Administration système

Nom de l’enseignant : RAKOTONDRAMANANA Radiarisainana Sitraka

Table des matières

CHAPITRE 1 GENERALITES DES SYSTEMES LINUX

1.1 Historique

Un élément essentiel à l'avènement de Linux est l'existence d'un système d'exploitation « éducatif » pour PC basé sur **un micronoyau Unix** baptisé **Minix**. Ce système créé à l'initiative de Andrew Tanenbaum fut distribué avec **la source en langage C** sur un jeu de quelques disquettes. Minix n'a jamais eu la prétention d'être commercialement viable et n'était destiné qu'à des fins pédagogiques.

Cependant, un étudiant finlandais **Linus TORVALDS** ne se satisfaisant pas de Minix, commença à le modifier et à créer un petit noyau performant pour un processeur i386. L'aventure Linus aurait pu s'arrêter à ce stade (projet d'étudiant) si Linus TORVALDS n'avait pas eu la volonté de le mettre à disposition de tous au travers d'Internet (téléchargement et forum de discussions). Cette idée chère à **Richard STALLMAN** et à son projet **Gnome Not Unix (GNU)** trouva en Linux une concrétisation. C'est donc du monde entier que viennent les corrections, les améliorations et les extensions.

Depuis, maintenant près de dix ans, Linux est bien le premier exemple réussi de développement à travers le monde (grâce à Internet). La stabilité du système provient de ce phénomène. En effet, ce style de développement et d'intégration pour un logiciel intéressant un large public (ce qui est le cas d'un OS) jouit de deux avantages remarquables.

- **La plus grande base de bêta-testeurs du monde** : il n'existe pas de restriction sur la diffusion puisqu'il n'y a aucune peur de « piratage », le logiciel étant libre de droit et fourni avec les sources. Le monde universitaire a été un grand vecteur de diffusion de par leur connexion à l'Internet et du fait de l'utilisation massive d'Unix. Rapidement, des milliers de personnes vont devenir utilisateurs de Linux, qui soit restent passifs en faisant remonter les informations de disfonctionnement, soit préfèrent les corriger eux-mêmes (on parle d'utilisateurs actifs) et distribuer la solution.

- **Une réactivité rapide pour la correction de bogues** : par ce principe d'utilisateurs passifs qui deviennent actifs, l'équipe de développement s'agrandit continuellement. Les mises à jour et les nouvelles versions peuvent se succéder très rapidement, puisqu'il n'y a pas d'étapes de commercialisation. Le cycle « découverte, résolution, intégration » est le plus court possible, bien que ces trois étapes puissent ne pas être le fait d'une même personne physique ni d'une même

équipe. La rapidité de réaction est un élément essentiel pour la survie d'un tel logiciel puisqu'il doit convaincre et ne peut pas jouer sur la compatibilité avec un système déjà existant et omniprésent. Cependant, le risque majeur d'un tel modèle de développement (libre et à travers Internet) est la divergence du code source qui n'a plus de réel responsable. La mise au point rapide de garde-fou a permis à Linux d'éviter cet écueil.

Ainsi la licence GPL sous laquelle est distribué le noyau Linux est un des meilleurs garants de cette stabilité, puisqu'il stipule que toute modification doit être publiée sous GPL. Il ne peut donc pas se produire de scission entre divers groupes de développement œuvrant chacun de leur côté pour l'amélioration de « leur nouveau produit ». Ils amélioreraient, par voie de conséquence, « le logiciel libre originel » étant contraints de divulguer leurs modifications.

De plus, les sociétés qui éditent des distributions Linux, qui font toutes parties du consortium Linux International, ont décidé récemment, sous l'égide de ce dernier, une norme commune, « Linux Standard Base » (http://www.linuxbase.org/) et, en même temps, de se rapprocher de la norme Unix98. Des procédures de tests de conformité sont actuellement en cours de développement. Malgré ces atouts, Linux ne s'implante réellement dans les premiers temps que dans le monde universitaire, où il est accueilli avec enthousiasme comme solution à très bas coût (le prix d'un PC) pour remplacer des stations Unix/ X-Window sur lesquelles tournaient la plupart des codes universitaires.

Ces mêmes universitaires, ayant un accès privilégié à Internet, ne se formalisaient pas du mode de distribution (et de mise à jour) ainsi que du manque d'interfaces conviviales lors de l'installation et de la maintenance. On peut réellement parler d'esprit de pionniers « bidouilleurs » qui ont laissé de côté l'aspect esthétique et pratique au profit de la robustesse du noyau. Un autre point principal qui expliquait l'absence d'adhésion du monde industriel est le manque de prise en charge des nouveaux périphériques ; alors que ces derniers sont fournis avec des drivers Windows lors de l'achat, il fallait des mois pour voir arriver un driver Linux (et même parfois le constructeur refusait de divulguer les informations nécessaires à la réalisation du driver). Consciente de ces faiblesses, la communauté Linux, forte d'un noyau stable et robuste, a réagi et propose dorénavant des solutions d'entreprises. L'installation est automatisée sous environnement graphique, avec détection de la plupart des périphériques. Il existe des possibilités d'installation en groupes (clones) par réseau pour un parc homogène.

De plus en plus d'accords sont signés entre les constructeurs de matériels pour soit fournir des drivers Linux, soit divulguer le maximum d'informations afin que le développement des drivers soit rapide. De plus en plus d'éditeurs de logiciels supportent Linux. Il est maintenant indéniable que Linux représente une alternative professionnelle dans le monde des systèmes d'exploitation.

Figure 1.01 : *Emblême GNU et Linux*

1.2 Installation système

1.2.1 Distribution Linux

- **Slackware :** une des plus anciennes distributions de Linux. Elle existe toujours aujourd'hui
- **Mandriva :** éditée par une entreprise française, elle se veut simple d'utilisation.
- **Red Hat :** éditée par une entreprise américaine "Red Hat", cette distribution est célèbre et très répandue, notamment sur les serveurs.
- **SuSE :** éditée par l'entreprise Novell.
- **Debian :** la seule distribution qui soit gérée par des développeurs indépendants au lieu d'une entreprise. C'est une des distributions les plus populaires.

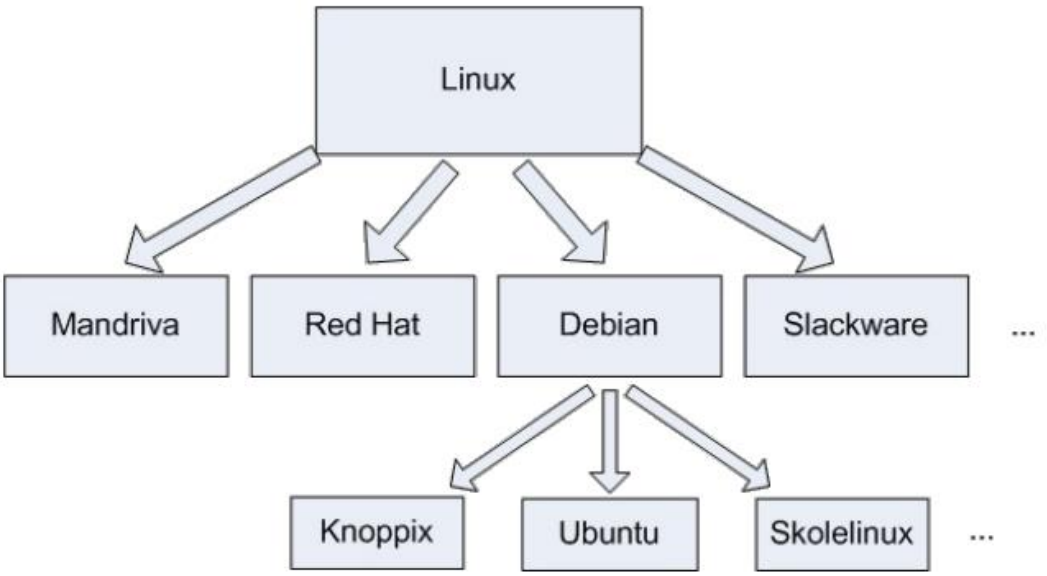


Figure 1.02 : *Distribution Linux*

1.2.2 Installation Ubuntu

Etape 1 :

Si vous devez modifier l'ordre de boot pour que votre ordinateur lise le CD, redémarrez. Pendant l'écran de boot (la toute première chose que vous voyez à l'écran), pressez la touche indiquée pour accéder au Setup, aussi appelé BIOS (c'est l'écran de configuration de votre carte mère). Généralement, la touche est F2 ou Suppr, mais cela peut varier selon la carte mère que vous avez. Vous devriez alors voir le superbe menu du BIOS (sigh !). D'un ordinateur à l'autre, cet écran peut être légèrement différent.

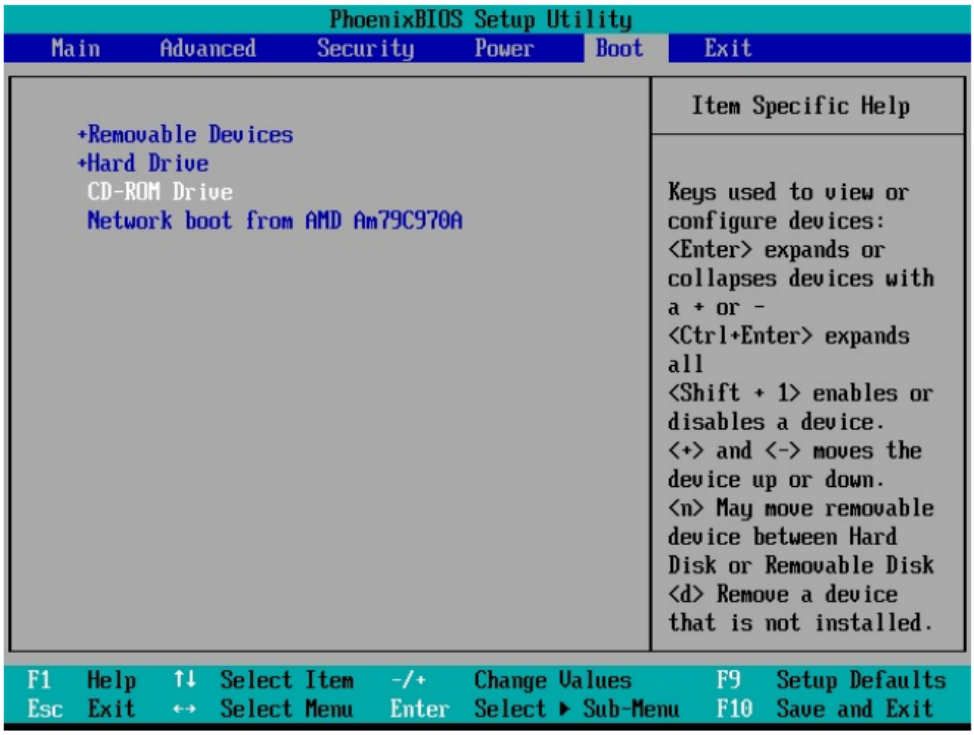


Figure 1.03 : *Répresentation de modification des ordres de boot*

Etape 2 :

Sélectionnez la langue dans le menu de gauche si les textes ne sont pas en français.

Vous voyez que vous avez 2 choix :

- **Essayer Ubuntu :** Ubuntu sera lancé sans toucher à votre disque dur. Vous pourrez donc l'essayer pour le tester. (Lancement de système sans installer dans la mémoire RAM)
- **Installer Ubuntu :** Ubuntu sera installé sur votre disque dur. Utilisez ce choix si vous êtes déjà certain de vouloir installer Ubuntu.

Figure 1.04 : *Commencement de l'installation*

Etape 3 : Préparation d'installation

Les 3 conditions sont posées :

- **Avoir un minimum d'espace disque disponible :** normal, si vous voulez avoir la place d'installer Ubuntu ! Ici, il s'agit du strict minimum, je vous conseille d'avoir 8-10 Go pour être suffisamment confortable. A priori, pas besoin de plus d'espace supplémentaire, sauf si vous prévoyez d'y stocker de gros fichiers : musique, photos, vidéos persos etc.
- **Etre branché sur le secteur :** cela concerne bien entendu les ordinateurs portables. Il est très fortement recommandé d'être branché, car installer un système d'exploitation sur batterie est tout simplement... Il serait très ennuyeux pour votre installation que celle-ci soit coupée en plein milieu à cause d'une batterie vide !
- **Etre connecté à Internet :** c'est facultatif, mais je le recommande fortement là aussi. Cela permettra à l'assistant d'installation de télécharger immédiatement les dernières mises à jour des programmes ainsi que les traductions françaises qui pourraient manquer sur le CD d'Ubuntu. Oubliez le wifi ici, qui peut être un peu compliqué à configurer : branchez-vous à Internet avec un vrai câble réseau (RJ45).

Deux options peuvent être cochées (récommendation pour cocher toutes les deux !) :

- **Télécharger les mises à jour pendant l'installation :** cela vous assurera que les programmes sont immédiatement le plus à jour possible. C'est préférable car les mises à jour

corrigent des failles de sécurité, des bugs et améliorent certaines fonctionnalités des programmes déjà présents sur votre CD d'Ubuntu. Bien entendu, il faut être connecté à Internet avec un câble réseau pour cela.

- **Installer ce logiciel tiers :** cette option vous permet d'installer certains programmes propriétaires. Pour qu'Ubuntu reste libre, ces programmes ne sont pas installés par défaut, mais vous pouvez demander leur installation en cochant cette case (ce que je vous recommande pour votre confort). Vous aurez ainsi la possibilité de lire des MP3, du Flash et d'autres fichiers multimédia protégés par des licences propriétaires. Cette option peut aussi améliorer la prise en charge de votre carte wifi. Bref, c'est forcément intéressant pour vous, sauf si vous ne voulez pas installer de programme propriétaire sur votre machine.

Etape 4 : Partitionnement des disques dures

Avant d'aller plus loin, il est très vivement conseillé d'effectuer **une défragmentation**. C'est une opération qui consiste en gros à mieux organiser les fichiers sur votre disque dur, à les rassembler pour éviter qu'ils ne soient éparpillés. Vos fichiers sont parfois placés un peu n'importe comment à la surface de votre disque dur. Sur la surface du disque, j'ai représenté une multitude de fichiers : ce sont les fichiers tels qu'ils sont placés sur votre disque actuellement. Un beau bordel. Parfois, certains fichiers sont **coupés en plusieurs morceaux et éparpillés sur votre disque** ! On dit que les fichiers sont **fragmentés** (coupés en plusieurs fragments).

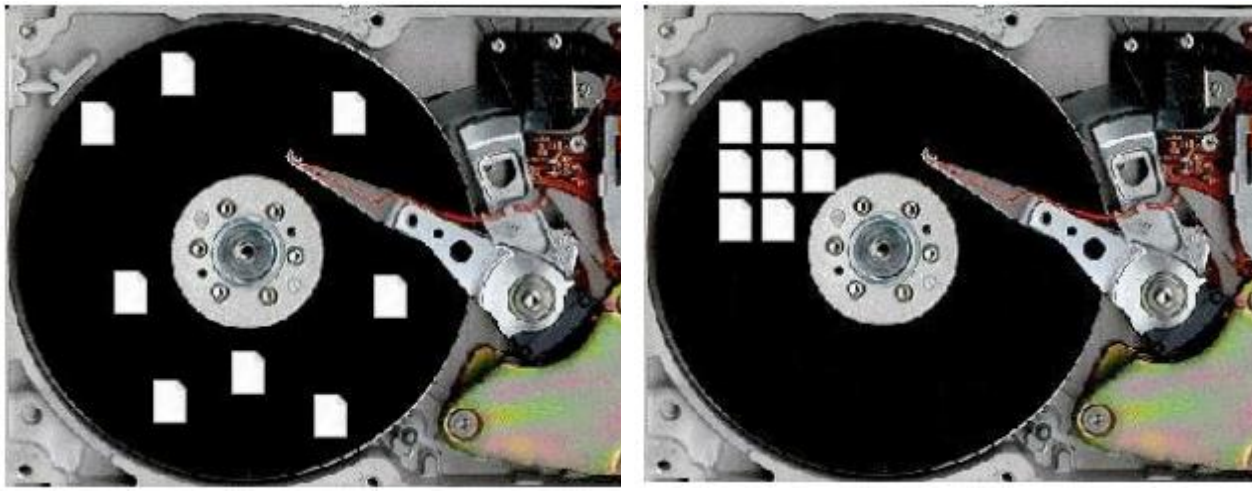


Figure 1.05 : *Comparaison des emplacements des morceaux de fichiers fragmentés*

Les systèmes de fichiers en linux sont différents de windows.

- **Ext2 :** c'est le système de fichiers qui a longtemps été utilisé sous Linux. Il a été développé par un français (Rémy Card) et présente la particularité de très peu se fragmenter. Ainsi, sous Linux et depuis longtemps, il n'y a pas besoin de faire de défragmentation.
- **Ext3 :** l'ext3 est très proche de l'ext2, à une différence majeure près, la journalisation. En effet, ext2 n'était pas journalisé, et en cas de crash disque on risquait plus facilement une perte de données. Ce n'est plus le cas avec l'ext3. A noter que l'ext2 et l'ext3 sont parfaitement compatibles entre eux, dans un sens comme dans l'autre.
- **Ext4 :** une amélioration de l'ext3, relativement récente, qui améliore la prise en charge des gros disques durs et diminue les problèmes de fragmentation des fichiers.

Pour partionner le système d'exploitation, 3 options sont proposées :

- **Installer à côté d'autres systèmes d'exploitation :** Ubuntu va se faire automatiquement de la place sur votre disque dur et créer les partitions pour vous. C'est la solution la plus simple que vous devriez choisir si vous ne voulez pas aller dans les détails. En revanche, vous n'aurez pas de partition spéciale pour les documents dans ce mode-ci. En bas de la fenêtre, vous pouvez déplacer le curseur pour décider de l'espace que vous attribuez à Windows et à Ubuntu.
- **Tout effacer et utiliser le disque entier :** tout le disque sera formaté, partition Windows comprise. Ne faites cela que si vous voulez supprimer Windows ! Ubuntu sera installé sur l'ensemble du disque dur.
- **Définir les partitions manuellement (avancé) :** choisissez cette option si vous voulez créer vous-même les partitions. C'est plus complexe mais cela vous donnera plus de choix.

En linux les systèmes sont representés par la lettre **hda** :

- **h** : la première lettre indique si le disque est de type IDE ou SCSI (un type de connexion différent à la carte mère). Si c'est une IDE, la lettre est un h, si c'est un SCSI (ou un S-ATA), la lettre est un **s**.
- **d** : cette lettre ne change pas. Pour designer la signification **drive**.
- **a :** c'est cette lettre qui indique les différents disques durs. hda représente le premier disque dur IDE, hdb le second, hdc le troisième etc.

Lorsqu'on crée des partitions, on ajoute généralement un chiffre représentant le numéro de la partition. Ainsi, si on a 3 partitions sur notre disque hda, elles seront nommées hda1, hda2, hda3...

L'outil de partitionnement manuel :

- **Créer une partition pour installer Ubuntu**

Ubuntu vous propose de créer 2 types de partitions :

Primaires : c'est la partition de base, classique. On ne peut en créer que 4 par disque.

Logiques : c'est un type de partition qui peut contenir de nombreuses sous-partitions. Celles-ci n'ont pas de limite de nombre comme les partitions primaires. Cliquez sur la partition libre du disque dur, puis cliquez sur le bouton "Nouvelle table de partition" en bas.

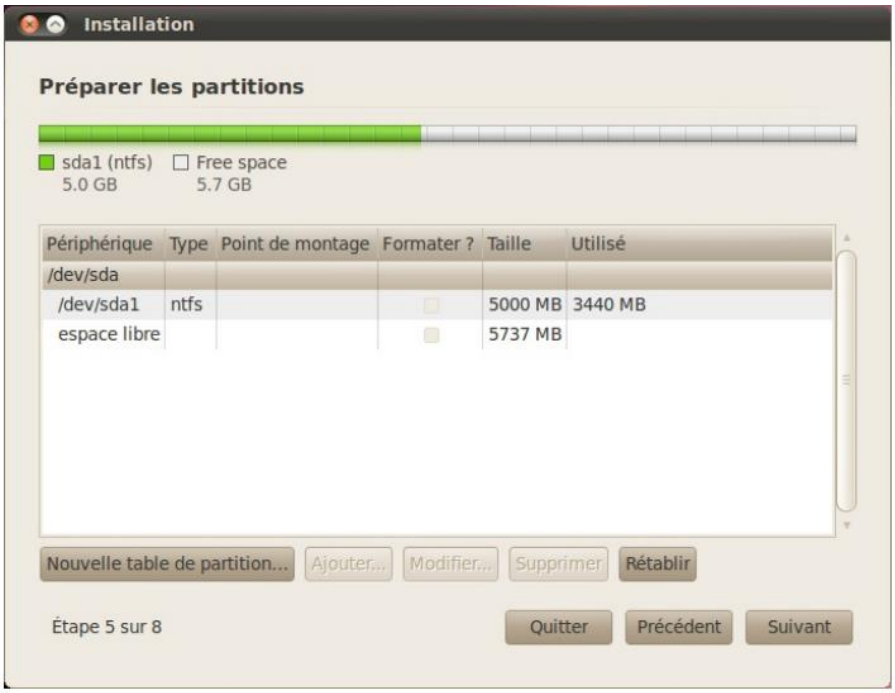


Figure 1.06 : *Préparation de partition*

Figure 1.07 : *Création d'une partition système*

- **Créer une partition pour les documents**

Cette fois, vous pouvez créer une partition bien plus grande. Ce sera la partition où vous stockerez vos documents, un peu comme le "Mes documents" de Windows qui est souvent vite rempli de musiques et de films gourmands en espace disque. Choisissez la taille que vous voulez pour cette partition mais veillez à laisser environ 1 Go (1000 Mo) de libre sur votre disque pour que l'on puisse créer une dernière partition après.

Figure 1.08 : *Partition pour les documents*

- **Créer une partition pour le swap**

Il faut enfin créer une partition d'environ 1 Go appelée **"swap".** Pour faire simple, il s'agit **d'une extension de la mémoire vive sur votre disque dur.** Lorsque votre mémoire vive est pleine, Linux continue à fonctionner mais passe par le disque dur, grâce à la partition swap.

Figure 1.09 : *Partition SWAP*

La finalisation de l'installation sera representé par la figure suivante.

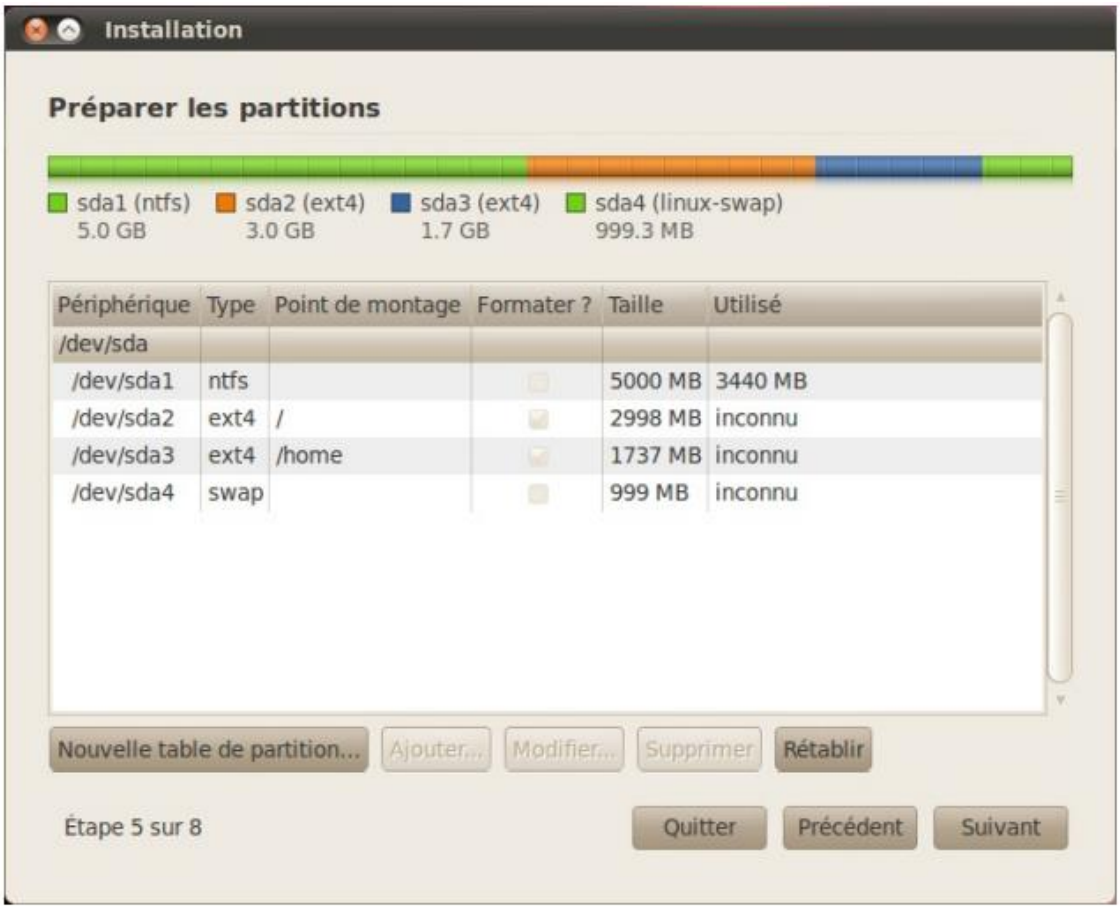


Figure 1.10 : *Finalisation de la partition du système*

Etape 5 : Selectionner le fuseau horaire

On vous demande sur cet écran près de quelle grande ville vous habitez pour régler le fuseau horaire. Cliquez sur la carte sur le point correspondant à la ville la plus proche. Vérifiez bien que l'heure indiquée est la bonne.

Etape 6 : Type de clavier

Dans la fenêtre qui suit, on vous demande quel type de clavier vous utilisez. Si vous habitez en France, vous avez un clavier dit "AZERTY", mais il se peut que vous habitiez un pays qui possède un clavier différent, comme la Suisse ou le Canada.

Etape 6 : Création de compte utilisateur

- La fenêtre suivante vous demande votre nom ainsi qu'un login (pseudonyme) qui vous identifiera sur votre ordinateur.

- Choisissez aussi un mot de passe. On vous demande le nom que vous voulez attribuer à votre ordinateur.
- On vous en propose un par défaut mais vous pouvez changer cela sans risque. Dans mon cas, je vais indiquer "mateo21-desktop" comme nom d'ordinateur.

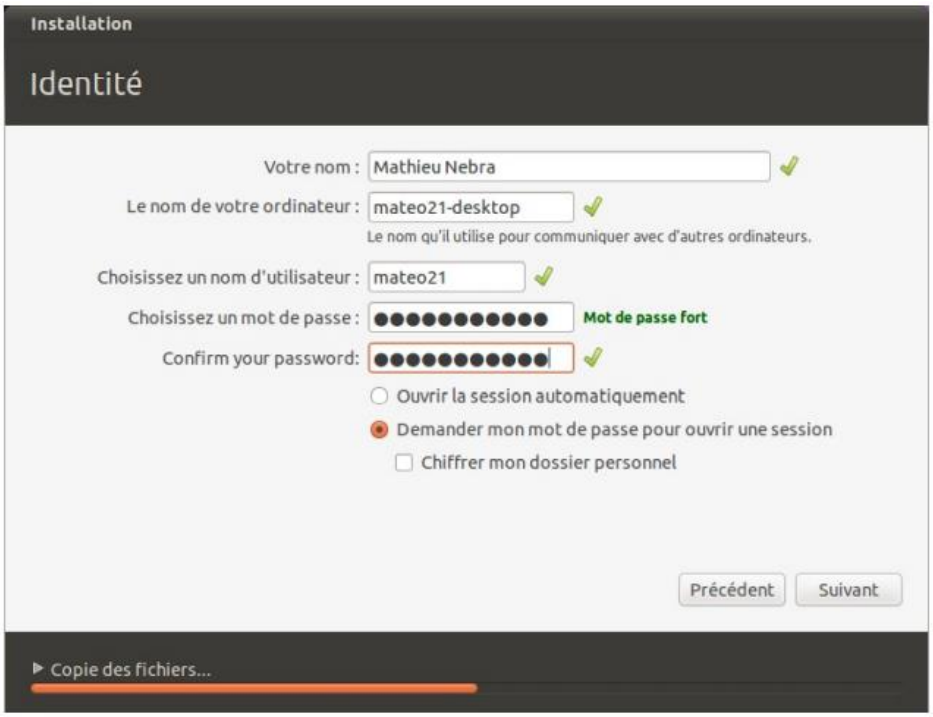


Figure 1.11 : *Extrait de finalisation d'installation*

Vous pouvez aussi choisir comment l'ordinateur doit être démarré :

- **Ouvrir la session automatiquement :** votre ordinateur démarrera entièrement sans vous demander de saisir votre mot de passe à l'allumage. N'utilisez cette option que si vous êtes sûr que personne d'autre que vous n'a accès à votre ordinateur. Evitez de sélectionnez cette option sur un ordinateur portable !
- **Demander mon mot de passe pour ouvrir une session :** on vous demandera votre mot de passe pour accéder à votre ordinateur. C'est l'option recommandée sur les ordinateurs portables en particulier comme je le disais.
- **Chiffrer mon dossier personnel :** si en plus vous êtes un peu parano et que vous avez des documents confidentiels, Ubuntu peut chiffrer vos documents, c'est-à-dire les crypter. Même un pro de l'informatique ne pourra pas lire vos documents s'il met la main sur votre ordinateur et qu'il analyse votre disque dur ! Je vous conseille de choisir un mot de passe solide pour que cela soit efficace.

