Administration Systèmes & Réseaux

Les systèmes de fichiers

May 3, 2015

Houcemeddine HERMASSI

houcemeddine.hermassi@enit.rnu.tn

École Nationale d'Ingénieurs de Carthage ENI-CAR
Université Carthage
Tunisie

Plan de cour

Les types de systèmes de fichiers

Manipulation des systèmes de fichiers

Contrôler le système de fichiers

Les quotas disques

Les droits d'accès

Les types de systèmes de fichiers
ext2

Définition

- Ext2 second extended filesystem est le premier système de fichiers développé spécifiquement pour Linux
- Prévu dès le début pour supporter les rajouts de fonctionnalités, il continue depuis 1993 à être utilisé et amélioré
- Il est rapide et nécessite moins d'écritures que les autres, donc il occasionne moins d'usure des supports de stockage, notamment les disques SSD, les clés USB ou les cartes mémoire.
- Les fichiers peuvent avoir jusqu'à une taille de 2To (2048 Go), tandis qu'une partition peut atteindre 32 To, voire 128 To , selon la taille des blocs et l'architecture.

Les types de systèmes de fichiers
ext3

Définition

- Ext3 third extended filesystem est le successeur de ext2 depuis 1999.
- il est entièrement compatible avec ext2.
- Il est possible d'utiliser un système de fichiers ext3 comme étant ext2, avec les mêmes commandes, les mêmes manipulations.
- Il est possible de transformer en quelques secondes un système ext2 en ext3.
- C'est l'un des systèmes de fichiers de choix pour Linux, et le plus utilisé pour sa souplesse.
- Comme pour ext2, la taille maximale des fichiers est de 2 To, et celle d'une partition de 32 To, suivant les mêmes restrictions.

Les types de systèmes de fichiers
reiserfs

Définition

- reiserfs a été le premier système de fichiers intégré à Linux, avant même ext3.
- Sa force réside, outre dans son journal, dans l'organisation indexée des entrées des répertoires (les tables catalogues contenant les associations inodes/fichiers) et la manipulation des fichiers de petite taille.
- Ses performances sont exceptionnelles en présence de milliers de fichiers, de faible à moyen volume
- Il est redimensionnable à chaud. Il devient plus lent sur des gros fichiers.
- Les fichiers peuvent atteindre 8 To, et les partitions 16 To.
- Les noms de fichiers peuvent avoir 4032 caractères mais sont limités par Linux à 255 caractères (plus précisément par le VFS, Virtual Filesystem Switch).
- reiserfs est moins utilisé malgré ses fortes qualités pour diverses raisons dont la principale est l'impossibilité de convertir un système de fichiers ext2/ext3 en reiserfs et vice versa, et ce à cause de la forte base de machines installées en ext2/ext3.

Les types de systèmes de fichiers
xfs

Définition

- xfs est créé par Silicon Graphics (sgi), il a été porté sous Linux en 2000
- Outre ses capacités de stockages encore inimaginables aujourd'hui,
- il a un système de journalisation très performant et des mécanismes avancés comme la défragmentation en ligne
- Possède la capacité d'effectuer des snapshots (figer létat d'un filesystem à un instant t pour le restaurer plus tard)
- La taille maximale théorique (parce que personne n'en a créé de si gros) des fichiers est de 8 Eo (Exaoctets). Sachant que 1 Eo vaut 1024 Po (Petaoctet) donc 1048576 To, soit, rendu à une unité plus appréhendable, environ 1000 milliards de DVD. La partition peut atteindre 16 Eo, soit la capacité maximale d'un contrleur sur 64 bits. En 32 bits, les tailles sont limitées à 16 To.
- L'utilisation de xfs est encore peu étendue sous Linux peut-être à cause de sa prétendue complexité pour un paramétrage avancé.

Les types de systèmes de fichiers
vfat

Définition

- vfat (Virtual File Allocation Table) est un terme générique regroupant les diverses versions de FAT supportant les noms de fichiers longs (255 caractères) sous Windows.
- La plupart des supports amovibles, disques externes, clefs USB et lecteurs MP3 utilisent un système de fichiers de ce type.
- Un système de fichiers adapté aux petits volumes.
- Un système de fichiers simple à implémenter, idéal pour des lecteurs multimédias.
- Une compatibilité entre diverses plates-formes (Windows, Linux, BSD, MacOS, etc.).

vfat: défauts

vfat souffre cependant de défauts inhérents à sa conception:

