En Debian podemos hacer que el contenido de una carpeta aparezca también dentro de otra ruta del sistema de archivos sin copiar los datos ni crear un enlace simbólico. Para ello podemos utilizar un **bind mount**, una funcionalidad del kernel de Linux que permite montar un directorio existente en otro punto del árbol de directorios. Este mecanismo resulta especialmente útil cuando una aplicación espera encontrar determinados archivos en una ruta concreta, cuando queremos reorganizar la estructura de directorios sin mover físicamente los datos o cuando queremos presentar el mismo directorio desde varias ubicaciones.

Supongamos que tenemos nuestros datos almacenados en:

```
/home/nipegun/Datos
```

y queremos que esos mismos datos también aparezcan en:

```
/mnt/Datos
```

Primero creamos el directorio que utilizaremos como punto de montaje:

```
sudo mkdir -p /mnt/Datos
```

Después realizamos el bind mount:

```
sudo mount --bind /home/nipegun/Datos /mnt/Datos
```

A partir de ese momento, al acceder a:

```
/mnt/Datos
```

veremos exactamente el mismo contenido que existe en:

```
/home/nipegun/Datos
```

No estamos copiando los archivos. Ambas rutas apuntan a los mismos objetos del sistema de archivos. Por tanto, si modificamos, creamos o eliminamos un archivo desde cualquiera de las dos rutas, el cambio será visible inmediatamente desde la otra.

Podemos comprobar que realmente se trata de un punto de montaje utilizando:

```
findmnt /mnt/Datos
```

También podemos verlo mediante:

```
mount | grep "/mnt/Datos"
```

Qué ocurre con los permisos

Un bind mount **no crea una nueva copia de los permisos ni modifica los propietarios de los archivos**. Los UID, GID y permisos Unix siguen siendo exactamente los mismos que los del directorio original. Por ejemplo, si tenemos:

```
/home/nipegun/Datos/documento.txt
```

con los siguientes permisos:

```
-rw-r--r-- 1 nipegun nipegun documento.txt
```

después del bind mount veremos esos mismos permisos al consultar:

```
/mnt/Datos/documento.txt
```

Por tanto, **un bind mount no sirve para saltarnos los permisos del sistema de archivos**. Si nuestro usuario no puede leer, escribir o ejecutar determinado archivo desde la ruta original, tampoco podrá hacerlo desde la ruta montada. También debemos tener en cuenta los permisos de los directorios padre que debemos atravesar para llegar al punto de montaje. Por ejemplo:

```
/mnt/Datos
```

puede contener archivos accesibles para nuestro usuario, pero necesitaremos también permiso de ejecución sobre:

```
/mnt
```

para poder atravesarlo.

Qué ocurre con el contenido previo de la carpeta de destino

Si el directorio utilizado como destino ya contiene archivos, estos **no se eliminan**, pero dejan temporalmente de ser visibles mientras el bind mount está activo. Por ejemplo, si antes del montaje tenemos:

```
/mnt/Datos/archivo-antiguo.txt
```

y posteriormente ejecutamos:

```
sudo mount --bind /home/nipegun/Datos /mnt/Datos
```

el contenido original de `/mnt/Datos` quedará oculto detrás del montaje. Cuando desmontemos el bind mount, volveremos a ver esos archivos.

Desmontar la carpeta

Para eliminar el bind mount utilizamos:

```
sudo umount /mnt/Datos
```

Los archivos originales de `/home/nipegun/Datos` permanecerán intactos. Podemos comprobar posteriormente que el punto de montaje ya no existe con:

```
findmnt /mnt/Datos
```

Hacer permanente el bind mount

El comando `mount –bind` crea un montaje temporal. Después de reiniciar Debian desaparecerá. Si queremos que se cree

automáticamente durante el arranque podemos declararlo en `/etc/fstab`. Editamos:

```
sudo nano /etc/fstab
```

y añadimos:

```
/home/nipegun/Datos  /mnt/Datos  none  bind  0  0
```

Podemos probar la configuración sin reiniciar ejecutando:

```
sudo mount -a
```

Conviene comprobar después que el montaje se ha realizado correctamente:

```
findmnt /mnt/Datos
```

De esta forma Debian realizará automáticamente el bind mount durante el arranque.

