Version : **2024.01**

Dernière mise-à-jour : 2024/11/15 07:59

RH12401 - Système de Fichiers

Contenu du Module

- **RH12401 - Système de Fichiers**
  - Contenu du Module
  - LAB #1 - Linux File Hierarchy System
    - 1.1 - Types de Fichiers
    - 1.2 - La Commande mount
    - 1.3 - La Commande umount
    - 1.4 - Le Fichier /etc/fstab
      - Options de Montage
  - LAB #2 - Système de Fichiers Unix
    - 2.1 - Superbloc
    - 2.2 - Inodes
    - 2.3 - Blocs de données
    - 2.4 - Liens Physiques
    - 2.5 - Liens Symboliques

LAB #1 - Linux File Hierarchy System

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**.

```
[trainee@redhat9 ~]$ cd /
```

```
[trainee@redhat9 /]$ ls -l
total 28
dr-xr-xr-x.   2 root root    6 Aug 10  2021 afs
lrwxrwxrwx.   1 root root    7 Aug 10  2021 bin -> usr/bin
dr-xr-xr-x.   5 root root 4096 Sep 25 12:30 boot
drwxr-xr-x.  20 root root 3320 Sep 25 12:44 dev
drwxr-xr-x. 133 root root 8192 Sep 25 12:44 etc
drwxr-xr-x.   3 root root   21 Oct 19  2023 home
lrwxrwxrwx.   1 root root    7 Aug 10  2021 lib -> usr/lib
lrwxrwxrwx.   1 root root    9 Aug 10  2021 lib64 -> usr/lib64
drwxr-xr-x.   2 root root    6 Aug 10  2021 media
drwxr-xr-x.   2 root root    6 Aug 10  2021 mnt
drwxr-xr-x.   2 root root    6 Aug 10  2021 opt
dr-xr-xr-x. 242 root root    0 Sep 25 12:44 proc
dr-xr-x---.   4 root root 4096 Oct 19  2023 root
drwxr-xr-x.  44 root root 1160 Sep 25 12:44 run
lrwxrwxrwx.   1 root root    8 Aug 10  2021 sbin -> usr/sbin
drwxr-xr-x.   2 root root    6 Aug 10  2021 srv
dr-xr-xr-x.  13 root root    0 Sep 25 12:44 sys
drwxrwxrwt.  15 root root 4096 Sep 25 12:48 tmp
drwxr-xr-x.  12 root root  144 Oct 19  2023 usr
drwxr-xr-x.  20 root root 4096 Oct 19  2023 var
```

- **/afs** : Andrew File System (AFS) est un système de fichiers distribué qui utilise un ensemble de serveurs de confiance pour présenter un espace de noms de fichiers homogène et transparent en termes d'emplacement.
- **/bin** : est une abréviation de ***bin****ary* ou binaires. Il contient des programmes tels ls. Sous RHEL 9 il s'agit d'un lien symbolique qui pointe vers /usr/bin.
- **/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 32 bits communes utilisées par les programmes ainsi que les modules. Sous RHEL 9 il s'agit d'un lien symbolique

qui pointe vers /usr/lib.
- **/lib64** : contient les bibliothèques 64 bits communes utilisées par les programmes ainsi que les modules. Sous RHEL 9 il s'agit d'un lien symbolique qui pointe vers /usr/lib64.
- **/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.
- **/run** : remplace le répertoire /var/run.
- **/sbin** : contient des binaires, donc programmes, pour l'administration du système local. Sous RHEL 9 il s'agit d'un lien symbolique qui pointe vers /usr/sbin.
- **/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.

1.1 - Types de Fichiers

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.

1.2 - 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.