- L'ensemble des informations est stockée au sein d'une table unique, y compris le nom du fichier et chaque adresse et longueur des blocs (appelés clusters) composant les données du fichier.
- De ce fait, FAT tente de regrouper les données d'un fichier sur le plus de clusters contigus du support. En cas de nombreuses écritures (ajout, suppression, etc.), le système se retrouve fortement fragmenté.
- Toujours de ce fait, plus le support à une taille importante, plus FAT est lent, car il doit vérifier toute la table FAT pour trouver des clusters disponibles.
- La gestion des noms longs est considérée comme une bidouille par de nombreuses personnes car FAT doit continuer à assurer une compatibilité (encore aujourd'hui) avec les noms courts en 8.3.
- FAT est limité à une taille de fichiers de 4 Go. Les fichiers plus gros (bases de données, archives, images ISO de DVD, etc.) doivent être découpés en conséquence et les logiciels (capture audio / vidéo, sgbd, etc.) doivent gérer cette limitation.

Bloc

- Le bloc est l'unité de base, atomique, de stockage du système de fichiers.
- Un fichier occupe toujours un nombre entier de blocs.
- si un fichier ne contient qu'un seul octet et qu'un bloc a une taille de 4096 octets, 4095 octets sont gâchés
- Il est possible de remplir un système de fichiers avec n fichiers de 1 octets, n représentant le nombre de blocs, alors que le volume total des données n'est que de n octets !

Superbloc

Chaque système de fichiers dispose d'au moins **un superbloc**. Un superbloc est une zone de méta-données qui contient plusieurs informations sur le système de fichiers :

- son type
- sa taille
- son état
- des informations (position) sur les autres zones de méta-données

Récupération des fichiers

Linux tente en premier lieu de lire le superbloc primaire, le premier du disque. Il peut arriver que celui-ci soit corrompu suite à de mauvaises manipulations, un crash, une panne. Dans ce cas les données du disque ne sont plus accessibles. Un système de fichiers Linux dispose de copies (backups) des superblocs à plusieurs endroits du disque. Les écritures sur les divers superblocs étant synchrones, ils sont tous identiques. En dernier recours si l'un deux est supprimé, il peut être recopié depuis un autre

Table d'inodes

Un **inode** est la contraction de index node, c'est-à-dire noeud d'index. C'est une structure de données contenant les informations décrivant et représentant un fichier. Ces informations sont appelées des attributs. Chaque fichier dispose d'un numéro d'inode (i-number). Tous les inodes sont présents au sein d'une table d'inodes. Cette table est généralement découpée en plusieurs morceaux répartis après chaque superbloc. Une table d'inode fait partie des méta-données.
Le contenu d'un inode varie d'un système de fichiers à un autre, mais la norme POSIX impose que chacun d'eux dispose au moins des attributs suivants pour chaque fichier :

- sa taille
- l'identifiant du périphérique le contenant
- son propriétaire
- son groupe
- son numéro d'inode
- son mode (ses droits) d'accès
- sa date de dernière modification d'inode (change time)
- sa date de dernière modification de contenu (modification time)
- sa date de dernier accès (access time)
- un compteur de hard links (liens physiques ou durs, voir plus loin).

Vous pouvez obtenir quelques informations sur un inode avec la commande **stat** :

```
# stat chapitre4.doc"
```

mkfs, syntaxe générale

Les commandes de formatage telles que celles présentes sous Microsoft n'existent pas de manière identique sous Linux. Un formatage de type Microsoft est en fait la création et la vérification d'un système de fichiers sur une partition. La première étape est le remplissage des différents secteurs, blocs, et clusters de zéros (ou d'un autre motif binaire) avec une vérification du support, et la seconde l'écriture d'un système de fichiers. Cette seule dernière opération suffit à la création d'un système de fichiers vierge sur le disque ou la partition.
La commande pour créer un système de fichiers est **mkfs**. **mkfs** appelle d'autres programmes en fonction du type de système de fichiers sélectionné.

```
mkfs -t typefs options périphérique"
```

C'est `typefs"` qui détermine le type de système de fichiers et donc le programme appelé. Il existe un programme par type de système de fichiers :

- **ext2** : mkfs.ext2
- **ext3** : mkfs.ext3
- **reiserfs** : mkfs.reiserfs
- **vfat** : mkfs.vfat (pour tous les formats FAT, mais mkfs.msdos existe)
- **ntfs** : mkfs.ntfs

Exemple-1: en ext2

Vous allez créer un système de fichiers de type **ext2** sur la première partition précédemment créée, à savoir **sdb1**. Voici la commande de base:

```
# mkfs -t ext2 /dev/sdb1"
```

Exemple-2: ext2 vers ext3

Ext3 est un système de fichiers ext2 auquel on a rajouté un journal. Vous pouvez convertir un système de fichiers ext2 en ext3 en utilisant `tune2fs".`

```
# tune2fs -j /dev/sdb1"
```

