

# APPRENTISSAGE LINUX- LP101

version 1.0 - 14 Janvier 2016





## NAISSANCE DE CE DOCUMENT

Ce document PDF a été rédigé à l'aide du logiciel lyx, inspiré à 99% sur la base du site :  
<http://x3rus.com/moodle/>.

Ce site a été élaboré par Thomas **BOUTRY**, français d'origine établi au Canada où il exerce sa profession. Passionné de logiciel libre, il nous fait partager gratuitement sa passion de Linux à travers son site et ses vidéos postées sur le Net.

Moi même intéressé par la distribution GNU-LINUX, pour apprendre l'essentiel de ce système d'exploitation, il m'a semblé nécessaire d'établir des notes qui ont donné naissance à ce polycopié, la lecture d'un document papier étant pour moi plus facile que la (re)lecture à l'écran.

J'adresse la première mouture de ce document à Thomas BOUTRY en remerciement. Il devrait vraisemblablement le mettre à disposition du plus grand nombre sur son site.

### Remarque :

Je remercie d'avance les lecteurs qui me feraient part des fautes de frappe, de grammaire et d'orthographe afin d'obtenir un document de meilleure qualité ( absence de relecteur ).

Retour d'informations pour correction à adresser à : pc-st-savin@orange.fr en notant la date du document, le numéro de la page et le paragraphe concerné (si possible).









# Première partie

## LINUX LP 101



# Table des matières

<b>I LINUX LP 101</b>	<b>7</b>
<b>1 INTRODUCTION À LINUX</b>	<b>15</b>
1.1 SYSTÈME D'EXPLOITATION . . . . .	15
1.2 UNIX ET LINUX . . . . .	18
<b>2 PRÉSENTATION DE GNU/LINUX</b>	<b>25</b>
2.1 CARACTÉRISTIQUES DE GNU/LINUX . . . . .	25
2.2 INSTALLATION DE GNU/LINUX ATELIER . . . . .	25
2.3 DESCRIPTION DE GNU/LINUX . . . . .	25
2.4 UTILISATEURS . . . . .	26
2.5 INTERFACES . . . . .	27
<b>3 INSTALLATION DE GNU/LINUX</b>	<b>37</b>
3.1 PRÉPARATION . . . . .	37
3.2 RÉALISATION DE L'INSTALLATION . . . . .	38
<b>4 APPRIVOISER LA LIGNE DE COMMANDE</b>	<b>39</b>
4.1 PRÉSENTATION DE LA LIGNE DE COMMANDE . . . . .	39
4.2 ATELIER LIGNE DE COMMANDE . . . . .	41
<b>5 COMMANDES DE BASE</b>	<b>45</b>
5.1 COMMANDES DE BASE . . . . .	45
5.2 ÉVOLUER DANS L'ARBORESCENCE ET MANIPULATION DE FICHIER . . . . .	46
5.3 AFFICHER LE CONTENU D'UN FICHIER . . . . .	51
5.4 LIER DES COMMANDES . . . . .	59
5.5 INTERRUPTION D'UNE COMMANDE . . . . .	60
5.6 VERSION VIDÉO . . . . .	60
5.7 AIDE-MÉMOIRE DES COMMANDES LINUX . . . . .	60

<b>6 ATELIER D'UTILISATION DES COMMANDES DE BASE</b>	<b>61</b>
6.1 UTILISATION DES COMMANDES DE BASE . . . . .	61
<b>7 COMMANDES DE BASE 2</b>	<b>63</b>
7.1 OBJECTIFS . . . . .	63
7.2 ARCHIVAGE ET EXTRACTION DE DONNÉES . . . . .	67
7.3 VERSION VIDÉO . . . . .	70
<b>8 AIDE DEPUIS LA LIGNE DE COMMANDE</b>	<b>73</b>
8.1 AIDE DEPUIS LA LIGNE DE COMMANDE . . . . .	73
8.2 TROUVER UNE COMMANDE AVEC MAN . . . . .	76
8.3 WHATIS - APROPOS . . . . .	76
8.4 INFO . . . . .	79
8.5 HELP . . . . .	80
8.6 RÉSUMÉ . . . . .	80
<b>9 ATELIER UTILISATION COMMANDES</b>	<b>83</b>
9.1 UTILISATION DE LA COMMANDE FIND . . . . .	83
9.2 ARCHIVAGE ET DÉCOMPRESSION . . . . .	83
9.3 MANUEL . . . . .	84
9.4 TRAVAUX PRATIQUES . . . . .	84
<b>10 TRUCS ET ASTUCES AVEC BASH</b>	<b>85</b>
10.1 TRUCS ET ASTUCES AVEC BASH . . . . .	85
10.2 RACCOURCIS CLAVIER . . . . .	85
10.3 CEINTURE JAUNE DE BASH : . . . . .	86
10.4 VERSION VIDÉO . . . . .	87
<b>11 PRÉSENTATION DU SYSTÈME PACKAGES</b>	<b>89</b>
11.1 INSTALLATION DE LOGICIEL . . . . .	89
11.2 DÉFINITION D'UN PACKAGE . . . . .	90
11.3 GESTION DES DÉPENDANCES . . . . .	90
11.4 SCRIPT D'INSTALLATION ET DE DÉSINSTALLATION . . . . .	91
11.5 LES PACKAGES ET INTERNET . . . . .	91
11.6 VERSION VIDÉO . . . . .	92

