Utilización extendida de BTRFS

1. Los subvolúmenes

a. Un sistema de archivos dentro de otro sistema de archivos

Los subvolúmenes son con frecuencia comparados a los volúmenes lógicos de LVM. Esto es un recurso rápido para explicar una cosa diferente. Un subvolumen BTRFS puede ser visto como un espacio de nombres (namespace) POSIX diferenciado. En la práctica, la creación de un subvolumen se parece a un subsistema de archivos dentro de un sistema de archivos: debe formar parte del sistema de archivos donde ha sido creado, pero es utilizable como si fuera un sistema de archivos totalmente aparte: puede ser montado, disponer de sus propios derechos y cuotas, y puede hacerse una instantánea (un snapshot).

Y sobre todo, al contrario que un volumen lógico, no se crea ningún dispositivo de tipo bloque. Cuando creamos un subvolumen, aparece como una simple carpeta en el entorno de su creación.

La raíz de un sistema de archivos btrfs es en sí un subvolumen, por defecto.

Los subvolumenes se manipulan con el comando **btrfs subvolume**.

b. Creación

Empleamos el comando **btrfs subvolume create**. Aquí un ejemplo de la creación de un subvolumen subvol1 y un ejemplo de su comportamiento:

```
# df -Th .
S. archivos      Tipo   Tamaño   Usados    Disp  Uso%   Montado en
/dev/sdh1        btrfs  10,0G     17M     9,5G    1%   /my_btrfs
# btrfs subvolume create subvol1
   Create subvolume './subvol1'
# ls
   subvol1
```

Puede listar los subvolúmenes de esta forma:

```
# btrfs subvolume list /my_btrfs
ID 257 gen 8 top level 5 path subvol1
```

c. Montaje

El principal interés de los subvolúmenes es el de poder montarlos en cadena. También se pueden utilizar opciones de montaje diferentes. Algunas no se admiten y es posible que su lista evolucione según las versiones del núcleo. La opción ro (read only) no se admite hasta la versión del núcleo 3.15. Aquí vemos una demostración. En primera instancia, montamos el subvolumen en /subvol1 mnt, luego escribimos un archivo allí. Posteriormente verificamos que el archivo está en la ubicación inicial.

```
# mkdir /subvol1_mnt
# mount -t btrfs -o subvol=subvol1 /dev/sdh1 /subvol1_mnt/
# df
...
/dev/sdh1       10484736   16736   9942976   1% /my_btrfs
/dev/sdh1       10484736   16736   9942976   1% /subvol1_mnt
# cd /subvol1_mnt
# touch toto
# ls /my_btrfs/subvol1
toto
# rm /subvol1_mnt/toto
```

Desmontamos un subvolumen como cualquier sistema de archivos:

```
# umount /subvol1_mnt
```

Volvemos a montar el subvolumen para solo lectura e intentamos las mismas operaciones. Constatamos que no podemos escribir en el nuevo punto de montaje, pero que si podemos sobre el subvolumen original:

```
# mount -t btrfs -o subvol=subvol1,ro /dev/sdh1 /subvol1_mnt/
# cd /subvol1_mnt/
# touch titi
touch: no se puede efectuar `touch' sobre « titi »: Sistema de archivos
de sólo lectura
# cd /my_btrfs/subvol1/
# touch titi
# ls titi
titi
```

d. Destrucción

Un subvolumen se destruye de la siguiente forma:

```
# btrfs subvolume delete /my_btrfs/subvol1
Delete subvolume (no-commit) '/my_btrfs/subvol1'
```

Por razones prácticas, el subvolumen ha sido conservado durante el resto del capítulo.

2. Los snapshots

a. Fundamentos

Un snapshot es una instantánea del sistema de archivos en un instante dado. Se trata de captar una «imagen» que represente su estado en el momento de su creación. De esta forma, las modificaciones aportadas a la instantánea o al sistema de archivos son independientes. El tamaño ocupado por una instantánea es el de los datos que se han modificado.