Exemple-3: Label

Vous pouvez afficher et changer le label du système de fichiers en tapant `e2label"`.

```
# e2label /dev/sdb1"
DATA"
# e2label /dev/sdb1 OLDDATA"
# e2label /dev/sdb1" OLDDATA"
```

mount

La commande `mount"` permet d'accéder aux périphériques de type blocs (les partitions) sur lesquels un système de fichiers existe. La commande mount attache le répertoire racine du système de fichiers à un répertoire pré-existant appelé point de montage (mountpoint).

```
mount -t typefs -o options périphérique point_de_montage"
```

La commande `mount"` utilisée seule donne tous les détails sur les systèmes de fichiers actuellement montés (périphériques, système de fichiers, point de montage, options)
Les mêmes informations sont accessibles en affichant le contenu du fichier **/etc/mtab**.

Montage par périphérique

La partition **sdb1** disposant de nouveau d'un système de fichiers **ext3**, la commande suivante rattache la racine du système de fichiers contenu dans **sdb1** au répertoire **/mnt/DATA**.

```
# mount -t ext3 /dev/sdb1 /mnt/DATA"
```

Montage par label

En cas de réorganisation des disques (déplacement dans une cha SCSI par exemple), l'ordonnancement des périphériques est modifié. L'utilisation des noms de périphériques oblige dans ce cas à modifier le fichier **/etc/fstab** à chaque modification. Ce n'est pas le cas avec les labels. Utilisez le paramètre **-L** de **mount**, suivi du nom du volume.

```
# mount -t ext3 -L DATA /mnt/DATA"
```

La liste des labels actuellement connus de Linux peut être obtenue en listant le répertoire **/dev/disk/by-label**. Notez que le label est un lien symbolique vers le fichier périphérique correspondant :

```
# ls -l /dev/disk/by-label/"
```

Montage par UUID

Chaque système de fichiers dispose d'un identifiant unique appelé **UUID** : Universal Unique Identifier, généralement un nombre aléatoire codé sur assez de bits pour que sur un ou plusieurs systèmes donnés, ils soient tous différents. Ainsi si le disque change de position logique, l'UUID ne varie pas et mount retrouve le système de fichiers, alors qu'il est théoriquement bien plus possible que deux systèmes de fichiers portent le même label.

Remarque: Il existe plusieurs moyens de connaître l'UUID d'une partition. Si udev est utilisé sur votre Linux, alors la commande `vol_id"` est probablement disponible. Il se peut qu'elle ne soit pas présente dans le path, par exemple sur une `/lib/udev"`.

```
# ./vol_id -u /dev/sdb1"
67f6e4b8-635c-4103-9a81-877fb7db29fe"
```

Si votre système de fichiers est en ext2 ou ext3 la commande `dumpe2fs"` retourne énormément d'informations dont l'UUID :

```
# dumpe2fs -h /dev/sdb1 | grep UUID"
dumpe2fs 1.40.2 (12-Jul-2007)"
Filesystem UUID: 67f6e4b8-635c-4103-9a81-877fb7db29fe"
```

Pour monter un système de fichiers par UUID, utilisez le paramètre -U de mount :

```
# mount -t ext3 -U 67f6e4b8-635c-4103-9a81-877fb7db29fe /mnt/DATA"
# mount"
/dev/sdb1 on /mnt/DATA type ext3 (rw)"
```

Remonter un système de fichier

Vous n'êtes pas obligé de démonter puis de remonter un système de fichiers si vous modifiez une option. Si vous modifiez une option de montage du système de fichiers (via le paramètre -o) vous pouvez passer l'option remount pour que la modification soit prise tout de suite en compte. Ne retapez pas la ligne de commande complète, mais seulement le périphérique ou le point de montage. Dans l'exemple suivant le système de fichiers est remonté en lecture seule :

```
# mount -o ro,remount /mnt/DATA"
# mount"
..."
/dev/sdb1 on /mnt/DATA type ext3 (ro)"
```

Options de montage

Chaque système de fichiers accepte un certain nombre d'options de montage qui peuvent être spécifiées après le paramètre -o de mount. Les options sont séparées par des virgules. Sauf indication contraire les options suivantes fonctionnent avec ext2 et ext3.