<b>TABLE DES MATIÈRES</b>	<b>11</b>
<b>12 SYSTÈME DE PACKAGE DEBIAN/UBUNTU</b>	<b>93</b>
12.1 SYSTÈME DE PACKAGE DEBIAN . . . . .	93
12.2 OUTILS DE GESTION DES PACKAGES . . . . .	94
12.3 PRÉSENTATION DE DPKG . . . . .	95
12.4 PRÉSENTATION DES OUTILS APT . . . . .	97
12.5 VERSION VIDÉO . . . . .	104
<b>13 INSTALLATION DE LOGICIEL SOUS DEBIAN</b>	<b>105</b>
13.1 MANIPULATION DE LOGICIEL SOUS UBUNTU . . . . .	105
13.2 SOLUTION . . . . .	106
<b>14 SYSTÈME DE PACKAGE REDHAT</b>	<b>107</b>
14.1 SYSTÈME DE PACKAGE REDHAT . . . . .	107
14.2 OUTILS DE GESTION DES PACKAGES . . . . .	108
14.3 PRÉSENTATION DE RPM . . . . .	108
14.4 PRÉSENTATION DE YUM . . . . .	109
<b>15 PRÉSENTATION DE VIM</b>	<b>111</b>
15.1 CONCEPT DE VIM . . . . .	111
15.2 UTILISATION DE VIM . . . . .	113
15.3 ASTUCES . . . . .	118
15.4 AVOIR L'AIDE DE VIM EN FRANÇAIS . . . . .	119
15.5 SITE DE RÉFÉRENCE VIM . . . . .	119
<b>16 TUTORIAL VIM</b>	<b>121</b>
16.1 VIMTUTOR . . . . .	121
16.2 ATELIER PRATIQUE . . . . .	121
<b>17 REDIRECTION ET FLUX DE DONNÉES</b>	<b>123</b>
17.1 LE SYSTÈME ENTRÉES / SORTIES . . . . .	123
17.2 REDIRECTION DU FLUX STANDARD ET DU FLUX D'ERREUR . . . . .	124
17.3 REDIRECTION DU FLUX D'ENTRÉE (STDIN) . . . . .	131
17.4 TUTORIEL VIDÉO . . . . .	133

<b>18 CHAÎNAGE DE COMMANDES</b>	<b>135</b>
18.1 DESCRIPTION . . . . .	136
18.2 LES COMMANDES FILTRES . . . . .	136
18.3 COMPLÉMENTS . . . . .	150
18.4 TUTORIEL VIDÉO . . . . .	150
<b>19 PRATIQUE DES PIPES ET DES FILTRES</b>	<b>153</b>
19.1 TRAVAIL SIMPLE . . . . .	153
19.2 TRAVAIL INTERMÉDIAIRE . . . . .	153
19.3 TRAVAIL EXPERT . . . . .	154
<b>20 PERMISSIONS</b>	<b>155</b>
20.1 PERMISSIONS ET PROPRIÉTAIRE DES FICHIERS . . . . .	155
20.2 LES PROPRIÉTAIRES . . . . .	156
20.3 LES PERMISSIONS . . . . .	156
20.4 MANIPULATION AVEC LA LIGNE DE COMMANDE . . . . .	157
20.5 VIDÉOS . . . . .	160
<b>21 FICHIERS DE LIENS</b>	<b>161</b>
21.1 DÉFINITIONS . . . . .	161
21.2 VIDEO . . . . .	165
<b>22 PRATIQUE PERMISSIONS ET LIENS</b>	<b>167</b>
22.1 PERMISSIONS . . . . .	167
22.2 LIENS . . . . .	167
22.3 VIDEO . . . . .	169
<b>23 ATELIER GESTION DES PERMISSIONS ET LIENS</b>	<b>171</b>
23.1 QUESTIONS PERMISSIONS . . . . .	171
23.2 QUESTIONS LIENS . . . . .	172
<b>24 PÉRIPHÉRIQUES SOUS GNU/LINUX</b>	<b>173</b>
24.1 RÉPERTOIRE /DEV . . . . .	173
24.2 LISTER LES PÉRIPHÉRIQUES . . . . .	176
24.3 LISTER LES INFORMATIONS MATÉRIELLES DU SYSTÈME . . . . .	183

24.4 LE NOYAU ET SES MODULES . . . . .	187
24.5 MESSAGE DU KERNEL . . . . .	193
24.6 VIDÉOS . . . . .	195
<b>25 PRATIQUE SUR PÉRIPHÉRIQUES N°1</b>	<b>197</b>
25.1 LES PÉRIPHÉRIQUES D'ENTRÉES . . . . .	197
25.2 UTILISATION DES PSEUDOS PÉRIPHÉRIQUES . . . . .	198
25.3 MANIPULATION DE PARTITION (CLEF USB) . . . . .	199
25.4 EXTRA . . . . .	201
25.5 VIDÉOS . . . . .	201
25.6 COMPLÉMENTS . . . . .	202
25.7 VIDÉO . . . . .	208
<b>26 PRATIQUE SUR PÉRIPHÉRIQUES N°2</b>	<b>209</b>
26.1 CRÉATION D'UN FICHIER SIMULANT UNE PARTITION . . . . .	209
<b>27 RÉVISION REDIRECTION ET FILTRES</b>	<b>211</b>
27.1 THÉORIE . . . . .	211
27.2 DÉMONSTRATION . . . . .	211
27.3 EXERCICES . . . . .	212
27.4 VIDÉOS . . . . .	213
27.5 SOLUTIONS . . . . .	213
27.6 WGET . . . . .	214
<b>28 SCRIPTING AVEC BASH</b>	<b>217</b>
28.1 INTRODUCTION AU SCRIPTING BASH . . . . .	217
28.2 RÉALISATION DE NOTRE PREMIER SCRIPT . . . . .	218
28.3 LES VARIABLES . . . . .	221
28.4 LES STRUCTURES DE CONTRÔLE . . . . .	225
28.5 DEBUG . . . . .	230
<b>29 GESTION DE PÉRIPHÉRIQUES - suite</b>	<b>231</b>
29.1 SYSTÈME UDEV . . . . .	231
29.2 CONCEPT . . . . .	231
29.3 PRÉSENTATION . . . . .	233
29.4 LES RÈGLES (RULES) UDEV . . . . .	236
29.5 VIDÉOS . . . . .	240



# Chapitre 1

## INTRODUCTION À LINUX

### Introduction

Objectif du chapitre :

- Expliquer le rôle d'un système d'exploitation,
- Expliquer les principales couches de l'architecture d'un système d'exploitation,
- Expliquer l'origine de GNU/Linux
- Expliquer les différentes versions de UNIX et GNU/Linux.

