Niveau : Utilisateur	Numéro de la Leçon	Dernière Modification
1/4	<progrecss 1/5 style=inline />	2020/01/30 03:28

Système de Fichiers

Le système de fichiers de Linux est organisé autour d'une arborescence unique ayant un point de départ appelé la **racine**, représenté par le caractère **/**. En dessous de cette racine se trouvent des répertoires contenant fichiers et sous-répertoires. L'organisation des répertoires est conforme à un standard, appelé le **Linux File Hierarchy System**.

Linux File Hierarchy System

- **/bin** : est une abréviation de ***bin****ary* ou binaires. Il contient des programmes tels ls.
- **/boot** : contient les fichiers nécessaires au démarrage du système.
- **/dev** : contient les nœuds utilisés pour accéder à tout type de matériel tel /dev/fd0 pour le lecteur de disquette. C'est le binaire *udev* qui se charge de créer et supprimer d'une manière dynamique les nœuds.
- **/etc** : contient des fichiers de configuration tels passwd pour les mots de passe et fstab qui est la liste des systèmes de fichiers à monter lors du démarrage du système.
- **/home** : contient les répertoires de chaque utilisateur sauf l'utilisateur root.
- **/lib** : contient les bibliothèques communes utilisées par les programmes ainsi que les modules.
- **/lost+found** : contient des fragments de fichiers endommagés et retrouvés par la commande *fsck*.
- **/media** : contient des répertoires pour chaque système de fichiers monté (accessible au système linux) tels floppy, cdrom etc.
- **/mnt** : contient des répertoires pour chaque système de fichiers monté temporairement par root.
- **/opt** : contient des applications optionnelles.
- **/proc** : contient un système de fichiers virtuel qui extrait de la mémoire les informations en cours de traitement. Le contenu des fichiers est créé dynamiquement lors de la consultation. Seul root peut consulter la totalité des informations dans le répertoire /proc.
- **/root** : le home de root, l'administrateur système
- **/sbin** : contient des binaires, donc programmes, pour l'administration du système local.
- **/selinux** : contient des fichiers propres à l'implémentation de SELINUX.
- **/srv** : contient des données pour les **s**e**rv**ices hébergés par le système tels ftp, bases de données, web etc.
- **/sys** : contient un système de fichiers virtuel dont le rôle est de décrire le matériel pour udev.

- **/tmp** : stocke des fichiers temporaires créés par des programmes.
- **/usr** : contient des commandes des utilisateurs dans /usr/bin, les HOWTO dans /usr/share/doc, les manuels dans /usr/share/man ainsi que d'autres entrées majeures.
- **/var** : contient des fichiers de taille variable.

Il existe trois types majeurs de fichier sous le système Linux :

- les fichiers normaux (ordinary files)
- les répertoires (directories)
- les fichiers spéciaux (special files ou Devices)

Le fichiers normaux sont des fichiers textes, des tableaux ou des exécutables.

La longueur du nom de fichier est limité à 255 caractères.

Il y a une distinction entres les majuscules et le minuscules.

Si le nom d'un fichier commence par un **.**, le fichier devient caché.

La commande mount

Pour que Linux soit informé de la présence d'un système de fichiers, ce système doit être monté. Pour monter un système de fichiers, on utilise la commande **mount** :

```
# mount /dev/<fichier_spécial> /mnt/<répertoire_cible>
```

ou **/dev/<fichier_spécial>** est le périphérique à monter et **/mnt/<répertoire_cible>** est le répertoire qui servira comme «fenêtre» pour visionner le contenu du système de fichiers. Ce répertoire doit impérativement exister avant d'essayer de monter le système de fichiers.