Options	Rôle
defaults	reprend les options rw, suid, dev, exec, auto, nouser, et async.
sync/async	Active ou désactive les écritures synchrones.
exec/noexex	Permet l'exécution/ou non des fichiers binaires sur le support.
noatime	Évite la mise à jour de l'horodatage à chaque accès à un fichier.
auto/noauto	montage automatique/ne peut être monté que explicitement.
user/nouser	Tout utilisateur peut monter le système de fichiers.
remount	Remontage du système de fichiers
ro/rw	Montage en lecture seule ou lecture et écriture.
dev/nodev	Interpréter/Ne pas interpréter les fichiers spéciaux.
noload	Pour ext3, ne charge pas le journal.
acl	Permet l'utilisation des Access Control Lists.
user_xattr	Pour ext3 et xfs, accepte les attributs étendus sur les fichiers,
umask	Pour FAT/NTFS, applique un autre masque global que celui par défaut
dmask=/fmask=	FAT/NTFS, différencie les masques pour les répertoires et les fichiers.
uid=/gid=	FAT/NTFS applique un utilisateur ou un groupe par défaut

unmount

La commande umount détache le système de fichiers du point de montage.

```
# umount /mnt/DATA"
```

Si un ou plusieurs fichiers du système de fichiers à démonter sont encore en cours d'utilisation, alors umount ne marchera pas. Vous devez vous assurer qu'aucun processus n'accède au système de fichiers.

```
# umount /mnt/DATA"
umount: /mnt/DATA: périphérique occupé"
```

La commande **lsof** vous aide à déterminer quel processus est actuellement en train d'utiliser un fichier du point de montage. Ici c'est le shell bash lancé par l'utilisateur seb qui y est présent (probablement que le répertoire courant est /**mnt**/**DATA**).

```
# lsof /mnt/DATA"
COMMAND PID USER FD TYPE DEVICE SIZE NODE NAME"
bash 5366 seb cwd DIR 8,17 4096 2 /mnt/DATA"
```

De manière très violente vous pouvez forcer l'arrêt des processus accédant au point de montage avec fuser. Il est fort probable que l'utilisateur concerné n'apprécie pas du tout (dans le cas présenté ici son shell sera arrêté et il sera déconnecté).

```
# fuser -km /mnt/DATA"
```

/etc/fstab

Le fichier **/etc/fstab** contient une configuration statique des différents montages des systèmes de fichiers. Il est appelé à chaque démarrage du système car c'est ici qu'on indique les périphériques et leurs points de montage. Il contient six champs.

`périphérique point_de_montage typefs options dump fsck"`

Champ	Description
périphérique	Le périphérique à monter. (/dev/hda1 par exemple),
point de montage	Le répertoire d'accès au système de fichiers monté.
typefs	Le type (ext2, etx3, reiser, vfat, etc.) du système de fichiers.
options	Les options, séparées par des virgules, vues précédemment.
dump	Fréquence de dump pour les outils de dump ou de sauvegarde.
fsck	Fréquence de vérification du système de fichiers. 0=ignorer. 1=en premier, 2 en second, etc.

/etc/fstab

Montage au boot: Lors de la séquence de démarrage le fichier `/etc/fstab"` est balayé par l'un des scripts, presque au tout début du boot, entre le chargement du noyau et le démarrage des services. Tous les systèmes de fichiers ne possédant pas noauto comme option sont automatiquement montés (le auto est implicite). Le premier à l'être est le système de fichiers racine /. Puis viennent ensuite le swap et les autres systèmes de fichiers s'ils sont spécifiés (ex : /home, /usr, etc.) ainsi que les systèmes de fichiers virtuels /proc, /sys , /dev/pts, etc.

/etc/fstab

Montage manuel: Le contenu de /**etc**/**fstab** peut être utilisé après l'initialisation du système pour monter et démonter ponctuellement les systèmes de fichiers qui n'ont pas par exemple l'option noauto, ou les supports de masse comme les lecteurs CD/DVD. Dans ce cas vous utilisez simplement les labels, les points de montage ou le périphérique sans avoir à réécrire toute la ligne de commande. Mount va chercher ses renseignements dans /**etc**/**fstab**.

```
mount /home"
mount -L /u01"
mount LABEL=/boot"
mount /dev/hda5"
```

/etc/fstab

Tout monter: Si vous avez effectué des modifications importantes dans la **fstab** comme le rajout de plusieurs nouveaux points de montage, vous pouvez, au lieu de monter chaque système de fichiers un par un, tous les monter d'un coup avec le paramètre **-a** de mount :

```
# mount -a"
```

Cas des CD et des images ISO

Les CD-Rom, DVD-Roms et autres supports de ce type se montent comme n'importe quel disque. Les CD-Roms et certains DVD-Roms utilisent le système de fichiers iso9660.

```
# mount -t iso9660 /dev/sr0 /media/cdrom"
```

La plupart des DVD-Roms utilisent plutt le format UDF (Universal Disk Format).

```
# mount -t udf /dev/sr1 /media/dvd"
```