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

```
[trainee@redhat9 /]$ su -
Password: fenestros
[root@redhat9 ~]# mount
proc on /proc type proc (rw,nosuid,nodev,noexec,relatime)
sysfs on /sys type sysfs (rw,nosuid,nodev,noexec,relatime,seclabel)
devtmpfs on /dev type devtmpfs (rw,nosuid,seclabel,size=4096k,nr_inodes=976019,mode=755,inode64)
securityfs on /sys/kernel/security type securityfs (rw,nosuid,nodev,noexec,relatime)
tmpfs on /dev/shm type tmpfs (rw,nosuid,nodev,seclabel,inode64)
devpts on /dev/pts type devpts (rw,nosuid,noexec,relatime,seclabel,gid=5,mode=620,ptmxmode=000)
tmpfs on /run type tmpfs (rw,nosuid,nodev,seclabel,size=1573912k,nr_inodes=819200,mode=755,inode64)
cgroup2 on /sys/fs/cgroup type cgroup2 (rw,nosuid,nodev,noexec,relatime,seclabel,nsdelegate,memory_recursiveprot)
pstore on /sys/fs/pstore type pstore (rw,nosuid,nodev,noexec,relatime,seclabel)
bpf on /sys/fs/bpf type bpf (rw,nosuid,nodev,noexec,relatime,mode=700)
/dev/mapper/rhel-root on / type xfs (rw,relatime,seclabel,attr2,inode64,logbufs=8,logbsize=32k,noquota)
selinuxfs on /sys/fs/selinux type selinuxfs (rw,nosuid,noexec,relatime)
systemd-1 on /proc/sys/fs/binfmt_misc type autofs
(rw,relatime,fd=29,pgrp=1,timeout=0,minproto=5,maxproto=5,direct,pipe_ino=12983)
tracefs on /sys/kernel/tracing type tracefs (rw,nosuid,nodev,noexec,relatime,seclabel)
debugfs on /sys/kernel/debug type debugfs (rw,nosuid,nodev,noexec,relatime,seclabel)
```

```
mqueue on /dev/mqueue type mqueue (rw,nosuid,nodev,noexec,relatime,seclabel)
hugetlbfs on /dev/hugepages type hugetlbfs (rw,relatime,seclabel,pagesize=2M)
none on /run/credentials/systemd-tmpfiles-setup-dev.service type ramfs
(ro,nosuid,nodev,noexec,relatime,seclabel,mode=700)
fusectl on /sys/fs/fuse/connections type fusectl (rw,nosuid,nodev,noexec,relatime)
configfs on /sys/kernel/config type configfs (rw,nosuid,nodev,noexec,relatime)
none on /run/credentials/systemd-sysctl.service type ramfs (ro,nosuid,nodev,noexec,relatime,seclabel,mode=700)
/dev/sda1 on /boot type xfs (rw,relatime,seclabel,attr2,inode64,logbufs=8,logbsize=32k,noquota)
none on /run/credentials/systemd-tmpfiles-setup.service type ramfs
(ro,nosuid,nodev,noexec,relatime,seclabel,mode=700)
tmpfs on /run/user/42 type tmpfs
(rw,nosuid,nodev,relatime,seclabel,size=786956k,nr_inodes=196739,mode=700,uid=42,gid=42,inode64)
tmpfs on /run/user/1000 type tmpfs
(rw,nosuid,nodev,relatime,seclabel,size=786956k,nr_inodes=196739,mode=700,uid=1000,gid=1000,inode64)
```

Cette information est stockée dans le fichier **/etc/mtab** :

```
[root@redhat9 ~]# cat /etc/mtab
proc /proc proc rw,nosuid,nodev,noexec,relatime 0 0
sysfs /sys sysfs rw,seclabel,nosuid,nodev,noexec,relatime 0 0
devtmpfs /dev devtmpfs rw,seclabel,nosuid,size=4096k,nr_inodes=976019,mode=755,inode64 0 0
securityfs /sys/kernel/security securityfs rw,nosuid,nodev,noexec,relatime 0 0
tmpfs /dev/shm tmpfs rw,seclabel,nosuid,nodev,inode64 0 0
devpts /dev/pts devpts rw,seclabel,nosuid,noexec,relatime,gid=5,mode=620,ptmxmode=000 0 0
tmpfs /run tmpfs rw,seclabel,nosuid,nodev,size=1573912k,nr_inodes=819200,mode=755,inode64 0 0
cgroup2 /sys/fs/cgroup cgroup2 rw,seclabel,nosuid,nodev,noexec,relatime,nsdelegate,memory_recursiveprot 0 0
pstore /sys/fs/pstore pstore rw,seclabel,nosuid,nodev,noexec,relatime 0 0
bpf /sys/fs/bpf bpf rw,nosuid,nodev,noexec,relatime,mode=700 0 0
/dev/mapper/rhel-root / xfs rw,seclabel,relatime,attr2,inode64,logbufs=8,logbsize=32k,noquota 0 0
selinuxfs /sys/fs/selinux selinuxfs rw,nosuid,noexec,relatime 0 0
systemd-1 /proc/sys/fs/binfmt_misc autofs
rw,relatime,fd=29,pgrp=1,timeout=0,minproto=5,maxproto=5,direct,pipe_ino=12983 0 0
tracefs /sys/kernel/tracing tracefs rw,seclabel,nosuid,nodev,noexec,relatime 0 0
debugfs /sys/kernel/debug debugfs rw,seclabel,nosuid,nodev,noexec,relatime 0 0
```

```
mqueue /dev/mqueue mqueue rw,seclabel,nosuid,nodev,noexec,relatime 0 0
hugetlbfs /dev/hugepages hugetlbfs rw,seclabel,relatime,pagesize=2M 0 0
none /run/credentials/systemd-tmpfiles-setup-dev.service ramfs ro,seclabel,nosuid,nodev,noexec,relatime,mode=700
0 0
fusectl /sys/fs/fuse/connections fusectl rw,nosuid,nodev,noexec,relatime 0 0
configfs /sys/kernel/config configfs rw,nosuid,nodev,noexec,relatime 0 0
none /run/credentials/systemd-sysctl.service ramfs ro,seclabel,nosuid,nodev,noexec,relatime,mode=700 0 0
/dev/sda1 /boot xfs rw,seclabel,relatime,attr2,inode64,logbufs=8,logbsize=32k,noquota 0 0
none /run/credentials/systemd-tmpfiles-setup.service ramfs ro,seclabel,nosuid,nodev,noexec,relatime,mode=700 0 0
tmpfs /run/user/42 tmpfs
rw,seclabel,nosuid,nodev,relatime,size=786956k,nr_inodes=196739,mode=700,uid=42,gid=42,inode64 0 0
tmpfs /run/user/1000 tmpfs
rw,seclabel,nosuid,nodev,relatime,size=786956k,nr_inodes=196739,mode=700,uid=1000,gid=1000,inode64 0 0
```

