del disco con *fdisk* o *gdisk*, deberemos ejecutar el comando *partprobe <disco>* para que el kernel cargue el nuevo esquema de particionado, no siendo necesario si el disco se particiona por primera vez o si se reinicia el sistema.

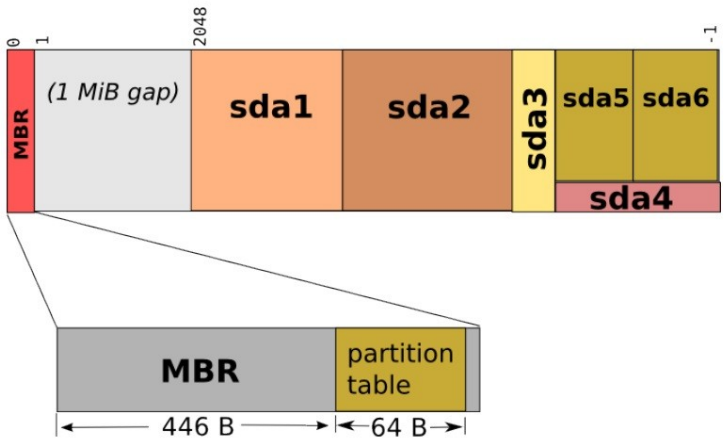
9.2.1.- Esquema de particionado MBR

MBR (**M**aster **B**oot **R**ecord) es el esquema tradicional que usan los sistemas con firmware BIOS con un máximo de 4 particiones primarias o 3 primarias y una extendida y posibilidad de crear particiones lógicas dentro de la extendida hasta un máximo de 15 particiones totales en el disco. Se utilizan 32 bits para direccionar los datos de los bloques del disco, luego el tamaño máximo del disco y/o de la partición es 2 TiB.

Dentro de la partición extendida, los metadatos de las particiones extendidas son una lista enlazada de forma que si se pierde un enlace, se pierden las particiones lógicas.

Es soportado por sistemas de 32 y 64 bits.

Los dispositivos extraíbles sólo pueden tener un esquema de particionado MBR.



Esquema MBR

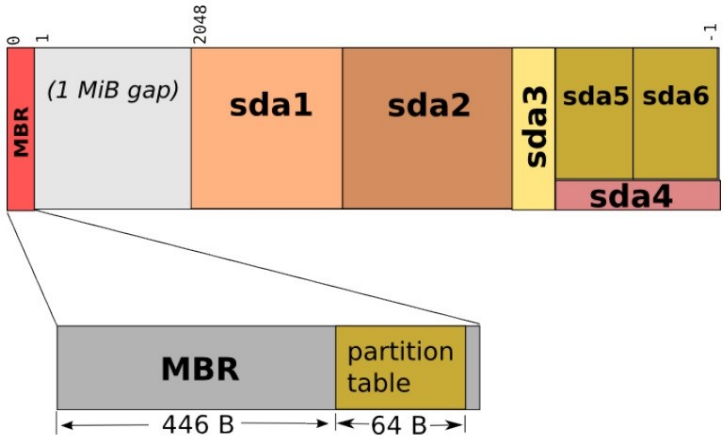
9.2.2.- Esquema de particionado GPT

GPT (**G**UID **P**artition **T**able) se utiliza en sistemas con firmware UEFI. Creado para mejorar las limitaciones de MBR, soporta 128 particiones ya que utiliza 64 bits para direccionar los bloques con lo que se puede llegar a un tamaño de disco de 8ZiB.

Almacena una copia de seguridad del encabezado y de la tabla de particiones al final del disco de forma que, en caso de que los primeros bytes del disco estén dañados, se pueden recuperar.

Soportado sólo por sistemas de 64 bits.

Es capaz de convertir discos MBR a GPT sin pérdida de datos, simplemente con abrir el disco con la utilidad *gdisk* y salir guardando cambios (*w*).