1.2.3 Visages de Ubuntu

La première version d'Ubuntu était basée sur Gnome. Le succès d'Ubuntu grandissant, les utilisateurs de KDE et de XFCE ont voulu eux aussi voir des versions d'Ubuntu basées sur leur gestionnaire de bureau favori. De là sont nées Kubuntu (basée sur KDE) et Xubuntu (basée sur XFCE).

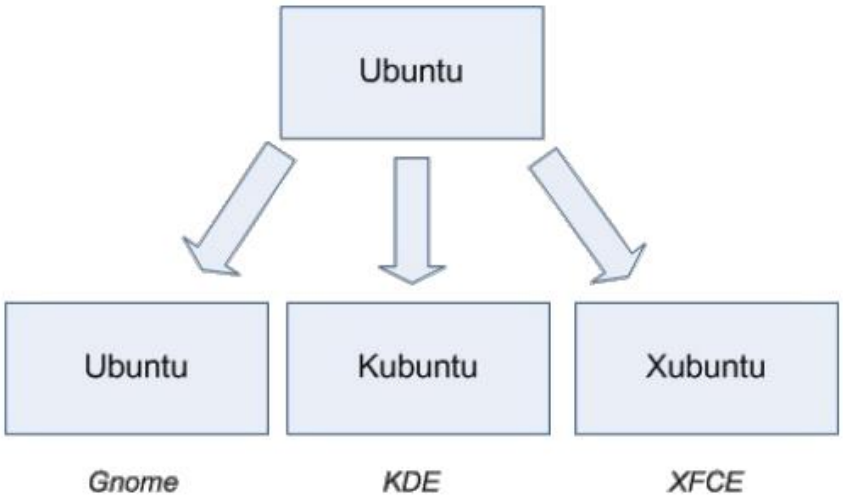


Figure 1.12 : *Les visages de ubuntu (les interfaces graphiques ubuntu)*

1.2.4 Installation logiciel en ubuntu

L'ajout et la suppression de programme est simple et intuitive. Rendez-vous dans **le menu Applications / Logithèque Ubuntu** :

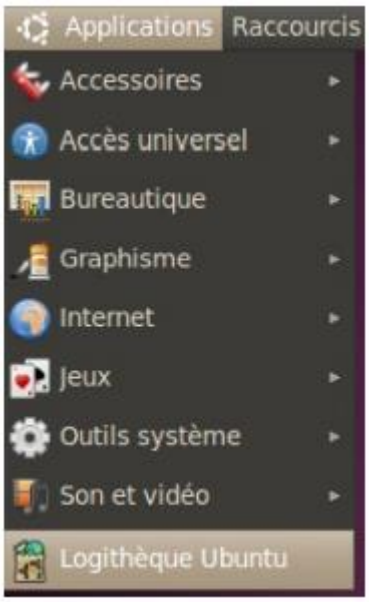


Figure 1.13 : *Ajouter et supprimer des programmes*

Pouvoir ajouter et supprimer des programmes c'est bien, mais il faut aussi les mettre régulièrement à jour pour profiter des nouvelles fonctionnalités et, surtout, corriger les failles de sécurité qui sont parfois détectées. Vous êtes automatiquement notifié dès qu'il y a des mises à jour disponibles. Il suffit de regarder la petite icône en haut à droite de l'écran (à gauche sur l'image qui suit) **: Cliquez dessus pour afficher le détail des mises à jour** :

Figure 1.14 : *Gestionnaire de mise à jour*

Si vous **ne voulez pas vous prendre la tête et être sûr d'avoir un système toujours à jour**, le mieux est **de configurer le gestionnaire de mises à jour pour qu'il installe les nouveautés sans demander votre autorisation.** Retournez dans **le menu Applications / Ajouter & Enlever.** Dans la fenêtre qui s'ouvre, **cliquez sur Préférences en bas**. **Cliquez ensuite sur l'onglet Mises à jour**, puis **sélectionnez Installer les mises à jour de sécurité sans confirmation.**

1.3 Commandes des bases

1.3.1 Invité de commande

Deux options permettent discuter avec l'ordinateur :

- En utilisant des gestes ou interfaces graphiques : Le mieux est d'ouvrir une console dans le mode graphique. Le programme Konsole sous KDE ou Terminal sous Gnome fera donc très bien l'affaire. L'interface graphique avec la souris c'est quand même plus intuitif
- En utilisant des ordres en tapant des commandes : mode terminal ou mode console

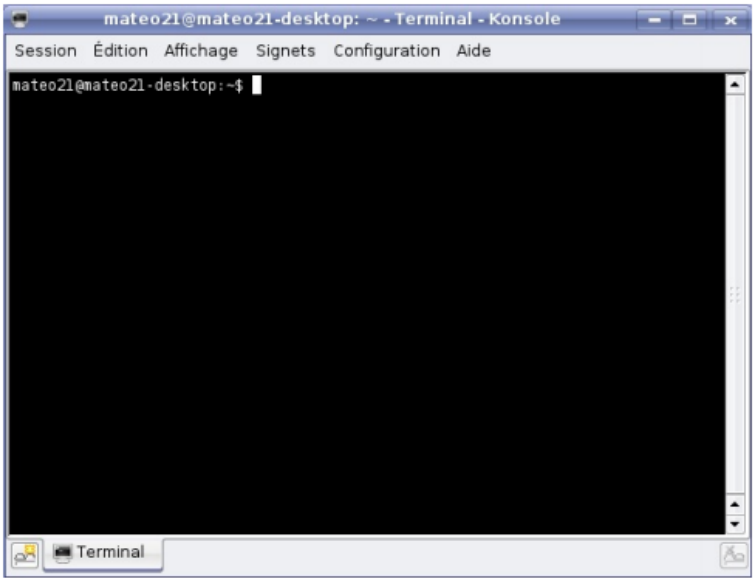


Figure 1.15 : *Répresentation de mode console*

```
mateo21@mateo21-desktop:~$
```

Cette interface s'appelle l'**invité de commande ou terminal ou mode console**. C'est un message qui vous invite à rentrer une commande en vous donnant par la même occasion une foule d'informations. Cette invite s'affiche avant chaque commande que vous tapez. Bien, décortiquons cette invite de commandes parce qu'elle est très intéressante :

- **mateo21 :** le premier élément est votre pseudonyme. C'est le pseudo sous lequel vous vous êtes loggé. En effet, rappelez-vous : on peut créer plusieurs comptes utilisateur sous Linux. Il est en général conseillé d'en faire un par personne susceptible d'utiliser l'ordinateur (un pour chaque membre de la famille par exemple). Nous verrons plus tard comment rajouter des comptes utilisateurs.

- **@ :** ce symbole n'indique rien de particulier. C'est le symbole "at" qui signifie "chez". Si on lit l'invite de gauche à droite, on doit donc comprendre "mateo21 chez".
- **mateo21-desktop :** ça c'est le nom de l'ordinateur sur lequel vous êtes en train de travailler. Dans mon cas il s'appelle mateo21-desktop, mais j'aurais pu l'appeler du nom que je voulais lors de l'installation. Par exemple, on a l'habitude de donner le nom d'un membre des Simpson à chacun des serveurs du Site du Zéro : Lisa, Bart, Itchy, Scratchy... Cela permet de savoir de quelle machine on parle quand on dit "Oulah Bart est surchargé, il faudrait voir quel est le programme qui ralentit tout". Si vous suivez toujours, la ligne d'invite de commandes se lit donc "mateo21 chez mateo21-desktop". En d'autres termes, je suis identifié en tant que mateo21 sur la machine mateo21-desktop.
- **:** : ce symbole à nouveau ne veut rien dire de spécial, c'est un séparateur.
- ~ : ça, c'est le dossier dans lequel vous vous trouvez actuellement. Vous pouvez naviguer de dossier en dossier dans la console et il est très utile qu'on vous rappelle systématiquement avant chaque commande où vous êtes. Pour information, le symbole ~ signifie que vous êtes dans votre dossier personnel, ce qu'on appelle le "Home" sous Linux. C'est l'équivalent du dossier "Mes documents" de Windows.
- **$** : ce dernier symbole est très important, il indique votre niveau d'autorisation sur la machine. Il peut prendre 2 formes différentes :

  $: signifie que vous êtes en train d'utiliser un compte utilisateur "normal", avec des droits limités (il ne peut pas modifier les fichiers système les plus importants). Mon compte mateo21 est donc un compte normal avec des droits limités.

  # : signifie que vous êtes en mode super-utilisateur, c'est-à-dire que vous êtes connecté sous le pseudonyme "root". Le root est l'utilisateur maître qui a le droit de tout faire sur sa machine (même de la détruire). Nous verrons le mode root plus en détails plus tard, pour l'instant nous restons dans un compte utilisateur limité car comme ça nous ne risquons pas de faire de bêtise.

1.3.2 Accéder à l'invité de commande

Sous toute machine Linux, il y a donc non pas une mais 6 consoles qui fonctionnent en simultané (d'où les 6 raccourcis différents de Ctrl + Alt + F1 à Ctrl + Alt + F6).
Au pire, vous changez de terminal jusqu'à retrouver celui où vous êtes. Et dès que vous en avez marre, vous pouvez retourner au mode graphique avec Ctrl + Alt + F7.

L'invité de commande peut être également accéssible via l'interface graphique.

Figure 1.16 : *Accéder à l'invité de commande via l'interface graphique*

1.3.3 Commande et paramètres

La commande est composée du nom de la commande et des paramètres correspondants. **Le manuel d'utilisation** permet de connaitre l'utilité de la commande géneralement elle notée en anglais *Read The Fucking Manual (RTFM).*

Les commandes simples et les plus utilisés sont la commande « date » et « ls ».

```
mateo21@mateo21-desktop:~$ date
vendredi 14 septembre 2007, 18:59:54 (UTC+0200)
```

```
mateo21@mateo21-desktop:~$ ls
Desktop Examples Images
```

Figure 1.17 : *Exemple des commandes et les résultats correspondants*

1.3.4 Paramètres

Les paramètres sont des options que l'on écrit à la suite de la commande. La commande et les paramètres sont séparés par un espace.

```
mateo21@mateo21-desktop:~$ commande parametres
```

Les paramètres peuvent eux-mêmes contenir des espaces, des lettres, des chiffres, en fait un peu de tout. Il n'y a pas de règle véritable sur la forme des paramètres, mais heureusement les programmeurs ont adopté une sorte de "convention" pour que l'on puisse reconnaître les différents types de paramètres.

1.3.4.1 Paramètres courts

Les paramètres les plus courants sont constitués d'une seule lettre, précédée d'un tiret. Comme exemple,

```
commande -d
```

Si on doit donner plusieurs paramètres, on peut faire comme ceci :

```
commande -d -a -U -h
```

Ou, plus court :

```
commande -daUh
```

La commande ls affiche les listes des fichiers et dossiers.

```
mateo21@mateo21-desktop:~$ ls -a
.               .gconfd             .mozilla-thunderbird
..              .gimp-2.2           .nautilus
.bash_history   .gksu.lock          .profile
.bash_logout    .gnome              .recently-used
.bashrc         .gnome2             .recently-used.xbel
.config         .gnome2_private     .ssh
```

Rémarque :

Attention à la casse des paramètres (majuscules / minuscules) ! Si vous écrivez -u, cela n'a en général pas du tout le même sens que -U !

1.3.4.2 Paramètres longs

Les paramètres constitués de plusieurs lettres sont précédés de 2 tirets.

```
commande --parametre
```

Cette fois pas le choix, si vous voulez mettre plusieurs paramètres longs il faudra mettre un espace entre chacun d'eux :

```
commande -daUh --autreparametre
```

On peut aussi combiner les paramètres longs et les paramètres courts dans une commande :

```
commande -daUh --autreparametre
```

Testons sur la commande ls avec le paramètre --all, qui signifie "tout" en anglais :

```
mateo21@mateo21-desktop:~$ ls --all
.              .gconfd            .mozilla-thunderbird
```

1.3.4.3 Valeurs des paramètres

Certains paramètres nécessitent que vous les complétiez avec une valeur. Cela fonctionne différemment selon si vous travaillez avec un paramètre long ou avec un paramètre court. Avec un paramètre court :

```
commande -p 14
```

Cela indique que l'on associe la valeur 14 au paramètre p. Avec ce genre de technique on peut par exemple faire comprendre à l'ordinateur "Je veux voir la liste de tous les fichiers de plus de 14 Mo". Si c'est un paramètre long, on fait en général comme ceci :

```
commande --parametre=14
```

Le résultat sera le même, il est juste plus lisible mais aussi plus long à écrire.

1.3.4.4 Autres paramètres

Il n'y a pas de règle absolue au niveau des paramètres et vous rencontrerez sûrement des paramètres qui fonctionnent différemment.
Heureusement les "conventions" que je viens de vous donner sont valables dans la grande majorité des cas, ce qui devrait vous permettre de vous repérer. Certains paramètres sont donc un peu différents et dépendent vraiment des commandes. Par exemple avec ls, si on ajoute le nom d'un dossier (ou sous-dossier) cela affichera le contenu de ce dossier au lieu du contenu du dossier courant :

```
mateo21@mateo21-desktop:~$ ls Examples
Experience ubuntu.ogg   logo-Ubuntu.png           oo-payment-schedule.ods
fables_01_01_aesop.spx  oo-about-these-files.odt  oo-presenting-
kubuntu.odp
gimp-ubuntu-splash.xcf  oo-about-ubuntu-ru.rtf    oo-presenting-
ubuntu.odp
kubuntu-leaflet.png     oo-cd-cover.odg           oo-trig.xls
logo-Edubuntu.png       oo-derivatives.doc        oo-welcome.odt
logo-Kubuntu.png        oo-maxwell.odt            ubuntu Sax.ogg
```

1.3.5 Retrouver une commande

Linux propose tellement de commandes différentes qu'il est facile de s'y perdre et d'en oublier une. Ça arrive très régulièrement personnellement, et heureusement ce n'est pas un drame. En effet, Linux propose toute une série de façons de retrouver une commande oubliée.

Lister les commandes correspondantes
Tapez juste "da" dans la console, puis **tapez 2 fois sur la touche "Tabulation"** située à gauche de votre clavier. Le résultat sera le suivant :

```
mateo21@mateo21-desktop:~$ da
dash date
mateo21@mateo21-desktop:~$ da
```

En tapant 2 fois sur Tabulation, vous avez demandé à l'ordinateur la liste des commandes qui commencent par "da". On vous a répondu "dash" et "date". Il y a donc 2 commandes qui commencent par "da", et vous venez de retrouver celle que vous cherchiez, c'est-à-dire "date"

L'autocomplétion
Plus sympa encore, s'il n'y a qu'un seul résultat correspondant à votre recherche, l'ordinateur complètera avec les lettres qui manquent et vous n'aurez plus qu'à taper sur Entrée ! Par exemple, il

n'y a qu'une commande qui commence par "dat". Tapez donc dat dans la console, **puis tapez 1 seule fois sur Tabulation**. La commande se complète comme par magie.

Trop de commandes

Parfois, il y a trop de commandes correspondant à votre recherche. Faites un essai un peu brutal : ne rentrez aucun début de commande et faites 2 fois Tab (Tabulation). Cela demande de faire la liste de toutes les commandes disponibles sur votre ordinateur.

```
mateo21@mateo21-desktop:~$
Display all 2173 possibilities? (y or n)
```

A cette question vous pouvez répondre "y" (yes), et la liste s'affichera page par page. Quelques raccourcis à connaître quand une liste s'affiche page par page :

- Tapez Espace pour passer à la page suivante
- Tapez Entrée pour aller à la ligne suivante
- Tapez q pour arrêter la liste

Si vous répondez "n" (no), rien ne se passera. C'est dans le cas où vous vous diriez "Oulah, 2173 possibilités autant chercher une aiguille dans une botte de foin, il faut affiner recherche"

1.3.6 Historique des commandes

On a très souvent besoin de retrouver une commande qu'on vient de taper il y a 5 minutes (ou même 5 secondes). Parfois c'est parce qu'on a oublié la commande, mais souvent c'est aussi parce qu'on a un énooorme poil dans la main comme moi et qu'on a vraiment la flemme de réécrire la commande en entier nous-même.

- Ce raccourci vaut de l'or : appuyez sur **la flèche directionnelle vers le haut** et vous verrez **apparaître la dernière commande** que vous avez tapée. Si vous réappuyez sur la flèche vers le haut, vous verrez l'avant-dernière commande, puis l'avant-avant-dernière etc. Si vous appuyez sur **la flèche vers le bas,** vous reviendrez vers **les commandes les plus récentes**.
- Si vous voulez "remonter" très longtemps en arrière dans l'historique de vos commandes, pas la peine de taper 100 fois sur la flèche vers le haut comme un forcené Il existe **la commande "history"** qui vous rappelle l'historique des commandes :

```
152  date
153  ls
154  ls -a
155  ls --all
156  history
```

La dernière commande tapée sera toujours history, forcément. Vous remarquerez que les commandes sont numérotées : ainsi, on peut savoir que date est la 152 ème commande que j'ai tapée dans le terminal.

- Crtl+R pour la recherche de commande tapée avec quelques lettres. Dans le cas où la flèche vers le haut et la commande history ne suffiraient pas à retrouver une vieille commande que vous avez tapée, il y a un raccourci super utile : Ctrl + R. Appuyez donc sur **les touches Ctrl et R** en même temps, et **l'ordinateur se mettra en mode "recherche d'une commande tapée"** (R comme Recherche). Là, vous pouvez taper n'importe quelle suite de lettres correspondant à une vieille commande. Par exemple, faites Ctrl + R puis tapez "all". Linux retrouve la commande "ls --all" qui contenait justement le mot "all". Vous n'avez plus qu'à taper Entrée pour relancer la commande

```
(reverse-i-search)`all': ls --all
```

1.3.7 Quelques raccourcis claviers pratiques

Ces raccourcis ne sont pas intuitifs, mais ça vaut vraiment le coup de les retenir. Les premiers temps vous reviendrez sûrement souvent ici pour les consulter. Commençons par quelques raccourcis généraux à connaître :

- CTRL+A : efface le contenu de la console. Utile pour faire un peu de ménage quand votre console est encombrée, ou quand votre boss passe derrière et que vous n'aimeriez pas qu'il voie ce que vous étiez en train de faire. A noter qu'il existe aussi **une commande, clear**, qui fait exactement la même chose.
- CTRL+D : envoie le message EOF (fin de fichier) à la console. Si vous tapez ce raccourci dans une ligne de commande vide (c'est-à-dire sans avoir écrit un début de commande avant), cela fermera la console en cours. A noter qu'il existe aussi **la commande exit** qui a le même effet.
- SHIFT + pgUp : vous permet de "remonter" dans les messages envoyés par la console. En mode graphique, la molette de la souris fait aussi très bien ça. La touche "Page Up" est

généralement représentée sur votre clavier par une flèche vers le haut barrée par plusieurs petites barres horizontales.

- SHIFT + pgDown : pareil, mais pour redescendre

Les raccourcis suivants sont utiles lorsque vous êtes en train d'écrire une longue commande :

- CTRL+A : : ramène le curseur au début de la commande. La touche "Origine" a le même effet (elle est située à côté de la touche fin et représentée par une flèche pointant en haut à gauche).
- CTRL+E : ramène le curseur à la fin de la ligne de commande. La touche "Fin" a le même effet.
- CTRL+U : supprime tout ce qui se trouve à gauche du curseur. Si le curseur est situé à la fin de la ligne, toute la ligne sera donc supprimée.
- CTRL+K : : supprime tout ce qui se trouve à droite du curseur. Si le curseur est situé au début de la ligne, toute la ligne sera donc supprimée.
- CTRL+W : supprime le premier mot situé à gauche du curseur. Un "mot" est séparé par des espaces. On s'en sert en général pour supprimer le paramètre situé à gauche du curseur
- CTRL+Y : si vous avez supprimé du texte avec une des commandes Ctrl + U, Ctrl + K ou Ctrl + W : qu'on vient de voir, alors le raccourci Ctrl + Y "collera" le texte que vous venez de supprimer. C'est un peu comme un couper-coller
- CTRCL+C : annulation de commande executée.

1.3.8 Liste des commandes de base

1.3.8.1 La commande pour les manuelles

Le manuel en ligne est accessible par la commande **man**. Il est organisé en différentes sections :

Section	Intitulé
1	les commandes utilisateur
2	les appels système
3	les bibliothèques de programmation
4	les fichiers spéciaux
5	les formats de fichiers
6	les jeux
7	divers
8	les commandes d'administration
9	le noyau

La syntaxe de la commande man est : `man {[}options{]} {[}section{]} commande`

Par défaut, les sections sont parcourues dans l'ordre des numéros. Il est possible de préciser le numéro de section pour résoudre les problèmes de synonymes :
Exemple : **man kill** et **man 2 kill**
La variable d'environnement MANPATH donne le chemin de recherche des pages de manuel. Par exemple : `MANPATH=/usr/man:/usr/local/man`
Il existe une base de mots clés pour la recherche dans les pages de manuel. Cette base est construite par la commande **/usr/sbin/makewhatis**. Les commandes **apropo**s et **whatis** permettent d'effectuer des recherches dans cette base de données.
Les commandes **apropo**s, **whatis et info** est en quelques sortes une commande avec des resumés des resultats de man.
Pour avoir les manuelles aussi les paramètres **--help** ou **-h** sont des bonnes options si vous connaissez le nom de commande.

1.3.8.2 Installation via commande

Sous Windows, vous connaissez ce qu'on appelle des "Programmes d'installation". En général il s'agit d'un .exe à lancer qui s'exécute et extrait les fichiers du programme dans un dossier "Program Files".

Figure 1.18 : *Exemple de programme d'installation en Linux*

Sous Ubuntu, on n'a pas de programmes d'installation. On a ce qu'on appelle **des paquets.**
Un paquet est une sorte de dossier zippé qui contient tous les fichiers du programme. Il se présente sous la forme d'un fichier **.deb**, en reférence à **DEBian**. Il contient toutes les instructions nécessaires pour installer le programme.

Un paquet .deb ressemble au .exe de windows ça fonctionne en fait très différemment notamment :

- Il y a une gestion des dépendances du programme.
- On n'a pas besoin de faire une recherche Google pour trouver un .deb. Tous les .deb sont rassemblés au même endroit sur un même serveur appelé dépôt (repository).

Les dépôts : tous les paquets sont regroupés au sein d'un même endroit appelé dépôt. Il s'agit d'un serveur qui propose tous les paquets qui existent. L'endroit où tous les paquets se trouvent est appelé dépôt (**repository** en anglais).

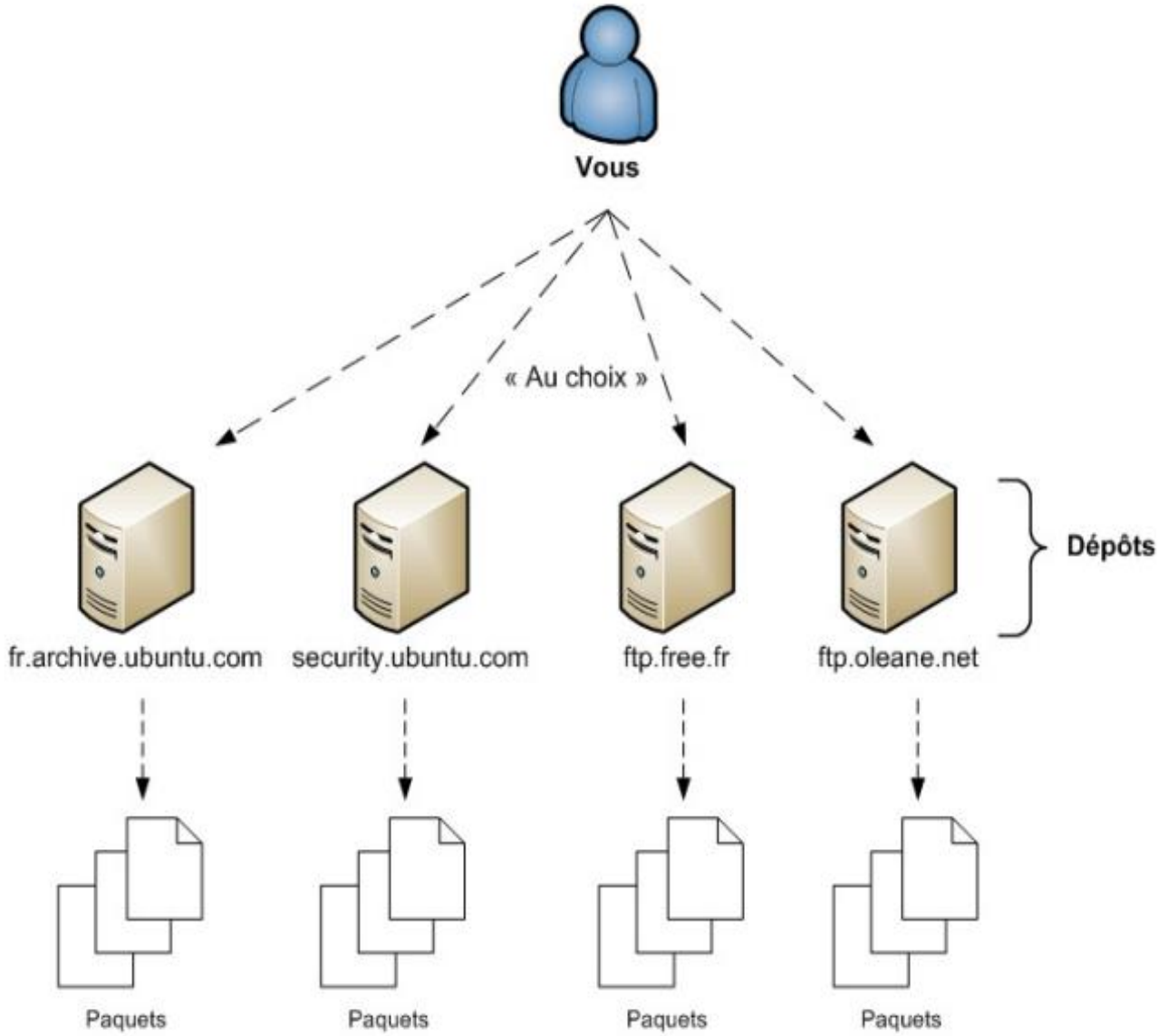


Figure 1.19 : *Illustration graphique des paquets et des dépôts*

Pour gérer des dépôts en Linux, il faut editer via des editeurs de texte le fichier dans le chemin suivant : `/etc/apt/sources.list`

```
sudo nano /etc/apt/sources.list
```

Normalement, chaque ligne du fichier commence par une de ces 2 directives :

- deb : pour télécharger la version compilée (binaire) des programmes. C'est ce que vous voudrez faire dans la plupart des cas, car c'est la version "prête à l'emploi".
- deb-src : permet de récupérer le code source du programme. Généralement, vous n'en avez pas besoin, sauf si vous êtes curieux et que vous voulez voir la source d'un programme (c'est l'avantage du logiciel libre de pouvoir voir la source des programmes !).

```
# See http://help.ubuntu.com/community/UpgradeNotes for how to upgrade to
# newer versions of the distribution.

deb http://fr.archive.ubuntu.com/ubuntu/ hardy main restricted
deb-src http://fr.archive.ubuntu.com/ubuntu/ hardy main restricted

## Major bug fix updates produced after the final release of the
## distribution.
deb http://fr.archive.ubuntu.com/ubuntu/ hardy-updates main restricted
deb-src http://fr.archive.ubuntu.com/ubuntu/ hardy-updates main restricted

## N.B. software from this repository is ENTIRELY UNSUPPORTED by the Ubuntu
## team, and may not be under a free licence. Please satisfy yourself as to
## your rights to use the software. Also, please note that software in
## universe WILL NOT receive any review or updates from the Ubuntu security
## team.
deb http://fr.archive.ubuntu.com/ubuntu/ hardy universe
deb-src http://fr.archive.ubuntu.com/ubuntu/ hardy universe
```

Figure 1.20 : *Extrait des listes de dépôt en Linux*

Remarque :

Il est possible aussi de gérer les paques et les dépôts via l'interface graphique.

Nous allons découvrir 3 nouveaux termes :

- **Paquet :** c'est un programme "prêt à l'emploi", en quelque sorte l'équivalent des programmes d'installation sous Windows.
- **Dépendance :** un paquet peut avoir besoin de plusieurs autres paquets pour fonctionner, on dit qu'il a des dépendances
- **Dépôt :** c'est le serveur sur lequel on va télécharger nos paquets.