<note> Connectez-vous à votre machine virtuelle Debian en tant que **trainee** avec le mot de passe **trainee**. Ouvrez un terminal via les menus **Applications > Accessoires > Terminal**. Tapez la commande **su -** et appuyez sur la touche ↵ Entrée. Indiquez le mot de passe **fenestros**. Vous êtes maintenant connecté en tant que l'administrateur **root** et vous pouvez reproduire les exemples qui suivent. </note>

Dans le cas où la commande **mount** est utilisée sans options, le système retourne une liste de tous les systèmes de fichiers actuellement montés :

```
root@debian:/# mount
/dev/sda1 on / type ext3 (rw,errors=remount-ro,acl)
tmpfs on /lib/init/rw type tmpfs (rw,nosuid,mode=0755)
proc on /proc type proc (rw,noexec,nosuid,nodev)
sysfs on /sys type sysfs (rw,noexec,nosuid,nodev)
udev on /dev type tmpfs (rw,mode=0755)
tmpfs on /dev/shm type tmpfs (rw,nosuid,nodev)
devpts on /dev/pts type devpts (rw,noexec,nosuid,gid=5,mode=620)
none on /selinux type selinuxfs (rw,relatime)
fusectl on /sys/fs/fuse/connections type fusectl (rw)
binfmt_misc on /proc/sys/fs/binfmt_misc type binfmt_misc (rw,noexec,nosuid,nodev)
```

Dans le cas où la commande **mount** est utilisée avec l'option **–a**, tous les systèmes de fichiers mentionnés dans un fichier spécial dénommé **/etc/fstab** seront montés en même temps :

```
root@debian:/# cat /etc/fstab
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point>   <type>  <options>       <dump>  <pass>
proc            /proc           proc    defaults        0       0
# / was on /dev/sda1 during installation
UUID=a42a1ddd-14bc-4dde-a537-e6c1b984a782 /               ext3    errors=remount-ro,acl 0       1
# swap was on /dev/sda5 during installation
UUID=e21d8931-21ca-4ab3-9fbb-bd71657b312e none            swap    sw              0       0
/dev/scd0       /media/cdrom0   udf,iso9660 user,noauto     0       0
```

Options de la commande

Les options de la commande **mount** sont :

```
trainee@debian:~$ mount --help
Usage: mount -V                : print version
       mount -h                : print this help
       mount                   : list mounted filesystems
       mount -l                : idem, including volume labels
So far the informational part. Next the mounting.
The command is `mount [-t fstype] something somewhere'.
Details found in /etc/fstab may be omitted.
       mount -a [-t|-O] ...    : mount all stuff from /etc/fstab
       mount device            : mount device at the known place
       mount directory         : mount known device here
       mount -t type dev dir   : ordinary mount command
Note that one does not really mount a device, one mounts
a filesystem (of the given type) found on the device.
One can also mount an already visible directory tree elsewhere:
       mount --bind olddir newdir
or move a subtree:
       mount --move olddir newdir
One can change the type of mount containing the directory dir:
       mount --make-shared dir
       mount --make-slave dir
       mount --make-private dir
       mount --make-unbindable dir
One can change the type of all the mounts in a mount subtree
containing the directory dir:
       mount --make-rshared dir
       mount --make-rslave dir
       mount --make-rprivate dir
       mount --make-runbindable dir
```

```
A device can be given by name, say /dev/hda1 or /dev/cdrom,
or by label, using  -L label  or by uuid, using  -U uuid .
Other options: [-nfFrsvw] [-o options] [-p passwdfd].
For many more details, say  man 8 mount .
```

La commande umount

Pour démonter un système de fichiers, on utilise la commande umount :

```
# umount /mnt/<répertoire_cible>
```

Options de la commande

Les options de la commande **umount** sont :

```
trainee@debian:~$ umount --help
Usage: umount -h | -V
       umount -a [-d] [-f] [-r] [-n] [-v] [-t vfstypes] [-O opts]
       umount [-d] [-f] [-r] [-n] [-v] special | node...
```

Systèmes de fichiers Unix

Chaque partition sous un système Unix peut héberger une des structures suivantes :

- superbloc
- inode
- bloc de données
- blocs d'indirections

Superbloc

Le superbloc contient :

- la taille des blocs
- la taille du système de fichiers
- le nombre de montages effectués pour ce système de fichiers
- un pointeur vers la racine du système de fichiers
- les pointeurs vers la liste des inodes libres
- les pointeurs vers la liste des blocs de données libres

Le Superbloc est dupliqué tous les 8 ou 16Mo sur le système de fichiers.