Esquema GPT

9.3.- Crear un sistema de archivos

Un sistema de archivos es una estructura organizada de datos que reside en un dispositivo de almacenamiento y que contiene archivos y directorios. Es una secuencia de bloques lógicos con un tamaño fijo, múltiplo de 512.

El kernel de de Linux utiliza los sistemas de archivos a un nivel lógico y no trata directamente con los dispositivos de almacenamiento a nivel físico. Cada dispositivo de almacenamiento se trata como un dispositivo lógico donde el controlador del disco se va a encargar de transformar las direcciones lógicas del kernel a direcciones físicas del disco.

Una vez que el dispositivo de bloque se ha creado con el particionado, hay que formatearlo en un determinado tipo de sistema de archivos. Se utiliza el comando *mkfs* con la sintaxis:

mkfs -t <tipo> <particion> ó mkfs.<tipo> <particion>

con *<tipo>: msdos,vfat, xfs, ext2, ext3, ext4*, … siendo los más utilizados en CentOS:

- *ext4*: sistema de archivos con *journaling* predeterminado en RHEL6 y CentOS6.
- *xfs*: sistema de archivos de 64 bits con *journaling* predeterminado en RHEL7 y CentOS7. Sus principales ventajas es la posibilidad de agrandarse y defragmentarse mientras está montado y activo, el tener incluidas en CentOS y RHEL herramientas de *backup* y recuperación específicas y el *journaling* de metadatos que permite una recuperación de datos más rápida.

NOTA: El *journaling* es un mecanismo por el cual un sistema de archivos implementa transacciones. Consiste en un registro en el que se almacena la información necesaria para restablecer los datos dañados por una transacción en el caso de que esta falle, p.e. en un fallo de alimentación del sistema.

9.4.- Montar y desmontar un sistema de archivos

Una vez formateado el dispositivo, hay que incluirlo en la estructura de directorios para que sea accesible, esto es a lo que se llama montar el sistema de archivos.

Para que deje de estar accesible, hay que desmontar el sistema de archivos.

9.4.1.- Montado

El montado del dispositivo se puede hacer de forma manual, con el comando *mount* o de forma persistente, utilizando el archivo */etc/fstab*, sobre un punto de montaje (directorio), creado previamente. Ambas formas requieren un usuario privilegiado.

- *Montado manual:* se realiza el montado utilizando el comando *mount*. No persiste en un reinicio del sistema. Sintaxis del comando *mount*:

  mount -o <opciones> <dispositivo> <punto_montaje>

- *Montado persistente*: se añade un línea al archivo */etc/fstab* con el formato:

<dispositivo> <punto_montaje> <tipo_fs> <opciones> <dump> <orden_fsck>

donde:

<*dispositivo*>: partición o disco a incluir en la jerarquía del sistema. Se puede montar en base al nombre del archivo en */dev*, al UUID de dispositivo, poniendo *UUID=<uuid_disp>* o en base a una etiqueta (que se ha definido al darle el formato con *mkfs* usando la opción *-L <etiqueta>*) poniendo *LABEL=<etiqueta>*.

<*punto_montaje*>: en qué directorio de la jerarquía se va a incluir el dispositivo. Debe existir previamente.

<*opciones*>: normalmente se pone *defaults.* Otras opciones son: *ro* para sólo lectura, *rw* para lectura y escritura (valor por defecto), *_netdev* si es un dispositivo que se monta por red, …

<*dump*>: lo utiliza el comando *dump* para realizar un backup de este sistema de archivos. Valores posibles: 1 ó 0, indicando si se realiza el backup o no.

<*orden_fsck*>: si ejecuta o no un chequeo del dispositivo (con *fsck)* en el arranque y el orden. Valores posibles, 0, 1, ...

NOTA: Una línea errónea en el */etc/fstab* puede hacer que el sistema no arranque de forma correcta en el siguiente reinicio, por lo que es importante que tras hacer cambios en el */etc/fstab*, ejecutemos el comando *mount -a* para que monte los sistemas de archivos que hemos añadido y nos de un error en caso de fallo en alguna línea.