Une image ISO est une image du contenu d'un CD ou d'un DVD. C'est un système de fichiers iso9660 ou udf dans un fichier. Il est possible d'utiliser cette image comme un périphérique, à l'aide des périphériques de loopback. L'image est rattachée à un périphérique de loopback, et les outils passent par ce périphérique comme s'il s'agissait d'un disque.

```
# mount -o loop -t iso9660 image.iso /mnt/iso"
```

Contrôler le système de fichiers
Statistiques d'occupation

Par système de fichiers

La commande `df"` permet d'obtenir des statistiques d'occupation de chaque système de fichiers monté. Sans argument, `df"` fournit des informations sur tous les systèmes de fichiers. Vous pouvez passer comme argument un périphérique monté ou un point de montage. Si vous passez un répertoire quelconque, df donne des informations sur le système de fichiers qui contient ce répertoire.

```
# df"
```

Le résultat est explicite. L'unité par défaut est le kilo-octet (identique au paramètre -k) bien que la norme POSIX définisse une unité de bloc à 512 octets. Vous pouvez modifier les paramètres pour demander le résultat en Mo (m).

```
# df -m /home"
```

Pour que ce soit plus lisible, rajoutez le paramètre -h (Human readable).

```
# df -h /home"
```

Ne confondez pas ce dernier paramètre avec -H qui affiche le résultat en unités **SI** (Système International).

```
# df -H /home"
```

Le -T rajoute l'affichage du type de système de fichiers.

```
# df -T /home"
```

La commande **df** permet aussi de fournir des statistiques pour l'utilisation des inodes. Vous pouvez cumuler le paramètre -i avec le paramètre -h.

```
# df -i /home"
# df -ih /home"
```

Contrôler le système de fichiers
Statistiques d'occupation

Par arborescence

La commande `du"` (disk usage) fournit des informations sur l'espace occupé par une arborescence (un répertoire et tout son contenu). Si rien n'est précisé, c'est le répertoire courant qui est utilisé. Les paramètres -k (Ko) et -m (Mo) déterminent l'unité. La taille est fournie pour chaque élément (voire arrondie). La taille totale de l'arborescence est sur la dernière ligne.

```
# du -m LIVRE_ALGO"
```

Pour n'avoir que le total et pas tous les détails, utilisez le -s.

```
# du -ks LIVRE_ALGO"
```

Notez que du n'est pas limitée à un seul système de fichiers et continue à calculer si elle tombe sur un point de montage dans l'arborescence qu'elle analyse. Si vous voulez limiter le calcul au système de fichiers courant sans rentrer dans les points de montage présents dans l'arborescence, précisez -x.

```
# du -msx /"
```

Contrôler le système de fichiers

Vérifier, régler et réparer

fsck

La commande `fsck"` permet de vérifier et de réparer un système de fichiers.

```
fsck -t typefs périphérique"
```

Le système de fichiers à vérifier ou réparer ne devrait pas être monté, ou alors seulement en lecture seule.

Tout comme mkfs, fsck appelle une autre commande selon le type de système de fichiers à vérifier : fsck.ext2, fsck.ext3, etc. Chacune peut prendre des options particulières. Si fsck ne reconnaas l'option qui lui est fournie, il la transmet au programme concerné. Si vous n'indiquez pas de type, fsck tente de le déterminer seul.

Pour cet exemple, le paramètre -f est passé à fsck pour forcer la vérification (il n'a pas été possible de produire une corruption) ainsi que le paramètre -V pour fournir tous les détails.

```
# fsck -fV /dev/sda2"
```

Lorsque le système de fichiers est endommagé, fsck vous pose des questions à chaque action nécessaire. Vous pouvez passer le paramètre -p pour tenter une réparation automatique, ou encore -y pour forcer les réponses à oui.

Lors du démarrage du système, celui-ci vérifie depuis combien de temps, ou au bout de combien de montage, le système de fichiers n'a pas été vérifié. Si l'intervalle de temps est trop important alors il va exécuter un fsck sur le système de fichiers concerné. Les intervalles peuvent être modifiés par la commande **tune2fs**.

Contrôler le système de fichiers
Vérifier, régler et réparer

badblocks

La commande `badblocks"` tente de vérifier les blocs défectueux sur le périphérique de stockage fourni en argument. Elle peut être appelée par mkfs ou fsck si le paramètre -c (check) leur est fourni.
Par défaut badblocks lit l'intégralité des blocs du support et retourne un erreur si un ou plusieurs d'entre eux sont illisibles. La commande peut être lancée même si le système de fichiers est monté, sauf si vous tentez un test en lecture et écriture, même non destructif.

```
# badblocks -v /dev/sda2"
```

Les paramètres -n (non destructif) et -w (write, avec motifs, destructif) tentent des écritures sur les blocs. Le premier lit et réécrit le bloc à l'identique, le second écrit plusieurs motifs (0xaa, 0x55, 0xff, 0x00) donc écrase l'ancien contenu.
L'exécution de badblocks peut être très longue, plusieurs heures sur quelques centaines de Go.