Sólo se puede hacer un snapshot de un subvolumen. Ahora bien, la raíz de un sistema de archivos btrfs es en sí misma un subvolumen, lo que permite hacer un snapshot. Los snapshots son vistos en sí mismos como subvolúmenes.

El interés es evidente: una instantánea permite por ejemplo verificar modificaciones antes

de efectuarlas realmente en el sistema de archivos de base. Con btrfs podemos incluso ir más lejos, ya que es posible, si el sistema de archivos raíz es de este formato, hacer una instantánea y utilizarla como raíz: de esta forma puede verificar la actualización de su distribución y si no funciona, volver a la raíz original. Si funciona, puede repartirla en la raíz y reiniciar la actualización, o sincronizar (con rsync por ejemplo) la instantánea con la verdadera raíz antes de reiniciar.

Las instantáneas tienen otros usos. Con el uso de contenedores (lxc, ocker), podemos presentar una instantánea que se utilizará para trabajar, siempre que se clone todo el contenido: ganar tiempo, ganar espacio. Con la destrucción del contenedor, eliminamos la instantánea: todas las modificaciones posiblemente aportadas desaparecerán. Podemos de esta forma, por ejemplo, crear una instantánea de un entorno de producción en un entorno de prueba, como plantilla, etc. De esta forma nos aseguramos de disponer del último estado de producción para trabajar.

b. Creación

Aquí tenemos el comando que creará una instantánea del subvolumen subvol1 que se montará en /my_btrfs/snap subvol1:

```
# btrfs subvolume snapshot /my_btrfs/subvol1 /my_btrfs/snap_subvol1
Create a snapshot of '/my_btrfs/subvol1' in '/my_btrfs/snap_subvol1'
# cd /my_btrfs/snap_subvol1
# ls
titi
```

Si creamos un archivo en la instantánea, este no aparece en el subvolumen original:

```
# touch loulou
# ls /my_btrfs/subvol1
titi
```

c. Montaje

El montaje funciona exactamente de la misma manera que para los subvolúmenes, con la

misma sintaxis de comando.

```
# mkdir /snap_mnt
# mount -t btrfs -o subvol=snap_subvol1, /dev/sdh1 /snap_mnt/
```

d. Destrucción

Un snapshot se suprime exactamente igual que un subvolumen:

```
# btrfs subvolume delete /my_btrfs/snap_subvol1
Transaction commit: none (default)
Delete subvolume '/my_btrfs/snap_subvol1'
```

e. Operaciones con los ID

Puede forzar, empleando el comando **list**, que cada subvolumen disponga de un identificador:

```
# btrfs subvolume list /my_btrfs
ID 258 gen 14 top level 5 path subvol1
ID 259 gen 14 top level 5 path snap_subvol1
```

Partamos de la idea que la raíz del sistema de archivos se encuentra en formato btrfs.

Imagine ahora que desea efectuar operaciones en la raíz, por ejemplo actualizar su distribución, pero que por razones prácticas puede dar marcha atrás en caso de problemas. El ideal es crear una instantánea y trabajar sobre esta. El problema es que desea trabajar sobre la ruta /, y no sobre la ruta de la instantánea.

Antes de nada debemos crear una instantánea de /:

```
# btrfs subvolume snapshot / /snapshot
Create a snapshot of '/' in '/snapshot'
```

Recupere el identificador de la instantánea creada:

```
btrfs subvolume list /snapshot
ID 257 gen 7 top level 5 path snapshot
```

Modifique el id predeterminado del subvolumen por defecto /:

```
# btrfs subvolume set-default 257 /
```

Reinicie. Desplácese a /, verifique que la instantánea (snapshot) no siga apareciendo. Esto es normal ya que / es de hecho la instantánea (snapshot): le ha dado su id.

```
# cd /
# ls
bin  boot  dev  etc  home  lib  lib64  media ...
```

Realice las modificaciones.

Ahora, desea hacer un rollback. Deberá volver al identificador original en /. Como se trata de una raíz del sistema de archivos, el identificador será siempre 0:

```
# btrfs subvolume set-default 0 /
```

Reinicie, y ¡ya !