9.4.2.- Desmontado

Tanto si el proceso de montado se hizo manual o persistente, se utiliza el comando *umount* para desmontar el sistema de archivos pasándole como argumento el punto de montaje o el nombre del dispositivo. Al igual que en caso del montado, se necesita un usuario privilegiado.

En el caso de un montado persistente, este desmontado sólo permanecerá hasta el siguiente reinicio, si no queremos que vuelva a montar el sistema de archivos en el arranque, deberemos eliminar esa línea del archivo */etc/fstab*.

Si exite algún proceso utilizando el sistema de archivos, el desmontado fallará; tendremos que esperar a que dicho proceso termine, forzar el desmontado usando la opción *-f* o bien, identificar con el comando *lsof* qué proceso está utilizando el sistema de archivos y obrar en consecuencia.

9.4.3.- Dispositivos extraíbles

En el caso de tener un sistema con entorno gráfico, los dispositivos de almacenamiento extraíbles cuando se insertan en el sistema, se montan automáticamente en el punto de montaje */run/media/<usuario>/<etiqueta>* donde *<usuario>* es el usuario con el que hayamos hecho *login* y *<etiqueta>* es la etiquetada que se le dio al sistema de archivos existente en el dispositivo cuando se creó.

En el caso de un sistema sin entorno gráfico, una vez insertado el usb, hay que revisar el archivo */var/log/messages* y en las últimas líneas, aparecerá el mensaje "[sdx] *Attached SCSI removable disk*" que nos indica en qué dispositivo del */dev* lo ha enlazado, en *sdx*. Otra forma es ejecutar el comando *lsblk* donde aparece un nuevo dispositivo de almacenamiento *sdx*. Pero no lo monta de forma automática, deberemos hacerlo con el comando *mount*.

En el caso de máquinas virtuales, deben tener el hardware *usb redirector* y con el visor de la máquina virtual en pantalla, el usb pasará a estar conectado en la máquina virtual no siendo visible desde la máquina física.

Antes de extraer estos dispositivos, hay que hacer un desmontado del sistema de archivos que se encuentra en ellos para que se terminen de escribir los datos pendientes, si no, podríamos perder datos incluso corromper el sistema de archivos.

9.5.- Comandos para examinar sistemas de archivos

Para obtener una visión de cómo está organizados los sistemas de archivos, utilizaremos principalmente los comandos:

- ***df***: muestra los sistemas de archivos montados, la cantidad de espacio usado y disponible en ellos, el porcentaje de ocupación y en qué punto de montajes están. Con la opción *-h* ó *-H* muestra el espacio disponible en lugar de en bytes en ks (K), megas (M), gigas(G), … La diferencia entre *-h* y *-H* es que con *-h* utiliza unidades internacionales (SI) y *-H* utiliza unidades IT.
- ***du***: muestra el tamaño de cada archivo y directorio en el sistema de archivos dado. También tiene las opciones *-h* y *-H*.
- ***blkid***: muestra cómo están las particiones de los dispositivos de almacenamiento y los sistemas de archivos montados en ellas junto con el UUID y formato del sistema de archivos.
- ***lsblk***: muestra los dispositivos, unidades, particiones y sus capacidades, estén montadas o no. Con la opción *-fm* muestra información más amplia.
- ***dd***: copia datos a nivel de bloque. Útil para duplicar dispositivos de almacenamiento bloque a bloque o incluso destruir un disco. Su sintaxis es:

  dd if=<origen> of=<destino> bs=<tamaño_bloque> count=<número_bloques>

  donde el <*origen*> y <*destino*> pueden ser dispositivos de almacenamiento o archivos, <*tamaño_bloque*> es el tamaño de bloque a utilizar y <*número_bloques*> es el número de bloques a copiar.