### 1.1 SYSTÈME D'EXPLOITATION

Un système informatique moderne est composé d'un ou plusieurs processeurs, d'une mémoire principale, de disques durs, d'imprimantes, d'un clavier, d'une souris, d'un écran, d'une carte réseau et de beaucoup d'autres périphériques d'entrée/sortie. En un mot un système complexe. Dans ce contexte, développer des programmes d'application qui doivent tenir compte correctement de toutes ces entrées/sorties n'est pas une mince tâche. C'est pour cette raison que les ordinateurs modernes sont équipés d'un « système d'exploitation ». Une des tâches du système d'exploitation est donc d'offrir aux utilisateurs une interface simple et conviviale avec le matériel.

#### Qu'est-ce qu'un système d'exploitation ?

Un système d'exploitation effectue deux tâches bien distinctes :

- Présenter une machine virtuelle à l'utilisateur.
- Gérer les ressources de l'ordinateur.

**Machine virtuelle** Être une machine virtuelle signifie transformer un ensemble de circuits électroniques en un outil moderne qui offre une abstraction simple au niveau des entrées/sorties, de l'utilisation de la mémoire, de la gestion des fichiers, de la protection et du contrôle des erreurs, de l'interaction des programmes entre eux et de leur contrôle.

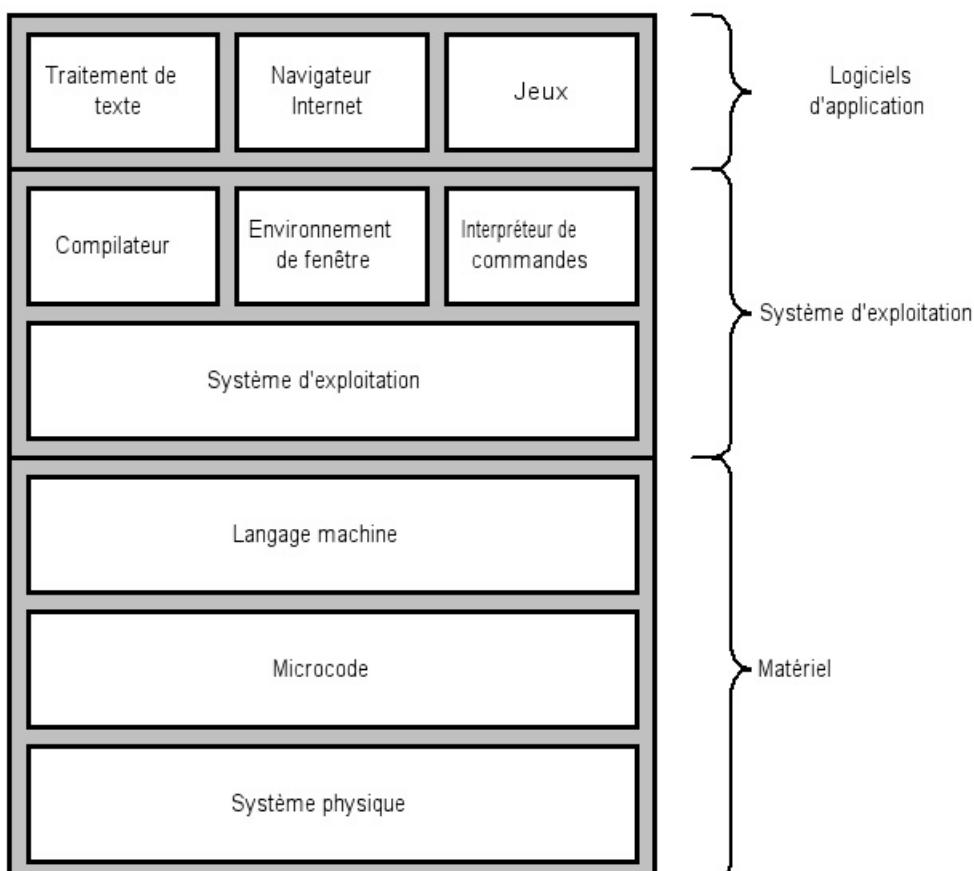
En deux mots : éviter au programmeur d'avoir à connaître les détails électroniques de tel ou tel microprocesseur et permettre à l'utilisateur de sauvegarder ses fichiers sans se soucier du type de disque utilisé pour stocker les informations.

**Gestionnaire de ressources** L'autre fonction du système d'exploitation est le partage des ressources. Le système d'exploitation joue un rôle de policier afin d'éviter les conflits d'utilisation de la mémoire, des périphériques d'entrée/sortie, des interfaces réseau, ... On peut facilement imaginer ce qui arriverait si trois programmes essayaient d'imprimer en même temps sans que des priorités aient été préalablement établies.

De plus, lorsque l'ordinateur est utilisé par plusieurs usagers, le partage de la mémoire et surtout sa protection demeurent une priorité absolue. À tout moment, un bon système d'exploitation connaît l'utilisateur d'une ressource, ses droits d'accès et son niveau de priorité.

**Architecture d'un système informatique** Tous les systèmes informatiques sont segmentés en « couches » pour permettre un meilleur contrôle de l'ensemble de l'ordinateur. La Figure 1 illustre les différentes couches d'un tel système.