Si desea aplicar los cambios, se podría montar el id 0 subvolumen en otro punto de montaje, y copiar (a través de rsync) su contenido, y luego poner todo de vuelta en la dirección correcta:

```
# mount -o subvolid=0 /mnt
```

3. Utilizar varios discos

BTRFS permite extender un sistema de archivos a varios discos. Esto es extremadamente simple: si nos falta espacio, añadimos un disco y el espacio estará disponible inmediatamente.

Cuando creamos un sistema de archivos btrfs en varios discos, sin opciones particulares, los metadatos en espejo (copiados idénticamente en cada disco), pero las escrituras son lineales: el primer disco se llena, luego el siguiente, y así sucesivamente. Aquí mostramos como crear un nuevo sistema de archivos en dos discos:

```
btrfs-progs v5.2.1
See http://btrfs.wiki.kernel.org for more information.
Label:              (null)
UUID:               e3d9e67d-a739-4374-89b6-7b6c818e5ac8
Node size:          16384
Sector size:        4096
Filesystem size:    400.80MiB
Block group profiles:
  Data:             RAID0            128.00MiB
  Metadata:         RAID1             32.00MiB
  System:           RAID1              8.00MiB
SSD detected:       no
Incompat features:  extref, skinny-metadata
Number of devices:  2
Devices:
   ID        SIZE  PATH
    1   200.40MiB  /dev/sdb
    2   200.40MiB  /dev/sdc
```

Este sistema de archivos puede montarse desde cualquier dispositivo de tipo bloc: sdb o sdd.

BTRFS soporta varios modos RAID. RAID5 y RAID6 son todavía experimentales (febrero de 2017) y no deben ser usados (https://btrfs.wiki.kernel.org/index.php/RAID56). RAID 0, 1 y 10 están perfectamente soportados. Aquí vemos como crear un RAID0 (stripping):

```
# mkfs.btrfs -f -d raid0 /dev/sdb /dev/sdc
```

Monte el sistema de archivos y cree en este un archivo de gran tamaño:

```
# mount /dev/sdc /my_btrfs
# cd /my_btrfs
# dd if=/dev/zero of=big_file bs=1024 count=2000000
2000000+0 registros leídos
2000000+0 registros escritos
204800000 bytes (205 MB,195 MiB) copiados, 0,845944 s, 242 MB/s
# df -H /my_btrfs/
/dev/sdb          421M    223M    130M  64%  /my_btrfs
```

El estado del sistema de archivo se obtiene con show:

```
# btrfs filesystem show /my_btrfs
Label: none uuid: d95ee15b-54e3-4a0e-87cd-cd51b3fbe345
   Total devices 2 FS bytes used 195.73MiB
   devid    1 size 200.40MiB used 168.00MiB path /dev/sdb
   devid    2 size 200.40MiB used 168.00MiB path /dev/sdc
```

Añada un disco en caliente de la siguiente forma:

```
# btrfs device add /dev/sdd /my_btrfs -f
```

Solo hay un pequeño problema: el espacio se añade pero los datos no se extienden al nuevo disco:

```
# btrfs filesystem show /my_btrfs
Label: none uuid: d95ee15b-54e3-4a0e-87cd-cd51b3fbe345
   Total devices 3 FS bytes used 195.73MiB
   devid    1 size 200.40MiB used 168.00MiB path /dev/sdb
   devid    2 size 200.40MiB used 168.00MiB path /dev/sdc
   devid    3 size 200.40MiB used 0.00B path /dev/sdd
```

Deberá repartir los datos en el grupo de discos de la siguiente manera:

```
# btrfs balance start -d -m /my_btrfs
```

```
Done, had to relocate 8 out of 8 chunks
# btrfs filesystem show /my_btrfs
Label: none uuid: d95ee15b-54e3-4a0e-87cd-cd51b3fbe345
    Total devices 3 FS bytes used 146.83MiB
    devid    1 size 200.40MiB used 96.00MiB path /dev/sdb
    devid    2 size 200.40MiB used 84.00MiB path /dev/sdc
    devid    3 size 200.40MiB used 116.00MiB path /dev/sdd
```