Important : Notez que le système de fichiers de /dev/sda1 et de /dev/mapper/rhel-root est **xfs**.

1.3 - La Commande umount

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

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

ou

```
# umount /dev/cdrom
```

1.4 - Le Fichier /etc/fstab

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@redhat9 ~]# cat /etc/fstab

#
# /etc/fstab
# Created by anaconda on Thu Oct 19 16:05:58 2023
#
# 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.
#
# After editing this file, run 'systemctl daemon-reload' to update systemd
# units generated from this file.
#
/dev/mapper/rhel-root   /                       xfs     defaults        0 0
UUID=6f6c5bb9-30be-4734-bc23-03fed8541616 /boot                   xfs     defaults        0 0
/dev/mapper/rhel-swap   none                    swap    defaults        0 0
```

Chaque ligne dans ce fichier contient 6 champs :

Champ 1	**Champ 2**	**Champ 3**	**Champ 4**	**Champ 5**	**Champ 6**
Fichier de bloc spécial ou UUID ou système de fichiers virtuel ou une étiquette	Point de montage	Type de système de fichiers	Options séparées par des virgules	Utilisé par *dump* (1 = à dumper, 0 ou vide = à ignorer)	L'ordre de vérification par *fsck* des systèmes de fichiers au moment du démarrage

L'**UUID** (*Universally Unique Identifier*) est une chaîne d'une longueur de 128 bits. Les UUID sont créés automatiquement et d'une manière aléatoire lors de la création du filesystem sur la partition. Ils peuvent être modifiés par l'administrateur.

Options de Montage

Les options de montage les plus importants sont :

Option	Systèmes de Fichier	Description	Valeur par Défaut
defaults	Tous	Egal à rw, suid, dev, exec, auto, nouser, async	S/O
auto/noauto	Tous	Montage automatique/pas de montage automatique lors de l'utilisation de la commande **mount -a**	auto
rw/ro	Tous	Montage en lecture-écriture/lecture seule	rw
suid/nosuid	Tous	Les bits SUID et SGID sont/ne sont pas pris en compte	suid
dev/nodev	Tous	Interprète/n'interprète pas les fichiers spéciaux de périphériques	dev
exec/noexec	Tous	Autorise:n'autorise pas l'exécution des programmes	exec
sync/async	Tous	Montage synchrone/asynchrone	async
user/nouser	Tous	Autorise/n'autorise pas un utilisateur à monter/démonter le système de fichier. Le point de montage est celui spécifié dans le fichier /etc/fstab. Seul l'utilisateur qui a monté le système de fichiers peut le démonter	S/O
users	Tous	Autorise tous les utilisateurs à monter/démonter le système de fichier	S/O
owner	Tous	Autorise le propriétaire du périphérique de le monter	S/O
atime/noatime	Norme POSIX	Inscrit/n'inscrit pas la date d'accès	atime
uid=valeur	Formats non-Linux	Spécifie le n° du propriétaire des fichiers pour les systèmes de fichiers non-Linux	root
gid=valeur	Formats non-Linux	Spécifie le n° du groupe propriétaire	S/O
umask=valeur	Formats non-Linux	Spécifie les permissions (droits d'accès/lecture/écriture)	S/O
dmask=valeur	Formats non-Linux	Spécifie les droits d'usage des dossiers (Obsolète, préférer dir_mode)	umask actuel
dir_mode=valeur	Formats non-Linux	Spécifie les droits d'usage des dossiers	umask actuel
fmask=valeur	Formats non-Linux	Spécifie les droits d'usage des fichiers (Obsolète, préférer file_mode)	umask actuel
file_mode=valeur	Formats non-Linux	Spécifie les droits d'usage des fichiers	umask actuel

LAB #2 - Système de Fichiers Unix

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

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

2.1 - Superbloc

Le superbloc contient :

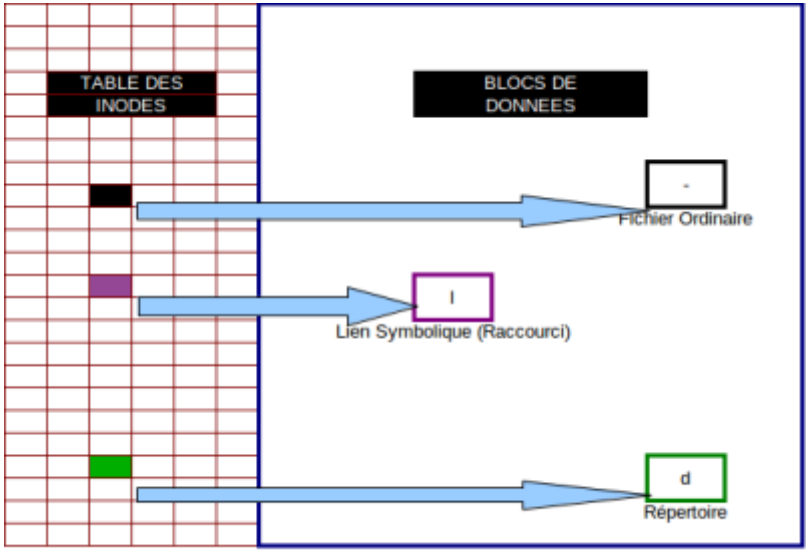
- la taille des blocs
- la taille du système de fichiers
- le nombre de montages effectués pour ce système de fichiers
- un pointeur vers la racine du système de fichiers
- les pointeurs vers la liste des inodes libres
- les pointeurs vers la liste des blocs de données libres

2.2 - 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 :