Sous Ubuntu, on utilise un programme qui gère les paquets pour nous. Il existe des programmes graphiques, comme Synaptic :

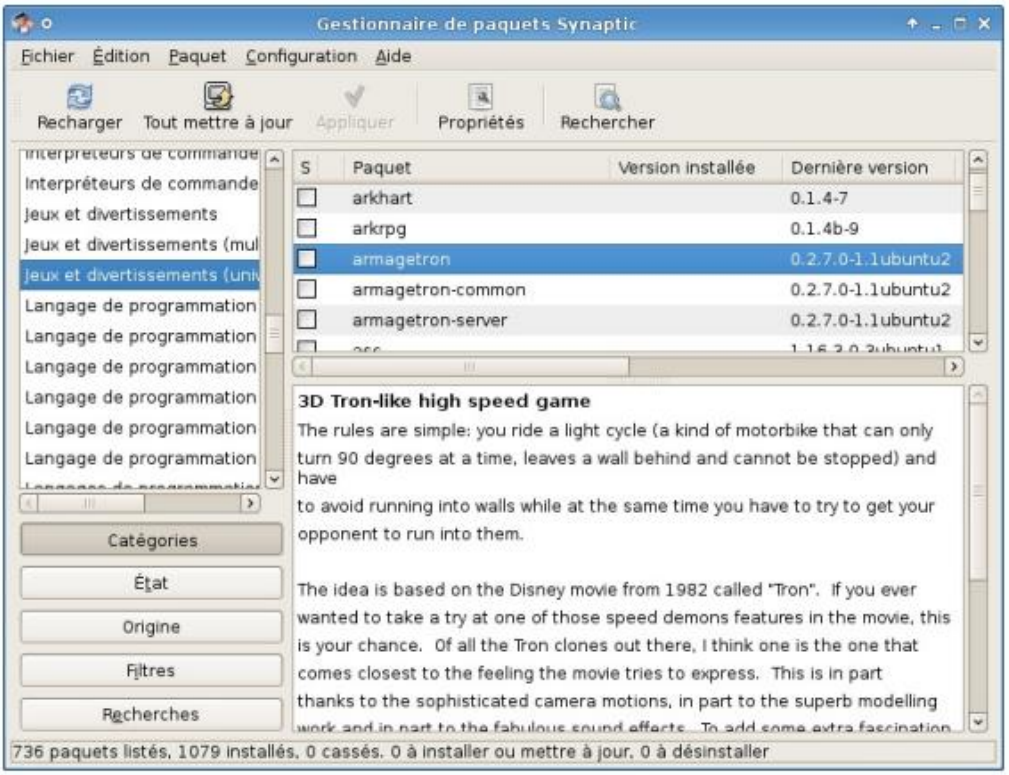


Figure 1.21 : *Mise à jour des dépôts en Linux*

a) Mettre à jour les paquets

La mise à jour des paquets nécessite le fait d'avoir une connexion Internet et se fait par la commande *apt-get update*. Il y a 2 cas où vous avez besoin de mettre à jour votre cache :

- Quand vous changez / ajoutez un dépôt à votre liste de dépôts.
- Si vous n'avez pas mis à jour votre cache depuis un moment (quelques semaines).

```
root@mateo21-desktop:~# apt-get update
Réception de : 1 http://wine.budgetdedicated.com hardy Release.gpg [191B]
Ign http://wine.budgetdedicated.com hardy/main Translation-fr
Atteint http://wine.budgetdedicated.com hardy Release
Atteint ftp://ftp.free.fr hardy Release.gpg
Ign http://wine.budgetdedicated.com hardy/main Packages
Atteint ftp://ftp.free.fr hardy/restricted Translation-fr
Atteint http://wine.budgetdedicated.com hardy/main Sources
Atteint ftp://ftp.free.fr hardy/main Translation-fr
Atteint http://wine.budgetdedicated.com hardy/main Packages
Atteint ftp://ftp.free.fr hardy/universe Translation-fr
```

Figure 1.22 : *Extrait de reponse de mise à jour*

b) Recherche de paquets

A moins que vous ne connaissiez déjà le nom exact du paquet que vous voulez, il va falloir faire une petite recherche.

On utilise pour cela la commande suivante :

```
apt-cache search votrerecherche
```

Cette commande effectue une recherche de paquet dans votre cache. Cela évite d'avoir à aller sur internet pour faire la recherche, ce qui aurait été lent.

```
root@mateo21-desktop:~# apt-cache search breakout
briquolo - Fast paced 3d Breakout
briquolo-data - Fast paced 3d Breakout data files
circuslinux - The clowns are trying to pop balloons to score points!
circuslinux-data - data files for circuslinux
gnome-breakout - Clone of the classic game Breakout, written for GNOME
lbreakout2 - A ball-and-paddle game with nice graphics
lbreakout2-data - A ball-and-paddle game with nice graphics (DATA FILES)
libfreebob0 - FreeBoB API
libfreebob0-dev - FreeBoB API - development files
tecnoballz - breaking block game ported from the Amiga platform
```

c) Installation des paquets

Pour installere un paquet, il faut utiliser la commande *apt-get install*

```
root@mateo21-desktop:~# apt-get install lbreakout2
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture de l'information d'état... Fait
Les paquets supplémentaires suivants seront installés :
  lbreakout2-data libsdl-mixer1.2 libsmpeg0
Les NOUVEAUX paquets suivants seront installés :
  lbreakout2 lbreakout2-data libsdl-mixer1.2 libsmpeg0
0 mis à jour, 4 nouvellement installés, 0 à enlever et 153 non mis à jour.
Il est nécessaire de prendre 2943ko dans les archives.
Après dépaquetage, 5358ko d'espace disque supplémentaires seront utilisés.
Souhaitez-vous continuer [O/n] ? O
Réception de : 1 ftp://ftp.free.fr feisty/main libsmpeg0 0.4.5+cvs20030824-
1.9build1 [105kB]
Réception de : 2 ftp://ftp.free.fr feisty/main libsdl-mixer1.2 1.2.6-
1.1build1 [145kB]
Réception de : 3 ftp://ftp.free.fr feisty/universe lbreakout2-data 2.5.2-
2.1ubuntu1 [2444kB]
Réception de : 4 ftp://ftp.free.fr feisty/universe lbreakout2 2.5.2-
2.1ubuntu1 [249kB]
2943ko réceptionnés en 6s (484ko/s)
Sélection du paquet libsmpeg0 précédemment désélectionné.
(Lecture de la base de données... 123350 fichiers et répertoires déjà installés.)
```

Figure 1.23 : *Installation de paquets sous linux*

d) Supprimer un paquet

Si vous voulez désinstaller un paquet, vous pouvez utiliser la commande *apt-get remove*.

```
apt-get remove lbreakout2
```

Pour supprimer également les dépendances inutiles avec la supression des paquets, vous pouvez utiliser la commande *apt-get autoremove*.

```
root@mateo21-desktop:~# apt-get autoremove lbreakout2
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture de l'information d'état... Fait
Les paquets suivants ont été automatiquement installés mais ne sont plus nécessaire
  libsdl-mixer1.2 libsmpeg0
Les paquets suivants seront ENLEVÉS :
  lbreakout2 lbreakout2-data libsdl-mixer1.2 libsmpeg0
0 mis à jour, 0 nouvellement installés, 4 à enlever et 0 non mis à jour.
Il est nécessaire de prendre 0o dans les archives.
Après dépaquetage, 5358ko d'espace disque seront libérés.
Souhaitez-vous continuer [O/n] ?
```

e) Mettre à jour tous les paquets

Une autre fonctionnalité particulièrement géniale d'apt-get est sa capacité à mettre à jour tous les paquets installés sur votre système d'un seul coup. Le programme ira chercher les nouvelles versions de tous vos programmes et les mettra à jour s'il y a une nouvelle version de disponible.

```
root@mateo21-desktop:~# apt-get upgrade
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture de l'information d'état... Fait
0 mis à jour, 0 nouvellement installés, 0 à enlever et 0 non mis à jour.
```

f) Les autres gestionnaires de paquets

La commande apt-get n'existe que sous Debian et ses dérivés, dont Ubuntu fait partie. Les autres distributions possèdent en général leur propre système de gestion des paquets. Citons :

- **rpm :** le système de gestion de paquets utilisé par la distribution Red Hat, qui reste très utilisé, mais qui ne gère malheureusement pas les dépendances
- **yum :** une surcouche de rpm gérant les dépendances, utilisé par la distribution Fedora.
- **urpmi :** une surcouche de rpm gérant les dépendances, utilisé par la distribution Mandriva.
- **emerge :** le gestionnaire de paquets de Gentoo, qui compile toujours à partir des sources (il ne télécharge jamais le programme binaire directement)

1.4 Configuration noyau

1.4.1 Evolution des noyeaux

La figure suivante represente l'évolution des noyeaux des ordinateures.

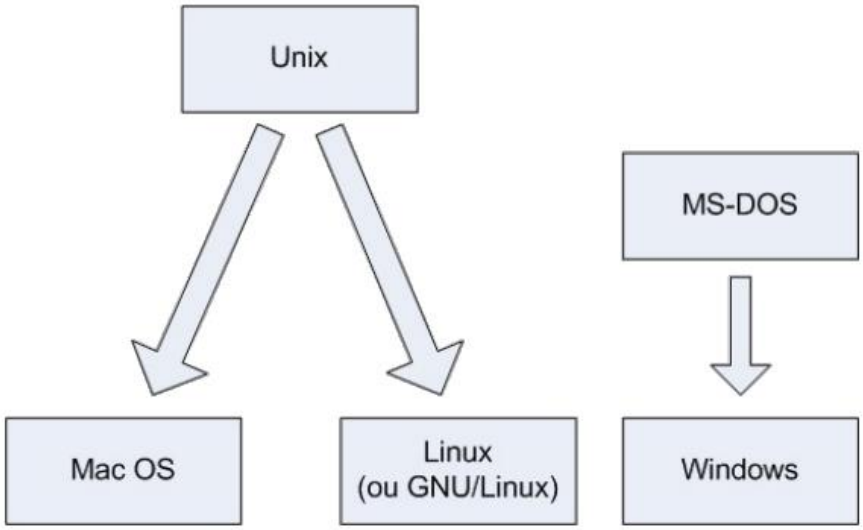


Figure 1.24 : *Evolution des noyaux des ordianteurs*

1.4.2 Structure du noyau et representation du code source

Ce paragraphe contient quelques informations importantes sur le noyau, mais ne constitue pas la référence en la matière. Des ouvrages entiers sont consacrés à ce sujet (un lien utile est le HOWTO du noyau : http://www.freenix.fr/unix/linux/HOWTO/Kernel-HOWTO.html). La maîtrise du noyau n'est pas une tâche facile et ne s'entend que par une étude approfondie. Il n'est heureusement pas nécessaire d'être un expert du noyau pour utiliser Linux.

Comme tout système d'exploitation, le noyau Linux est une interface entre des programmes utilisateurs et le matériel physique. L'accès à ce matériel se fait par l'intermédiaire d'appels systèmes qui sont identiques quelle que soit la machine. Cette encapsulation du matériel libère les développeurs de logiciels de la gestion complexe des périphériques : c'est le système d'exploitation qui s'en charge. En bref, le noyau Linux, vu par l'utilisateur comme un seul et unique fichier, est composé de quatre sous-systèmes principaux. On peut le représenter graphiquement sous la forme d'une couche s'intercalant entre le matériel et le logiciel.

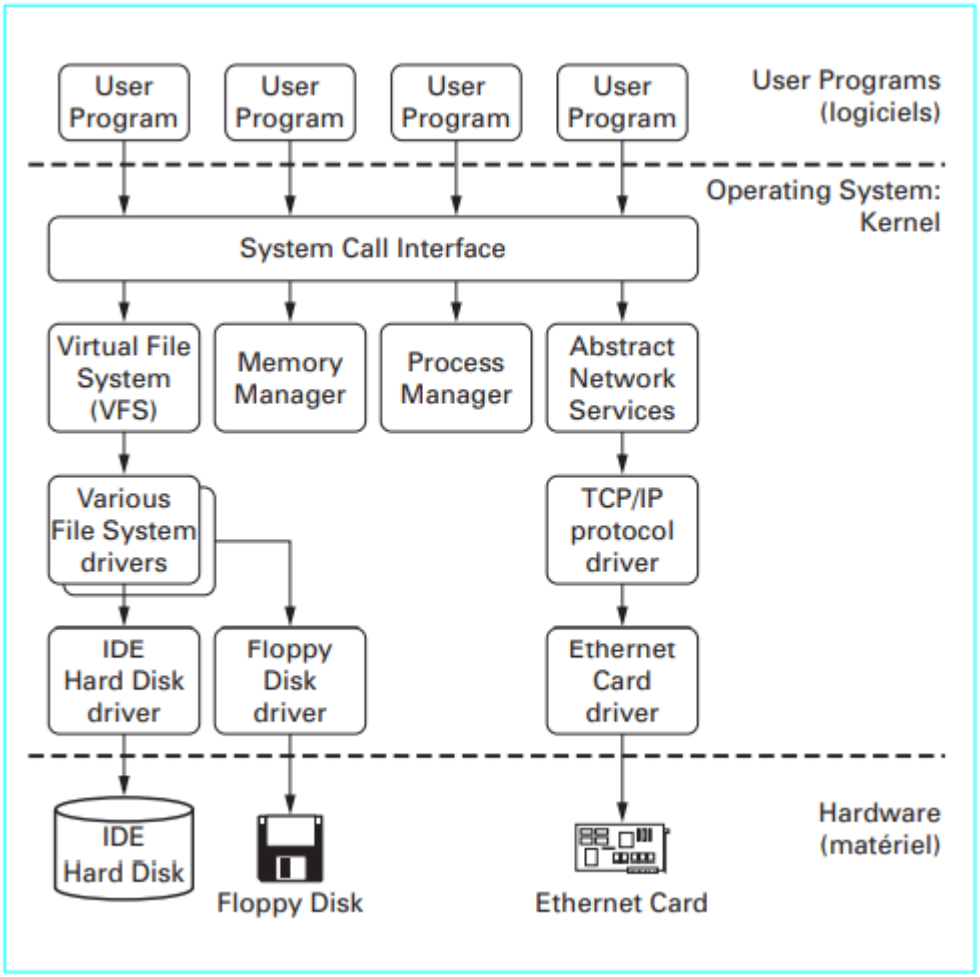


Figure 1.25 : *Structure du noyeau*

1.4.3 Version du noyau

La commande `uname` permet d'avoir les informations sur le système et dont la version du noyau qui tourne au moment de la commande.

#uname -a

La commane affichera :

- Le nom du noyau : Linux
- Le nom de l'hôte : mateo21
- Le « release » du noyau (kernel release) : 2.6.32.43-generic
- La version du noyau (kernel version) : #91-Ubuntu SMP Wed Sep 5 16 :43 :09 UTC 2012
- L'architecture materielle : i686
- Le système d'exploitation : GNU/Linux

Mais plus facilement, la version du noyau en cours peut être trouvé facilement avec la commande.
#cat /proc/version
La version d'un noyau linux est formée d'une suite de 3 ou 4 nombres sous forme suivante : 3.0.x ou 3.0.y

- 3 : numéro de version majeure du noyau. Le nombre correspondant à la « génération » du noyau. La sortie de cette troisième génèration a été annoncée par Linus Torvalds en 2011
- 0 : Numéro de version mineure du noyau
- x : Numéro de révision du noyau
- y : Numéro de « patch level » du noyau

1.4.4 Compilation d'un noyau

Compiler un noyau consiste à :

- Acquérir les sources du noyay à utiliser
- Configurer selon les paramètres de votre choix et selon les modules à rendre disponible au niveau du noyeau
- Compiler ce programme selon l'architecture utilisée
- Faire en sorte de pouvoir démarrer avec le noyau ainsi obtenu (compilé et compressé)

Pour ce faire, il faut donc plusieurs outils de récupération de source (selon les methodes qui conviennent : par un des protocoles du réseau ou par copie des fichiers), de configuration (choix par une interface en texte ou sous forme de menu de choix ou sous forme graphique), de compilation, de compression et de configuration du chargeur de demarrage (bootloader).

1.4.4.1 Récupération des sources

#cd /usr/src
wget ft://ftp.kernel.org/pub/linux/kernel/v3.0/linux-3.025.tar.bz2

1.4.4.2 Decompression est archivage

#tar -xjfv linux-3.025.tar.bz2
#rm linux
#ln -s linux-3.0.25 linux

1.4.4.3 Configuration

cd linux
#make menuconfig
Veuillez notez qu'il faut l'application ncurses, ncurses-dev, automake, autoconf, Qt3 (pour la configuration graphique) sont requis. Les configurations suivantes sont également valables :

- make config propose une longue suite de question reponse (y/n/m)
- make xconfig propose une configuraiton en mode graphique (Qt)
- make gconfig propose une configuration en mode graphique (GTK)

1.4.4.4 Compilation

#make

1.4.4.5 Installation

#make modules_install
#make install

1.4.4.6 Chargeur de demarrage

#make initrd
Cette commande recrée une nouvelle ram disk pour chacun des noyeaux trouvés dans /boot/
Il est possible de vérifier les configurations manuellement dans les fichiers de configuration de GRUB par exemple.

1.4.4.7 Redémarrage

#reboot

1.4.5 Appliquer un « patch » au noyau

Il s'agit du procédé qui consiste à ajouter de(s) fonctionnalité(s) sans changer la version de noyau. Voici un exemple de commande

1.4.5.1 Récuperer le patch

#wget patch-<version>.tar.bz2

#tar -xjvf patch-<version>.tar.bz2 /usr/src/linux

1.4.5.2 Appliquer le patch

#cd /usr/src/linux
#patch -pl < patch-<version>.

1.4.5.3 Compiler

Recommencer ici toute la chaine de recompilation : configuration, compilation, installation, et configuration du chargeur de demarrage.

1.4.6 Modification des paramètres du noyau sans recompilation

1.4.6.1 Module

Une fois l'option « modules » activée lors de la configuration d'un noyau, il est possible d'inserer à chaud des modules ou drivers utiles au fonctionnement du système. On utilisera les commandes : *lsmod, insmod, modprobe et depmod* pour gerer les modules.

Les modules en question se trouvent d'habitudes dans le repertoire /lib/modules/<version_du_kernel> et se présentent sous forme de *nom_de_module.o* ou *nom_de_module.ko*

Un module chargé en cours de fonctionnement du système ne se trouvera pas chargé au prochain redémaragge. Noter que le fichier /etc/modules et le contenu du repertoire /etc/modprobe.d/ servent à rendre persistant les chargements ou paramètres à passer au noyau.

1.4.6.2 Paramètres du noyau

Les paramètres courant se trouvent dans le répertoire /proc/sys, il est possible demodifier ces paramètres. Par exemple
#echo 1 > /proc/sys/net/ipv4/ip_forward

Le fichier /etc/sysctl.conf permet de rendre persistant ces valeurs. La commande sysctl permet également de manipuler ces valeurs.

#sysctl -a
(Affiche tous les paramètres courants du noyau)
#sysctl -p
(Permet de changer les paramètres du noyau selon les valeurs définies dans le fichier /etc/sysctl.conf)

1.4.7 Gestion des bibliothèques

Une bibliothèque est un ensemble de fonctions d'habitudes d'utilisation courante rassemblés dans un « paquétage ». Les programmeurs utilisent ces fonctions déjà existantes afin de ne plus réinventer les roues.

La manière dont les programmes utilisent les bibliothèques peut être de façons statiques quand les bibliothèques sont intégrées dans le programme au moment de sa compilation. Ces bibliothèques se présentent sous forme de <fichier.a> tandis que les bibliothèques dynamiques (ou partagées) sont appelées par les programmes au moment de leurs exécutions, elles sont de la forme <fichier.so>
Pour avoir des informations sur les bibliothèques utilisées par un programme donnée, il y a la commande ldd. Elle peermet également de résoudre les problèmes rélatifs à une bibliothèque manquante par exemple.

Pour le cas des bibliothèques partagées, un programme cherche ceux dont il a besoin dans certains répertoires du système. Ces chemins sont définis dans la variable LD_LIBRARY_PATH. Il est donc possible d'ajouter certains chemins à la valeur de certaines variables pour résoudre momentanément un problème de bibliothèque sans que la configuration soit persistante. Le contenu de cette variable peut être configuré dans le fichier /etc/ld.so.conf.

La commande ldconfig lit le fichier /etc/ld.so.conf et génère une cache dans le fichier /etc/ld.so.cache. L'option -p de ldconfig permet d'avoir la liste et les informations sur les bibliothèques actuellement insèrées en cache.

CHAPITRE 2 ADMINISTRATION LINUX

2.1 Gestion de comptes utilisateurs

La grande puissance de Linux, c'est d'être :

- Multitâche
- Multi-utilisateurs

Multitâche signifie qu'on peut démarrer plusieurs programmes à la fois. Ça on a déjà l'habitude de le faire avec Windows. Multi-utilisateurs signifie que plusieurs personnes peuvent travailler sur le même OS en même temps. Ça normalement c'est un peu nouveau. En effet, lorsque vous utilisez l'ordinateur, vous avez l'habitude d'être "seul" dessus. Sous Linux, on peut très bien être 15 personnes à utiliser la même machine en même temps.

2.1.1 La commande sudo (executer en tant que root)

Lorsque vous avez installé Ubuntu, on vous a demandé le nom du compte utilisateur que vous vouliez créer. Moi j'ai créé l'utilisateur "mateo21" par exemple. Dans la plupart des distributions Linux, on vous proposera de créer un compte utilisateur avec des droits limités, comme c'est le cas pour mon compte "mateo21".

C’est une sécurité. Comme c'est vous l'patron, vous pouvez à tout moment dire "Bon allez je passe en mode chef-qui-peut-tout-faire". Mais c'est une sécurité de ne pas avoir le droit de tout faire par défaut, car certaines commandes peuvent être dangereuses pour la stabilité et la sécurité de votre ordinateur. Avoir des droits limités, ça veut dire aussi par exemple qu'on s'empêche d'exécuter la commande de la Mort (la commande rm -rf /* par exemple c.f. paragraphe la commande rm).

2.1.2 Organisation des utilisateurs en linux

On peut créer autant d'utilisateurs que l'on veut, eux-mêmes répartis dans des groupes. Il y a un utilisateur "spécial" appelé root, aussi appelé super-utilisateur. Il a tous les droits sur la machine.

Au départ chez moi, 2 utilisateurs sont créés : root et mateo21. On ne se connecte en root **que très rarement, lorsque c'est nécessaire**. **Certaines commandes** de Linux ne sont accessibles **qu'à root**.

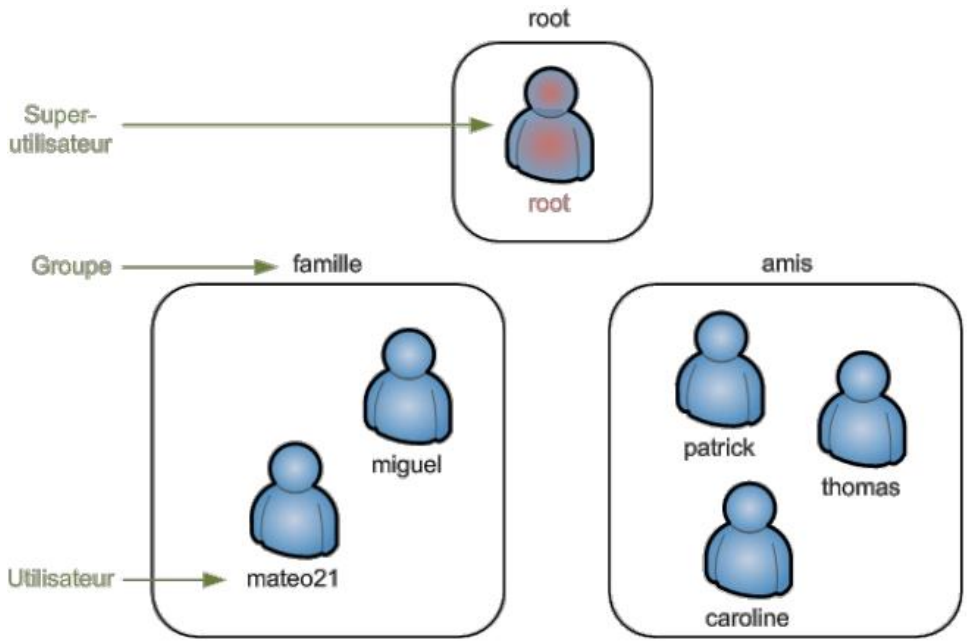


Figure 2.01 : *Organisateur des utilisateurs sous linux*

2.1.3 Commande sudo devenir root en un instant

Par défaut, vous êtes connecté sous **votre compte limité** (mateo21 pour ma part). Il est impossible sous Ubuntu de se connecter directement en root au démarrage de l'ordinateur.

Comment faire alors pour exécuter des commandes que seul root a le droit d'exécuter ?

On peut devenir **root temporairement** à l'aide de la commande sudo. Cette commande signifie "Faire en se substituant à l'utilisateur" : **S**ubstitute **U**ser **DO.**

```
sudo commande
```

```
mateo21@mateo21-desktop:/home$ sudo ls
[sudo] password for mateo21:
autredossier   Desktop    Examples  Images  Modèles     Musique  tutos
autresanimaux  Documents  images    log     mondossier  Public   Vidéos
```

2.1.4 Commande sudo su devenir root tout en restant

Si vous tapez sudo su (tout court), vous passerez root indéfiniment.

```
mateo21@mateo21-desktop:/home$ sudo su
[sudo] password for mateo21:
root@mateo21-desktop:/home#
```

Le symbole # à la fin de l'invite de commandes vous indique que vous êtes devenu super-utilisateur. Vous pouvez alors exécuter autant de commandes en root que vous le voulez.

Pour quitter le "mode root", tapez la **commande exit** (ou faites **la combinaison Ctrl + D**).

```
root@mateo21-desktop:/home/mateo21# exit
exit
mateo21@mateo21-desktop:~$
```

Voici la traduction de la Figure 1.27 :

- Fais-moi un sandwich.
- Quoi ? Fais-le toi-même.
- Sudo fais-moi un sandwich.
- Ok.

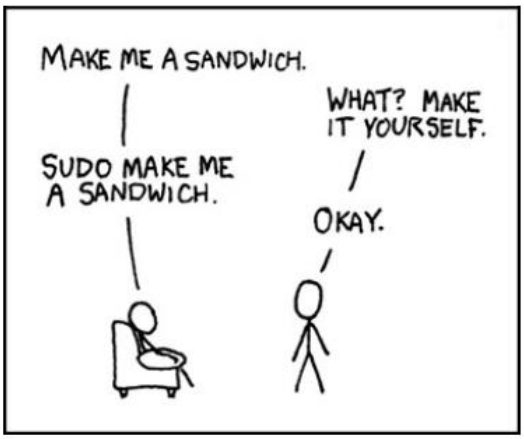


Figure 2.02 : *Illustration de la commande sudo*

2.1.5 Commande adduser pour ajouter des utilisateurs

Maintenant que vous savez passer root (temporairement ou indéfiniment), nous allons pouvoir découvrir des commandes qui sont réservées à root. La commande *adduser et deluser* sont de celles-là. Si vous essayez de les appeler avec votre utilisateur normal, on vous dira que vous n'avez pas le droit de les utiliser. Seul root peut gérer les utilisateurs.

```
root@mateo21-desktop:/home# adduser patrick
Ajout de l'utilisateur « patrick »...
Ajout du nouveau groupe « patrick » (1001)...
Ajout du nouvel utilisateur « patrick » (1001) avec le groupe « patrick »...
Création du répertoire personnel « /home/patrick »...
Copie des fichiers depuis « /etc/skel »...
```

La commande adduser permet d'ajouter un utilisateur. Vous devez au minimum fournir un paramètre : le nom de l'utilisateur à créer. Par exemple, pour créer un compte pour patrick.
Si vous tentez d'exécuter la commande avec votre compte limité, vous aurez une erreur de ce genre : *"adduser : Seul le superutilisateur peut ajouter un utilisateur ou un groupe sur le système*". Le répertoire personnel de patrick est automatiquement créé (/home/patrick) et son compte est préconfiguré. On vous demande ensuite de taper son mot de passe :

```
Entrez le nouveau mot de passe UNIX :
Retapez le nouveau mot de passe UNIX :
passwd : le mot de passe a été mis à jour avec succès
```

Tapez le mot de passe de Patrick, puis faites Entrée. Retapez le mot de passe pour valider. Encore une fois, si vous ne voyez pas d'étoiles * quand vous tapez le mot de passe, c'est normal. C'est une sécurité pour qu'on ne puisse pas compter le nombre de caractères derrière votre épaule.

2.1.6 Commande passwd pour changer le mot de passe

S'il était nécessaire de changer le mot de passe de patrick par la suite, utilisez la commande passwd en indiquant en paramètre le nom du compte à modifier.

```
root@mateo21-desktop:/home# passwd patrick
Entrez le nouveau mot de passe UNIX :
Retapez le nouveau mot de passe UNIX :
passwd : le mot de passe a été mis à jour avec succès
```

2.1.7 Commande deluser pour supprimer un compte

Patrick vous ennuie ? Patrick est parti ? Si son compte n'est plus nécessaire (ou que vous voulez vous venger). Vous pouvez le supprimer avec deluser.

```
deluser patrick
```

Aucune confirmation ne vous sera demandée !
Remarque :
Surtout, ne supprimez pas votre compte utilisateur ! Par exemple, je ne dois surtout pas supprimer le compte mateo21. En effet, si je le fais, il n'y aura plus que root sur la machine... et Ubuntu interdit de se logger en root. Par conséquent, au prochain démarrage de la machine vous ne pourrez pas vous connecter... et vous serez complètement coincé !

Toutefois, cette commande seule ne supprime pas le répertoire personnel de Patrick. Si vous voulez supprimer aussi son home et tous ses fichiers personnels, utilisez le paramètre `--remove-home` :

```
deluser --remove-home patrick
```

Remarque :
Les commandes adduser et deluser sont des commandes qui n'existent que dans Debian et tous ses descendants.

2.1.8 Commande pour un groupe

Chaque utilisateur appartient à un groupe. Dans ce cas, à quel groupe appartiennent les utilisateurs mateo21 et patrick ? On n'a rien défini nous !