dumpe2fs

La commande `dumpe2fs"` accepte en argument un périphérique contenant un système de fichiers ext2 ou ext3. Elle retourne un grand nombre d'informations sur le système de fichiers.

```
# dumpe2fs /dev/sda2"
```

Cette sortie est très longue (elle a pourtant été tronquée), mais vous donne tous les détails possibles sur le système de fichiers. Vous pouvez isoler uniquement l'en-tête (jusqu'à la taille du journal) avec le paramètre -h. Le mieux, si vous cherchez une information précise, est d'utiliser la commande **grep**.

```
# dumpe2fs -h /dev/sda2|grep -i "block size" "
```

Les quotas disques
Définitions

quotas

Les **quotas** permettent de poser des limites à l'utilisation de systèmes de fichiers. Ces limites sont de deux types :

- **inodes** : limite le nombre de fichiers.
- **blocs** : limite la taille disque.

Les quotas sont implémentés par système de fichiers individuel et pas pour l'ensemble des systèmes de fichiers. Chaque utilisateur peut être géré de manière totalement indépendante. Il en est de mêmepour les groupes. Pour chaque utilisation (inode ou bloc), vous pouvez mettre en place deux limites dans le temps :

- **Limite dure** (hard) : quantité maximale d'inodes ou de blocs utilisés que l'utilisateur ou le groupe ne peuvent absolument pas dépasser.
- **Limite douce** (soft) : quantité maximale d'inodes ou de blocs utilisés que l'utilisateur ou le groupe peuvent temporairement dépasser.
- **Un délai de grâce** est mis en place. Durant ce temps, l'utilisateur peut continuer à travailler sur le système de fichiers. Le but est qu'il revienne à terme sous la limite douce. Le délai dépassé, la limite douce devient la limite dure.

Les quotas sont implémentés dans le noyau Linux et au sein des systèmes de fichiers. Pour les utiliser, les outils de quotas (packages quota) doivent être installés. Les manipulations suivantes sont effectuées sur un système de fichiers ext3.

Les quotas disques
Mise en place

quotas

Vous allez mettre en place les quotas sur la partition /**home** en respectant les étapes suivantes :
1- Modifiez les options de partition dans `/etc/fstab"`. On rajoute dans les options usrquota (utilisateur) ou grpquota (groupe), ou les deux.

```
LABEL=/home /home ext3 defaults,usrquota 1 2"
```

2- Remontez le système de fichiers.

```
# mount -o remount /home"
```

3- Créez les fichiers contenant les informations de quota (base de données de quotas).

```
# cd /home"
aquota.user aquota.group"
```

4- Mettez à jour la base de données avec la commande **quotacheck**.

```
# quotacheck -c /home"
```

5- Démarrez (ou arrêter) les quotas. Cette opération n'est pas nécessaire après un redémarrage de Linux car la mise en place des quotas est comprise dans les scripts de démarrage. La commande quotaon démarre les quotas pour le système de fichiers indiqué (-a pour tous). La commande quotaoff stoppe les quotas.

```
quotaon /home"
```

6- Éditez les quotas pour les utilisateurs ou les groupes. La commande **edquota** est utilisée. En pratique, si tous les utilisateurs doivent avoir les mêmes quotas ou avec quelques variantes, on crée un utilisateur lambda dont on recopiera les propriétés. Établir les quotas pour roger :

```
# edquota roger # = edquota -u roger"
```

Les quotas de arthur sont identiques à ceux de roger :

```
# edquota -p roger arthur"
```

Les quotas disques
Mise en place

quotas

7- Établissez le délai de grâce. Le délai accepte les unités seconds , minutes , hours , days , weeks , monthes .

```
# edquota -t"
```

8- Vérifiez les quotas. Les utilisateurs peuvent vérifier l'état de leurs quotas avec la commande **quota**. L'administrateur peut générer un rapport avec **repquota**. Enfin la commande **warnquota** qui peut être exécutée via cron peut envoyer un mail aux utilisateurs pour les prévenir en cas de dépassement.