Hacer el montaje sin sudo

Normalmente un usuario corriente no puede ejecutar directamente:

```
mount --bind /home/nipegun/Datos /home/nipegun/Montado
```

porque crear montajes requiere el privilegio `CAP_SYS_ADMIN`. Una posibilidad es autorizar previamente ese montaje mediante `/etc/fstab`. Por ejemplo:

```
/home/nipegun/Datos  /home/nipegun/Montado  none  bind,user  0  0
```

La modificación de `/etc/fstab` tendremos que realizarla inicialmente como root, pero una vez configurada podremos montar el directorio como usuario normal mediante:

```
mount /home/nipegun/Montado
```

Y desmontarlo mediante:

```
umount /home/nipegun/Montado
```

La opción `user` permite que un usuario no privilegiado realice específicamente ese montaje definido en `/etc/fstab`, sin concederle permiso general para realizar cualquier montaje del sistema. Además, el usuario que realiza el montaje será normalmente quien pueda desmontarlo.

Si queremos permitir que distintos usuarios puedan realizar o desmontar ese montaje podemos utilizar `users` en lugar de `user`:

```
/home/nipegun/Datos  /home/nipegun/Montado  none  bind,users  0  0
```

Debemos tener en cuenta que las opciones `user` y `users` aplican también restricciones de seguridad por defecto, como `nosuid`, `nodev` y `noexec`, salvo que configuremos explícitamente otras opciones cuando sean necesarias.

Realizar un bind mount dentro de un namespace de usuario

Linux también permite crear espacios de nombres aislados. En sistemas donde los **unprivileged user namespaces** estén permitidos podemos crear un namespace en el que nuestro usuario disponga de capacidades administrativas únicamente dentro de ese entorno. Podemos iniciar uno mediante:

```
unshare --user --map-root-user --mount
```

Dentro del nuevo namespace apareceremos como root desde el punto de vista de ese namespace:

```
id
```

Y, si la configuración y las restricciones del kernel lo permiten, podremos realizar determinados montajes dentro de ese namespace:

```
mount --bind /home/nipegun/Datos /home/nipegun/Montado
```

La diferencia fundamental es que **el montaje sólo existirá dentro de ese mount namespace**. Otra terminal normal de nuestro sistema no verá necesariamente ese montaje y, cuando finalice el namespace, el montaje desaparecerá. Esta técnica resulta especialmente interesante para entornos aislados, sandboxes, procesos de compilación y contenedores.

Utilizar bindfs cuando necesitamos modificar la representación de los permisos

Si además de montar una carpeta dentro de otra queremos que desde la nueva ruta aparezcan propietarios o permisos diferentes, un bind mount convencional no es suficiente. Podemos utilizar `bindfs`, que implementa este comportamiento mediante FUSE. En Debian podemos instalarlo mediante:

```
sudo apt update
sudo apt install bindfs
```

Por ejemplo, podemos montar:

```
bindfs /home/nipegun/Datos /home/nipegun/Montado
```

Al utilizar FUSE podemos realizar determinados montajes desde espacio de usuario sin necesitar los privilegios que requiere un bind mount convencional. Además, `bindfs` permite transformar propietarios, grupos y permisos visibles desde el punto de montaje, lo que lo convierte en una alternativa interesante cuando necesitamos presentar los mismos archivos de forma diferente a determinadas aplicaciones o usuarios.

Bind mount frente a enlace simbólico

Para muchos casos sencillos también podríamos utilizar un enlace simbólico:

```
ln -s /home/nipegun/Datos /home/nipegun/Montado
```

Sin embargo, conceptualmente no funcionan de la misma forma.

- Un enlace simbólico es simplemente un archivo especial que contiene una referencia a otra ruta.
- Un bind mount, en cambio, hace que el kernel presente directamente un directorio existente en otro punto del sistema de archivos.

Esto significa que determinadas aplicaciones, servicios, chroots, sandboxes o mecanismos que inspeccionan puntos de montaje pueden comportarse de manera diferente ante un enlace simbólico y ante un bind mount.

Podemos comprobar claramente la diferencia utilizando `findmnt`. Un enlace simbólico no aparecerá como un sistema de archivos montado, mientras que un bind mount sí:

```
findmnt /home/nipegun/Montado
```

Cuando simplemente necesitamos acceder cómodamente a otra ruta, un enlace simbólico suele ser suficiente y no requiere privilegios especiales. Cuando necesitamos que otra ruta del árbol de directorios se comporte realmente como un punto de montaje del mismo directorio, utilizaremos un bind mount.