  Ejemplos útiles:

  - Clonar un disco:

    dd if=/dev/sdx of=/dev/sdy bs=1024

  - Hacer un disco inaccesible eliminando el contenido del primer bloque:

    dd if=/dev/zero of=/dev/sdx bs=512 count=1

  - Destruir todo el contenido de un disco:

    dd if=/dev/zero of=/dev/sdx ó dd if=/dev/urandom of=/dev/sdx

  - Crear una iso de un CD:

    dd if=/dev/cdrom of=<archivo>.iso

  - Crear copia de seguridad del sector de arranque de un disco:

    dd if=/dev/sdx of=<archivo> bs=512 count=1

- Restaurar copia de seguridad del sector de arranque de un disco:

  dd if=<archivo> of=/dev/sdx bs=512 count=1

- ***lsof***: lista los archivos abiertos y el proceso que está accediendo a ellos. Útil cuando desmontamos un sistema de archivos y nos da un error porque hay un proceso utilizando algún archivo que se encuentra en él.

9.6.- Enlaces entre archivos

Los enlaces permiten dar varios nombres a un mismo archivo. En Linux, los archivos del sistema se representa por un inodo único, un bloque que almacena metainformación del archivo (permisos, usuario y grupo propietario, ubicación en el sistema de archivos, creación, modificación, ...).

Un directorio tiene la lista de números de inodo con sus nombres correspondientes de los archivos que contiene.

9.6.1.- Enlaces físicos

Los enlaces físicos son distintos nombres que apuntan al mismo número de inodo. Podemos ver cuantos enlaces físicos tiene un determinado archivo utilizando el comando *ls -l,* es el número que aparece entre los permisos y el propietario del archivo. Por defecto todos los archivos tienen 1.

Para ver el número de inodo de un archivo ejecutaremos *ls -i <archivo>*.

Si hay varios archivos que tienen el mismo número de inodo, tienen la misma información, si elimino uno de ellos, la información permanece ya que existen otros archivos que apuntan al mismo número de inodo. El archivo será eliminado definitivamente cuando no hay enlaces a él.

Crear un enlace físico en lugar de copiar un archivo, es una forma de ahorrar espacio en disco, la información está una sóla vez pero se referencia varias veces.

Los enlaces físicos tienen que estar el mismo sistema de archivos ya que cada uno de ellos tiene su propia tabla de inodos para evitar que un mismo número de inodo apunte a dos ubicaciones distintas.

Para crear enlaces físicos, se ejecuta el comando *ln <archivo_original> <archivo_nuevo>*, una vez creado el nuevo archivo, no se sabe cuál es el original y cuál es el nuevo. Si hacemos algún cambio en uno de los archivos, ya sea en los metadatos o en el contenido del archivo, todos los enlaces físiccos que existan del mismo inodo, tendrán reflejado el cambio.

9.6.2.- Enlaces simbólicos

Los enlaces simbólicos o enlaces *soft* son archivos especiales que simplemente apuntan a otro archivo que, a diferencia de los enlaces físicos, si pueden apuntar a un archivo de otro sistema de archivos. Si se realiza una modificación sobre el enlace simbólico, se verán reflejado en el archivo original pero si se elimina el archivo original, el enlace simbólico deja de funcionar

Con el comando *ls -l* del archivo enlace simbólico, veremos a qué archivo apunta.

Para crear un enlace simbólico, utilizaremos el comando:

```
ln -s <archivo_original> <archivo_enlace_simbólico>
```

9.7.- Localizar archivos en los sistemas de archivos

Un administrador de sistemas necesitará en algún momento realizar búsquedas de archivos que cumplan cierto criterio. Las dos herramientas más importantes de que dispone son:

- ***locate*** <*archivo*>: busca el archivo en la base de datos interna de *locate* que tiene registrados en qué sistema de archivos está cada archivo. Con la opción *-i*, no diferencia entre mayúsculas y minúsculas y con la opción *-n <número>* limita la lista de resultados al número dado. Si se ejecuta como un usuario regular, sólo devuelve los resultados en las rutas donde ha podido leer.

  La base de datos interna de *locate* se actualiza diariamente, se puede actualizar en un momento dado con el comando *updatedb* ejecutado como *root*.