Pour visualiser l'emplacement du Superbloc primaire et ses sauvegardes, utilisez la commande suivante :

```
root@debian:~# dumpe2fs /dev/sda1 | grep -i superbloc
dumpe2fs 1.41.12 (17-May-2010)
 superbloc Primaire à 0, Descripteurs de groupes à 1-1
 superbloc Secours à 32768, Descripteurs de groupes à 32769-32769
 superbloc Secours à 98304, Descripteurs de groupes à 98305-98305
 superbloc Secours à 163840, Descripteurs de groupes à 163841-163841
 superbloc Secours à 229376, Descripteurs de groupes à 229377-229377
 superbloc Secours à 294912, Descripteurs de groupes à 294913-294913
 superbloc Secours à 819200, Descripteurs de groupes à 819201-819201
 superbloc Secours à 884736, Descripteurs de groupes à 884737-884737
```

Pour réparer un système de fichiers en restaurant un Superbloc, utilisez la commande suivante :

```
# e2fsck -f -b 32768 /dev/sda1 [Enter]
```

Inodes

Chaque fichier est représenté par un **inode**. L'inode contient :

- le type de fichier, soit **-**, **d**, **l**, **b**, **c**, **p**, **s**
- les droits d'accès, par exemple **rwx rw- r–**
- le nombre de liens physiques soit le nombre de noms
- l'UID du créateur ou l'UID affecté par la commande **chown** s'il y a eu une modification
- le GID du processus créateur ou le GID affecté par la commande **chgrp**
- la taille du fichier en octets
- la date de dernière modification de l'inode, soit le **ctime**
- la date de dernière modification du fichier, soit le **mtime**
- la date du dernier accès, soit le **atime**
- les adresses qui pointent vers les blocs de données du fichier

Graphiquement, on peut schématiser cette organisation de la façon suivante :

Pour mieux comprendre, tapez la commande suivante :

```
# ls -ld /dev/console /dev/initctl /dev/loop0 /etc /etc/passwd [Entrée]
```

Vous obtiendrez un résultat similaire à la suivante :

```
root@debian:/# ls -ld /dev/console /dev/initctl /dev/loop0 /etc /etc/passwd
crw-------.   1 root root  5, 1 19 oct.  07:21 /dev/console
prw-------.   1 root root     0 19 oct.  07:21 /dev/initctl
brw-rw----.   1 root disk  7, 0 19 oct.  07:21 /dev/loop0
drwxr-xr-x. 128 root root 12288 20 oct.  14:36 /etc
-rw-r--r--.   1 root root  1298 27 avril 16:36 /etc/passwd
```

Le premier caractère de chaque ligne peut être un des suivants :

- **-** - un fichier
- **d** - un répertoire

- **l** - un lien symbolique
- **b** - un périphérique du type bloc
- **c** - un périphérique du type caractère
- **p** - un tube nommé pour la communication entre processus
- **s** - un socket dans un contexte réseau

Pour visualiser le numéro d'inode, utilisez l'option **-i** :

```
root@debian:/# ls -ldi /dev/console /dev/initctl /dev/loop0 /etc /etc/passwd
  1300 crw-------.   1 root root  5, 1 19 oct.  07:21 /dev/console
  3496 prw-------.   1 root root     0 19 oct.  07:21 /dev/initctl
  3573 brw-rw----.   1 root disk  7, 0 19 oct.  07:21 /dev/loop0
194689 drwxr-xr-x. 128 root root 12288 20 oct.  14:36 /etc
201193 -rw-r--r--.   1 root root  1298 27 avril 16:36 /etc/passwd
```

Blocs de données

Les données sont stockées dans des blocs de données. Dans le cas d'un répertoire, le bloc de données contient une table qui référence les inodes et les noms des fichiers dans le répertoire. Cette table s'appelle une **table catalogue**.