En effet, si vous ne définissez rien, un groupe du même nom que l'utilisateur sera automatiquement créé.
Ainsi, mateo21 appartient au groupe mateo21, et patrick au groupe patrick. On peut le vérifier en regardant à qui appartiennent les dossiers dans /home via un `ls -l` :

```
root@mateo21-desktop:~# cd /home
root@mateo21-desktop:/home# ls -l
total 24
drwx------  2 root    root    16384 2007-09-19 18:22 lost+found
drwxr-xr-x 65 mateo21 mateo21  4096 2007-11-15 22:40 mateo21
drwxr-xr-x  2 patrick patrick  4096 2007-11-15 23:00 patrick
```

La 3ème colonne indique le propriétaire du fichier ou dossier, et la 4ème indique le groupe qui possède ce fichier ou dossier.

- Ainsi, le dossier mateo21 appartient à l'utilisateur mateo21 et au groupe mateo21.
- De même pour patrick. On constatera par ailleurs que lost+found appartient à root, et qu'il y a un groupe root (root fait donc partie du groupe root).

Bon, si tout le monde est dans son propre groupe, quel intérêt me direz-vous ?

Vous pourriez très bien vous contenter de ce fonctionnement (un utilisateur = un groupe), mais au cas où vous ayez beaucoup d'utilisateurs, il faut utiliser la commande addgroup.

2.1.8.1 Commande addgroup pour ajouter un groupe

La commande addgroup crée un nouveau groupe. Vous avez juste besoin de spécifier le nom du groupe en paramètre :

```
root@mateo21-desktop:/home# addgroup amis
Ajout du groupe « amis » (identifiant 1002)...
Terminé.
```

2.1.8.2 Commande usermode pour modifier un utilisateur

La commande usermod permet d'éditer un utilisateur. Elle possède plusieurs paramètres, on va en retenir 2 :

- -l : renomme l'utilisateur (le nom de son répertoire personnel ne sera pas changé par contre)
- -g : change de groupe

Si je veux mettre patrick dans le groupe amis, je ferai donc comme ceci :

```
usermod -g amis patrick
```

Et pour remettre patrick dans le groupe patrick comme il l'était avant :

```
usermod -g patrick patrick
```

Remarque :

Il est aussi possible de faire en sorte qu'un utilisateur appartienne à plusieurs groupes. Pour ce faire, utilisez le paramètre -G (majuscule).

Exemple : `usermod -G amis, paris, collegues patrick`

Séparez les noms des groupes par une virgule, sans espace entre chaque nom de groupe.

Rémarque :

Faites très attention en utilisant usermod ! Lorsque vous utilisez -G, l'utilisateur change de groupe peu importe les groupes auxquels il appartenait auparavant. Si vous voulez ajouter des groupes à un utilisateur (sans perdre les groupes auxquels il appartenait auparavant),

Utilisez -a : `usermod -aG amis patrick`

2.1.8.3 Commande delgroup pour supprimer un groupe

Si vous voulez supprimer un groupe, c'est tout simple :

```
delgroup amis
```

Remarque :
Les commandes addgroup et delgroup n'existent pas ailleurs que sous Debian et ses dérivés (même remarque que pour adduser et deluser). Les commandes "traditionnelles" qui marchent partout sont *groupadd et groupdel*, mais elles offrent moins d'options.

2.1.8.4 Commande pour changer le propriétaire d'un fichier

L'utilisateur root, et seulement lui, peut changer le propriétaire d'un fichier. Par exemple, supposons que mateo21 possède dans son répertoire personnel un fichier appelé "rapport.txt". Voici le résultat d'un `ls -l` pour ce fichier :

```
mateo21@mateo21-desktop:~$ ls -l rapport.txt
-rw-r--r-- 1 mateo21 mateo21 0 2007-11-15 23:14 rapport.txt
```

Petite astuce : comme vous venez de le voir, si on précise un nom de fichier en dernier paramètre de la commande ls, on ne verra que ce fichier dans les résultats. Le joker * est là aussi utilisable : ls -l *.jpg afficherait uniquement les images JPEG contenues dans ce dossier.
Ce fichier, je souhaite le "donner" à patrick. C'est là qu'intervient la commande chown.

2.1.8.5 Commande chown pour changer le proprietaire d’un fichier

La commande chown, qui doit être utilisée en tant que root, attend 2 paramètres au moins :

- Le nom du nouveau propriétaire
- Le nom du fichier à modifier

```
chown patrick rapport.txt
```

On peut voir ensuite que patrick est bien le nouveau propriétaire du fichier :

```
root@mateo21-desktop:/home/mateo21# ls -l rapport.txt
-rw-r--r-- 1 patrick mateo21 0 2007-11-15 23:14 rapport.txt
```

Seulement... il appartient toujours au groupe mateo21 !

2.1.8.6 Commande chgrp pour changer le groupe proprietaire d'un fichier

La commande *chgrp* s'utilise exactement de la même manière que chown, à la différence près qu'il affecte cette fois le groupe propriétaire d'un fichier.

```
chgrp amis rapport.txt
```

Le but est d'affecter le fichier rapport.txt au groupe "amis".
Un petit *ls -l* nous confirmera que rapport.txt appartient désormais à patrick et au groupe amis :

```
root@mateo21-desktop:/home/mateo21# ls -l rapport.txt
-rw-r--r-- 1 patrick amis 0 2007-11-15 23:14 rapport.txt
```

La commande *chown* peut aussi changer le groupe proprietaire d'un fichier.

```
chown patrick:amis rapport.txt
```

Le but est d'affecter le fichier à l'utilisateur patrick et au groupe amis.
Il suffit de séparer par un symbole deux-points ":" le nom du nouvel utilisateur (à gauche) et le nom du nouveau groupe (à droite).
L'option -R dans la commande *chown* permet aussi de faire la modification de manière recursive.

2.1.9 Commande chmod pour attribuer des droits d'accès

2.1.9.1 Fonctionnement des droits

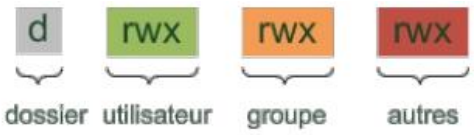
Chaque fichier et chaque dossier possèdent une liste de droits. C'est une liste qui dit qui a le droit de voir le fichier, de le modifier et de l'exécuter. Vous avez déjà vu des listes de droits, oui oui ! Lorsque vous faites un `ls -l`, c'est la première colonne :

```
mateo21@mateo21-desktop:~$ ls -l
total 40
drwxr-xr-x 2 mateo21 mateo21 4096 2007-11-13 21:53 Desktop
drwxr-xr-x 2 mateo21 mateo21 4096 2007-11-13 13:46 Documents
lrwxrwxrwx 1 mateo21 mateo21   26 2007-09-19 18:31 Examples -
> /usr/share/example-content
drwxr-xr-x 2 mateo21 mateo21 4096 2007-09-25 20:28 images
drwxr-xr-x 2 mateo21 mateo21 4096 2007-10-19 01:21 Images
drwxr-xr-x 3 mateo21 mateo21 4096 2007-09-25 11:11 log
drwxr-xr-x 2 mateo21 mateo21 4096 2007-10-19 01:21 Modèles
drwxr-xr-x 2 mateo21 mateo21 4096 2007-10-19 01:21 Musique
drwxr-xr-x 2 mateo21 mateo21 4096 2007-10-19 01:21 Public
-rw-r--r-- 1 mateo21 mateo21    0 2007-11-15 23:14 rapport.txt
drwxr-xr-x 3 mateo21 mateo21 4096 2007-09-19 19:51 tutos
drwxr-xr-x 2 mateo21 mateo21 4096 2007-10-19 01:21 Vidéos
```

Vous voyez tous ces d, r, w et x au début ? Ce sont ce qu'on appelle les droits d'accès du fichier ou dossier. On peut voir 5 lettres différentes. Voici leurs significations :

- d (Directory) : indique si l'élément est un dossier.
- l (Link) : indique si l'élément est un lien (raccourci).
- r (Read) : indique si on peut lire l'élément.
- w (Write) : indique si on peut modifier l'élément.
- x (eXecute) : si c'est un fichier, "x" indique qu'on peut l'exécuter. Ce n'est utile que pour les fichiers exécutables (programmes et scripts). Si c'est un dossier, "x" indique qu'on peut le "traverser", c'est-à-dire qu'on peut voir les sous-dossiers qu'il contient si on a le droit de lecture dessus.

Si la lettre apparaît, c'est que le droit existe. S'il y a un tiret à la place, c'est qu'il n'y a pas de droit.
Les droits sont découpés en fonction des utilisateurs :



Le premier élément ("d") mis à part, on constate que r, w et x sont répétés 3 fois en fonction des utilisateurs :

- Le premier triplet rwx indique les droits que possède le **propriétaire** du fichier sur ce fichier.
- Le second triplet rwx indique les droits que possèdent les autres membres **du groupe** sur ce fichier.
- Enfin, le dernier triplet rwx indique les droits que possèdent tous **les autres** utilisateurs de la machine sur ce fichier. Prenons un cas concret, le fichier rapport.txt :

```
mateo21@mateo21-desktop:~$ ls -l rapport.txt
-rw-r--r-- 1 mateo21 mateo21 0 2007-11-15 23:14 rapport.txt
```

Ses droits sont :

- **- :** le premier tiret indique qu'il ne s'agit pas d'un dossier. S'il y avait eu un "d" à la place, cela aurait indiqué qu'il s'agissait d'un dossier.
- **rw- :** indique que le propriétaire du fichier, mateo21 en l'occurence, peut lire et modifier (et donc supprimer) le fichier. En revanche, il ne peut pas l'exécuter car il n'a pas de x à la fin. Je rappelle que quiconque peut modifier un fichier a aussi le droit de le supprimer.
- **r-- :** tous les gens qui font partie du groupe "mateo21" mais qui ne sont pas "mateo21" peuvent seulement lire le fichier. Ils ne peuvent ni le modifier, ni l'exécuter. Je reconnais qu'avoir un nom de groupe identique au nom d'utilisateur peut embrouiller : si vous êtes aussi bien organisé que sur mon premier schéma, on parlerait plutôt du groupe "famille".
- **r-- :** tous les autres (ceux qui ne font pas partie du groupe "mateo21") peuvent seulement lire le fichier.

En résumé, ces droits nous apprennent que l'élément est un fichier, que mateo21 peut le lire et le modifier, et que tous les autres utilisateurs peuvent seulement le lire.

Remarque :

Root a TOUS les droits. Il peut tout faire : lire, modifier, exécuter n'importe quel fichier.

2.1.9.2 Commande chmod pour modifier les droits d'accès

Une précision importante pour commencer : contrairement aux commandes précédentes, vous n'avez pas besoin d'être root pour utiliser *chmod*. Vous devez juste être propriétaire du fichier dont vous voulez modifier les droits d'accès. La commande *chmod* est un petit peu délicat à utiliser. En effet, on peut attribuer les droits sur un fichier / dossier via plusieurs méthodes différentes, la plus courante étant celle des chiffres.

- Attribuer des droits avec des chiffres (chmod absolu)

Il va falloir faire un petit peu de calcul mental. En effet, on attribue un chiffre à chaque droit :

Droit	Chiffre
r	4
w	2
x	1

Si vous voulez combiner ces droits, il va falloir additionner les chiffres correspondants. Ainsi, pour attribuer le droit de lecture et de modification, il faut additionner 4 + 2, ce qui donne 6. **Le chiffre 6** signifie donc **"Droit de lecture et d'écriture".** Voici la liste des droits possibles et la valeur correspondante :

Droits	Chiffre	Calcul
---	0	0+0+0
r--	4	4+0+0
-w-	2	0+2+0
--x	1	0+0+1
rw-	6	4+2+0
-wx	3	0+2+1
r-x	5	4+0+1
rwx	7	4+2+1

Comme exemple :

```
chmod 600 rapport.txt
```

Un petit ls -l pour voir le résultat :

```
mateo21@mateo21-desktop:~$ ls -l rapport.txt
-rw------- 1 mateo21 mateo21 0 2007-11-15 23:14 rapport.txt
```

- Attribuer des droits avec des lettres (chmod relatif)

Il existe un autre moyen de modifier les droits d'un fichier. Il revient un peu au même mais permet parfois de paramétrer plus finement, droit par droit. Dans ce mode, il faut savoir que :

  - **u** = user (propriétaire)
  - **g** = group (groupe)
  - **o** = other (autres)

Et que :

- + signifie "ajouter le droit".
- - signifie "supprimer le droit".
- = signifie "affecter le droit".

Exemples :

"Ajouter le droit d'écriture au groupe"

```
chmod g+w rapport.txt
```

"Enlever le droit de lecture aux autres"

```
chmod o-r rapport.txt
```

"Ajouter les droits de lecture et d'exécution au propriétaire"

```
chmod u+rx rapport.txt
```

"Ajouter le droit d'écriture au groupe et l'enlever aux autres"

```
chmod g+w,o-w rapport.txt
```

"Enlever le droit de lecture au groupe et aux autres"

```
chmod go-r rapport.txt
```

"Ajouter le droit d'exécution à tout le monde"

```
chmod +x rapport.txt
```

"Affecter tous les droits au propriétaire, juste la lecture au groupe, rien aux autres"

```
chmod -R 700 /home/mateo21
```

Et toujours... -R pour affecter récursivement

Le paramètre -R existe aussi dans chmod. Si vous affectez des droits sur un dossier avec -R, tous ses fichiers et sous-dossiers récupèreront le même droit.

2.2 Opérations sur les disques et les fichiers

Le système qui gère les fichiers sous Linux est un peu déroutant au début, surtout quand on est habitué à celui de Windows. En effet, ici vous ne trouverez pas de "C:\", "D:\" ou que sais-je encore. Les fichiers sont organisés d'une manière complètement différente. Au lieu de séparer chaque disque dur, lecteur cd, lecteur de disquettes, lecteur de carte mémoire... Linux place en gros tout au même endroit.

2.2.1 Types des fichiers

Pour faire simple, il existe 2 grands types de fichiers sous Linux :

- **Les fichiers classiques :** ce sont les fichiers que vous connaissez, ça comprend les fichiers texte (.txt, .doc, .odt...), les sons (.wav, .mp3, .ogg), mais aussi les programmes. Bref, tout ça ce sont des fichiers que vous connaissez et que vous retrouvez dans Windows.
- **Les fichiers spéciaux :** certains autres fichiers sont spéciaux car ils représentent quelque chose. Par exemple, votre lecteur CD est un fichier pour Linux. Là où Windows fait la distinction entre ce qui est un fichier et ce qui ne l'est pas, Linux lui dit que tout est un fichier. C'est une conception très différente, un peu déroutante comme je vous l'ai dit, mais pas de panique vous allez vous y faire.

2.2.2 Racine

Dans un système de fichiers, il y a toujours ce qu'on appelle une racine, c'est-à-dire un "gros dossier de base qui contient tous les autres dossiers et fichiers". Sous Windows, il y a en fait plusieurs racines. "C:\" est la racine de votre disque dur, "D:\" est la racine de votre lecteur CD (par exemple). Sous Linux, il n'y a qu'**une et une seule racine : "/".** Comme vous le voyez, il n'y a pas de lettre de lecteur car justement Linux ne donne pas des noms aux lecteurs comme le fait Windows. Il dit juste "La base, c'est / ".

2.2.3 Architecture des dossiers

Sous Windows, on a l'habitude de trouver souvent les mêmes dossiers à la racine : "Documents and Settings", "Program Files", "Windows"... Sous Linux, vous vous en doutez, les dossiers sont complètement différents (et on ne risque pas de trouver de dossier qui s'appelle Windows)

Il y a une liste des dossiers les plus courants que l'on retrouve à chaque fois à la racine de Linux. La description de chaque dossier sera rapide, mais c'est juste pour que vous puissiez vous repérer au début.

- **bin :** contient des programmes (exécutables) qui sont susceptibles d'être utilisés par tous les utilisateurs de la machine.
- **boot :** fichiers permettant le démarrage de Linux.
- **dev :** fichiers contenant les périphériques. En fait, on en reparlera plus tard, mais ce dossier contient des sous-dossiers qui "représentent" chacun un périphérique. On y retrouve ainsi par exemple le fichier qui représente le lecteur CD.
- **etc :** fichiers de configuration.
- **home :** répertoires personnels des utilisateurs. On en a déjà parlé un peu avant : c'est dans ce dossier que vous placerez vos fichiers personnels, à la manière du dossier "Mes documents" de Windows. Chaque utilisateur de l'ordinateur possède son dossier personnel. Par exemple, dans mon cas mon dossier personnel se trouve dans "/home/mateo21/". S'il y avait un autre utilisateur (appelons-le Patrick) sur mon ordinateur, il aurait eu droit lui aussi à son propre dossier : "/home/patrick/".
- **lib :** dossier contenant les bibliothèques partagées (généralement des fichiers .so) utilisées par les programmes. C'est en fait là qu'on trouve l'équivalent des .dll de Windows.
- **media :** lorsqu'un périphérique amovible (comme une carte mémoire SD ou une clé USB) est inséré dans votre ordinateur, Linux vous permet d'y accéder à partir d'un sous-dossier de "media". On parle de montage. C'est un peu compliqué, on en reparlera dans un chapitre plus tard.
- **mnt :** c'est un peu pareil que media, mais pour un usage plus temporaire.
- **opt :** répertoire utilisé pour les add-ons de programmes.
- **proc :** contient des informations système.
- **root :** c'est le dossier personnel de l'utilisateur "root". Normalement, les dossiers personnels sont placés dans "home". Mais celui de root fait exception. En effet, comme je vous l'ai dit dans le chapitre précédent root est le super-utilisateur, le "chef" de la machine en quelque sorte. Il a droit à un espace spécial sbin : contient des programmes système importants.
- **tmp :** dossier temporaire utilisé par les programmes pour stocker des fichiers.
- **usr :** c'est un des plus gros dossiers, dans lequel vont s'installer la plupart des programmes demandés par l'utilisateur.

- **var :** ce dossier contient des données "variables", souvent des fichiers, des logs (traces écrites de ce qui s'est passé récemment sur l'ordinateur), etc.

Cette liste de dossiers est en fait présente sur tous les OS de type Unix, et pas seulement sous Linux.

2.2.4 Schéma architecture d'un fichier

Pour bien que vous vous repériez, sachez qu'on peut présenter l'organisation des dossiers de Linux de cette manière :

Figure 2.03 : *Architecture des fichiers en linux*

La racine tout en haut est /, elle contient plusieurs dossiers, qui contiennent chacun eux-mêmes plusieurs dossiers, qui contiennent des dossiers et fichiers, etc etc

2.2.5 Commande pwd et which : où… où suis-je ?

Le nombre de dossiers et de fichiers présents après l'installation d'Ubuntu est tellement grand qu'il serait facile de s'y perdre. Un grand nombre de programmes sont en effet préinstallés pour que vous puissiez profiter rapidement des possibilités de Linux.

En revanche, repérer dans l'arborescence des dossiers est nécessaire. C'est un peu comme avoir une carte routière en quelque sorte. **Pwd** est l'abréviation de **"Print Working Directory"**, c'est-à-dire "Afficher le dossier actuel".

```
mateo21@mateo21-desktop:~$ pwd
/home/mateo21
```

Which permet de connaitre **l'emplacement d'une commande.** Une commande n'est rien d'autre qu'un programme qu'on peut appeler n'importe quand n'importe où dans la console. La commande which prend un paramètre : le nom de la commande dont vous voulez connaître l'emplacement.

```
mateo21@mateo21-desktop:~$ which pwd
/bin/pwd
```

pwd se trouve donc dans le dossier /bin/ ! Le "pwd" à la fin n'est pas un dossier mais le nom du programme lui-même.

Rémarque : Vous noterez que les programmes sous Linux ne possèdent en général pas d'extension (contrairement à Windows où l'extension utilisée est en général .exe).

2.2.6 Commande ls (lister les fichiers et les dossiers)

La commande ls permet de lister les fichiers et les dossiers.
Si la couleur ne s'affiche pas, vous pouvez rajouter le paramètre --color=auto, comme ceci : ls --color=auto
Si vous ne voulez pas de la couleur au contraire, essayez le paramètre --color=none.

Commande ls	
Options	Déscriptions
-a	Afficher les fichiers et dossiers cachés
-F	Indique le type d'élément
-l	Liste détaillée
-h	Afficher la taille en Ko, Mo, Go
-t	Trier par date selon la dernière modification

```
mateo21@mateo21-desktop:~$ ls -larth
total 380K
-rw-------  1 mateo21 mateo21   26 2007-09-19 16:40 .dmrc
-rw-r--r--  1 mateo21 mateo21   89 2007-09-19 16:40 .gtkrc-1.2-gnome2
-rw-------  1 mateo21 mateo21   16 2007-09-19 16:40 .esd_auth
drwx------  2 mateo21 mateo21 4,0K 2007-09-19 16:40 .update-notifier
lrwxrwxrwx  1 mateo21 mateo21   26 2007-09-19 18:31 Examples -
> /usr/share/example-content
-rw-r--r--  1 mateo21 mateo21  220 2007-09-19 18:31 .bash_logout
drwxr-xr-x  4 root    root    4,0K 2007-09-19 18:31 ..
drwxr-xr-x 10 mateo21 mateo21 4,0K 2007-09-25 16:03 .jedit
-rw-r--r--  1 mateo21 mateo21 1,1K 2007-09-25 16:03 .pgadmin3
drwxr-xr-x 47 mateo21 mateo21 4,0K 2007-09-25 16:03 .
-rw-------  1 mateo21 mateo21 1,8K 2007-09-25 16:38 .bash_history
-rw-------  1 mateo21 mateo21  17K 2007-09-25 16:52 .recently-used
drwx------  2 mateo21 mateo21 4,0K 2007-09-25 16:54 .gconfd
-rw-------  1 mateo21 mateo21   39 2007-09-25 17:18 .lesshst
-rw-r--r--  1 mateo21 mateo21  53K 2007-09-25 17:21 .xsession-errors
```

2.2.7 Commande cd : changer de dossier

Si on veut aller à la racine, il suffit de taper cd /

Si on veut aller dans le dossier usr dans le dossier racine :

```
mateo21@mateo21-desktop:/$ cd usr
```

Si on veut aller dans le dossier games dans le dossier usr :

```
mateo21@mateo21-desktop:/usr$ cd games
mateo21@mateo21-desktop:/usr/games$
```

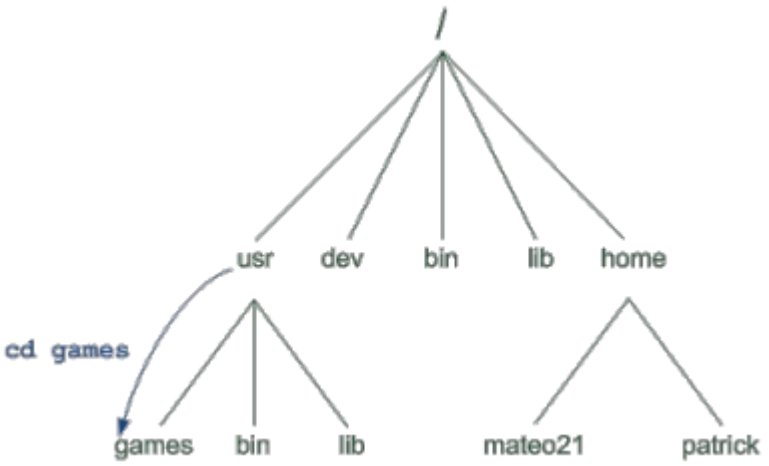
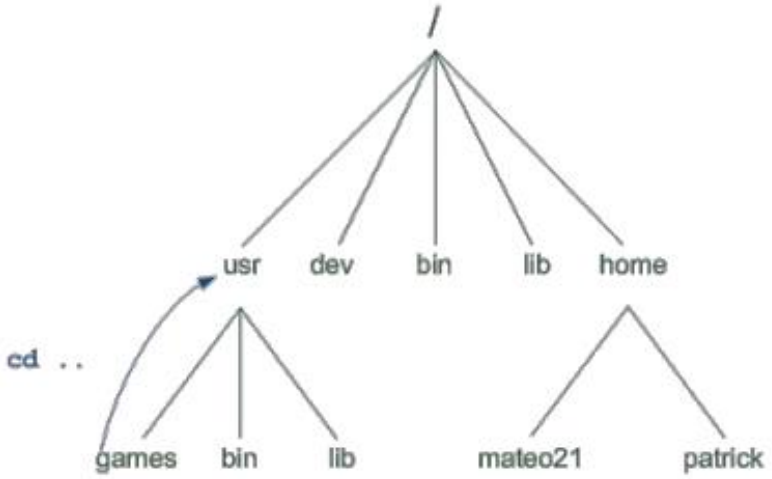


Figure 2.04 : *Répresentation de dossier pour tester la commande cd*

Pour révernir dans le parcours de dossier, il suffit d'utiliser .. :



L'utilisation d'une telle technique est aussi possible :

```
mateo21@mateo21-desktop:/usr/games$ cd ../..
mateo21@mateo21-desktop:/$
```

2.2.8 Chemins relatifs

Un chemin relatif est un chemin qui dépend du dossier dans lequel vous vous trouvez. Tout à l'heure, on est allé dans le sousdossier games de /usr en tapant juste son nom.

```
mateo21@mateo21-desktop:/usr$ cd games
```

En faisant cela, on utilise un chemin relatif, c'est-à-dire relatif au dossier actuel. Quand on met juste le nom d'un dossier comme ici, cela indique que l'on veut aller dans un sous-dossier. Si on fait cd games depuis la racine, ça va planter :

```
mateo21@mateo21-desktop:/$ cd games
bash: cd: games: Aucun fichier ou répertoire de ce type
```

L'erreur est dû sur le fait que le dossier games n'existe pas dans le dossier racine.
Pour se rendre dans games, il faut d'abord indiquer le dossier qui le contient (usr) :

```
mateo21@mateo21-desktop:/$ cd usr/games
mateo21@mateo21-desktop:/usr/games$
```

2.2.9 Chemins absolus

Contrairement aux chemins relatifs, les chemins absolus fonctionnent quel que soit le dossier dans lequel on se trouve. Un chemin absolu est facile à reconnaître : il commence toujours par la racine (/). Vous devez faire ensuite la liste des dossiers dans lesquels vous voulez entrer. Par exemple, supposons que je sois dans /home/mateo21 et que je souhaite aller dans /usr/games. Avec un chemin absolu :

```
mateo21@mateo21-desktop:~$ cd /usr/games
mateo21@mateo21-desktop:/usr/games$
```

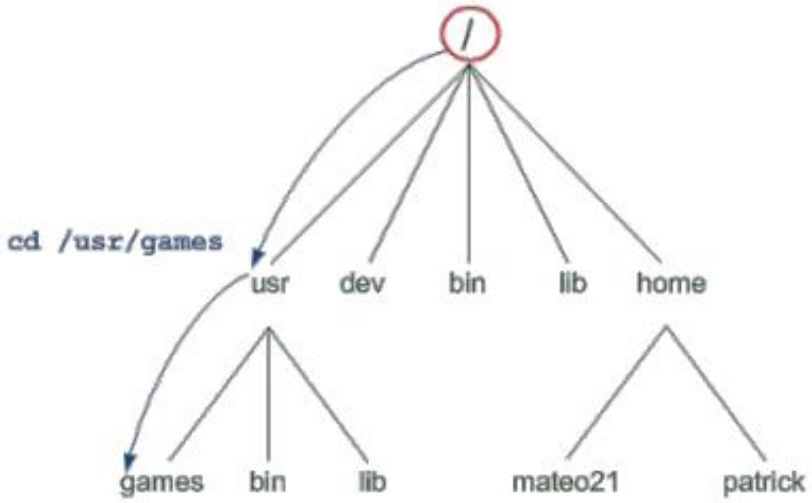


Figure 2.05 : *Exemple d'arborescente pour le chemin absolu*

Le schéma montre bien qu'on part de la racine / pour indiquer où on veut aller. Si on avait voulu faire la même chose à coup de chemin relatif, il aurait fallu écrire :

```
mateo21@mateo21-desktop:~$ cd ../../usr/games/
mateo21@mateo21-desktop:/usr/games$
```

Ce qui signifie "reviens en arrière (donc dans /home) puis reviens en arrière (donc dans /), puis va en avant dans usr, puis va en avant dans games".

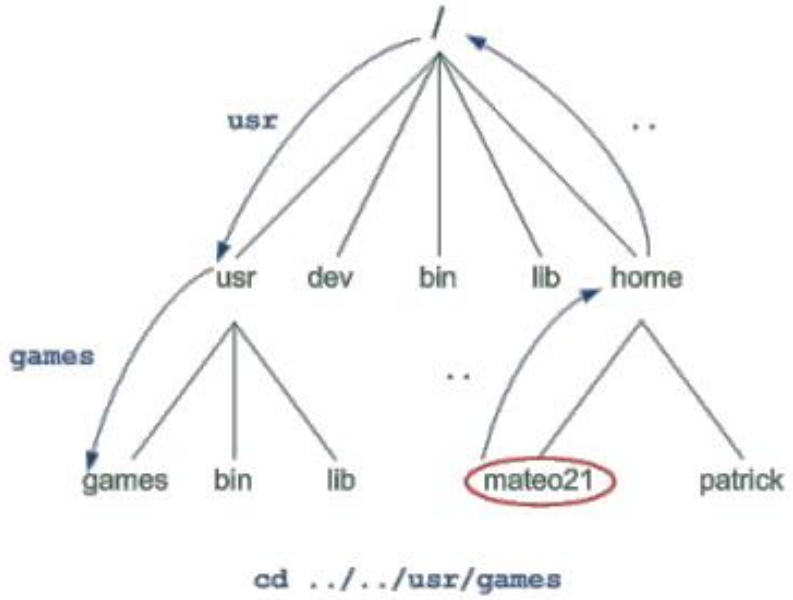


Figure 2.06 : *Exemple d'arborescente pour la commande cd*

Pour retourner au repertoire home, il existe trois possibilités : cd /home/mateo21 ou cd ~ ou cd L'autocompletion en utilisant tab une fois aussi marche avec la spécification du chemin

2.2.10 Commande du (taille ocupée par un dossier)

La commande "du", pour Disk Usage (utilisation du disque) vous donne des informations sur la taille qu'occupe les dossiers sur votre disque. Placez-vous pour commencer dans /usr/games, et tapez "du" :

```
mateo21@mateo21-desktop:~$ cd /usr/games
mateo21@mateo21-desktop:/usr/games$ du
5732    .
```

Commande du	
options	Déscriptions
-a	Afficher la taille des fichiers et dossiers caches
-h	Afficher pour les humains
-s	Afficher la taille totale

2.2.11 Commande mount

La connexion des systèmes de fichiers s'effectue avec la commande mount. Cette commande permet d'associer un fichier spécial à un répertoire dans le système de fichiers en spécifiant le type de système sur la partition en cours de connexion.