  Esta base de datos interna está en */var/lib/mlocate/mlocate.db*.
- ***find*** <*directorio*> *-name* <*archivo*>: hace una búsqueda de un archivo en los sistemas de archivos del sistema en base a unos criterios dados. Hay que tener permisos de lectura y ejecución sobre los directorios para poder buscar en ellos. Si se omite el directorio, se toma el directorio actual. En <*archivo*> se pueden utilizar comodines (*; ?, …) para encontrar coincidencias parciales pero hay que entrecomillarlo. La opción *-iname* hace que no se diferencie entre mayúsculas y minúsculas. El comando *find* tiene muchas otras opciones como *-user <usuario>*, *-group <grupo>*, *-uid <uid>*, *-gid <gid>* para buscar los archivos pertenecientes a un determinado usuario o grupo por nombre de usuario o grupo o usando el uid y gid, *-perm <permisos>* para hacer búsquedas de archivos con determinados permisos, *-size <tamaño>* búsquedas de archivos de un determinado tamaño, ...

9.8.- Swap

La memoria virtual de un sistema está formada por la memoria física (RAM) y la memoria swap. Con la memoria virtual, el kernel de Linux es capaz de ejecutar procesos que requieren de más memoria de la que se encuentra disponible físicamente en el sistema.

La memoria swap, también llamada área de swap o memoria de intercambio, es una partición de un disco o un archivo en un sistema de archivos (que no sea NFS) que el kernel utiliza para incrementar la memoria física del sistema.

Cuando se excede el uso de la memoria física del sistema, el kernel desaloja de dicha memoria las páginas correspondientes a un proceso poco activo pasándolas a swap utilizando el espacio liberado para un proceso nuevo. Cuando estas páginas pasadas a swap vuelvan a ser necesarias, se pasan a la RAM para que el proceso correspondiente las pueda usar.

La swap es mucho más lenta que la RAM, ya que es un acceso/escritura en disco por ello debe limitarse su uso a lo mínimo imprescindible.

9.8.1.- Uso de la memoria swap

Se puede modificar con que frecuencia los procesos pueden ser movidos de la memoria física a la swap con un valor que define el peso relativo de uso de swap. El parámetro es ***vm.swappiness*** que tiene un valor de 0 a 100, donde 0 desactiva por completo el uso de swap y 100 hace que se haga un uso de swap intensivo. Se recomienda un valor bajo de este parámetro. La recomendación es que los valores indicados para estaciones de trabajo sean de 10-20 y en servidores de 20-60.

Se puede consultar el valor de este parámetro con un ***cat /proc/sys/vm/swappiness*** o con el comando ***sysctl vm.swappiness***.

Para modificar este valor en *runtime* se puede modificar directamente el archivo ***/proc/sys/vm/swappiness*** poniendo el nuevo valor en lugar del que hay o utilizando el comando ***sysctl -w vm.swappiness=n***, siendo ***n*** el nuevo valor del parámetro. Este cambio permanece hasta el siguiente reinicio del sistema; para hacer el cambio persistente, se debe añadir en el archivo /***etc/sysctl.conf*** la línea ***vm.swappiness=n***, donde ***n*** es el nuevo valor del parámetro.

9.8.2.- Crear y activar espacio de swap

Para crear un nuevo espacio de swap, hay que seguir los pasos:

1. Crear la partición o archivo de swap:

Para crear una partición en un disco con formato de swap, se utilizan las utilidades ***fdisk*** o ***gdisk*** y en el tipo de la partición se le pone tipo *"Linux swap"* (82 en fdisk y 8200 en gdisk).

Para crear un archivo de swap, se crea un archivo con un determinado tamaño llenándolo de ceros con el comando ***dd if=/dev/zero of=<archivo> bs=1024 count=<tamaño_bytes>***. Por ejemplo, para crear un archivo de swap llamado ***new_swap*** en **/** con tamaño ***512Mib***, ejecutaremos:

dd if=/dev/zero of=/new_swap bs=1024 count=512000

2. Formatear la partición o archivo con formato de swap con el comando mkswap con el formato:

mkswap <opciones> <partición> o ***mkswap <opciones> <archivo>***

Las opciones más usadas son:

-c: chequea el dispositivo (si es de bloque) y busca bloques dañados y los marca para no utilizarlos.