L'édition des quotas se fait avec l'éditeur par défaut du système qui est généralement vi (le comportement peut être modifié via les variables EDITOR et VISUAL). Les blocs de quotas sont des blocs de 1 Ko.

```
# edquota roger"
# edquota -t"
# repquota /home"
```

Une dernière nécessité est d'utiliser régulièrement la commande **quotacheck** pour maintenir la cohérence des informations de quotas des systèmes de fichiers. En effet, en cas arrêt des quotas ou de problème (arrêt inopiné par exemple), il peut parfois être nécessaire de vérifier et de réactualiser les informations.

```
# quotacheck -avug"
```

Les droits d'accès
Les droits de base

Droits et utilisateurs

À sa création par l'administrateur, un utilisateur se voit affecter un **UID** (User Identification) unique. Les utilisateurs sont définis dans le fichier /**etc**/**passwd**. De même chaque utilisateur est rattaché à un groupe au moins (groupe principal), chaque groupe possédant un identifiant unique, le **GID** (Group Identification). Les groupes sont définis dans /**etc**/**group**.
La commande `id"` permet d'obtenir ces informations. En interne, le système travaille uniquement avec les UID et GID, et pas avec les noms eux-mêmes.

```
$ id"
uid=1000(seb) gid=100(users) groupes=7(lp),16(dialout),33(video), 100(users)"
```

À chaque fichier (inode) sont associés un UID et un GID définissant son propriétaire et son groupe d'appartenance. Vous affectez des droits pour le propriétaire, pour le groupe d'appartenance et pour le reste du monde. On distingue trois cas de figure :

- UID de l'utilisateur identique à l'UID défini pour le fichier. Cet utilisateur est propriétaire du fichier.
- Les UID sont différents : le système vérifie si le GID de l'utilisateur est identique au GID du fichier. Si oui l'utilisateur appartient au groupe associé au fichier.
- Dans les autres cas (aucune correspondance) : il s'agit du reste du monde (others), ni le propriétaire, ni un membre du groupe.

d	rwxr-xr-x	29	seb	users	4096	Mar 15 22:13	Documents

Sur cette ligne du tableau, le répertoire Documents appartient à l'utilisateur seb et au groupe users, et possède les droits rwxr-xr-x.

Les droits d'accès
Les droits de base

Signification

Droit/Général	Signification
r	Readable (lecture).
w	Writable (écriture).
x	Executable (exécutable comme programme).
Droit/Fichier normal	Signification
r	Le contenu du fichier peut être lu , chargé en mémoire, visualisé, recopié.
w	Le contenu du fichier peut être modifié.
x	Le fichier peut être exécuté depuis la ligne de commande,
Droit/Répertoire	Signification
r	Les éléments du répertoire sont accessibles en lecture.
w	Les éléments du répertoire sont modifiables
x	Le catalogue peut être accédé par CD et listé.

rwx	r-x	r–
Droits de l'utilisateur en lecture, écriture et exécution.	Droits pour les membres du groupe en lecture et exécution.	Droits pour le reste du monde en lecture uniquement.

Les droits d'accès
Modification des droits

chmod

Lors de sa création, un fichier ou un répertoire dispose de droits par défaut. Utilisez la commande **chmod** (change mode) pour modifier les droits sur un fichier ou un répertoire. Il existe deux méthodes pour modifier ces droits : par la forme symbolique et par la base 8. Seul le propriétaire d'un fichier peut en modifier les droits (plus l'administrateur système). Le paramètre -R change les droits de manière récursive.

Par symbole

La syntaxe est la suivante :

```
chmod modifications Fic1 [Fic2...]"
```

S'il faut modifier les droits de l'utilisateur, utilisez le caractère **u**, pour les droits du groupe le caractère **g**, pour le reste du monde le caractère **o** et pour tous le caractère **a**.
Pour ajouter des droits, on utilise le caractère **+**, pour en retirer le caractère **-**, et pour ne pas tenir compte des paramètres précédents le caractère **=**.
Enfin, le droit d'accès par lui-même : **r**, **w** ou **x**.
Vous pouvez séparer les modifications par des virgules et cumuler plusieurs droits dans une même commande.
Si vous voulez supprimer tous les droits, ne précisez rien après le signe **=**

Par base 8

La syntaxe est identique à celle des symboles. À chaque droit correspond une valeur octale, positionnelle et cumulable. Pour encoder trois dro
its rwx, il faut trois bits, chacun prenant la valeur 0 ou 1 selon que le droit est absent ou présent. $2^3 = 8$, d'o une notation octale possible. Le r vaut 4, le w vaut 2 et le x vaut 1. Le tableau suivant vous aide :

r	w	x	r	w	x	r	w	x
400	200	100	40	20	10	4	2	1

Pour obtenir le droit final il suffit d'additionner les valeurs. Par exemple si vous voulez rwxrw-rw- alors obtenez 400+200+100+40+10+4+1=755, et pour rw-r–r– 400+200+40+4=644

```
$ chmod 755 fic1"
$ chmod 644 fic2"
$ ls -l fic1 fic2"
```

Restreindre des droits automatiquement

Lors de la création d'un fichier ou d'un répertoire, des droits leur sont automatiquement assignés. Généralement, c'est rw-r–r– (644) pour un fichier et rwxr-xr-x (755) pour un répertoire. Ces valeurs sont contrlées par un masque, lui-même modifiable par la commande **umask**. La commande **umask** prend comme paramètre une valeur octale dont chaque droit individuel sera supprimé des droits d'accès maximum du fichier ou du répertoire.