La syntaxe est :

```
mount [-t type] fichier_spécial répertoire
```

La commande umount permet de terminer la connexion.

```
umount [répertoire | fichier_special]
```

Le fichier /etc/fstab contient une description des connexions devant être effectuées au démarrage du système.

2.2.12 Commande fsck

La commande fsck donne l'ordre d'examen des systèmes de fichiers pour fsck

2.2.13 Commande cat et less (Afficher un fichier)

Nous allons d'abord voir comment afficher le contenu d'un fichier. Il y a en gros 2 commandes basiques sous Linux qui permettent de faire cela : cat et less. Aucune de ces commandes ne permet d'éditer un fichier, elles permettent juste de le voir.

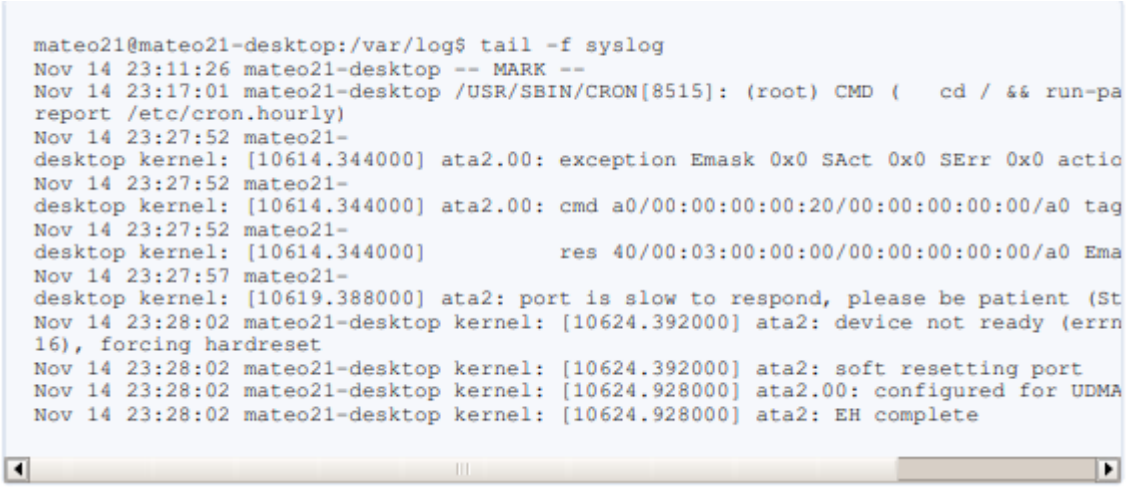
- La commande **cat** permet d'afficher tout le contenu d'un fichier dans la console **d'un coup**. La commande cat avec -n est un affichage avec le numéro de ligne.
- La commande **less** affiche **un fichier page par page**. La commande cat est rapide. Trop rapide. Tout le fichier est lu et affiché d'un coup dans la console, ce qui fait qu'on n'a pas le temps de le lire s'il est très gros. C'est là qu'une autre commande comme less devient vraiment indispensable. La grosse différence entre less et cat, c'est que less affiche progressivement le contenu du fichier, page par page. Ca vous laisse le temps de le lire dans la console.

2.2.14 Commande head et tail

Ces deux commandes sont un peu le contraire l'une de l'autre : la première permet d'afficher le début du fichier, la seconde permet d'afficher la fin.

- La commande head ("tête" en anglais) affiche seulement les premières lignes du fichier. Elle ne permet pas de se déplacer dans le fichier comme less, elle permet juste de récupérer les premières lignes. La commande peut être associée avec l'option numéro de ligne -n.

- Très intéressante aussi (voire même plus), la commande tail vous renvoie la fin du fichier, donc les dernières lignes. Il y a un autre paramètre à côté duquel vous ne pouvez pas passer : -f (f pour "follow", "suivre" en anglais). Ce paramètre magique ordonne à tail de "suivre" la fin du fichier au fur et à mesure de son évolution. C'est extrêmement utile pour suivre un fichier de log qui évolue souvent. Vous pouvez tester sur syslog par exemple :



2.2.15 Commande touch et mkdir (Créer de fichiers et des dossiers)

La commande *touch* permet de créer un nouveau fichier tandisque mkdir permet de créer un nouveau dossier. Vous pouvez ajouter l'extension ou le type de fichier et créer plusieurs fichiers en une seule commandeavec la commande *touch*.

```
touch fichierbidon autrefichierbidon.txt
```

```
mkdir mondossier
```

Pour que le nom de fichier contient d'espace, il faut utiliser une double entre quote.

```
touch "Fichier bidon"
```

La commande *mkdir* prend comme paramètre -p pour les dossiers intermediaires tout en créant les dossiers et les sous-dossiers specifiés.

```
mkdir -p animaux/vertebres/chat
```

2.2.16 Commande cp et mv

La commande cp (abréviation de "CoPy", "copier" en anglais) vous permet comme son nom l'indique de copier un fichier... mais aussi de copier plusieurs fichiers à la fois, et même de copier des dossiers.

```
cp fichierbidon fichiercopie
```

Pour copier dans un dossier rélatif ou absolu, il suffit de specifier en second paramètre le chemin du dossier

```
cp fichierbidon mondossier/
```

```
cp fichierbidon /var/log/
```

Avec l'option -R (un R majuscule !) ; Linux peut copier des dossiers.

```
cp -R animaux autresanimaux
```

L'utilisation de joker representée par le symbole * facilite la manipulation de la commande cp. Le symbole * est appelé joker, ou encore wildcard en anglais sous linux. Le Joker represente tous les elements.

```
cp *.jpg mondossier/
```

```
cp so* mondossier/
```

La commande mv permet de faire quelque chose d'assez étonnant : deplacer un fichier, renommer un fichier, deplacer et renommer un fichier en même temps. En effet, il n'existe pas de commande spéciale pour renommer un fichier en console sous linux, c'est la commande mv qui est utilisée pour ça.

```
mv fichierbidon superfichier
```

```
mv fichierbidon mondossier/superfichier
```

2.2.17 Commande rm pour supprimer les fichiers et dossier

La commande rm (pour "ReMove", "supprimer" en anglais) peut supprimer un fichier, 2 fichiers, plusieurs fichiers, des dossiers, voire même votre ordinateur entier si vous le voulez. Il faut l'utiliser avec précaution donc. Commençons par des choses simples, supprimons ce fichierbidon :

```
rm fichierbidon
```

Commande rm	
Options	Déscriptions
-i	Demander confirmation
-f	Forcer la suppression
-v	En mode verbeux. (Dis moi-ce que tu fais)
-r	Supprimer les dossiers et ses contenus

La commande *rm* peut être combiné avec le joker.

NE JAMAIS FAIT CETTE COMMANDE SURTOUT EN MODE SUPER-UTILISATEUR.

```
NON NON NON NE FAITES JAMAIS CA !!! => rm -rf /*
```

- rm : commande la suppression
- -r : suppression récursive de tous les fichiers et dossiers
- -f : force la suppression sans demander la moindre confirmation
- /* : supprime tous les fichiers et dossiers qui se trouvent à la racine (/) quel que soit leur nom (joker *)

2.2.18 Commande ln pour créer des liens entre les fichiers

Bien qu'un peu moins courante, la commande ln vous sera certainement utile un jour ou l'autre. Elle permet de créer des liens entre des fichiers, c'est-à-dire (pour employer des mots que vous connaissez) qu'elle permet de créer des raccourcis. Ces "raccourcis", qu'on appelle des liens sous

Linux, sont un peu plus complexes que ceux que vous avez l'habitude de voir sous Windows. En effet, on peut créer 2 types de liens :

- Des liens physiques
- Des liens symboliques

Ces 2 types ne fonctionnent pas de la même manière. Pour comprendre ce qui les différencie, il faut savoir comment un OS tel que Linux gère les fichiers sur le disque dur.

Sur le disque dur, chaque fichier est grosso-modo séparé en 2 parties :

- Son nom
- Son contenu

Chaque contenu de fichier se voit attribuer un numéro d'identification appelé inode. Chaque nom de fichier est donc associé à un inode (son contenu).

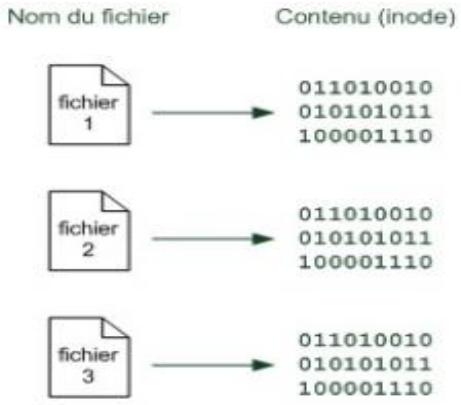


Figure 2.07 : *Répresentation des inodes*

- Les liens physiques

Ce type de lien est plus rarement utilisé que le lien symbolique, mais il faut tout de même le connaître car il peut se révéler pratique. Un lien physique permet d'avoir 2 noms de fichiers qui partagent exactement le même contenu, c'est-à-dire le même inode.

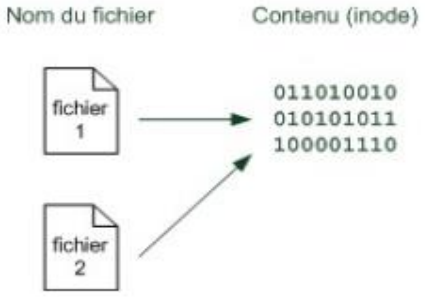


Figure 2.08 : *Répresentation des liens physiques*

Ainsi, que vous passiez par fichier1 ou par fichier2, vous modifiez exactement le même contenu. En quelque sorte, le fichier est le même. On peut juste y accéder via 2 noms de fichiers différents.

```
ln fichier1 fichier2
```

- Les liens symboliques

Les liens symboliques ressemblent plus aux "raccourcis" dont vous avez peut-être l'habitude sous Windows. La plupart du temps, on crée des liens symboliques sous Linux pour faire un raccourci, et non des liens physiques qui sont un peu particuliers. Le principe du lien symbolique est que l'on crée un lien vers un autre nom de fichier. Cette fois, on pointe vers le nom de fichier et non vers l'inode directement.

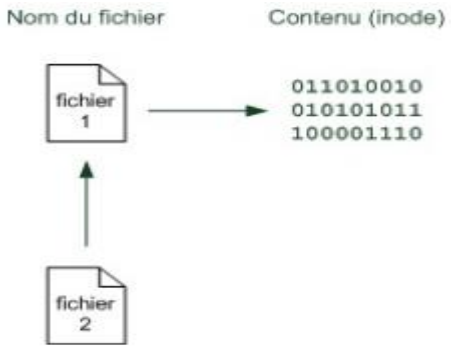


Figure 2.09 : *Répresentation d'un lien symbolique*

```
ln -s fichier1 fichier2
```

2.2.19 Recherche des fichiers

2.2.19.1 Utilisation de la commande locate

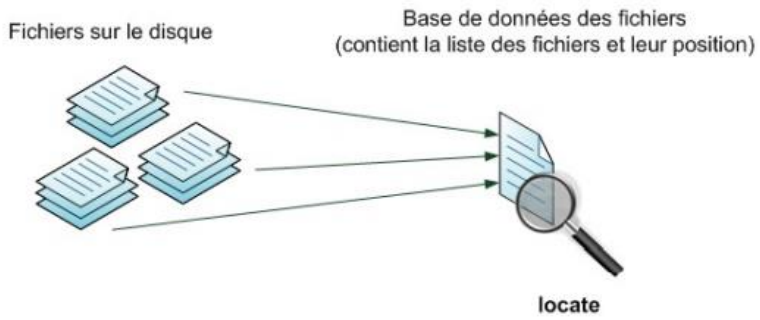


Figure 2.10 : *Explcation detaillée de la commande locate*

Votre problème, c'est que les fichiers viennent tout juste d'être créés et n'ont pas encore été répertoriés dans la base de données. Ils ne seront donc pas découverts par locate.
Vous pouvez forcer la commande locate à reconstruire la base de données des fichiers du disque dur. Cela se fait avec la commande updatedb, à exécuter en root (avec sudo) :

```
sudo updatedb
```

```
mateo21@mateo21-desktop:~$ locate notes.txt
/home/mateo21/notes.txt
```

2.2.19.2 Utilisation de la commande find pour la recherche approfondie

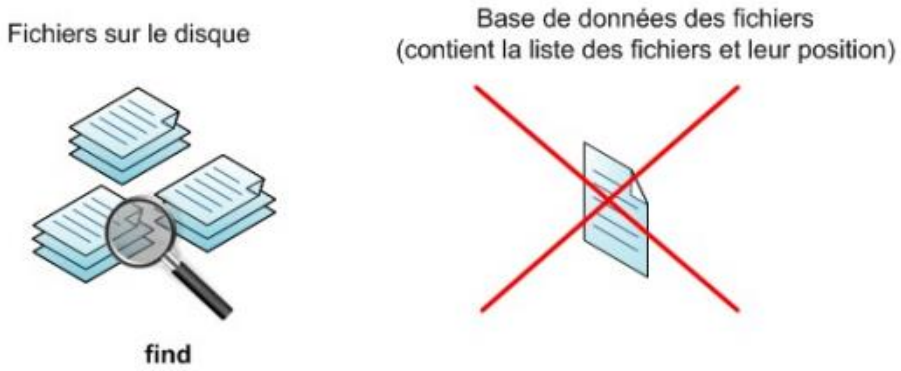


Figure 2.11 : *Recherche avec find*

Contrairement à locate, **find ne va pas lire dans une base de données** mais va au **contraire parcourir tout votre disque dur**. Cela peut être très long si vous avez plusieurs Go de données.
La commande find s'utilise de la façon suivante :

find "où" "quoi" "que faire avec"
(seul le paramètre "quoi" est obligatoire)

- Où : c'est le nom du dossier dans lequel la commande va faire la recherche. Tous les sous-dossiers seront analysés. Contrairement à locate, il est donc possible de limiter la recherche à /home par exemple. Par défaut, si ce paramètre n'est pas précisé, la recherche s'effectuera dans le dossier courant et ses sous-dossiers.
- Quoi : c'est le fichier à rechercher. On peut rechercher un fichier par son nom, mais aussi en fonction de sa date de dernière création, de sa taille, etc. Ce paramètre est obligatoire.

- Que faire avec : il est possible d'effectuer des actions automatiquement sur chacun des fichiers trouvés (on parle de "post-traitement"). L'action la plus courante consiste à afficher simplement la liste des fichiers trouvés, mais nous verrons que nous pouvons faire bien d'autres choses

Exemple 1 : Recherche tous les fichiers qui commençaient par syslog dans le dossier /var/log

```
mateo21@mateo21-desktop:~$ find /var/log/ -name "syslog*"
/var/log/syslog.3.gz
/var/log/syslog.5.gz
/var/log/syslog.4.gz
/var/log/syslog
/var/log/syslog.6.gz
/var/log/syslog.2.gz
/var/log/syslog.1.gz
/var/log/installer/syslog
/var/log/syslog.0
```

Exemple 2 : rechercher tous les fichiers qui font plus de 10 Mo

```
mateo21@mateo21-desktop:/var$ find ~ -size +10M
/home/mateo21/souvenirs.avi
/home/mateo21/backups/backup_mai.gz
/home/mateo21/backups/backup_juin.gz
```

Exemple 3 : Rechercher tous les fichiers odt dont le nombre de jours qui vous séparent du dernier accès à un fichier est de 7jours.

```
mateo21@mateo21-desktop:~$ find -name "*.odt" -atime -7
/home/mateo21/ecriture/resume_infos_juin.odt
```

Exemple 4 : Rechercher uniquement des répertoires ou des fichiers

On peut aussi rechercher uniquement des répertoires ou des fichiers. Utilisez :

- **-type d** : pour rechercher uniquement **des répertoires** (directories).
- **-type f** : pour rechercher uniquement **des fichiers** (files).

```
find /var/log -name "syslog" -type d
```

Exemple 5 : Rechercher et afficher les fichiers (ou afficher de façon formatée)

```
find ~ -name "*.jpg" -print
```

```
mateo21@mateo21-desktop:~$ find . -name "*.jpg" -printf "%p - %u\n"
./photos/australie1.jpg - mateo21
./photos/australie2.jpg - mateo21
./photos/australie3.jpg - mateo21
```

Exemple 6 : Rechercher et supprimer les fichiers trouvés

```
find ~ -name "*.jpg" -delete
```

Exemple 7 : Rechercher et appeler une commande

```
find ~ -name "*.jpg" -exec chmod 600 {} \;
```

- Pour chaque fichier .jpg trouvé, on exécute la commande qui suit -exec : Cette commande ne doit PAS être entre guillemets.
- Les accolades {} seront remplacées par le nom du fichier.
- La commande doit finir par un \; obligatoirement.

2.2.20 Filter les données

La commande *grep* peut s'utiliser de nombreuses façons différentes. Pour le moment, nous allons suivre le schéma ci-dessous :

```
grep texte nomfichier
```

Les paramètres courants de la commande *grep* sont :

- **-i :** signifie ne pas tenir compte de la casse (majuscules / minuscules)
- **-v :** inverser la recherche : ignorer un mot
- **-r** : rechercher dans tous les fichiers et sous-dossiers

Pour faire des recherches plus poussées, pour ne pas dire des recherches très poussées, vous devez faire appel aux expressions régulières. C'est un ensemble de symboles qui va vous permettre de dire à l'ordinateur très précisément ce que vous recherchez. Le tableau suivant represente les principaux caractères spéciaux qu'on utilise dans les expressions régulières :

Caractère spécial	Signification
.	Caractère quelconque
^	Début de ligne
$	Fin de ligne
[]	Un des caractères entre les crochets
?	L'élément précédent est optionnel (peut être présent 0 ou 1 fois)
*	L'élément précédent peut être présent 0, 1 ou plusieurs fois
+	L'élément précédent doit être présent 1 ou plusieurs fois
|	Ou
()	Groupement d'expressions

Figure 2.12 : *Illustration des symboles en expressions regulières*

2.2.21 Trier des données

La commande *sort* se révèle bien utile lorsqu'on a besoin de trier le contenu d'un fichier.

Exemple 1 : Le sujet consiste à créer un nouveau fichier (avec nano par exemple) appelé "noms.txt" et d'y placer le texte suivant :

```
François
Marcel
Albert
Jean
Stéphane
patrice
Vincent
jonathan
```

En executant la commande suivante :

```
$ sort noms.txt

Albert
François
Jean
jonathan
Marcel
patrice
Stéphane
Vincent
```

- **-o** : écrire le resultat dans un autre fichier

```
sort -o noms_tries.txt noms.txt
```

- -r : trier en ordre inverse

```
$ sort -r  noms.txt

Vincent
Stéphane
patrice
Marcel
jonathan
Jean
François
Albert
```

- -R : trier de manière aléatoire

```
$ sort -R noms.txt

patrice
François
Marcel
jonathan
Jean
Albert
Vincent
Stéphane
```

- **-n :** trier des nombres

Le tri **avec l'option -n** sera donc :

```
36
16
42
129
27
364
```

Sans option -n, le resultat sera donc :

```
129
16
27
36
364
42
```

2.2.22 Compter le nombre des lignes

La commande **wc** signifie **"Word Count".** C'est donc a priori un compteur de mots, mais en fait on lui trouve plusieurs autres utilités : compter le nombre de lignes (très fréquent) et compter le nombre de caractères.

```
$ wc noms.txt
 8  8 64 noms.txt
```

Ces 3 nombres signifient, dans l'ordre :

1. Le nombre de lignes
2. Le nombre de mots
3. Le nombre d'octets

- **-l :** compter le nombre de lignes
- **-w :** compter le nombre de mots
- **-c :** compter le nombre d'octets
- **-m :** compter le nombre de caractères

```
$ wc -l noms.txt
8 noms.txt
```

2.2.23 Supprimer les doublons

La commande *uniq* permet de supprimer et d'indiquer les doublons.

- **-c :** compter le nombre d'occurrences
- **-d :** afficher uniquement les lignes présentes en double

Soit un fichier trié :

```
Albert
François
François
François
Jean
jonathan
Marcel
Marcel
patrice
Stéphane
Vincent
```

Il y a des noms en double (et même en triple) dans ce fichier. Appliquons un petit coup de uniq là-dessus pour voir ce qu'il en reste :

```
$ uniq doublons.txt

Albert
François
Jean
jonathan
Marcel
patrice
Stéphane
Vincent
```

La liste de noms sans les doublons s'affiche alors dans la console. Vous pouvez demander à ce que le résultat sans doublons soit écrit dans un autre fichier plutôt qu'affiché dans la console :

```
uniq doublons.txt sans_doublons.txt
```

Les autres resultats des exemples sont :

```
$ uniq -c doublons.txt
      1 Albert
      3 François
      1 Jean
      1 jonathan
      2 Marcel
      1 patrice
      1 Stéphane
      1 Vincent
```

```
$ uniq -d doublons.txt

François
Marcel
```

2.2.24 Commande cut pour couper une partie de fichier

Comme, vous avez déjà coupé du texte dans un éditeur de texte, la commande *cut* vous propose de faire cela au sein d'un fichier, afin de conserver uniquement une partie de chaque ligne.
Si vous souhaitez conserver uniquement les caractères 2 à 5 de chaque ligne du fichier :

```
$ cut -c 2-5 noms.txt

ran
arce
lber
ean
tép
atri
ince
onat
```

Pour conserver du 1er au 3ème caractère :

```
$ cut -c -3 noms.txt

Fra
Mar
Alb
Jea
St
pat
Vin
jon
```

De même, pour conserver du 3ème au dernier caractère :

```
$ cut -c 3- noms.txt

ançois
rcel
bert
```

Il est possible également **de couper selon un delimiteur.** Prenons un cas pratique : les fichiers CSV. Ils sont générés par des tableurs tels que Excel et Openoffice.org pour faciliter l'échange et le traitement de données. Imaginons que vous ayez une (petite) classe et que vous rendiez les notes du dernier contrôle. Vous avez fait un joli tableur et vous avez enregistré le document au format CSV. Le fichier sur lequel nous allons nous baser sera le suivant :

```
Fabrice,18 / 20,Excellent travail
Mathieu,3 / 20,Nul comme d'hab
Sophie,14 / 20,En nette progression
Mélanie,9 / 20,Allez presque la moyenne !
Corentin,11 / 20,Pas mal mais peut mieux faire
Albert,20 / 20,Toujours parfait
Benoît,5 / 20,En grave chute
```

Comme le nom CSV l'indique, les virgules servent à séparer les colonnes. Ces colonnes contiennent, dans l'ordre :

- Le prénom
- La note
- Un commentaire

C'est un exemple tout à fait fictif bien entendu. Créez un nouveau fichier avec le texte que je viens de vous donner, que vous appellerez par exemple "notes.csv". Le but est d'extraire de ce fichier la liste des prénoms. Vous allez avoir besoin d'utiliser 2 paramètres :

- **-d :** indique quel est le délimiteur dans le fichier
- **-f :** indique le numéro du ou des champs à couper

Dans notre cas, le délimiteur qui sépare les champs est la virgule. Le numéro du champ à couper est 1 (c'est le premier).

```
$ cut -d , -f 1 notes.csv

Fabrice
Vincent
Sophie
Mélanie
Corentin
Albert
Benoît
```

Après le -d, nous avons indiqué quel était le délimiteur (à savoir la virgule ","). Après le -f, nous avons indiqué le numéro du champ à conserver (le premier). Si nous voulons juste les commentaires :

```
$ cut -d , -f 3 notes.csv

Excellent travail
Nul comme d'hab
En nette progression
Allez presque la moyenne !
Pas mal mais peut mieux faire
Toujours parfait
En grave chute
```

Pour avoir les champs n°1 et n°3 (le prénom et le commentaire) :

```
$ cut -d , -f 1,3 notes.csv

Fabrice,Excellent travail
Vincent,Nul comme d'hab
Sophie,En nette progression
Mélanie,Allez presque la moyenne !
Corentin,Pas mal mais peut mieux faire
Albert,Toujours parfait
Benoît,En grave chute
```

De même, il est possible de conserver toute une série de champs avec le tiret comme tout à l'heure : `cut -d , - f 2-4 notes.csv` a pour effet de conserver les champs n°2, 3 et 4.
D'autre part, `cut -d , -f 3- notes.csv` conserve les champs du n°3 jusqu'à la fin.

2.2.25 Les flux de redirections

2.2.25.1 Description des flux de redirections

Vous devriez maintenant avoir l'habitude d'un certain nombre de commandes que propose la console de Linux. Le fonctionnement est toujours le même :

1. Vous tapez la commande (par exemple *ls*).
2. Le résultat s'affiche dans la console.

Il est pourtant possible de rediriger ce résultat. Au lieu que le résultat s'affiche dans la console, vous allez pouvoir l'envoyer ailleurs : dans un fichier, ou en entrée d'une autre commande pour "chaîner des commandes". Ainsi, le résultat d'une commande peut en déclencher une autre à l'aide de petits symboles spéciaux, appelés flux de redirection.

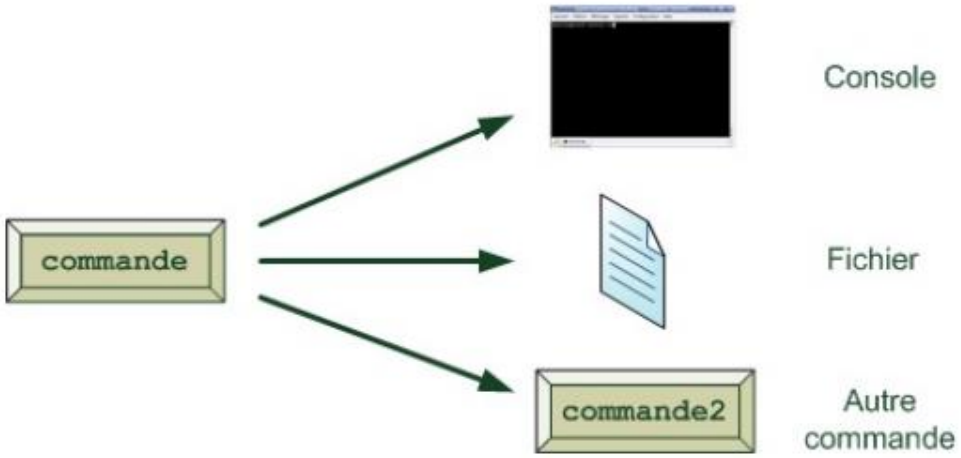


Figure 2.13 : *Principe de redirection des commandes*

Jusqu'ici, nous n'avons donc exploité que la première possibilité (celle par défaut) : afficher le résultat dans la console. Il nous reste donc bien d'autres techniques à découvrir !

Les flux de redirection sont une composante essentielle de la console sous Linux, et ce depuis l'époque d'Unix. Ils vont très certainement changer votre façon de "voir" comment la console fonctionne et démultiplier votre contrôle sur les commandes que vous lancez.

2.2.25.2 Rediriger le résultat dans un fichier

La manipulation la plus simple va nous permettre d'écrire le résultat d'une commande dans un fichier, au lieu de l'afficher bêtement dans la console. Nous avions travaillé sur un petit fichier de

type "CSV" que les tableurs peuvent générer. Ce sont les notes des élèves d'une classe à un contrôle :

```
Fabrice,18 / 20,Excellent travail
Mathieu,3 / 20,Nul comme d'hab
Sophie,14 / 20,En nette progression
Mélanie,9 / 20,Allez presque la moyenne !
Corentin,11 / 20,Pas mal mais peut mieux faire
Albert,20 / 20,Toujours parfait
Benoît,5 / 20,En grave chute
```

La commande cut nous avait permis de "couper" une partie du fichier et d'afficher le résultat dans la console. Par exemple, nous avions demandé à cut de prendre tout ce qui se trouvait avant la première virgule afin d'avoir la liste des noms de tous les élèves présents à ce contrôle :

```
$ cut -d , -f 1 notes.csv

Fabrice
Vincent
Sophie
Mélanie
Corentin
Albert
Benoît
```

Ce résultat s'est affiché dans la console. C'est ce que font toutes les commandes par défaut... à moins que l'on utilise un flux de redirection.

2.2.25.3 Redirection dans un nouveau fichier

Supposons que nous souhaitions écrire la liste des prénoms dans un fichier, afin de garder sous le coude la liste des élèves présents au contrôle. C'est là qu'intervient le petit symbole magique > (appelé chevron) que je vous laisse trouver sur votre clavier.