-f: fuerza el formato

-L <*etiqueta*>: Le pone a la partición de swap la etiqueta <*etiqueta*> para luego activar dicha partición en base a la etiqueta en lugar del nombre del dispositivo o UUID.

3. Activar la partición de swap para que pueda ser utilizada:

Una vez formateada la partición, se activa en *runtime* con el comando ***swapon <particion>*** o ***swapon <archivo>***, o utilizando ***swapon -L <etiqueta>*** en el caso de haberle asignado una con el comando mkswap.

Se puede añadir la opción *-p <prioridad>* para establecer una prioridad de uso a este nuevo espacio de swap. La prioridad en CentOS 7 es un valor de -1 a 32767, siendo -1 el valor por defecto.

Si se quiere hacer persistente este nuevo espacio de swap, deberemos añadir en el archivo ***/etc/fstab*** la línea:

<*origen*> swap swap defaults 0 0

donde <*origen*> puede ser: la partición a usar, el archivo de swap, LABEL=<etiqueta> en el caso de haber asignado una etiqueta al ejecutar *mkswap* o UUID=<uuid> poniendo en <uuid> el uuid que le asigna el comando *mkswap* tras formatear el archivo o partición.

En el caso de querer establecer una prioridad, se cambia el cuarto campo de las opciones de *defaults* a *defaults,pri=<prioridad>*.

Una vez añadido al archivo ***/etc/fstab*** la línea con el nuevo espacio de swap a utilizar, se ejecuta el comando ***swapon -a*** para que las líneas existentes en dicho archivo con el segundo y tercer campo con "*swap*" y que no tengan en el campo de las opciones la opción "*noauto*" se activen.

4. Comprobar que se está utilizando el nuevo espacio de swap con el comando ***swapon -s*** que muestra la información de la memoria swap activa o con el comando ***free*** mirando la cantidad de memoria swap existente.

9.8.3.- Desactivar espacio de swap

Utilizaremos el comando:

swapoff <partición> o swapoff <archivo>

Si la hemos activado de forma persistente en el archivo */etc/fstab*, habría que eliminar la línea correspondiente.

9.8.4.- Recomendaciones del tamaño de swap

Es importante analizar qué aplicaciones en el sistema se van a ejecutar y cuál va a ser su carga para determinar la cantidad de memoria virtual necesaria.

Existía una recomendación general sobre utilizar el doble de la memoria RAM que tuviese el sistema pero con los sistemas actuales que pueden llevar terabytes de RAM esto es innecesario, con lo que las nuevas recomendaciones son:

- *RAM de hasta 2 Gb*: swap=2xRAM o swap=3xRAM si el sistema utiliza hibernación.
- *RAM de 2Gb-8Gb:* swap=RAM o swap=2xRAM si el sistema utiliza hibernación.
- *RAM de 8Gb-64Gb*: swap de al menos 4Gb o swap=1.5xRAM si el sistema utiliza hibernación.
- *RAM de más de 64Gb*: swap de al menos 4Gb y no se recomienda el uso de hibernación.

9.9.- Caso práctico

En **server1**, utilizando el disco *vdb*, se va a crear un sistema de archivos *xfs* montado persistente en */bkup* de *500MiB* con la etiqueta *BACKUP*. Además, se creará otro sistema de archivos *ext4* de *250MiB* montado persistente en */material* con la etiqueta *MATERIAL*, su montado se hará en base al UUID.

También en **server1**, se creará un enlace simbólico al archivo */var/log/messages* en el home del usuario *admin* con el nombre de *mensajes_sistema*.

En **server1**, buscar todos los directorios en los que el usuario *admin* tenga permisos de escritura y los directorios del sistema que tengan activados todos los permisos. Localizar el archivo *logrotate.conf*.