**Matériel** Au plus bas niveau, on retrouve la première couche qui contient les composantes physiques constituées des circuits intégrés, des fils, des sources de courant, ...



La couche suivante regroupe des outils logiciels primitifs qui permettent de contrôler directement les composantes physiques sous-jacentes, comme les registres internes du processeur et l'unité arithmétique et logique. Cette couche est appelée le microcode et réside bien souvent dans le processeur de l'ordinateur.

L'autre couche est celle du langage machine qui est interprété par le microcode. Ce langage de bas niveau regroupe 50 à 300 instructions pour permettre de déplacer des bits, de calculer ou de comparer des valeurs à l'aide des registres internes du processeur.

**Système d'exploitation** Le système d'exploitation qui se trouve juste au-dessus, offre aux programmeurs et aux utilisateurs un ensemble de fonctions du genre « lire le fichier » ou « afficher à l'écran ». Il s'agit ici d'un niveau d'abstraction élevé qui évite ainsi au programmeur de devoir écrire, par exemple, du code pour déplacer les têtes de lecture d'un disque rigide. Il s'agit du niveau d'exécution des pilotes de périphériques (contrôleurs d'interruptions, de disques, de cartes graphiques, ...).

En haut de la hiérarchie, il y a la couche où l'on retrouve les interpréteurs de commandes, les compilateurs et les logiciels d'applications. Il est clair que ces programmes ne font pas partie du système d'exploitation, même s'ils sont livrés avec celui-ci dans bien des cas.

**Logiciels d'applications** Finalement, au-dessus de toutes ces couches se trouvent les logiciels d'applications qui permettent à un utilisateur d'effectuer des tâches particulières sans qu'il ait à tenir compte des couches inférieures.

**Fonctions d'un système d'exploitation** Aujourd'hui, l'informatique, aussi bien dans les entreprises que dans l'enseignement, utilise des machines plus petites fonctionnant avec des systèmes d'exploitation à caractère universel. Parmi ces systèmes d'exploitation, deux se distinguent particulièrement, un système Windows, et un autre multi-utilisateurs et multi-tâches, UNIX (Linux). D'une manière contestable, on peut affirmer que le premier système est destiné à des ordinateurs individuels, tandis que l'autre est réservé au travail en groupe.

Parmi les nombreux systèmes d'exploitation, UNIX/Linux est celui qui offre le plus de richesse, le plus d'homogénéité et le plus de souplesse. Pour cette raison, dans ce livre, Linux a été choisi comme système d'exploitation pour illustrer les concepts théoriques. Par ailleurs, le système MS-DOS puis Windows, en évoluant, a incorporé beaucoup de caractéristiques de UNIX/Linux.

On peut diviser les fonctions des systèmes d'exploitation classiques en quatre parties principales :

- La gestion des processus (programmes).
- La gestion de la mémoire.
- Le système de fichiers.
- La gestion des entrées/sorties.

Les systèmes d'exploitation modernes intègrent par ailleurs d'autres caractéristiques. Ces dernières concernent notamment deux évolutions majeures des systèmes informatiques. La première est l'interconnexion des différentes machines et des différents systèmes par des réseaux locaux ou étendus. La seconde est la disparition des écrans de texte et leur remplacement par des dispositifs à fenêtres multiples disposant de propriétés graphiques.

## 1.2 UNIX ET LINUX

UNIX est un système d'exploitation très populaire parce qu'il est présent sur un grand nombre de plates-formes, du micro-ordinateur à l'ordinateur central (mainframe). L'avantage de cela, c'est que les programmes développés sous UNIX peuvent être transférés d'une plate-forme à une autre avec un minimum de modifications.

Ce système est multitâche, c'est-à-dire qu'il est capable de gérer et d'exécuter plusieurs programmes simultanément. De plus, il est multi-utilisateurs, c'est-à-dire que plusieurs personnes peuvent s'y connecter en même temps et travailler ; le système partage alors toutes les ressources logicielles et matérielles de l'ordinateur entre les différents usagers.

L'histoire d'Unix est unique dans le monde des systèmes d'exploitation. En effet, alors que la plupart des systèmes d'exploitation ont été conçus par des fabricants d'ordinateurs pour vendre leurs machines, UNIX n'a pas été conçu dans un but commercial. Il l'est devenu parce qu'il constitue une norme en matière de système d'exploitation.

Contrairement à un système d'exploitation commercial complètement contrôlé par son fabricant, le système UNIX est aujourd'hui distribué par plusieurs intervenants dont voici les principaux :

- AT&T, à qui on attribue la paternité de UNIX ;
- L'université de Berkeley, qui a fait évoluer UNIX dans plusieurs domaines ;
- SUN Microsystems, à qui l'on doit les améliorations importantes de l'interface graphique ;
- Santa Cruz Operation et Microsoft, le XENIX/UNIX fut la première version pour PC de UNIX.

À cause de cette situation de développement, le système d'exploitation UNIX a mis plusieurs années à être standardisé. Actuellement, il en existe deux principales variantes, incompatibles entre elles :

- Le UNIX SYSTEM V,
- Le UNIX BSD.

Il existe aussi une multitude de variations mineures dérivées d'une des deux ou des deux principales variantes ; on a ainsi, en les regroupant :

- Les systèmes Unix-Based ;
- Les systèmes Unix-Like.

### Systèmes Unix-Based

Les systèmes Unix-Based ont obtenu une licence d'utilisation d'AT&T. Il s'agit d'une adaptation de UNIX. Ils restent compatibles avec la version d'AT&T parce qu'ils partagent le même noyau. On retrouve dans cette catégorie :

- XENIX/UNIX provenant de SCOMicrosoft ;
- AIX provenant d'IBM ;
- Mac OsX provenant d'APPLE ;
- SunOS/Solaris provenant de SUN MicroSystems ;
- IRIX provenant de Silicon Graphics ;
- ULTRIX provenant de DIGITAL.