```
[root@redhat9 ~]# ls -ld /dev/console /dev/sda1 /etc /etc/passwd
crw--w----.   1 root tty  5, 1 Sep 25 12:44 /dev/console
brw-rw----.   1 root disk 8, 1 Sep 25 12:44 /dev/sda1
drwxr-xr-x. 133 root root 8192 Sep 25 12:44 /etc
-rw-r--r--.   1 root root 2109 Oct 19  2023 /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@redhat9 ~]# ls -ldi /dev/console /dev/sda1 /etc /etc/passwd
      12 crw--w----.   1 root tty  5, 1 Sep 25 12:44 /dev/console
     273 brw-rw----.   1 root disk 8, 1 Sep 25 12:44 /dev/sda1
67154049 drwxr-xr-x. 133 root root 8192 Sep 25 12:44 /etc
68914044 -rw-r--r--.   1 root root 2109 Oct 19  2023 /etc/passwd
```

2.3 - 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**.

2.4 - 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 :

```
[root@redhat9 ~]# cd /tmp; mkdir inode; cd inode; touch fichier1; ls -ali
total 4
33625083 drwxr-xr-x.  2 root root   22 Sep 25 13:08 .
33554561 drwxrwxrwt. 16 root root 4096 Sep 25 13:08 ..
33625154 -rw-r--r--.  1 root root    0 Sep 25 13:08 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** :

33625154 -rw-r–r–. 1 root root 0 Sep 25 13:08 fichier1

Créez maintenant un lien physique :

```
[root@redhat9 inode]# ln fichier1 fichier2
[root@redhat9 inode]# ls -ali
total 4
33625083 drwxr-xr-x.  2 root root   38 Sep 25 13:09 .
33554561 drwxrwxrwt. 16 root root 4096 Sep 25 13:08 ..
33625154 -rw-r--r--.  2 root root    0 Sep 25 13:08 fichier1
33625154 -rw-r--r--.  2 root root    0 Sep 25 13:08 fichier2
```

Notez les deux lignes suivantes :

```
33625154 -rw-r--r--.  2 root root    0 Sep 25 13:08 fichier1\\
33625154 -rw-r--r--.  2 root root    0 Sep 25 13:08 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).

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.

2.5 - 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 :

```
[root@redhat9 inode]# ln -s fichier1 fichier3
```

```
[root@redhat9 inode]# ls -ali
total 4
33625083 drwxr-xr-x.  2 root root   54 Sep 25 13:10 .
33554561 drwxrwxrwt. 16 root root 4096 Sep 25 13:08 ..
33625154 -rw-r--r--.  2 root root    0 Sep 25 13:08 fichier1
33625154 -rw-r--r--.  2 root root    0 Sep 25 13:08 fichier2
33625156 lrwxrwxrwx.  1 root root    8 Sep 25 13:10 fichier3 -> fichier1
```

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

Important : Un lien symbolique peut être créé 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.