En el sistema **central** vamos a añadir un espacio de swap extra utilizando una partición en */dev/vdb* de 250 Mib con una prioridad de ***10***. Se añadirá este espacio de swap de forma persistente en base a la etiqueta "***masswap***". Además se cambiará el valor de la variable de ***swappiness*** a ***40*** de forma persistente.

RESOLUCIÓN

- Sistema de archivos *xfs* montado persistente en */bkup* de *500Mib* con etiqueta *BACKUP*:

```
[root@server1 ~]# echo -en "n\np\n\n\n+500M\nw\n" | fdisk /dev/vdb
[root@server1 ~]# fdisk -l /dev/vdb
Disk /dev/vdb: 1073 MB, 1073741824 bytes, 2097152 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0x207b3620
   Device Boot      Start         End      Blocks   Id  System
/dev/vdb1            2048     1026047      512000   83  Linux
[root@server1 ~]# mkfs.xfs -L BACKUP /dev/vdb1
meta-data=/dev/vdb1              isize=512    agcount=4, agsize=32000 blks
         =                       sectsz=512   attr=2, projid32bit=1
         =                       crc=1        finobt=0, sparse=0
data     =                       bsize=4096   blocks=128000, imaxpct=25
         =                       sunit=0      swidth=0 blks
naming   =version 2              bsize=4096   ascii-ci=0 ftype=1
log      =internal log           bsize=4096   blocks=855, version=2
         =                       sectsz=512   sunit=0 blks, lazy-count=1
realtime =none                   extsz=4096   blocks=0, rtextents=0
[root@server1 ~]# mkdir /bkup
[root@server1 ~]# echo "LABEL=BACKUP /bkup xfs defaults 0 0" >> /etc/fstab
root@server1 ~]# mount -a
[root@server1 ~]# mount | grep /bkup
/dev/vdb1 on /bkup type xfs (rw,relatime,seclabel,attr2,inode64,noquota)
```

- Sistema de archivos *ext4* montado persistente en */material* de *250 MiB* con la etiqueta *MATERIAL*:

[root@server1 ~]# echo -en "n\np\n\n\n+250M\nw\n" | fdisk /dev/vdb

[root@server1 ~]# fdisk -l /dev/vdb

```
Disk /dev/vdb: 1073 MB, 1073741824 bytes, 2097152 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0x207b3620
   Device Boot      Start         End      Blocks   Id  System
/dev/vdb1            2048     1026047      512000   83  Linux
/dev/vdb2         1026048     1538047      256000   83  Linux
```

[root@server1 ~]# partprobe

[root@server1 ~]# mkfs.ext4 -L MATERIAL /dev/vdb2

```
mke2fs 1.42.9 (28-Dec-2013)
Filesystem label=MATERIAL
OS type: Linux
Block size=1024 (log=0)
Fragment size=1024 (log=0)
Stride=0 blocks, Stripe width=0 blocks
64000 inodes, 256000 blocks
12800 blocks (5.00%) reserved for the super user
First data block=1
Maximum filesystem blocks=33816576
32 block groups
8192 blocks per group, 8192 fragments per group
2000 inodes per group
Superblock backups stored on blocks:
        8193, 24577, 40961, 57345, 73729, 204801, 221185

Allocating group tables: done
Writing inode tables: done
Creating journal (4096 blocks): done
Writing superblocks and filesystem accounting information: done
```

[root@server1 ~]# mkdir /material

[root@server1 ~]# blkid

```
/dev/vda1: UUID="065285f5-a5a2-4eaf-a6da-62adc631fb04" TYPE="xfs"
/dev/vda2: UUID="0e0mPU-yxkG-8xy8-p1WD-F5gv-uDar-cJnJ6f" TYPE="LVM2_member"
/dev/vdb1: LABEL="BACKUP" UUID="a93e7ddd-1886-4da4-b14b-9549142c6330"
TYPE="xfs"
/dev/vdb2: LABEL="MATERIAL" UUID="e27d29c6-9437-4332-be3f-2156faa6a426"
TYPE="ext4"
/dev/mapper/centos-root: LABEL="root" UUID="8aa04869-9ca4-4f50-90b4-a6fb5f891638"
TYPE="xfs"
/dev/mapper/centos-swap: UUID="15530ecb-e589-46d4-98ee-9a0117634a1c"
```