Ce symbole permet de rediriger le résultat de la commande dans le fichier de votre choix. Essayez par exemple de taper ceci :

```
cut -d , -f 1 notes.csv > eleves.txt
```

Comme vérficiation que le fichier est bien créé :

```
$ ls -l
total 20
-rw-r--r-- 1 mateo21 mateo21  91 2008-04-19 19:36 doublons.txt
-rw-r--r-- 1 mateo21 mateo21  56 2008-09-26 12:01 eleves.txt
-rw-r--r-- 1 mateo21 mateo21  35 2008-04-19 17:06 fichier_trie.txt
-rw-r--r-- 1 mateo21 mateo21  20 2008-04-19 19:03 nombres.txt
-rw-r--r-- 1 mateo21 mateo21 253 2008-09-26 12:01 notes.csv
```

Pour afficher le contenu, il est possible d'utiliser la commande *cat* ou *less*.

2.2.25.4 Redirection à la fin d'un fichier

Le double chevron >> sert lui aussi à rediriger le résultat dans un fichier, mais cette fois à la fin de ce fichier. Avantage : vous ne risquez pas d'écraser le fichier s'il existe déjà. Si le fichier n'existe pas, il sera créé automatiquement. Normalement, vous devriez avoir créé un fichier eleves.txt lors des manipulations précédentes. Si vous faites :

```
cut -d , -f 1 notes.csv >> eleves.txt
```

Les noms seront ajoutés à la fin du fichier, sans écraser le résultat précédent. Bon du coup, on a des noms en double maintenant.

```
$ cat eleves.txt
Fabrice
Mathieu
Sophie
Mélanie
Corentin
Albert
Benoît
Fabrice
Mathieu
Sophie
Mélanie
Corentin
Albert
Benoît
```

Vous pouvez utiliser la commande *sort* et *uniq* pour faire la distinction des doublons.
Nous venons de découvrir 2 flux de redirection dans des fichiers :

- > : redirige dans un fichier et l'écrase s'il existe déjà.
- >> : redirige à la fin d'un fichier et le crée s'il n'existe pas.

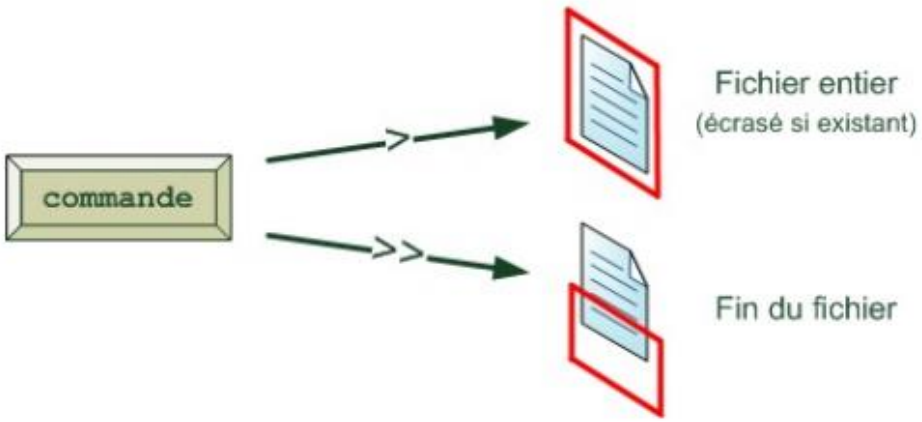


Figure 2.14 : *Les redirections de sorties en Linux*

2.2.25.5 Rediriger les erreurs 2>, 2>> et 2>&1

Il faut savoir que toutes les commandes produisent 2 flux de données différents :

- **La sortie standard :** pour tous les messages (sauf les erreurs).
- **La sortie d'erreurs :** pour toutes les erreurs.

Prenons un exemple concret pour voir comment ça se passe. Supposons que vous fassiez un cat du fichier notes.csv pour afficher son contenu. Il y a 2 possibilités :

- **Si tout va bien :** le résultat (le contenu du fichier) s'affiche sur la sortie standard.
- **S'il y a une erreur :** celle-ci s'affiche dans la sortie d'erreurs.

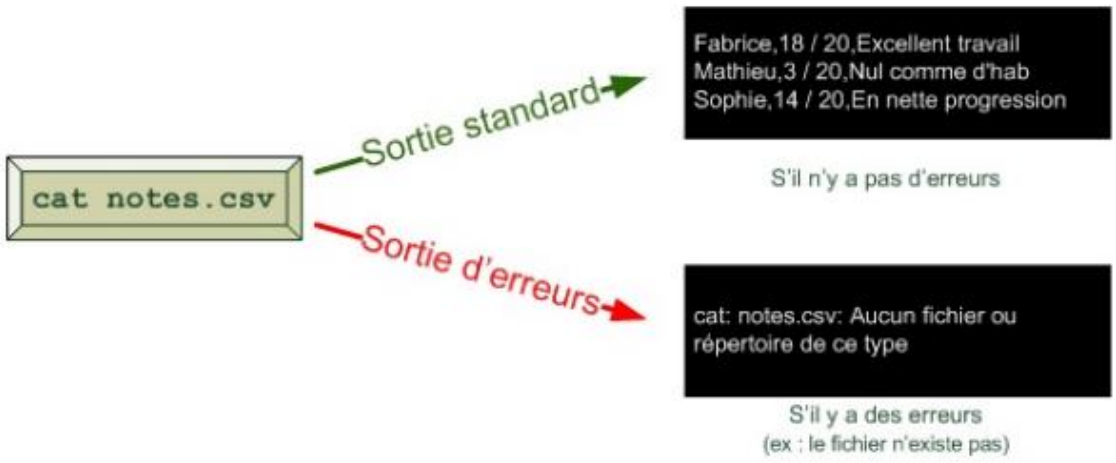


Figure 2.15 : *Flux de redirection avec consideration des erreurs*

Par défaut, tout s'affiche dans la console : la sortie standard comme la sortie d'erreurs.

Cela explique pourquoi vous ne faisiez pas la différence entre ces 2 sorties jusqu'ici : elles avaient l'air identiques. Tout à l'heure, nous avons vu comment rediriger la sortie standard dans un fichier. Toutefois, les erreurs continuent d'être affichées dans la console. Faites le test :

```
cut -d , -f 1 fichier_inexistant.csv > eleves.txt
cut: fichier_inexistant.csv: Aucun fichier ou répertoire de ce type
```

Le fichier "fichier_inexistant.csv" n'existe pas. L'erreur s'est affichée dans la console au lieu d'avoir été envoyée dans eleves.txt.

a) Rédiriger les erreurs dans un fichier à part

On pourrait souhaiter "logger" les erreurs dans un fichier d'erreurs à part pour ne pas les oublier et pour pouvoir les analyser ensuite. Pour cela, on utilise l'opérateur 2>. Vous avez bien lu : c'est le chiffre 2 collé au chevron que nous avons utilisé tout à l'heure. Faisons une seconde redirection à la fin de cette commande cut :

```
cut -d , -f 1 fichier_inexistant.csv > eleves.txt 2> erreurs.log
```

Il y a deux redirections ici :

- **> eleves.txt :** redirige le résultat de la commande (sauf les erreurs) dans le fichier eleves.txt. C'est la sortie standard.
- **2> erreurs.log :** redirige les erreurs éventuelles dans le fichier erreurs.log. C'est la sortie d'erreurs. Vous pouvez vérifier : si "fichier_inexistant.csv" n'a pas été trouvé, l'erreur aura été inscrite dans le fichier "erreurs.log" au lieu d'être affichée dans la console.

Remarque :

Notez qu'il est aussi possible d'utiliser 2>> pour ajouter les erreurs à la fin du fichier.

b) Fusionner les sorties :

Notez qu'il est aussi possible d'utiliser 2>> pour ajouter les erreurs à la fin du fichier. Il faut utiliser le code suivant : 2>&1 Cela a pour effet de rediriger toute la sortie d'erreurs dans la sortie standard. Traduction pour l'ordinateur : "envoie les erreurs au même endroit que le reste. Essayez donc ceci :

```
cut -d , -f 1 fichier_inexistant.csv > eleves.txt 2>&1
```

Tout ira désormais dans eleves.txt : le résultat (si ça a marché) de même que les erreurs (s'il y a eu un problème).

Nous avons découvert 3 symboles :

c) 2> : redirige les erreurs dans un fichier (s'il existe déjà il sera écrasé).
d) 2>> : redirige les erreurs à la fin d'un fichier (s'il n'existe pas, il sera créé).
e) 2>&1 : redirige les erreurs au même endroit et de la même façon que la sortie standard.

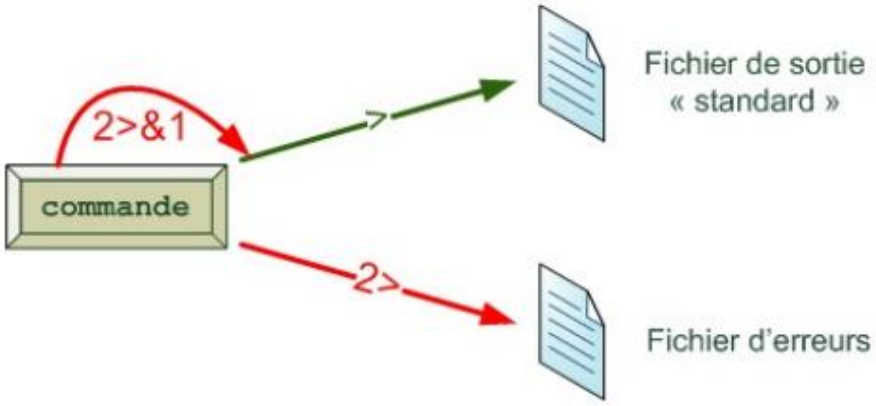


Figure 2.16 : *Illustration des trois flux de sorties par défaut*

2.2.25.6 Lire depuis un fichier ou un clavier

Pour le moment, nous avons redirigé uniquement la sortie des commandes. Nous avons décidé où envoyer les messages issus de ces commandes. Pour faire un peu l'inverse, c'est-à-dire de décider d'où vient l'entrée d'une commande. Jusqu'alors, l'entrée venait des paramètres de la commande... on peut faire en sorte qu'elle vienne d'un fichier ou d'une saisie au clavier !

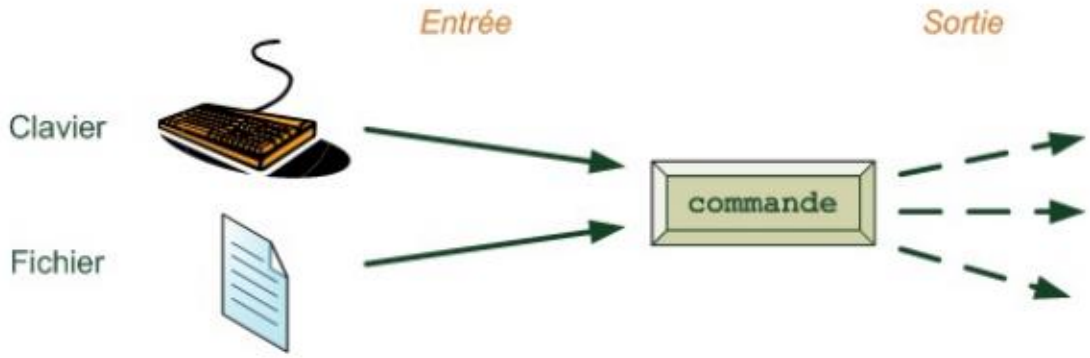


Figure 2.17 : *Entrée de commande*

a) Lire depuis un fichier

Le chevron ouvrant < (à ne pas confondre avec le chevron fermant que nous avons utilisé tout à l'heure) permet d'indiquer d'où vient l'entrée qu'on envoie à la commande. On va prendre un exemple tout bête : la commande *cat*.

```
$ cat < notes.csv
Fabrice,18 / 20,Excellent travail
Mathieu,3 / 20,Nul comme d'hab
Sophie,14 / 20,En nette progression
Mélanie,9 / 20,Allez presque la moyenne !
Corentin,11 / 20,Pas mal mais peut mieux faire
Albert,20 / 20,Toujours parfait
Benoît,5 / 20,En grave chute
```

Cela aura pour effet d'afficher le contenu du fichier envoyé en entrée :

```
$ cat < notes.csv
Fabrice,18 / 20,Excellent travail
Mathieu,3 / 20,Nul comme d'hab
Sophie,14 / 20,En nette progression
Mélanie,9 / 20,Allez presque la moyenne !
Corentin,11 / 20,Pas mal mais peut mieux faire
Albert,20 / 20,Toujours parfait
Benoît,5 / 20,En grave chute
```

b) Lire depuis un clavier de manière progressive

Le double chevron ouvrant << fait quelque chose d'assez différent : il vous permet d'envoyer un contenu à une commande avec votre clavier. Cela peut s'avérer très utile. Je vous propose un exemple concret pour bien voir ce que ça permet de faire en pratique. Essayez de taper ceci :

```
sort -n << FIN
```

La console vous propose alors de taper du texte

```
$ sort -n << FIN
>
```

Comme sort -n sert à trier des nombres, on va justement écrire des nombres, un par ligne (en appuyant sur la touche Entrée à chaque fois)

```
$ sort -n << FIN
> 13
> 132
> 10
> 131
```

Continuez ainsi jusqu'à ce que vous ayez terminé. Lorsque vous avez fini, tapez FIN pour arrêter la saisie. Tout le texte que vous avez écrit est alors envoyé à la commande (ici sort) qui traite cela en entrée. Et, comme vous pouvez vous en douter, la commande sort nous trie nos nombres !

```
$ sort -n << FIN
> 13
> 132
> 10
> 131
> 34
> 87
> 66
> 68
> 65
> FIN
10
13
34
65
66
68
87
131
132
```

Cela vous évite d'avoir à créer un fichier si vous n'en avez pas besoin. Vous pouvez faire la même chose avec une autre commande comme par exemple wc pour compter le nombre de mots ou de caractères.

```
$ wc -m << STOP
> Combien de caractères dans cette phrase ?
> STOP
42
```

Ce qui compte, c'est que vous définissiez un mot-clé qui servira à indiquer la fin de la saisie. Notez par ailleurs que rien ne vous oblige à écrire ce mot en majuscules.

Nous pouvons donc "alimenter" des commandes de 2 manières différentes : < : envoie le contenu d'un fichier à une commande. << : passe la console en mode saisie au clavier, ligne par ligne. Toutes ces lignes seront envoyées à la commande lorsque le mot-clé de fin aura été écrit.

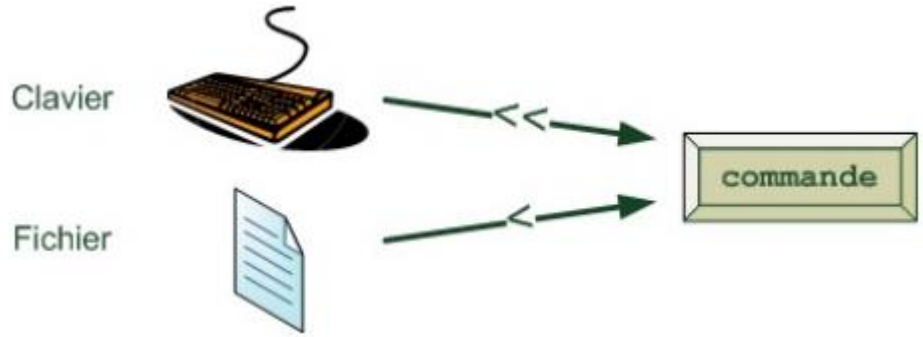


Figure 2.18 : *Illustration de redirection d'entrée*

Vous pouvez tout à fait combiner ces symboles avec ceux qu'on a vus précédemment.

```
$ sort -n << FIN > nombres_tries.txt 2>&1
> 18
> 27
> 1
> FIN
```

2.2.25.7 Chainer les commandes

Passons maintenant au symbole le plus intéressant que vous utiliserez le plus souvent : le pipe | (prononcez "païpe", comme un bon anglais). Son but ? Chaîner des commandes.

a) Théorie

"Chaîner des commandes" ? Cela signifie connecter la sortie d'une commande à l'entrée d'une autre commande.

Figure 2.19 : *Illustration pour chainer des commandes*

En gros, tout ce qui sort de la commande1 est immédiatement envoyé à la commande2. Et vous pouvez chaîner des commandes comme cela indéfiniment ! Cette fonctionnalité est vraiment une des plus importantes et elle décuple littéralement les possibilités offertes par la console. Souvenez-vous : dans le chapitre précédent je vous disais que chaque commande Unix avait un et un seul rôle, mais qu'elle le remplissait bien. Parfois, l'utilité de certaines commandes seules peut paraître limitée, mais celles-ci prennent en général tout leur sens lorsqu'on les combine à d'autres commandes.

b) Pratique

Exemple 1 : Trier les élèves par nom

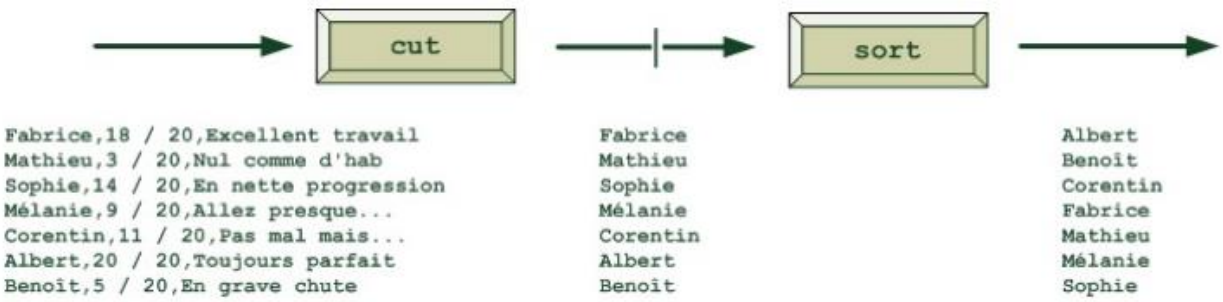
Si vous vous souvenez bien, nous avons toujours un fichier notes.csv qui contient la liste des élèves et leurs notes :

```
Fabrice,18 / 20,Excellent travail
Mathieu,3 / 20,Nul comme d'hab
Sophie,14 / 20,En nette progression
Mélanie,9 / 20,Allez presque la moyenne !
Corentin,11 / 20,Pas mal mais peut mieux faire
Albert,20 / 20,Toujours parfait
Benoît,5 / 20,En grave chute
```

Avec cut, on peut récupérer les noms. Avec sort, on peut les trier par ordre alphabétique. Pourquoi ne pas connecter cut à sort pour avoir la liste des noms triés ?

```
$ cut -d , -f 1 notes.csv | sort
Albert
Benoît
Corentin
Fabrice
Mathieu
Mélanie
Sophie
```

Le « pipe » effectue la connexion entre la sortie de cut (des noms dans le désordre) et l'entrée de sort.



On peut même aller plus loin et écrire cette liste triée dans un fichier :

```
cut -d , -f 1 notes.csv | sort > noms_tries.txt
```

Exemple 2 : Trier les répertoires par taille

La commande du permet d'obtenir la taille de chacun des sous-répertoires du répertoire courant (je vous conseille de vous placer dans votre home en tapant d'abord cd)

```
$ du
4       ./.gnome2_private
40       ./.local/share/Trash/files
4       ./.local/share/Trash/info
12      ./.local/share/Trash
160      ./.local/share
20      ./.local
...
```

Il existe 2 problèmes : cette liste est parfois très longue et elle n'est pas triée. Un problème à la fois. Tout d'abord, on aimerait par exemple avoir cette même liste dans l'ordre décroissant de taille des répertoires pour repérer plus facilement les plus gros d'entre eux qui prennent de la place sur notre disque. Pour avoir cette liste du plus grand au plus petit, il nous suffit d'écrire :

```
du | sort -nr
```

On envoie tout le contenu de la commande *du* à la commande *sor*t qui se charge de trier les nombres au début de chacune des lignes

```
$ du | sort -nr

...

4       ./.evolution/memos/config
4       ./.evolution/calendar/config
4       ./.evolution/cache
4       ./bin
```

Problème : comme les plus gros répertoires ont été affichés en premier, et que j'ai beaucoup de sous-répertoires, je dois remonter très haut dans la console pour retrouver les plus gros d'entre eux. ... Que diriez-vous de connecter cette sortie à head ? Cette commande permet de filtrer uniquement les premières lignes qu'elle reçoit, nous l'avons déjà étudiée dans un chapitre précédent.

```
$ du | sort -nr | head
120920  .
59868   ./.ies4linux
43108   ./.ies4linux/ie6
41360   ./.ies4linux/ie6/drive_c
41248   ./.ies4linux/ie6/drive_c/windows
40140   ./Desktop
34592   ./.ies4linux/ie6/drive_c/windows/system32
16728   ./.ies4linux/downloads
13128   ./.mozilla
13124   ./.mozilla/firefox
```

Vous pouvez paramétrer le nombre de résultats affichés avec l'option -n de head. Si vous avez oublié comment l'utiliser, direction le manuel ou le chapitre qui en parlait. Si vous voulez naviguer à travers

tous les résultats, vous pouvez connecter la sortie à less. Cette commande permet d'afficher des résultats page par page, ça nous est justement utile dans le cas présent où nous avons beaucoup de résultats !

```
du | sort -nr | less
```

Vous allez vous retrouver avec un affichage de less, page par page.

```
120920  .
59868   ./.ies4linux
43108   ./.ies4linux/ie6
41360   ./.ies4linux/ie6/drive_c
41248   ./.ies4linux/ie6/drive_c/windows
40140   ./Desktop
34592   ./.ies4linux/ie6/drive_c/windows/system32
16728   ./.ies4linux/downloads
13128   ./.mozilla
13124   ./.mozilla/firefox
13112   ./.mozilla/firefox/v5p4a55d.default
12604   ./.ies4linux/downloads/ie6
11808   ./.ies4linux/downloads/ie6/FR
5848    ./.mozilla/firefox/v5p4a55d.default/Cache
3656    ./.ies4linux/ie6/drive_c/windows/profiles
3616    ./.ies4linux/ie6/drive_c/windows/profiles/mateo21
3496    ./.ies4linux/ie6/drive_c/windows/profiles/mateo21/Local Settings
3416    ./.ies4linux/ie6/drive_c/windows/profiles/mateo21/Local Settings/Tempora
ry Internet Files
3408    ./.ies4linux/ie6/drive_c/windows/profiles/mateo21/Local Settings/Tempora
ry Internet Files/Content.IE5
2220    ./.ies4linux/ie6/drive_c/windows/fonts
2012    ./ies4linux-2.99.0.1
:
```

Vous pouvez maintenant voir les premiers fichiers (les plus gros) et descendre progressivement vers les fichiers plus petits page par page avec la touche Espace ou ligne par ligne avec la touche Entrée (ou les flèches du clavier).

CHAPITRE 3 INTRODUCTION ADMINISTRATION DES SYSTEMES WINDOWS SERVER

3.1 Gestion de l'Active Directory (AD)

3.1.1 Rappel

La création du premier domain controller crée automatique une forêt et un arbre. Les serveurs suivants pourront donc rejoindre cette forêt et cet arbre. Ils pourront également créer d'autres arbres dans cette forêt. Notez que l'administrateur de la racine (du premier domain controller) peut tout faire sur l'ensemble de la forêt.

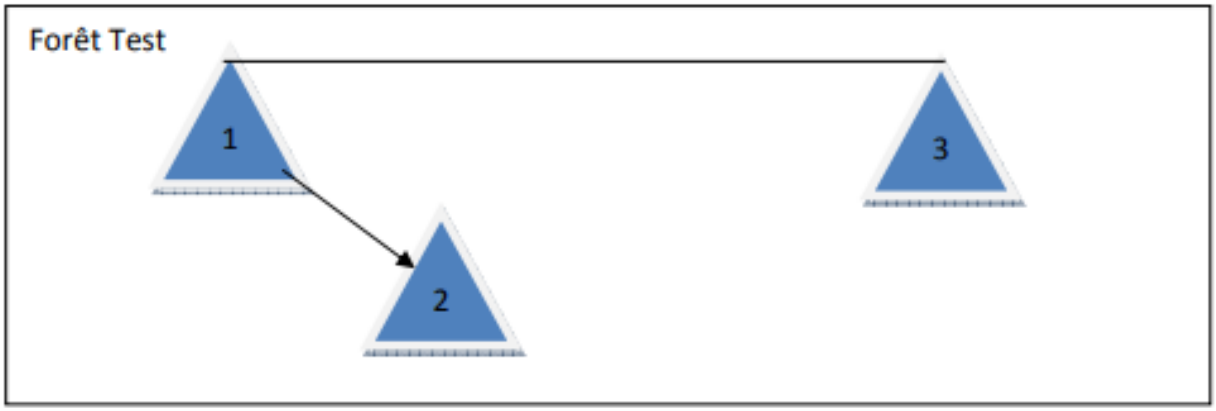


Figure 3.01 : *Répresentation des forêts d'Active Directory*

1. Domain Controler (Racine)
2. Domain enfant
3. Domain d'un autre arbre

Constitution idéal et minimal pour une infrastructure professionnelle. Deux Serveurs qui jouent le rôle de domaine Controller, de global catalog et de DNS. La synchronisation entre ces deux serveurs fonctionne d'une manière incrémentale.

3.1.2 Organization Unit (O.U)

Une OU sert principalement à structure l'active directory en rendant de grosse structure plus visible mais elle sert aussi à appliquer des groupes Policy. Un exemple d'une bonne structure :

Belgique \ Admins

\ Users

\ Computers \ Servers

\ Workstations

\ Groupes

Pour une meilleure visibilité et pour avoir accès à l'ensemble des options, il est conseillé de cocher dans View l'option advanced et voir en tant que container.

3.1.3 Users

Les **users** représentent les comptes des utilisateurs mais aussi des services tiers (Exemple : BackupExec). Il est conseillé de réaliser un compte par utilisateur, un compte par service tiers, deux comptes pour les administrateurs (Un simple et un admin). Notez qu'il y a possibilité de créer des requêtes sur les utilisateurs. Pour ce faire, il faut aller dans Saved Queries ; News Queries.

3.1.4 InetOrgPerson

L'inetOrgPerson est identique à un user classique de Windows et présente les mêmes options mais ce type d'objet est plus compatible avec les autres annuaires comme ceux de Novel, Mac, etc. Il est donc plus que conseillé de créer ce type de compte dans les réseaux hétérogènes

3.1.5 Computers

Représente une machine ou un serveur. Il s'ajoute automatiquement dans l'active directory lors de l'ajout dans le domaine. Par défaut, ils tombent tous dans le container Computers mais il y a possibilité de les faire tomber ailleurs via une ligne de commande.

3.1.6 Les groupes

Il y a deux types de groupe et trois types de scope.

- Le type **distribution** qui n'est important que dans les organisations qui possède Exchange.
- Le type **Security** qui sert à appliquer des permissions. Notez que le type Security fait automatiquement le même qu'un type distribution.

Les scopes : Domain local (permission), Global (réunir des users), universal (pour plusieurs arbres).

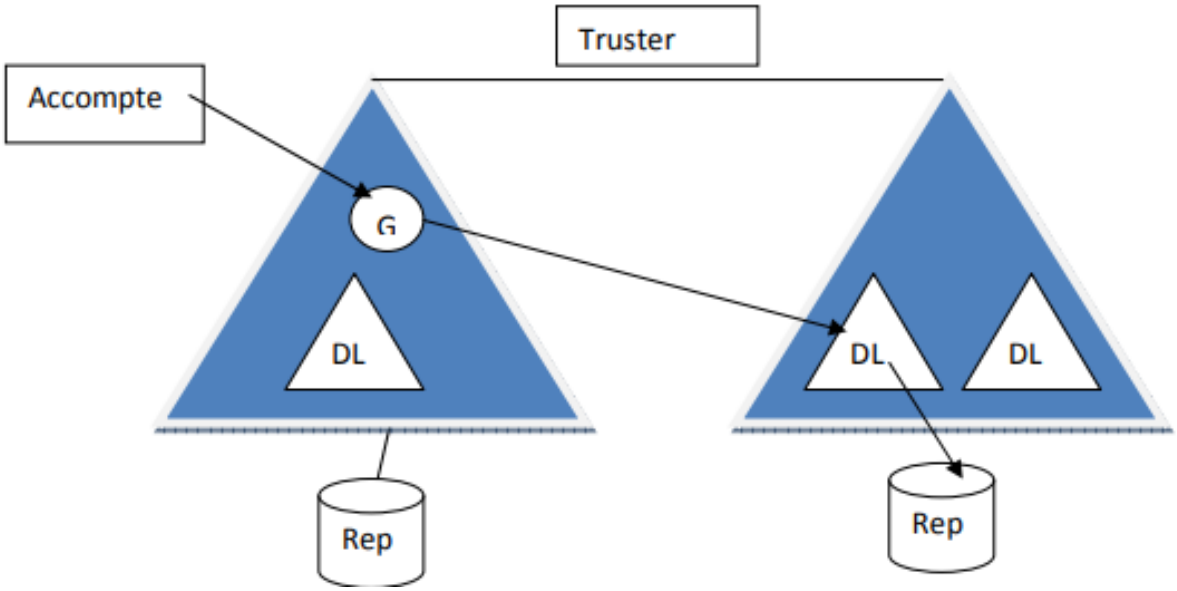


Figure 3.02 : *Illustration des groupes*

Microsoft recommande pour des raisons de sécurité de toujours mettre des accomptes dans un Global et ensuite d'appliquer celui-ci dans un domaine local. Vous ajoutez de cette manière une couche de sécurité.