## Systèmes Unix-Like

Les systèmes Unix-Like reproduisent les mêmes fonctionnalités que la version AT&T, mais le noyau du système est incompatible parce qu'il a été réécrit pour éviter le versement de droit d'auteurs à AT&T. On retrouve dans cette catégorie :

- Minix ;
- Linux ;
- FreeBSD ;
- QNX.

Cette prolifération de produits a fait apparaître un certain nombre de différences entre les systèmes, dont les principales sont :

- Les communications inter-programmes ;
- La gestion de la mémoire (segmentation ou pagination) ;
- Divers paramètres du système ;
- Divers outils qui peuvent être intégrés dans un produit et absents dans un autre.

## Naissance du noyau Linux

En 1991, les compatibles PC dominent le marché des ordinateurs personnels et fonctionnent généralement avec les systèmes d'exploitation MS-DOS, Windows ou OS/2. Les PC basés sur le microprocesseur Intel 80386, vendus depuis 1986, commencent à être abordables. Mais les systèmes grand public restent attachés à la compatibilité avec les anciens processeurs 16 bits d'Intel et exploitent mal les capacités 32 bits et l'unité de gestion mémoire du 80386.



C'est cette année que l'étudiant finlandais Linus Torvalds, indisposé par la faible disponibilité du serveur informatique UNIX de l'université d'Helsinki, entreprend le développement d'un noyau de système d'exploitation, qu'on appellera plus tard le « noyau Linux ». Linus désire alors surtout comprendre le fonctionnement de son ordinateur fondé sur un Intel 80386.

Linus Torvalds fait son apprentissage avec le système d'exploitation Minix. Comme le concepteur de Minix — Andrew Tanenbaum — refuse d'intégrer les contributions visant à améliorer Minix, Linus décide de programmer un remplaçant de Minix. Il commence par développer un simple émulateur de terminal, qu'il utilise pour se connecter via un modem au serveur informatique de son université. Après l'ajout de diverses fonctionnalités dont un système de fichiers compatible avec celui de Minix, Linus oriente son projet vers quelque chose de plus ambitieux : un noyau aux normes POSIX. À

ce noyau, il adapte de nombreux composants disponibles du système d'exploitation GNU pour obtenir un système d'exploitation plus complet.

Le 26 août 1991, il annonce sur le forum Usenet news :comp.os.minix qu'il écrit un système d'exploitation, mais en tant que « hobby », qui ne sera pas grand et professionnel comme gnu'. Le 5 octobre 1991, il annonce la disponibilité d'une ébauche de la version 0.02 de son noyau, la version 0.01 ayant eu une diffusion plus que confidentielle. Enfin en février 1992, la version 0.12 est diffusée sous la Licence publique générale GNU (GNU GPL) à la place de la licence ad hoc qui interdisait jusque-là la redistribution commerciale.

## Le Projet GNU

Le 27 septembre 1983, Richard Stallman dévoile dans la pure tradition hacker son projet de développer un système d'exploitation compatible UNIX appelé GNU, en invitant la communauté hacker à le rejoindre et participer à son développement. Cette annonce succède à la « guerre » déclarée par Symbolics au laboratoire d'intelligence artificielle du MIT et à la disparition de la communauté hacker Lisp. Il annonce que le système pourra être utilisé et partagé librement par tous comme ce fut le cas avec Emacs.



Concrètement il relate l'effort à accomplir, dont on distingue déjà en 1985 certaines pièces maîtresses : le compilateur GCC finalisé dès juin 1984, une version emacs compatible UNIX, etc. L'effort sera opiniâtrement poursuivi, et au début des années 90, le projet GNU possède une version utilisable de tous les éléments nécessaires à la construction d'un système d'exploitation (outre ceux cités précédemment : un shell, des bibliothèques, les composants de base, les outils de développement...) à l'exception du plus central : le noyau.

## Principales distributions de Linux

Il existe une très grande variété de distributions, ayant chacune des objectifs et une philosophie particulière. Les éléments différenciants principalement les distributions sont : la convivialité (facilité de mise en œuvre), l'intégration (taille du parc de logiciels validés distribués), la notoriété (communauté informative pour résoudre les problèmes), l'environnement de bureau (GNOME, KDE, ...), le type de paquet utilisé pour distribuer un logiciel (principalement DEB et RPM) et le mainteneur de la distribution (généralement une entreprise ou une communauté). Le point commun est le noyau (kernel) et un certain nombre de commandes.

Les distributions rassemblent les composants d'un système dans un ensemble cohérent et stable dont l'installation, l'utilisation et la maintenance sont facilitées. Elles comprennent donc le plus

souvent un logiciel d'installation et des outils de configuration. Il existe de nombreuses distributions, chacune ayant ses particularités. Certaines sont dédiées à un usage spécifique (pare-feu, routeur, grappe de calcul, édition multimédia...), d'autres à un matériel spécifique.

Les distributions généralistes destinées au grand public pour un usage en poste de travail (bureautique) sont les plus connues (Debian, Gentoo, Mandriva Linux, RedHat/Fedora, Slackware, SUSE Linux Enterprise/openSUSE, Ubuntu). Le mainteneur de la distribution peut être une entreprise (comme dans le cas de RedHat et Ubuntu, Canonical) ou une communauté (comme Debian, Gentoo ou Slackware).