Le nom d'un fichier n'est pas stocké dans l'inode mais dans une **table catalogue**. Cette particularité nous permet de donner deux noms différents au même fichier. Pour ajouter un nouveau nom à un fichier, il convient de créer un **lien physique**.

Liens Physiques

Un lien physique se crée en utilisant la commande suivante :

- ln nom_du_fichier nom_supplémentaire

Pour illustrer ce point, tapez la ligne de commande suivante :

```
# cd /tmp; mkdir inode; cd inode; touch fichier1; ls -ali [Entrée]
```

Vous obtiendrez un résultat similaire à celui-ci :

```
root@debian:/# cd /tmp; mkdir inode; cd inode; touch fichier1; ls -ali
total 8
61576 drwxr-xr-x.  2 root root 4096 20 oct.  14:44 .
24337 drwxrwxrwt. 20 root root 4096 20 oct.  14:44 ..
61577 -rw-r--r--.  1 root root    0 20 oct.  14:44 fichier1
```

Notez bien le numéro de l'inode du fichier **fichier1**. Notez aussi que le numéro dans le troisième champs de la ligne de fichier1 a la valeur **1** :

61577 -rw-r–r–. 1 root root 0 20 oct. 14:44 fichier1

Créez maintenant un lien physique :

```
# ln fichier1 fichier2 [Entrée]
```

Visualisez le résultat :

```
root@debian:/tmp/inode# ln fichier1 fichier2
root@debian:/tmp/inode# ls -ali
total 8
61576 drwxr-xr-x.  2 root root 4096 20 oct.  14:44 .
24337 drwxrwxrwt. 20 root root 4096 20 oct.  14:44 ..
61577 -rw-r--r--.  2 root root    0 20 oct.  14:44 fichier1
61577 -rw-r--r--.  2 root root    0 20 oct.  14:44 fichier2
```

Notez les deux lignes suivantes :

61577 -rw-r–r–. 2 root root 0 20 oct. 14:44 fichier1
61577 -rw-r–r–. 2 root root 0 20 oct. 14:44 fichier2

Les deux fichiers, fichier1 et fichier2, sont référencés par le même inode. Le nombre de liens est donc augmenté de 1 (le numéro dans le troisième champs).

<note important> Un lien physique ne peut être créé que dans le cas où les deux fichiers se trouvent dans le même filesystem et que le fichier source existe. </note>

Liens Symboliques

Un lien symbolique est un **raccourci** vers un autre fichier ou répertoire. Un lien symbolique se crée en utilisant la commande suivante :

- ln -s nom_du_fichier nom_raccourci

Pour illustrer ce point, tapez la ligne de commande suivante :

```
# ln -s fichier1 fichier3 [Entrée]
```

Vous obtiendrez un résultat similaire à celui-ci :

```
root@debian:/tmp/inode# ln -s fichier1 fichier3
root@debian:/tmp/inode# ls -ali
total 8
61576 drwxr-xr-x.  2 root root 4096 20 oct.  14:46 .
24337 drwxrwxrwt. 20 root root 4096 20 oct.  14:44 ..
61577 -rw-r--r--.  2 root root    0 20 oct.  14:44 fichier1
61577 -rw-r--r--.  2 root root    0 20 oct.  14:44 fichier2
61578 lrwxrwxrwx.  1 root root    8 20 oct.  14:46 fichier3 -> fichier1
```

Notez que le lien symbolique est référencé par un autre inode. Le lien symbolique pointe vers le fichier1.

<note important> Un lien symbolique peut être crée même dans le cas où les deux fichiers se trouvent dans deux filesystems différents et même dans le cas où le fichier source n'existe pas. </note>

~~DISCUSSION:off~~

Donner votre Avis

{(rater>id=debian_6_l101|name=cette page|type=rate|trace=user|tracedetails=1)}