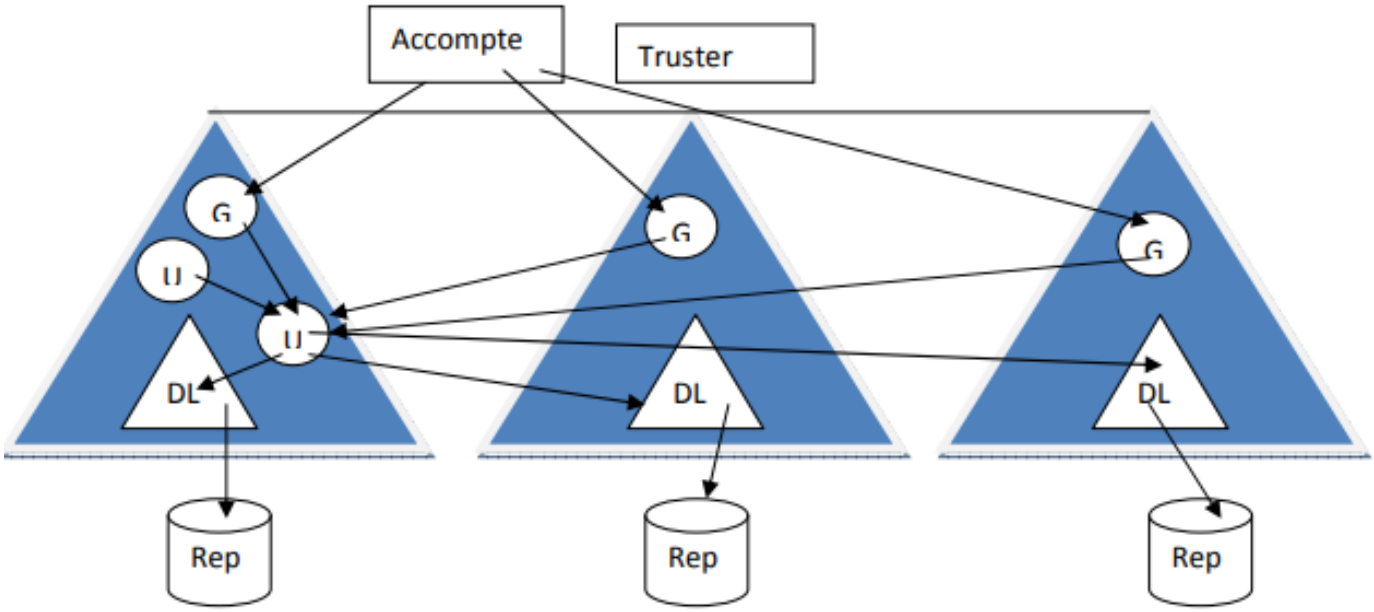


Figure 3.03 : *Illustration des accomptes et truster*

Toujours pour une question de sécurité, vous devez créer un DL par ressources et par type de ressource.

L'universal est utilisé pour les grandes forêts ou la gestion de la sécurité doit être plus poussée.

3.1.7 Permission dans l'Active Directory

Les Bultin contiennent tous les groupes de domaines locaux pour attribuer des droits aux utilisateurs ayant un rôle administratif.

Les plus utilisés :

- Entreprise admins : Droit sur l'ensemble de la forêt
- Schema admins : Responsable de l'AD
- Acompte operators : Création des groupes, users.
- Server operators : Opération de maintenance mais aucun accès à l'AD.

Configuration par défaut dans une forêt.

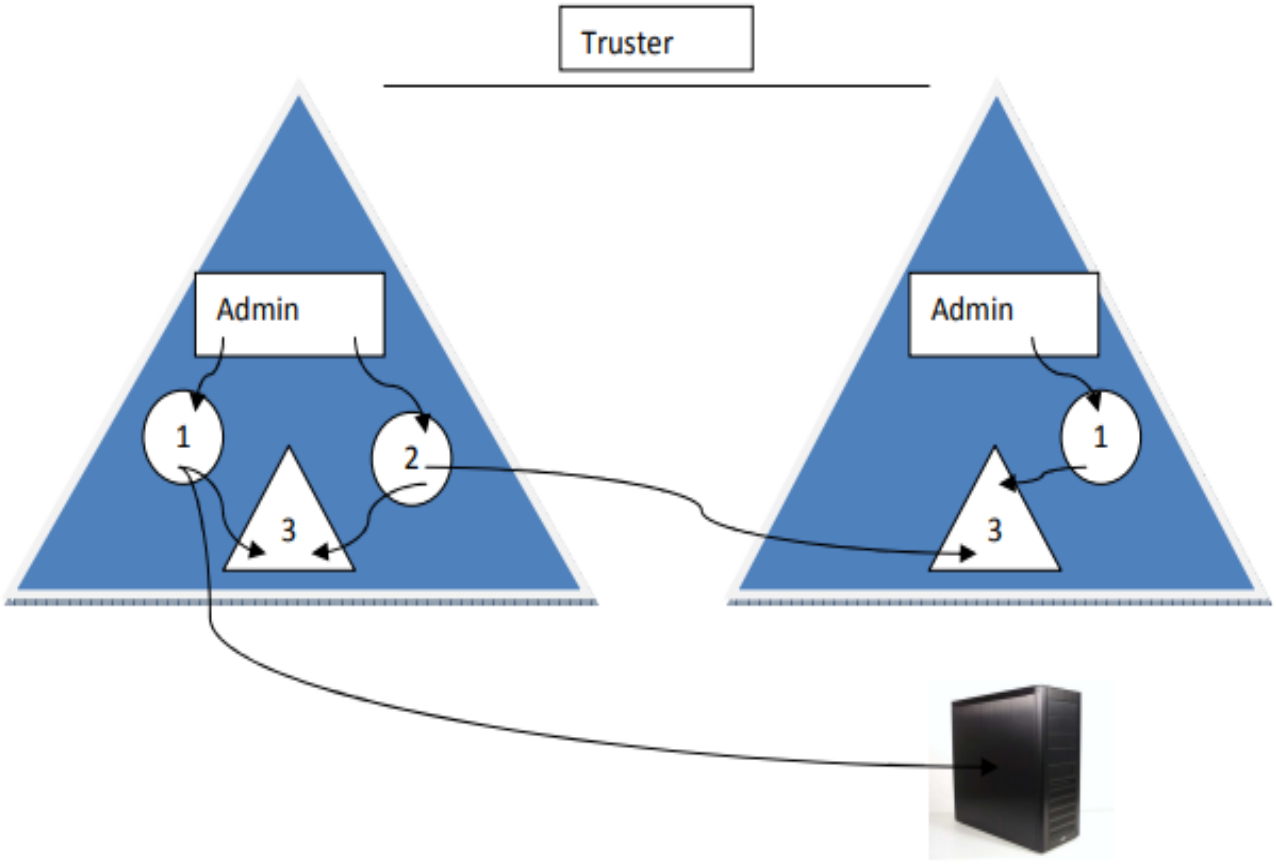


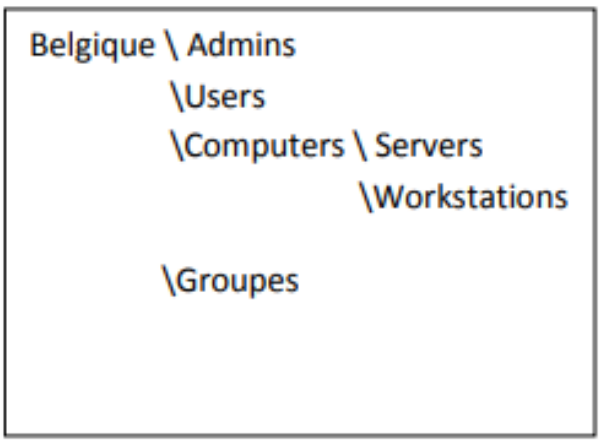
Figure 3.04 : *C*Illustration de configuration dans une forêt*

1. Domain Admin (groupe global)
2. Entreprise admin (Groupe universel)
3. Administrators (Domain Local)

3.1.8 Délégation des droits AD

Comme son nom l'indique ceci permet de déléguer des rôles dans l'active directory. Une délégation peut se faire sur une OU, un type d'objet, une tâche, une propriété d'un objet.

Pour une question de sécurité, les OU doivent être bien structuré et les personnes qui auront des rôles déléguer ne devront jamais avoir la possibilité de s'en ajouter des nouveaux
Exemple de bonne OU Exemple de bonne OU :



3.2 Accès aux ressources

L'accès aux ressources se configure en trois étapes : Le partage, la publication et la permission.

3.2.1 Le partage

Pour réaliser un partage, il suffit de cliquer sur propriétée et d'aller dans l'onglet share. On peut ensuite configurer le nom de celui-ci ainsi que ses permissions que nous verrons plus loin. Vous pouvez bien entendu créer des partages invisibles par les utilisateurs en ajoutant le signe $ à la fin du nom de partage. Ce genre de partage est très utile pour créer des répertoires contenant des sources, des packages d'installations, etc.

Notez qu'on voit toujours l'ensemble des partages via le computer management.

3.2.2 La publication

La publication représente la possibilité de rendre facilement un partage accessible à un utilisateur. Soit via un mapping, soit via un raccourci. Cette procédure se réalise souvent via un script de démarrage ou via une policy.

Vous pouvez également publier le partage dans l'active directory. On peut voir rapidement et d'une manière centraliser les partages sur une machine ou surtout sur les serveurs.

Pour ce faire, il faut aller dans le computer management et cliquer sur publish dans les propriétés du partage.

3.2.3 Les permissions

L'utilisateur reçoit une série de clé (token) lors de son identification sur le réseau. Ses clés sont représentées par des SID. Exemple : User test : SID …..- 1307, global compta : SID …..-1334

Quand on demande une ressource, il y a vérification via les ACL ou sont vérifié les permissions (grant).

Notez qu'un deny explicite est toujours le plus restrictif concernant les ACL.

3.2.4 La gestion des permissions

Il y a deux possibilités de gérer les permissions, soit via le sharing, soit via les droits NTFS. Ceux-ci sont complémentaires et la meilleure représentation est sans doute une double porte.

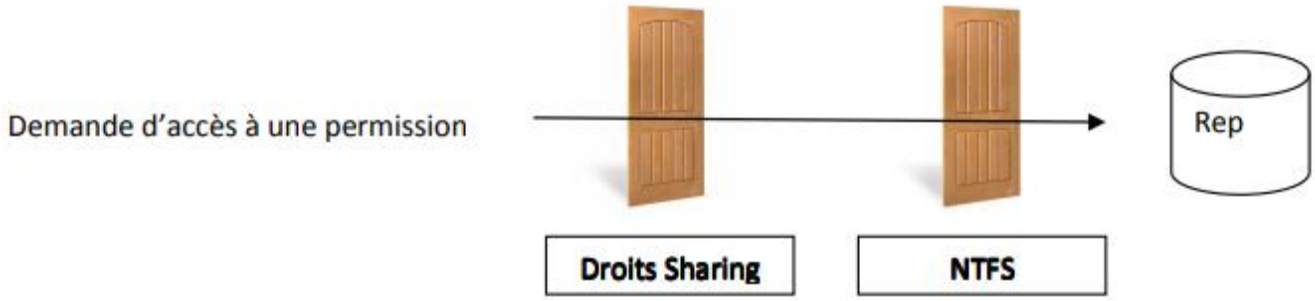


Figure 3.05 : *Illustration de partage*

C'est toujours le plus restrictif qui l'emporte. Même si le sharing vous laisse passer en écriture si le NTFS ne l'autorise pas vous serez bloqué et inversement.

Droits Sharing

- **Read :** Lecture
- **Change :** Lecture, écriture, suppression
- **Full Control :** Lecture, écriture, suppression + modification des permissions

Conseil : Au lieu de laisser Everyone, il vaut mieux mettre Authenticated Users ce qui renforce un peu la sécurité.

NTFS

- **Read :** On peut voir le contenu d'un dossier et ses propriétés ainsi que des fichiers.
- **Read& Exécute & Exécute & Exécute :** On peut exécuter des fichiers exécutable (Exe,bat, cmd, vbs,…) et on peut lire les fichiers. Au niveau des répertoires, on a les droits de traverse.
- **List Folder Contents :** Traverse et read des dossiers.
- **Write :** Création de fichier, répertoires, possibilité de renommer et de modifier les contenus.
- **Modify :** Toutes les options et en plus, on peut supprimer les fichiers. Idem sur les répertoires mais ceux-ci doivent être vide.
- **Full control :** Toutes les options + modification des permissions. On peut également ajouter des permissions spéciales via l'onglet advanced. On peut par exemple empêcher de lire les propriétés des fichiers et répertoires. Les options sont vastes et permettent une configuration très adaptée.

3.3 La notion d'héritage

Par défaut, l'héritage sur les dossiers est activé. Un nouveau dossier héritera donc des permissions de son parent. On peut couper cet héritage en vidant les permissions ou en recopiant les permissions dans l'onglet advanced du NTFS.

Attention au groupe users qui a des droits par défaut sur tous les répertoires. Il est plus que conseiller de le changer. Attention également au Creator Owner qui a des droits fulls control par défaut. Il vaut mieux limiter le creator owner en lui donnant les mêmes permissions que son groupe sur la ressource.

3.4 Gestion des données

3.4.1 Limitation de l'espace disque

Il y a deux manières pour limiter l'espace disque consommée par les utilisateurs.

Quotas	File Server Ressource Manager
Basé sur le propriétaire des fichiers	Installer le service « File service Ressour Manager » du rôle File Server.
Une gestion par partition	Limite la taille de certain dossier
	Bloquer des extensions (File screening manager)
	Reporting d'activité

Solution idéale :

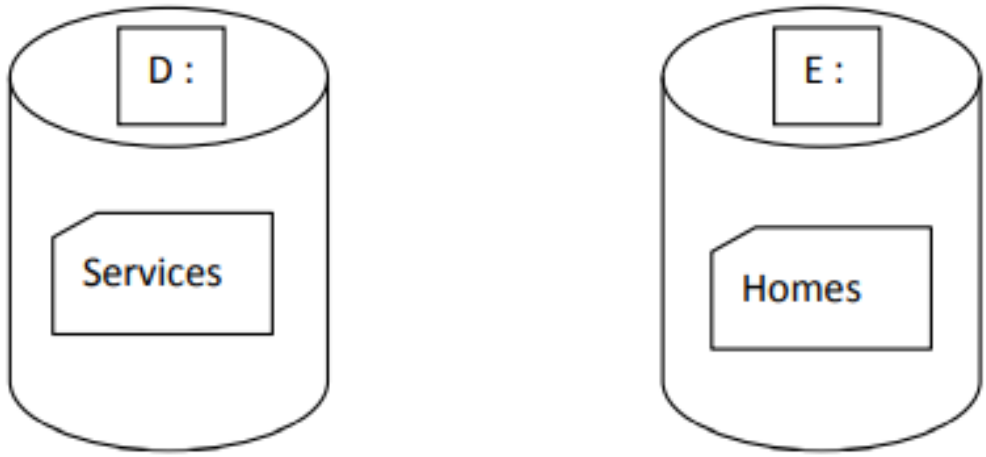


Figure 3.06 : *Solution idéale pour la limitation d'espcace disque*

Du File Server Ressource Manager sur les services et de la gestion de quotas classique sur les homes.

Solution moins idéal :

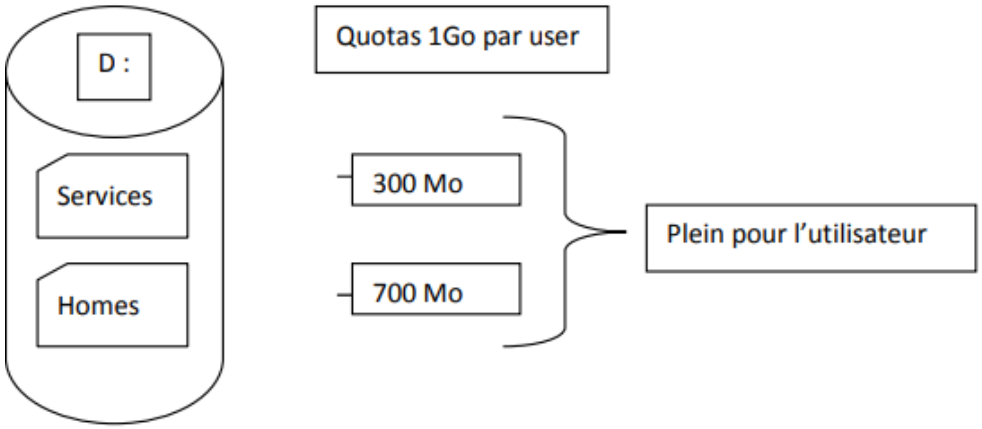


Figure 3.07 : *Solution moins idéale pour la limitation d'espace disque*

3.4.2 Les quotas

Les quotas s'activent directement sur la partition. C'est une configuration générale pour l'ensemble de la partition mais on peut néanmoins spécifier une taille plus grande pour certains utilisateurs. On a également une partie qui peut servie de « reporting » sur l'espace déjà consommé.

Si vous décidez de modifier les quotas, les nouvelles modifications ne seront effectivement que sur les nouveaux utilisateurs. Il faudra repasser manuellement sur les utilisateurs déjà existants.

3.4.3 File Server Ressource Manager

Vous trouverez le « File Server Ressource Manager » dans les outils d'administration à condition que le service soit installé. Vous pourrez soit utiliser un Quota Template qui est définit soit en créer un nouveau et l'appliquer sur un répertoire.
Vous pouvez également envoyer des messages à l'administrateur ou l'utilisateur quand celui-ci arrive à la taille limite. C'est également via le « File Server Ressource Manager » que vous pourrez empêcher l'écriture de fichier à extension bien définie. Comme par exemple des MP3 ou des fichiers vidéo.

3.4.4 Le Shadow Copy

Le service Shadow-Copy est destiné au stockage de fichiers sur des dossiers partagés.

Il permet :
• De récupérer les fichiers accidentellement effacés.
• De récupérer les fichiers malencontreusement écrasés ou mis à jour.
• D'implémenter la notion de version sur les documents sur lesquels on travaille.
• De limiter les accès physiques aux serveurs pour effectuer des backups de fichiers à restaurer, en offrant des possibilités de récupération de fichiers directement sur un poste client supportant le client VSS.

Lors de la configuration de ce service, il faut lui attribuer une taille maximale comme par exemple 15% de la partition.

On ne sait pas déplacer un « Shadow Copy » ou même en faire un backup. En cas de crash disque, les données sont perdues et il faudra que les utilisateurs repartent de 0. Il est donc bien important de ne pas confondre le « Shadow Copy » avec un système de backup. Notez qu'on peut très bien dédier un disque dur en SATA pour jouer ce rôle. Ce qui permet donc de délocaliser les données.

3.4.5 DFS : Distributed File System

DFS (Distributed File System) fournit aux utilisateurs un moyen simple d'accéder à des données reparties et distribuées sur un réseau. Un dossier partagé DFS sert de point d'accès à d'autres dossiers sur le réseau. Il permet par exemple de regrouper différents partages stockés sur des différents serveurs à un seul point.

Il permet également une réplication de données entre deux serveurs.

• Idéal pour avoir les données proches des utilisateurs.

• Un backup centralisé.

• Fault Tolerance mais il est loin d'être idéal dans cette tâche.

Le mieux sera toujours d'avoir un système de clustering. Il faut d'abord définir un « name space » ou un « root » sur le Domain Controler. Ensuite, on crée le lien qui se rattache à ce « name space ». C'est évidement un répertoire partagé.

Il suffit ensuite de browser le réseau directement via le nom du domaine. \\NomDomain

Dans le cas de la synchronisation :

1. Créer deux « name space » identiques sur chaque serveur.
2. Créer des links sur les serveurs.
3. On crée la réplication en définissant un domain controler comme primaire et la topology de la synchronisation.

3.5 Group Policy

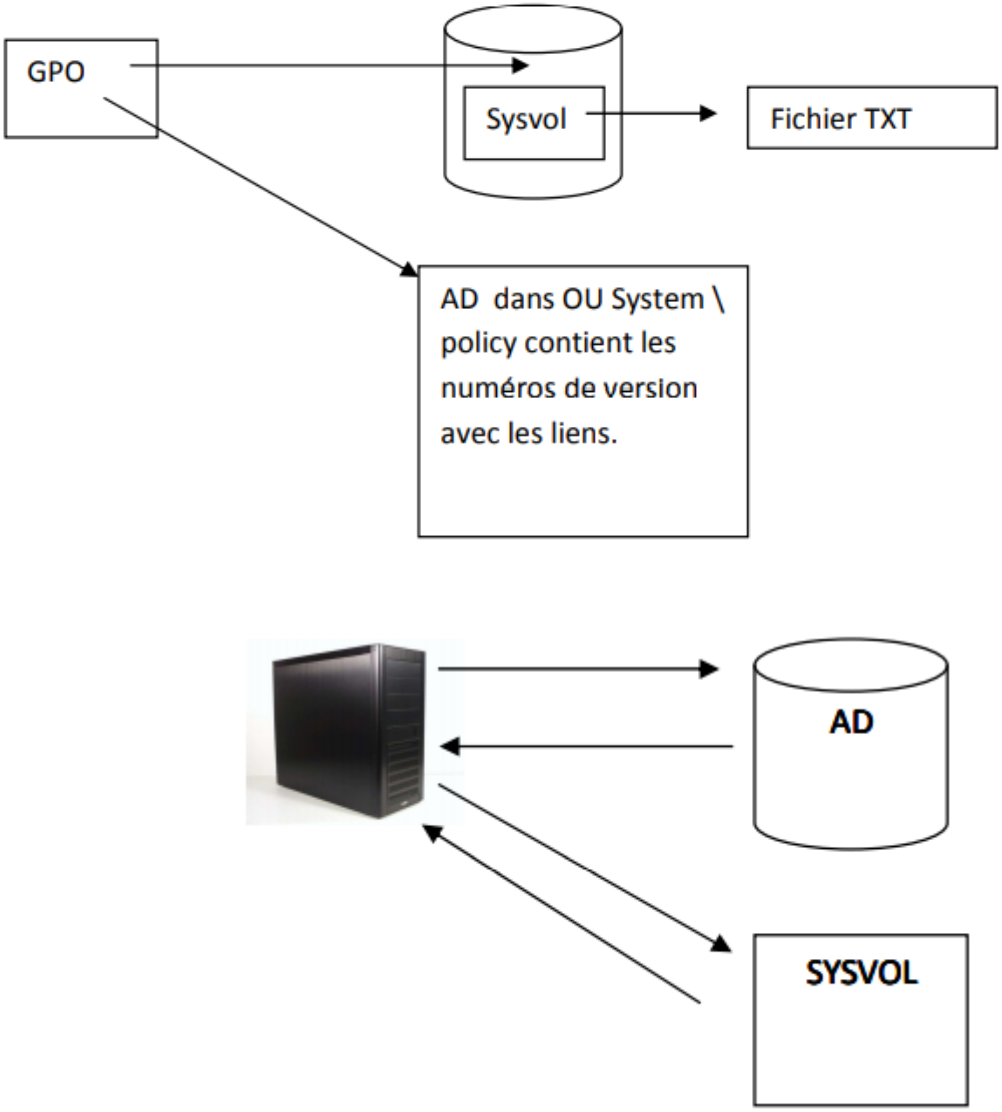
Elles sont utilisées pour la restriction, les scripts, la configuration.

Dans la catégorie configuration, on retrouve les services, les softs, la sécurité, le paramétrage.

Elles peuvent s'appliquer à une machine, une OU, un groupe, un objet. Elles s'appliquent automatiquement toute les 90 à 120 minutes mais elles peuvent être forcée via la commande **"gpupdate /force "**

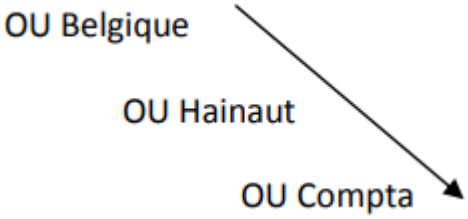
L'outil pour les administrer est Group policy Management. Le concept consiste à créer un template de police et puis de l'appliquer sur une OU.

3.5.1 Fonctionnement



Les polices de sécurité ne sont en fait que des fichiers textes qui sont indexé dans l'active Directory. A chaque fois qu'un pc démarre ou qu'un utilisateur s'authentifie, il y a échange avec l'active Directory pour voir s'il y a eu des changements. Si c'est le cas, la machine télécharge les fichiers qui ont été ajoutés ou qui ont été modifiés. La commande **« gpupdate /force »** oblige la machine à télécharger de nouveau l'entièreté des fichiers textes.

Par défaut, une GPO s'applique à tout ce qui est en-dessous d'elle.



On peut néanmoins forcer le système pour que ce soit une police ascendante qui domine avec la commande force.

On peut également appliquer une policy à un groupe de personne. Pour ce faire, pendant l'édition de la GPO, il faut sélectionner propriété sur le nom de la GPO et mettre les droits nécessaires dans l'onglet sécurité.

Exemple de policy Exemple de policysur les computers sur les computers sur les computers :

On peut par exemple renommer les comptes de l'administrator local des machines mais on ne peut malheureusement pas changer son mot de passe. Pourquoi faire ça ? Simplement par ce que ce compte est connu par tout le monde et qu'il a tous les droits. On ne peut pas le bloquer car son SSID finit par 500.

On pourrait aussi mettre le Number of previous logon pour empêcher les machines de se loger sans la présence d'un domaine. Surtout les Workstation qui n'ont pas besoin de cette option car une personne qui aurait son compte désactivé pourrait très bien se connecter à sa machine en retirant simplement le câble réseau. En désactivant cette option, il ne sera plus en mesure de se loger sur la machine.

On pourrait aussi injecter des groupes d'utilisateurs dans un groupe local sur des machines via le Restricted groups. Par exemple, l'équipe de l'Help-Desk qui a besoin d'avoir l'ensemble des droits admins en local pour effectuer des tâches de maintenance.

Il arrive parfois que certains programmes mal conçus doivent avoir les droits administrateurs pour s'exécuter. Il est bien entendu très dangereux de donner ces droits à des simples utilisateurs. Via les GP dans "File System", vous pouvez donner des droits sur certains répertoires des machines de votre parc sans pour autant donner plus de droits. Pour se rendre compte exactement des droits d'un programme, il faut le faire tourner en même temps que « FileMon » et « RegMon » (produit gratuit de Microsoft) qui donnent un rapport exact des accès du programme sur la machine.

On peut également bloquer des programmes via le software restriction policies qui se définit via des règles. Ces règles sont divisées suivant des certificats, des « network zone », des « path rule » et des « hash rule ». Le path rule permet de dire qu'on ne peut plus exécuter tel fichier exécutable mais on peut toujours contourner le problème en renommant l'exécutable à la différence du « Hash Rue » qui fait d'une certaine façon une photo de l'exécutable. Il est alors impossible de lancer l'exécutable et ce même en le renommant.
On peut limiter la bande passante d'un programme en upload sur le réseau via la « policy-based QOS ».

On peut alléger la charge réseau générée par les « Vista » et les « Seven » en désactivant les services Link-layer Topologie et Microsoft peer to peer dans l'administrative.

3.5.2 Password policy

Une Password policie ne peut s'attacher que sur le domaine. Si on l'attache sur une OU plus bas, ça ne sera appliqué que sur les comptes locaux. On peut quand même créer une police **Fine Grained PWD s**ur des groupes d'utilisateurs mais il faut êtr PWD e totalement en domaine level 2008.

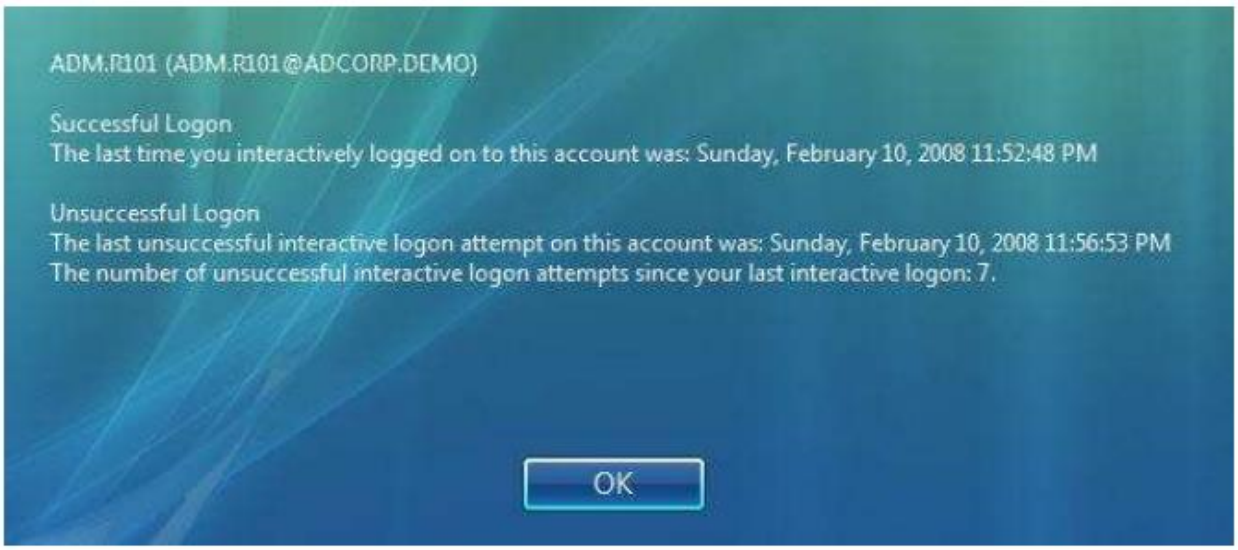
Quelques options :
Account lockout duration : Bloque le compte pendant X minutes. Si on le met sur 0, c'est un blocage infini. Recommandation 30 min.

Account lockout threshold : Nombre d'erreurs qu'on permet à l'utilisateur. Recommandation entre 3 à 10 erreurs.