Leurs orientations particulières permettent des choix selon les besoins et les préférences de l'utilisateur. Certaines sont plus orientées vers les utilisateurs débutants (Mageïa, Ubuntu, etc.), car plus simples à mettre en œuvre. Debian, en revanche, reste prisée pour les serveurs ou plutôt considérée comme une méta-distribution, c'est-à-dire pour servir de base à une nouvelle distribution. Diverses distributions en dérivent, comme Ubuntu, Knoppix, MEPIS... L'installation de Debian est devenue plus facile depuis la version 3.1 (Sarge), néanmoins des compétences en shell et une culture des projets libres restent nécessaires pour obtenir le GNU/Linux de ses rêves ; en revanche la mise à jour et la maintenance du système sont très aisées grâce aux outils Debian. La distribution Gentoo, destinée à des utilisateurs plus connaisseurs, à la recherche de mises à jour fréquentes, a pour particularité d'être compilée depuis le code source sur le poste même de l'usager, en tenant compte des nombreux paramètres locaux. Ceci en fait le système d'exploitation le plus optimisé pour chaque configuration individuelle. Certaines distributions sont commerciales, comme celles de RedHat, Novell/SUSE, alors que d'autres sont l'œuvre d'une fondation à but non lucratif comme Gentoo et Debian.

## Historique

### GNU/Linux

La première distribution apparue en 1992 était assemblée sur quelques dizaines de disquettes. En raison de la très forte croissance de GNU/Linux, une distribution actuelle peut occuper de quelques méga-octets (pour être installée sur une clé USB par exemple) à plusieurs giga-octets.

Avant l'existence des distributions, les utilisateurs de GNU/Linux devaient composer eux-mêmes leur système en réunissant tous les éléments nécessaires.

En 1992, Linux (version 0.96) est pleinement fonctionnel. C'est la naissance des premières distributions GNU/Linux : Yggdrasil Linux, MCC Interim Linux, TAMU. Au milieu de l'année, Softlanding Linux System (SLS) est créée : Slackware, la plus ancienne distribution encore en activité aujourd'hui, est dérivée de cette distribution. Elle est aussi la première, longtemps avant les autres, à permettre un usage direct depuis le CD-ROM, sans installation préalable.

Vous pouvez voir la multiplication des distributions avec cette image :

[http://upload.wikimedia.org/wikipedia/commons/1/1b/Linux\\_Distribution\\_Timeline.svg](http://upload.wikimedia.org/wikipedia/commons/1/1b/Linux_Distribution_Timeline.svg)

Voici une liste non exhaustive des distributions disponibles :

## Distribution grand public



Fedora est une distribution communautaire supervisée par RedHat. Elle est basée sur le système de gestion de paquetages logiciels RPM.



Mageia est une distribution GNU/Linux éditée par une association à but non lucratif : Mageia.Org. C'est une distribution dérivée de Mandriva Linux qui s'adresse à tous, utilisateurs débutants comme avertis. De création récente (2011) et offrant la plupart des environnements graphiques elle s'est rapidement montrée apte à toucher un large public. <http://distrowatch.com/dwres>. Caractéristiques, comparatifs et actualités des distributions.



SUSE a été créée en 1993 à Nuremberg en Allemagne, elle a été rachetée par la société Novell à la fin de l'année 2003. Elle propose deux distributions principales : SUSE Linux Enterprise orientée vers les entreprises (certifications matérielles et logicielles nombreuses) et openSUSE orientée vers le grand public.



Ubuntu, basée sur Debian. Distribution commerciale orientée vers le grand public et constituée par Canonical, édite des versions stables tous les six mois. Elle est disponible en live CD. Cette distribution dispose d'une communauté d'utilisateurs dans le monde entier très dynamique.

## Distribution large



ArchLinux est une distribution communautaire sans version : elle est en mise à jour permanente. Elle dispose toujours des dernières versions des logiciels disponibles, grâce à une communauté de développeurs très active. Cette distribution ultra-légère a été inspirée par Crux Linux, suivant le principe KISS de simplicité technique. Son absence d'outils spécifiques (excepté son gestionnaire de paquets, pacman), en fait une distribution adaptée à la découverte de l'administration des systèmes GNU/Linux.



Debian est une distribution non commerciale régie par le contrat social Debian. Elle se distingue par le très grand nombre d'architectures supportées, son importante logithèque et par son cycle de développement relativement long, gage d'une certaine stabilité. Sa qualité et son sérieux sont unanimement reconnus, mais elle garde l'image d'une distribution réservée aux experts, alors que son ergonomie a bien évolué. C'est la distribution qui a été retenue pour le déploiement de GNU/Linux dans la ville de Munich (14 000 PC). Elle est très utilisée sur les serveurs.



Gentoo est une distribution caractérisée par sa gestion des paquetages à la manière des ports BSD, effectuant généralement la compilation des logiciels (X, OpenOffice, etc.) sur l'appareil de l'utilisateur. Elle est dédiée aux utilisateurs avancés, aux développeurs et aux passionnés. La compilation des logiciels in situ donne une liberté de choix de fonctionnalités et de dépendances très poussée, apportant davantage de souplesse dans la gestion des paquets que dans une distribution utilisant des paquets binaires. Enfin, Gentoo se fait forte d'une communauté d'utilisateurs particulièrement active et d'une documentation complète et centralisée.



RedHat (officiellement RedHat Enterprise Linux ou RHEL) est une distribution commerciale largement répandue dans les entreprises (surtout aux États-Unis). La société RedHat qui la supervise a développé RPM, un gestionnaire de paquets sous licence GPL que d'autres distributions utilisent.



Slackware est l'une des plus anciennes distributions existantes. Slackware a été historiquement une des premières permettant de faire tourner GNU/Linux in situ depuis un CD-ROM, dès 1995. Slackware est toujours activement maintenue par son créateur Patrick Volkerding. Slackware Linux est particulièrement adaptée aux utilisations de type serveur, on peut la considérer comme la distribution la plus pure de GNU/Linux.

Le site distrowatch permet d'avoir plus d'informations sur les distributions, incluant celles moins populaires.