- Par défaut, tous les fichiers sont créés avec les droits 666 (rw-rw-rw-).
- Par défaut tous les répertoires sont créés avec les droits 777 (rwxrwxrwx).
- Puis le masque est appliqué.
- Le masque est le même pour l'ensemble des fichiers.
- Un masque ne modifie pas les droits des fichiers existants, mais seulement ceux des nouveaux fichiers.

Le masque par défaut est 022, soit —-w–w-. Pour obtenir cette valeur, tapez umask sans paramètre.

```
$ umask"
0022"
```

Calcul de masque

Pour un fichier

```
Défaut rw-rw-rw- (666)"
Retirer --w-w- (022)"
Reste rw-r-r- (644)"
```

Pour un répertoire

```
Défaut rwxrwxrwx (777)"
Retirer --w-w- (022)"
Reste rwxr-xr-x (755)"
```

Notez qu'appliquer un masque n'est pas soustraire, mais supprimer des droits de ceux par défaut, droit par droit. Par exemple :

```
Défaut rw-rw-rw- (666)"
Retirer --wxrwx (037)"
Reste rw-r (640)"
```

Et non 629, ce qui est impossible en octal...

chown & chgrp

Il est possible de changer le propriétaire et le groupe d'un fichier à l'aide des commandes **chown** (*change owner*) et **chgrp** (*change group*). Le paramètre -R change la propriété de manière récursive.

```
chown utilisateur fic1 [Fic2...]"
chgrp groupe fic1 [Fic2...]"
```

En précisant le nom d'utilisateur (ou de groupe), le système vérifie d'abord son existence. Vous pouvez préciser un UID ou un GID, dans ce cas le système n'effectuera pas de vérification.
Pour les deux commandes, les droits précédents et l'emplacement du fichier ne sont pas modifiés.
Il est possible de modifier en une seule commande à la fois le propriétaire et le groupe.

```
chown utilisateur[:groupe] fic1 [fic2...]"
chown utilisateur[.groupe] fic1 [fic2...]"
```

Seul root a le droit de changer le propriétaire d'un fichier. Mais un utilisateur peut changer le groupe d'un fichier s'il fait partie du nouveau groupe.

```
$ chgrp video fic1"
$ ls -l fic1"
-rwxr-xr-x 1 seb video 0 mar 21 22:03 fic1"
```

SUID et SGID

Il est possible d'établir des **droits d'accès étendus** à l'exécution d'une commande. Ces droits d'accès étendus appliqués à une commande permettent à cette commande de s'exécuter avec les droits du propriétaire ou du groupe d'appartenance de la commande, et non plus avec les droits de l'utilisateur l'ayant lancée.
L'exemple le plus simple est le programme **passwd** permettant de changer son mot de passe. Si la commande était exécutée avec les droits d'un utilisateur classique, **passwd** ne pourrait pas ouvrir et modifier les fichiers `/etc/passwd"` et `/etc/shadow"` :

```
$ ls -l /etc/passwd"
-rw-r-r- 1 root root 1440 fév 24 10:35 /etc/passwd"
```

Vous constatez que ce fichier appartient à root, et que seul root peut y écrire. Un utilisateur simple ne peut lire que son contenu sans interagir. La commande **passwd** ne devrait donc pas pouvoir modifier les fichiers. Voyez les droits de la commande **passwd** (/bin/passwd ou /usr/bin/passwd) :

```
> ls -l /usr/bin/passwd"
-rwsr-xr-x 1 root shadow 78208 sep 21 23:06 /usr/bin/passwd"
```

Un nouveau droit est apparu : le droit s pour les droits de l'utilisateur root. Ce nouvel attribut permet l'exécution de la commande avec des droits d'accès étendus. Le temps du traitement, le programme est exécuté avec les droits du propriétaire du fichier ou de son groupe d'appartenance. Dans le cas de passwd, il est lancé avec les droits de root le temps de son traitement.
Le droit s sur l'utilisateur est appelé le **SUID-Bit** (Set User ID Bit), et sur le groupe le **GUID-Bit** (Set Group ID Bit)
La commande **chmod** permet de placer les SUID-Bit et GUID-Bit.

```
chmod u+s commande"
chmod g+s commande"
```

Les valeurs octales sont 4000 pour le SUID-Bit et 2000 pour le GUID-Bit.

```
chmod 4755 commande"
chmod 2755 commande"
```

Merci pour votre attention!