Reset account lockout counter after : Temps d'observation des erreurs. Exemple : 30 minutes

Fine Grained PWD : Pour ce faire, il faut utiliser Fine Grained PWD **ADSI Edit** (sorte de registery pour l'AD) ADSI Edit qui permet de voir l'ensemble des attributs de l'AD et de les modifier. Il faut aller sur le container System, Password Settings container, création d'un nouvel objet. La première valeur permet seulement de donner un ordre entre différentes policie. Les autres valeurs représentent la configuration de celle-ci. Une fois la police créée, il faut l'appliquer dans l'active directory au même endroit et aller sur les propriétés et puis dans l'onglet Attribute Editor à la ligne msDS-PSOAppliesTO et rajouter le compte ou le groupe. La dernière étape consiste à forcer les gens à changer les mots de passe pour que celle-ci soit appliquée.

Public last logon information " ('windows componeme Public last logon information nt \ windows logon options à activer sur le domain et sur OU des PC) qui ne fonctionne qu'avec Vista ou Seven. Cette option permet de voir quand on s'est authentifié la dernière fois avec succès et quand la dernière fois, il y a eu quelqu'un qui a essayeé de se connecter sans succès avec son login.



Outils pratique : Admodify.net qui permet d'éditer Outils pratique : les attributs de tous les objets qu'on désire.

EXERCICES

ATELIER 1 : Arborescence et fichiers

Exercice 1.1 : Création de répertoires

• Afficher le répertoire courant.

```
[stagiaire]$ pwd
/home/stagiaire
```

• Se déplacer de deux façons différentes sous le répertoire /home.

Chemin absolu :

```
[stagiaire]$ cd /home/
```

Chemin relatif :

```
[stagiaire]$ cd ..
```

• Vérifier que /home soit bien le nouveau répertoire courant.

```
[stagiaire]$ pwd
/home
```

• Retourner dans le répertoire de connexion, et vérifier.

```
[stagiaire]$ cd
[stagiaire]$ pwd
/home/stagiaire
```

• Créer les répertoires suivants : /home/stagiaire/tickets/ /home/stagiaire/tickets/pierre/ /home/stagiaire/tickets/jacques/

```
[stagiaire]$ mkdir -p tickets/pierre tickets/jacques
[stagiaire]$ ls tickets/
jacques pierre
```

Exercice 1.2 : Gestion des fichiers

• Créer le fichier /home/stagiaire/tickets/listing_en_cours.

```
[stagiaire]$ touch tickets/listing_en_cours
```

• Copier ce fichier dans les répertoires /home/stagiaire/tickets/pierre et /home/stagiaire/tickets/jacques. Vérifier la taille de ces fichiers.

```
[stagiaire]$ cp tickets/listing_en_cours tickets/pierre/
[stagiaire]$ cp tickets/listing_en_cours tickets/jacques/
```

Vérifier la copie en comparant les tailles :

```
[stagiaire]$ ls -lh tickets/listing_en_cours tickets/pierre/listing_en_cours
tickets/jacques/listing_en_cours
-rw-r--r-- 1 stagiaire users 0 [···] tickets/listing_en_cours
-rw-r--r-- 1 stagiaire users 0 [···] tickets/pierre/listing_en_cours
-rw-r--r-- 1 stagiaire users 0 [···] tickets/jacques/listing_en_cours
```

La taille des fichiers est identique, 0 octet (ils sont vides).

• Renommer le fichier /home/stagiaire/tickets/jacques/listing_en_cours en listing_fini.

```
[stagiaire]$ mv tickets/jacques/listing_en_cours tickets/jacques/listing_fini
```

• Déplacer et renommer le fichier /home/stagiaire/listing_en_cours en /STAGE/commandes/archive_listing.

Pour déplacer le fichier listing_en_cours du répertoire /home/stagiaire/tickets vers /STAGE/commandes, il faut d'abord créer ce dernier dossier :

```
[stagiaire]$ mkdir -p /STAGE/commandes
```

Puis le déplacer :

```
[stagiaire]$ mv tickets/listing_en_cours /STAGE/commandes/archive_listing
```

Exercice 1.3 : Gestion des répertoires

• Copier le répertoire /home/stagiaire/tickets/pierre/ et son contenu en le renommant /home/stagiaire/tickets/sauvegarde.

```
[stagiaire]$ cp -r tickets/pierre/ tickets/sauvegarde
```

• Renommer le répertoire /home/stagiaire/tickets/sauvegarde/ en /home/stagiaire/tickets/archives.

```
[stagiaire]$ mv tickets/sauvegarde/ tickets/archives
```

• Copier le répertoire /home/stagiaire/tickets/ dans le répertoire /STAGE/commandes/.

```
[stagiaire]$ cp -r tickets/ /STAGE/commandes/
```

Exercice 1.4 : Suppression de fichiers et répertoires

• Afficher le contenu des répertoires /home/stagiaire/tickets/jacques/ et /home/stagiaire/tickets/pierre/.

```
[stagiaire]$ ls tickets/jacques/ tickets/pierre/
tickets/jacques/:
listing_fini

tickets/pierre/:
listing_en_cours
```

• Supprimer le répertoire /home/stagiaire/tickets/jacques/ avec la commande rmdir.

```
[stagiaire]$ rmdir tickets/jacques/
rmdir : échec de suppression de « tickets/jacques/ » : Le dossier n'est pas vide
[root]# rm -f tickets/jacques/listing_fini
[root]# rmdir tickets/jacques/
```

• Supprimer le répertoire /home/stagiaire/pierre/ en une seule commande.

```
[stagiaire]$ rm -rf tickets/pierre/
```

Vérifier les suppressions :

```
[stagiaire]$ ls -R tickets/
tickets/:
archives

tickets/archives:
listing_en_cours
```

https://www.youtube.com/watch?v=QySrZv5_cis (YouTube video)

ATELIER 2 : Recherches et filtres

Exercice 2.1 : Affichage et filtres

• Copier dans le répertoire de connexion /home/stagiaire le fichier /etc/passwd.

Dorénavant, travailler sur cette copie.

```
[stagiaire]$ cp /etc/passwd ./
```

• Afficher les 7 premières lignes puis les 3 dernières.

```
[stagiaire]$ head -n 7 /home/stagiaire/passwd
[stagiaire]$ tail -n 3 /home/stagiaire/passwd
```

• Retrouvez la ligne contenant alain.

```
[stagiaire]$ grep "^alain" /home/stagiaire/passwd
alain:x:500:500::/home/GroupeA/alain:/bin/bash
```

Ou,

```
[stagiaire]$ less /home/stagiaire/passwd
```

Puis,

```
/alain
```

• Trier ce fichier par ordre d'`UID` croissant.

```
[stagiaire]$ sort -k3 -t: -n /home/stagiaire/passwd
```

• Combien y a-t-il d'utilisateurs créés sur le serveur ?

```
[stagiaire]$ wc -l /home/stagiaire/passwd
39 /home/stagiaire/passwd
```

Le fichier passwd contient 39 lignes, il y a donc 39 utilisateurs créés sur le serveur.

• Déplacer ce fichier dans le répertoire /STAGE/commandes

```
[stagiaire]$ mv /home/stagiaire/passwd /STAGE/commandes
```

• Afficher les fichiers passwd présents dans le dossier /STAGE en précisant leur type.

```
[stagiaire]$ find /STAGE -name "passwd" –exec file {} \;
/STAGE/commandes/passwd: ASCII text
```

https://www.youtube.com/watch?v=0kiGdvjOrDA (YouTube video)

ATELIER 3 : tubes et redirections

Exercices 3.1

• Créer un fichier /home/stagiaire/suivi_admin.

```
[stagiaire]$ touch /home/stagiaire/suivi_admin
```

• Se connecter sur le terminal 2 et suivre les modifications du fichier en direct. Se connecter sur le terminal 2 avec CTRL+SHIFT+ALT+F2 et afficher le fichier en temps réel

```
[stagiaire]$ tail -f /home/stagiaire/suivi_admin
```

Retourner sous le terminal 1 et ajouter au fichier suivi_admin le texte Voici les répertoires de /STAGE/commandes/gestion/ :

• Retourner sur l'interface graphique avec ALT+F1 et modifier le fichier :

```
[stagiaire]$ echo "Voici les répertoires de /STAGE/commandes/gestion/ :" >
/home/stagiaire/suivi_admin
```

• Toujours dans suivi_admin, ajouter la liste des répertoires de /STAGE/commandes/gestion/ en faisant apparaître les tailles avec l'indication Ko, Mo, Go ou To.

```
[stagiaire]$ find /STAGE/commandes/gestion/ -type d -exec ls -sdh '{}' \; >>
/home/stagiaire/suivi_admin
```

• Vérifier le contenu du fichier en basculant sur le terminal 2. Se connecter sur le terminal 2 avec CTRL+SHIFT+ALT+F2 • Retourner sous terminal 1 et ajouter au fichier suivi_admin le texte Voici les personnes ayant un fichier listing_en_cours sous /STAGE/commandes/gestion/ :
Retourner sur l'interface graphique avec ALT+F1.

```
[stagiaire]$ echo "Voici les personnes ayant un fichier listing_en_cours sous
/STAGE/commandes/gestion/:" >> /home/stagiaire/suivi_admin
```

• Tapez la commande :

```
[stagiaire]$ find /STAGE/commandes/tickets -listing_en_cours >>
/home/stagiaire/suivi_admin 2>/home/stagiaire/erreur
```

• Basculer sur le terminal 2 et vérifier que la commande se soit bien exécutée. Sur le terminal 2, rien n'a été modifié. En fait, la commande saisie comporte une erreur. Son affichage a donc été redirigé sur le canal d'erreur, le fichier erreur, et non suivi_admin.
• Corriger la commande pour remplir le fichier suivi_admin. Il faut donc corriger la commande :

```
[stagiaire]$ find /STAGE/commandes/tickets -name listing_en_cours >>
/home/stagiaire/suivi_admin 2> /home/stagiaire/erreur
```

• Afficher parmi les 3 dernières lignes du fichier suivi_admin celles qui contiennent pierre

```
[stagiaire]$ tail -n3 /home/stagiaire/suivi_admin | grep "pierre"
/STAGE/commandes/tickets/pierre/listing_en_cours
```

• Retourner sous le terminal 2 et se déconnecter

Taper Ctrl+d puis :

```
[stagiaire]$ exit
```

ATELIER 4 Gestion des utilisateurs et des groupes

Exercice 4.1 : Fichiers de configuration

• Sauvegarder les fichiers de configuration des groupes et des utilisateurs en les copiant dans le répertoire /STAGE/utilisateurs.

```
[root]# mkdir /STAGE/utilisateurs
[root]# cp /etc/passwd /etc/shadow /etc/group /etc/gshadow /STAGE/utilisateurs/
```

Exercice 4.2 : Groupes et utilisateurs

• Créer les groupes et les utilisateurs suivant

Groupe	GID	Utilisateurs
LINUX	1001	antoine, xavier
WINDOWS	1002	vincent, david
GroupeD	503	
GroupeV	504	
GroupeX	505	

Utilisateur	UID	REPERTOIRE
antoine	2001	/home/linux/antoine
xavier	2002	/home/linux/xavier
vincent	2003	/home/windows/vincent
david	2004	/home/windows/david

Créer dans un premier temps les 5 groupes en précisant leur GID :

```
[root]# groupadd -g 1001 LINUX
[root]# groupadd -g 1002 WINDOWS
[root]# groupadd -g 503 GroupeD
[root]# groupadd -g 504 GroupeV
[root]# groupadd -g 505 GroupeX
```

Créer ensuite les répertoires parents des répertoires de connexion des utilisateurs :

```
[root]# mkdir /home/{linux,windows}
```

Ajouter enfin les 4 utilisateurs en précisant leur UID, le GID de leur groupe principal, leur répertoire de connexion ainsi que leur shell :

```
[root]# useradd -u 2001 -g 1001 -d /home/linux/antoine antoine
[root]# useradd -u 2002 -g 1001 -d /home/linux/xavier xavier
[root]# useradd -u 2003 -g 1002 -d /home/windows/vincent vincent
[root]# useradd -u 2004 -g 1002 -d /home/windows/david david
```

Exercice 4.3 : Commentaires

• Modifier les commentaires des utilisateurs et visualiser les modifications

Utilisateur	Commentaire
antoine	Antoine LM
xavier	Xavier S
vincent	Vincent B
david	David B

```
[root]# usermod -c "Antoine LM" antoine
[root]# usermod -c "Xavier S" xavier
[root]# usermod -c "Vincent B" vincent
[root]# usermod -c "David B" david
```

Visualiser les modifications dans le fichier /etc/passwd :

```
[root]# tail -n 4 /etc/passwd
antoine:x:2001:1001:Antoine LM:/homme/linux/antoine:/bin/bash
xavier:x:2002:1001:Xavier S:/homme/linux/xavier:/bin/bash
vincent:x:2003:1002:Vincent B:/homme/linux/vincent:/bin/bash david:x:2004:1002:David
B:/homme/linux/david:/bin/bash
```

ATELIER 5 : Gestion avancée des utilisateurs

Exercice 5.1 : Groupes secondaires

Inviter les utilisateurs suivant dans le groupe adéquat.

Utilisateur	Groupe
antoine	GroupeA
xavier	GroupeX
vincent	GroupeV
david	GroupeD

Ajouter le groupe secondaire à l'utilisateur :

```
[root]# usermod -aG GroupeA antoine
[root]# usermod -aG GroupeX xavier
[root]# usermod -aG GroupeV vincent
[root]# usermod -aG GroupeD david
```

Ou ajouter l'utilisateur au groupe secondaire :

```
[root]# gpasswd -a antoine GroupeA
[root]# gpasswd -a xavier GroupeX
[root]# gpasswd -a vincent GroupeV
[root]# gpasswd -a david GroupeD
```

Inviter ces utilisateurs dans le groupe users :

```
[root]# gpasswd -a antoine users
[root]# gpasswd -a xavier users
[root]# gpasswd -a vincent users
[root]# gpasswd -a david users
```

Rediriger les informations à propos des groupes de ces utilisateurs vers le fichier /STAGE/utilisateurs/modifications. Rediriger le résultat de la commande id :

```
[root]# id antoine >> /STAGE/utilisateurs/modifications
[root]# id xavier >> /STAGE/utilisateurs/modifications
[root]# id vincent >> /STAGE/utilisateurs/modifications
[root]# id david >> /STAGE/utilisateurs/modifications
```

Exercice 5.2 : Mots de passe

• Définir les mots de passe des utilisateurs.

Utilisateur	MDP
antoine	tuxtux
xavier	tuxone
vincent	formatux
david	itstime

```
[root]# passwd antoine
[root]# passwd xavier
[root]# passwd vincent
[root]# passwd david
```

Exercice 5.3 : Pérennité du compte et du mot de passe

• Configurer les paramètres du mot de passe et du compte de l'utilisateur antoine.

◦ Durée maximale du mot de passe : 60 jours

◦ Durée minimale du mot de passe : 45 jours

◦ Délai avant expiration du mot de passe : 5 jours

◦ Inactivité du mot de passe : 10 jours

◦ Durée de validité du compte : 365 jours

```
[root]# chage -m 45 -M 60 -I 10 -W 5 -E `date --date '1 year' +%Y-%m-%d` antoine
```

Ou en mode interactif

```
[root]# chage antoine
```

Exercice 5.4 : Verrouillage du compte

• Verrouiller le compte david puis le compte de vincent avec deux commandes distinctes. Verrouiller le compte utilisateur :

```
[root]# passwd -l david
```

Et,

```
[root]# usermod -L vincent
```

• L’utilisateur vincent devra modifier son mot de passe à la première connexion. Demander un changement de mot de passe à la prochaine connexion :

```
[root]# chage -d 0 vincent
```

Exercice 5.5 : Validation des changements

Afficher les 4 dernières lignes du fichier /etc/shadow et visualiser les changements apportés par cet exercice.

```
[root]# tail -n 4 /etc/shadow
antoine:$6$...:16897:45:60:7:10:1726
xavier:$6$...:16897:0:99999:7:::
vincent:!$6$...:0:0:99999:7:::
david:!!$6$...:16897:0:99999:7:::
```

ATELIER 6 : Gestion avancée des comptes

Exercice 6.1 : Création par défaut

• Configurer les paramètres par défaut pour qu’à la création d’un utilisateur :

◦ Le répertoire de connexion soit dans /home/utilisateurs

◦ L’UID minimum soit 3000 et le GID minimum 3000

◦ Le mot de passe ne soit valable que 60 jours

◦ Les répertoires privé, travail et partage soient créés dans le répertoire de connexion Dans un premier temps, vérifier la valeur de la variable HOME contenue dans le fichier /etc/default/useradd.

```
[root]# grep HOME /etc/default/useradd
HOME=/home
```

Sans utiliser un éditeur de texte, affecter la nouvelle valeur de HOME et vérifier.

```
[root]# useradd -D -b /home/utilisateurs
[root]# grep HOME /etc/default/useradd
HOME=/home/utilisateurs
```

• Modifier le fichier /etc/login.defs.

Modifier le fichier /etc/login.defs pour configurer les options d’UID, de GID et de mot de passe :

```
[root]# vim /etc/login.defs
UID_MIN         3000
GID_MIN         3000
PASS_MAX_DAYS       60
```

• Automatiser la création des dossiers privé, travail, partage lors de la création des comptes. Créer les répertoires privés, travail et partage dans le répertoire /etc/skel :

```
[root]# mkdir /etc/skel/{privé,travail,partage}
```

• Rediriger les paramètres par défaut d'ajout d'utilisateurs vers le fichier /STAGE/utilisateurs/defaut. Rediriger les paramètres par défaut de la commande useradd :

```
[root]# useradd –D > /STAGE/utilisateurs/defaut
```

Exercice 6.2 : Message à la connexion

Faire afficher le message suivant à la connexion du compte xavier : "Tout est fichier, sauf le café !"

• Solution 1 :

```
[root]# echo 'echo "Tout est fichier, sauf le café !"' >> /home/linux/xavier/.bashrc
```

• Solution 2 :

```
[root]# vim /home/linux/xavier/.bashrc
```

• Sur le terminal 3, connectez-vous en tant que xavier et vérifier :

```
localhost login: xavier
Password:
Tout est fichier, sauf le café !
[xavier@localhost ~]$
```

Exercice 6.3 : Ajout d'un utilisateur

• Ajouter l'utilisateur tanguy.

```
[root]# useradd tanguy
```

Si vous n'avez pas pensé à créer le répertoire /home/utilisateurs, vous obtiendrez l'erreur suivante :

```
useradd : impossible de créer le répertoire /home/utilisateurs/tanguy
```

Vérifier que les modifications des exercices précédents ont bien été prises en compte.

```
[root]# id tanguy
uid=3000(tanguy) gid=3000(tanguy) groupes=3000(tanguy)
[root]# tail -n 1 /etc/passwd
tanguy:x:3000:3000::/home/utilisateurs/tanguy:/bin/bash
[root]# tail -n 1 /etc/shadow
tanguy:!!:16897:0:60:7:::
```

ATELIER 7 : Ajout et préparation d'un disque

Exercice 7.1 : Ajouter un disque

• Ajouter un disque de 10 Go à votre serveur virtuel.

Exercice 7.2 : Préparer le nouveau disque

Démarrer à nouveau votre machine virtuelle et connectez-vous avec l'utilisateur root.

• Visualiser les disques accessibles :

```
[root]# fdisk -l
```

• Créer 6 partitions : 4 de 2 Go et 2 de 1 Go.

```
[root]# cfdisk /dev/sdc
```

Créer une nouvelle partition :

Choisir l'option [Primaire] :

Donner une taille à la partition :

Répéter l'opération 3 fois :

Faire un premier lecteur logique :

Répéter l'opération 3 fois :

Penser à écrire la nouvelle table de partition :

Exercice 7.3 : Formater les nouvelles partitions

• Formater les nouvelles partitions en ext4.

```
[root]# mkfs -t ext4 /dev/sdc1
[root]# mkfs -t ext4 /dev/sdc2
[root]# mkfs -t ext4 /dev/sdc3
[root]# mkfs -t ext4 /dev/sdc5
[root]# mkfs -t ext4 /dev/sdc6
[root]# mkfs -t ext4 /dev/sdc7

[root]# ls -l /dev/sdc*
```

Exercice 7.4 : Monter les disques dans l'arborescence

Monter les volumes sur /Disque [1-6].

Création des points de montage :

```
[root]# mkdir /Disque1 /Disque2 /Disque3 /Disque4 /Disque5 /Disque6
```

Montage des partitions :

```
[root]# mount /dev/sdc1 /Disque1
[root]# mount /dev/sdc2 /Disque2
[root]# mount /dev/sdc3 /Disque3
[root]# mount /dev/sdc5 /Disque4
[root]# mount /dev/sdc6 /Disque5
[root]# mount /dev/sdc7 /Disque6
```

Visualisation des systèmes de fichiers :

```
[root]# mount
...
/dev/sdc1 on /Disque1 type ext4 (rw)
...
```

• Faire en sorte qu'à chaque redémarrage, le disque ajouté (ses partitions) soit pris en compte.

```
[root]# vim /etc/fstab
/dev/sdc1    /Disque1    ext4    defaults    0  0
...
/dev/sdc7    /Disque6    ext4    defaults    0  0
```

• Rediriger le contenu du fichier de configuration des montages et la table associée dans le fichier « Resultat » que vous placerez sur la première partition du disque ajouté dans un répertoire « SuiviInstallation ».

```
[root]# cd /Disque1
[root]# mkdir SuiviInstallation
[root]# cat /etc/fstab /etc/mtab > SuiviInstallation/Resultat
```

ATELIER 8 : Droits particuliers

Exercice 8.1 : Création d'une boîte aux lettres

• Créer l'arborescence /home/BAL/resultats/.

```
[root]# mkdir -p /home/BAL/resultats
```

• Faire en sorte que les utilisateurs puissent uniquement déposer des fichiers dans « resultats » en ne pouvant que passer dans BAL.

```
[root]# chmod 711 /home/BAL
[root]# cd /home/BAL
[root]# chmod 733 resultats
```

• Lister les droits de « BAL » ainsi que ceux de ses sous répertoires

```
[root]# cd ..
[root]# ls -ld BAL
drwx--x--x 5 root root 1024 fev 11 08:21 BAL

[root]# cd BAL
[root]# ls –l
drwx-wx-wx 5 root root 1024 fev 11 08:21 resultats
```

Exercice 8.2 : Droits d'endossement

La commande « chfn » permet de formater le champ commentaire des comptes utilisateurs. Cette commande écrit dans le fichier « /etc/passwd ». • Visualiser les droits de la commande et ceux du fichier.

```
[root]# ls -l /usr/bin/chfn
-rwx--x--x 1  root  root  16464  oct  16  2007  /usr/bin/chfn

[root]# ls -l /etc/passwd
-rw-r--r-- 1  root  root  1638  fev  5  13:23  /etc/passwd
```

• Faire en sorte que les utilisateurs puissent utiliser cette commande et ainsi changer le commentaire les concernant.

```
[root]# chmod u+s /usr/bin/chfn
```

• Listez à nouveau les droits de la commande et ceux du fichier.

```
[root]# ls -l /usr/bin/chfn
-rws--x--x 1  root  root  16464  oct  16  2007  /usr/bin/chfn

[root]# ls -l /etc/passwd
-rw-r--r-- 1  root  root  1638  fev  5  13:23  /etc/passwd
```

ATELIER 9 : Sécuriser les fichiers

Exercice 9.1 : Evaluer ses droits sur un répertoire

• Créer le répertoire « cours » dans votre répertoire de connexion.

```
[stagiaire]$ cd
[stagiaire]$ mkdir cours
```

• Observer les droits d'accès appliqués à ce répertoire :

```
[stagiaire]$ ls -lisad cours
drwxr-xr-x 5 stagiaire users 4096 fev 11 08:21 cours
```

• Quel utilisateur êtes-vous vis-à-vis de ce répertoire ?

Je suis le propriétaire

• De quels droits disposez-vous sur ce répertoire ? Je dispose des droits de lecture, écriture et de passage dans le répertoire

Exercice 9.2 : Interdire la modification du contenu d'un répertoire

• Modifier les droits du répertoire cours pour qu'ils deviennent dr-xr-xr-x.

```
[stagiaire]$ chmod u-w cours
```

• Créer un fichier linux.txt dans ce répertoire.

```
[stagiaire]$ touch cours/linux.txt
```

• Que se passe-t-il ?

cp: ne peut créer le fichier régulier 'cours/linux.txt' permission non accordée.

• Créer un sous répertoire « windows » dans le répertoire cours.

```
[stagiaire]$ mkdir cours/windows
```

• Que se passe-t-il ?

mkdir : ne peut créer le répertoire 'windows' : permission non accordée. • Rétablir les droits drwxr-xr-x du répertoire cours. Créer un fichier linux.txt dans cours puis créer le répertoire cours/windows.

```
[stagiaire]$ chmod u+w cours
[stagiaire]$ touch cours/linux.txt
[stagiaire]$ mkdir cours/windows
```

• Remettre les droits dr-xr-xr-x sur le répertoire cours. Tenter de détruire le fichier linux.txt et le sous-répertoire wndows.

```
[stagiaire]$ chmod u-w cours
[stagiaire]$ rm –f cours/linux.txt
rm: ne peut enlever 'cours/linux.txt' permission non accordée.

[stagiaire]$ rm -Rf cours/windows
rm: ne peut détruire le répertoire 'cours/windows' : permission non accordée.
```

• Copier le fichier cours/linux.txt dans le répertoire cours/windows en le renommant srv2k8.txt.

```
[stagiaire]$ cp cours/linux.txt cours/windows/srv2k8.txt
```

• Commenter le résultat. Les droits du répertoire cours permettent le passage, et ceux sur windows la copie du fichier.

Exercice 9.3 : Interdire l'accès à un répertoire

• Modifier les droits du répertoire cours pour qu'ils deviennent drw-r-xr-x.

```
[stagiaire]$ chmod 655 cours
```

• Se positionner dans cours

```
[stagiaire]$ cd cours
```

• Que se passe-t-il ?

```
bash : cd : cours : permission non accordée
```

Il manque le droit x qui permet le positionnement.

• Afficher le contenu du fichier cours/linux.txt.

```
[stagiaire]$ less cours/linux.txt
less: cours/linux.txt : permission non accordée
```

• Afficher ou tenter d'accéder au sous répertoire windows.

```
[stagiaire]$ cd cours/windows
```

• Que se passe-t-il ?

```
bash : cd : cours/windows : permission non accordée
```

Exercice 9.4 : Autoriser les droits en lecture seule sur un fichier

• Mettre les droits afin de pouvoir se positionner dans le répertoire cours et donner les droits de lecture seule au fichier « linux.txt ».

```
[stagiaire]$ chmod 755 cours
[stagiaire]$ cd cours
[stagiaire]$ chmod u-w linux.txt
```

• Afficher le contenu de ce fichier.

```
[stagiaire]$ less linux.txt
```

• Tenter de modifier son contenu (à l'aide d'un éditeur de texte ou par la commande « cat >>linux.txt »).

```
[stagiaire]$ vim linux.txt
[stagiaire]$ cat >>linux.txt
```

• Tenter de détruire le fichier « linux.txt ».

```
[stagiaire]$ rm -f linux.txt
```

• Que se passe-t-il ? Pourquoi ?

La destruction du fichier est possible car le répertoire où il est situé possède le droit « w » pour « stagiaire ».

Exercice 9.5 : Droits d'exécution sur un fichier exécutable

• Copier le fichier « /bin/ls » dans votre répertoire de travail. Renommer le fichier en « ls1 »

```
[stagiaire]$ cd
[stagiaire]$ cp /bin/ls ls1
```

• Quel est son type ?

```
[stagiaire]$ file ls1
```

• Exécuter ce fichier par la commande « ./ls1 ».

```
[stagiaire]$ ./ls1
```

• Enlever les droits d'exécution à ce fichier.

```
[stagiaire]$ chmod a-x  ls1
```

• Tenter de le lancer à nouveau « ./ls1 ». Que se passe-t-il ?

```
[stagiaire]$ ./ls1
bash: ./ls1: permission non accordée
```

Exercice 9.6 : Droits d'exécution sur un script

• Créer un script nommé « Prog » dans ce script vous écrirez ceci :

```
#!/bin/bash
clear
echo "essai de script Shell"
```

• Vérifier les droits du fichier « Prog ».

```
[stagiaire]$ ls -l Prog
```

• Afficher le type de ce fichier.

```
[stagiaire]$ file Prog
```

• Modifier ses droits pour qu'il soit exécutable par le propriétaire.

```
[stagiaire]$ chmod u+x Prog
```

Exécuter ce fichier «./Prog » et vérifier le résultat.

```
[stagiaire]$ ./Prog
```

Affichage à l'écran du message « essai de script Shell ».

Bibliographie

[1] Gilles CHAMILLARD « Administration d'un système Linux », Editions ENI, 2012

[2] Sébastien Namèche, « Les bases de l'administration du système Linux », 2004, http://-sebastien.nameche.fr/

[3] cours - « Administration Linux à 200% »

[4] Rob Flickenger, collectif – O'Reilly

[5] Matt Welsh, Matthias Kalle Dalheimer, Terry Dawson et Lar Kaufman, « Le système Linux », Editions O'Reilly [4] Frisch, « Les bases de l'administration système », Editions O'Reilly

[6] coma94 et ShigeruM, « Débuter en informatique avec Windows 7 », www.openclassroom.com

[7] Fabrice Chrzanowski, « Administration de windows server 2012R2 », www.alaphorm.com

[8] Raphaël Roose, « Windows Server 2008 » pour Alka SA