# Chapitre 2

## PRÉSENTATION DE GNU/LINUX

Objectif du chapitre :

- Présenter les caractéristiques de GNU/Linux,
- Mode d’interaction (texte, Graphique)
- Structure du système de fichier
- Concept des partitions et des accès au périphérique mobile
- Démonstration de GNU/Linux

### 2.1 CARACTÉRISTIQUES DE GNU/LINUX

Présentation générale du système d’exploitation

Nous verrons ce qui distingue GNU/Linux, la structure des fichiers et des répertoires, afin d’être en mesure de comprendre le système dans son ensemble quand on parcourt les fichiers. De plus, nous verrons comment s’intègrent les partitions du système ainsi que les équipements mobiles.

Ce chapitre est très important car il permet de bâtir de bonnes bases en vue de la compréhension de GNU/Linux.

### 2.2 INSTALLATION DE GNU/LINUX ATELIER

Nous allons réaliser l’installation de GNU/Linux, plus particulièrement la distribution Ubuntu de canonical. De nos jours, l’installation est très simple, elle a beaucoup évolué avec les années. De plus, la détection du matériel est meilleure, donc plus agréable.

### 2.3 DESCRIPTION DE GNU/LINUX

Tel que mentionné dans la section UNIX et Linux, GNU/Linux est un système d’exploitation proche du système UNIX. GNU/Linux a l’avantage d’être disponible sur un grand nombre d’architectures,

que l'on parle d'Intel / AMD , Sparc, PowerPC, Alpha, ARM, ainsi que des processeurs dédiés « à l'embarqué » et bien d'autres ...

Linux est le noyau du système d'exploitation qui fait l'interaction entre le matériel et le logiciel. La partie GNU est l'ensemble des logiciels qui permet à l'utilisateur d'utiliser le système. Le couple GNU/Linux forme le système d'exploitation. Autour d'eux se regroupent de nombreux programmes.

GNU/Linux est :

- multi-platformes
- multi-utilisateurs
- multi-tâches
- multi-processeurs

Le système d'exploitation est orienté réseau. L'origine de UNIX et donc de GNU/Linux provient du monde des MainFrames. À cette époque, le système d'exploitation était sur un ordinateur central et les utilisateurs établissaient une connexion à ce dernier avec des terminaux. Cette logique fut conservée bien que l'ensemble des communications soient maintenant locales sur une seule machine ; ceci offre une grande flexibilité.

Le système GNU/Linux a l'avantage d'être "**scalable**". En d'autres mots, il peut être exécuté aussi bien sur un système avec peu de ressources que sur une machine surpuissante. Bien entendu, si nous exécutons GNU/Linux sur un Pentium 100Mhz, il ne faut pas espérer jouer aux derniers jeux sortis. Cependant, il sera possible d'utiliser le système dans la limite de ses ressources. Le Noyau Linux est très modulaire. Ceci offre la possibilité de réduire l'empreinte mémoire et les fonctionnalités qu'il utilise.

**La sécurité de GNU/Linux est réputée** ; ce n'est pas simplement parce que c'est GNU/Linux et que les programmeurs sont meilleurs que les autres ... L'architecture du système aide à la sécurité, cependant ce n'est pas la principale raison. Si nous regardons les « bugs » et les failles de sécurités qui sont publiés, il y en a autant que sur les autres systèmes. La particularité est que le code source est disponible, par le fait même un grand nombre de personnes relisent le code et le décortiquent ; si un problème est trouvé, il est signalé. Ceci offre un panel énorme de personnes qui réalisent du "code review" et même parfois proposent le correctif.

GNU/Linux utilise la segmentation des processus par utilisateur. Ceci permet d'augmenter la sécurité du système ; chaque utilisateur ne pouvant accéder qu'aux données auxquelles il a droit.

## 2.4 UTILISATEURS

Chaque utilisateur est identifié par un **nom d'utilisateur (login)** et un **mot de passe (password)**.

Le **login** est associé à un numéro unique (le **UID : User IDentifiant**) et permet d'identifier chaque utilisateur. Il ne contient ni espace, ni caractères spéciaux.

Le **mot de passe** doit être choisi judicieusement : il doit inclure des caractères en minuscule et majuscule, des chiffres et des caractères spéciaux.

Chaque utilisateur dispose d'un répertoire de travail (le Home Directory) dans lequel il a le droit de créer ses propres fichiers et répertoires. Ce répertoire de travail se situe généralement dans le répertoire **/home** et porte le nom du login. Exemple : le répertoire de travail de l'utilisateur alex est **/home/alex**.

Il existe un utilisateur particulier : le **root**. Le **root** est l'administrateur du système, il dispose de tous les droits et s'occupe de la gestion du système : ajout et suppression des utilisateurs, installation et configuration du système et des logiciels ...

Le root a tous les pouvoirs sur le système, il peut tout faire, y compris tout casser. De ce fait, il faut donc choisir un mot de passe très sécurisé, et toujours vérifier deux fois avant d'exécuter une opération avec l'utilisateur root.

Le root n'est pas un utilisateur à proprement parler, il s'agit d'une fonction. Ainsi, il ne faut jamais travailler en permanence avec le compte root, mais utiliser son compte habituel, et ne passer root (via les commandes **su** ou **sudo**) que si l'on a besoin de réaliser une opération d'administration.

Contrairement aux autres utilisateurs, le répertoire de travail du **root** se situe à la racine du système (**/**). L'explication est simple : en cas de problème avec la partition **/home**, l'utilisateur root pourra quand même accéder à son répertoire de travail.

## 2.5 INTERFACES

### LA LIGNE DE COMMANDE

De part la filiation avec UNIX, la ligne de commande (ou shell UNIX) est toujours disponible dans Linux, quelle que soit la distribution.