TYPE="swap"

[root@server1 ~]# echo “UUID= e27d29c6-9437-4332-be3f-2156faa6a426 /material ext4 defaults 0 0” >> /etc/fstab

root@server1 ~]# mount -a

[root@server1 ~]# mount | grep material

/dev/vdb2 on /material type ext4 (rw,relatime,seclabel,data=ordered)

- Enlace simbólico al archivo */var/log/messages* en el home de *admin* llamado *mensajes_sistema*:

[root@server1 ~]# su - admin

[admin@server1 ~]$ ln -s /var/log/messages ~/mensajes_sistema

- Directorios donde el usuario *admin* puede escribir:

[root@server1 ~]# find /home -type d -perm -200 -user admin/home/admin

/home/admin/web_publica

/home/admin/musica

/home/admin/videos

- Directorios del sistema con todos los permisos activados:

[root@server1 ~]# find / -type d -perm -777

/dev/mqueue
/dev/shm
/var/tmp
/tmp
/tmp/.font-unix
/tmp/.XIM-unix
/tmp/.X11-unix
/tmp/.ICE-unix
/tmp/.Test-unix

- Localizar el archivo *logrotate.conf*, si vamos a utilizar el comando *locate*, hay que instalar previamente el paquete que lo proporciona ya que en un instalación mínima no se instala.

[root@server1 ~]# yum provides locate

Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
Available Packages
Name : **mlocate**
Arch : x86_64
Version : 0.26
Release : 6.el7
Size : 113 k
Repo : centos7
Summary : An utility for finding files by name
URL : https://fedorahosted.org/mlocate/
License : GPLv2
Description : mlocate is a locate/updatedb implementation. It keeps a database

```
: of all existing files and allows you to lookup files by name.
:
: The 'm' stands for "merging": updatedb reuses the existing
: database to avoid rereading most of the file system, which makes
: updatedb faster and does not trash the system caches as much as
: traditional locate implementations.
```

[root@server1 ~]# yum install -y mlocate

[root@server1 ~]# locate logrotate.conf

locate: can not stat () `/var/lib/mlocate/mlocate.db': No such file or directory

Como se acaba de instalar el comando, no hay base de datos creada, hacemos:

[root@server1 ~]# updatedb

[root@server1 ~]# locate logrotate.conf

/etc/logrotate.conf

/usr/share/man/man5/logrotate.conf.5.gz

- En **central**, creamos la nueva partición de swap en disco */dev/vdb* de *250 Mib* con formato *"Linux Swap"*:

[root@central ~]# echo -en "n\np\n\n\n+250M\nt\n82\nw\n" | fdisk /dev/vdb

Formateamos la *swap* y compruebo la *swap* existente antes de añadir el nuevo espacio de swap:

[root@central ~]# mkswap -L masswap /dev/vdb3

[root@central ~]# swapon -s

Filename	Type	Size	Used	Priority
/dev/dm-1	partition	2097148	0	-1

Añadimos el nuevo espacio de forma persistente en el */etc/fstab* y la activo:

[root@central ~]# echo "LABEL=masswap swap swap defaults,pri=10 0 0" >> /etc/fstab

[root@central ~]# swapon -a

Verificamos que en la *swap* existente aparece el nuevo espacio de *swap*:

[root@central ~]# swapon -s

Filename	Type	Size	Used	Priority
/dev/dm-1	partition	2097148	0	-1
/dev/vdb3	partition	511996	0	10

Modificamos el valor del parámetro *swappiness* a *40*:

[root@central ~]# sysctl -w vm.swappiness=40

[root@central ~]# echo vm.swappiness=40 >> /etc/sysctl.conf