---

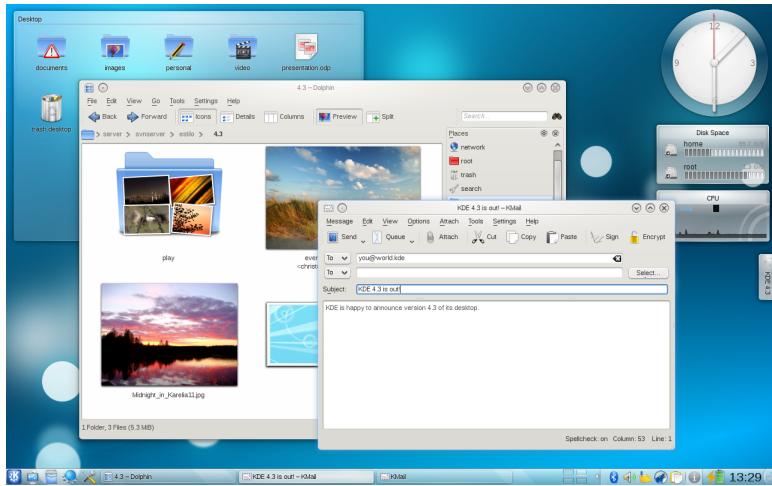
```
pat@amd-a10:~$ ls -l
total 524
drwxr-xr-x 3 pat pat 4096 nov. 19 18:49 Bureau
-rw-r--r-- 1 pat pat 7822 nov. 1 09:55 certification.lyx
-rw-r--r-- 1 pat pat 4784 nov. 1 09:46 certification.lyx~
-rw-r--r-- 1 pat pat 110612 nov. 1 17:16 config materiel.lyx~
-rw-r--r-- 1 pat pat 3911 nov. 1 09:38 dma.lyx~
drwxr-xr-x 3 pat pat 4096 nov. 19 19:45 Documents
drwxr-xr-x 3 pat pat 4096 nov. 20 09:05 images
drwxr-xr-x 2 pat pat 4096 oct. 12 16:48 Images
```

Certaines distributions, notamment celles spécialisées dans les serveurs ou certaines tâches d'administration, utilisent uniquement la ligne de commande, en particulier pour sa faible consommation de ressources, due à l'absence d'interface graphique, mais surtout pour sa puissance d'action, liée à l'interopérabilité des commandes et la possibilité de générer des scripts.

Longtemps, de nombreuses opérations de configuration nécessitaient son utilisation, ce qui n'est plus vrai avec les distributions récentes dédiées à l'utilisation familiale. Néanmoins, les aides en ligne mentionnent souvent la démarche à suivre en ligne de commande, même lorsqu'une configuration graphique est possible : cette méthode est plus universelle dans le monde Linux, et souvent plus facile à expliquer pour la personne qui aide, et son interlocuteur n'a qu'à « copier-coller » l'indication.

## GESTIONNAIRES X WINDOW

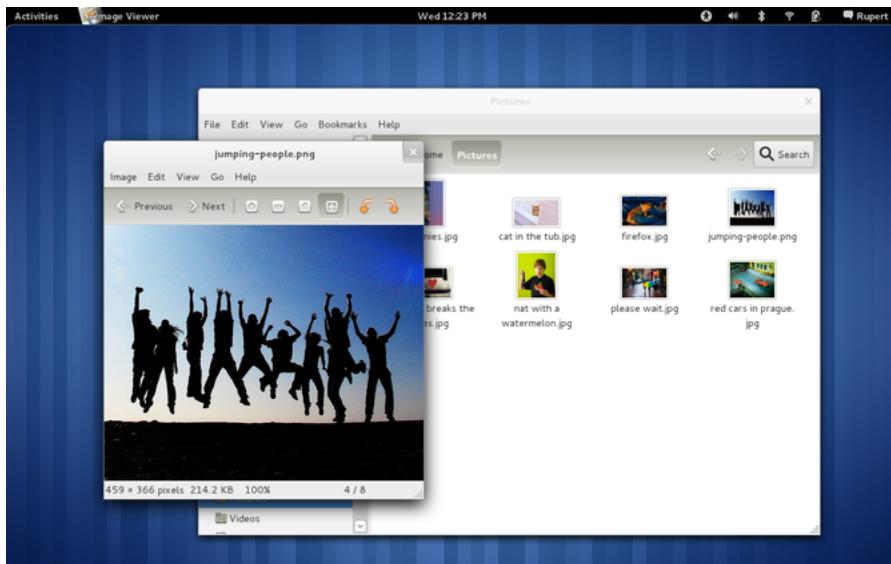
L'emploi du terme générique Linux est trompeur, s'agissant de l'utilisation d'un ordinateur personnel. En effet, il existe plusieurs interfaces (ou gestionnaire de fenêtres) aux caractéristiques différentes, comme KDE, GNOME ou Xfce



Cependant, comme toutes ces interfaces sont fondées sur X Window, leurs applications peuvent cohabiter et elles offrent des points communs, dont l'affichage de fenêtres à distance (y compris via des protocoles compressés et chiffrés comme **ssh** et **nox**) et le « copier-coller » simplifié : un texte sélectionné par la souris est automatiquement copié, un clic milieu (ou un clic molette, ou sur les deux boutons en même temps) suffit alors pour coller le texte à l'endroit désiré. Il n'y a donc jamais besoin du clavier pour effectuer un copier-coller sous X.

Deux environnements de bureau ont atteint récemment une maturité certaine ; citons l'année 2003 pour KDE, un peu plus tard pour GNOME. Très actifs, ces deux projets ont néanmoins l'intention de s'améliorer nettement pour leurs prochaines versions majeures. Les efforts dans ce sens sont concentrés au sein des projets Appeal pour KDE, et ToPaZ pour GNOME.

Techniquement, ils reposent tous deux sur de nombreuses technologies communes, au premier rang desquelles le système de fenêtrage X11.



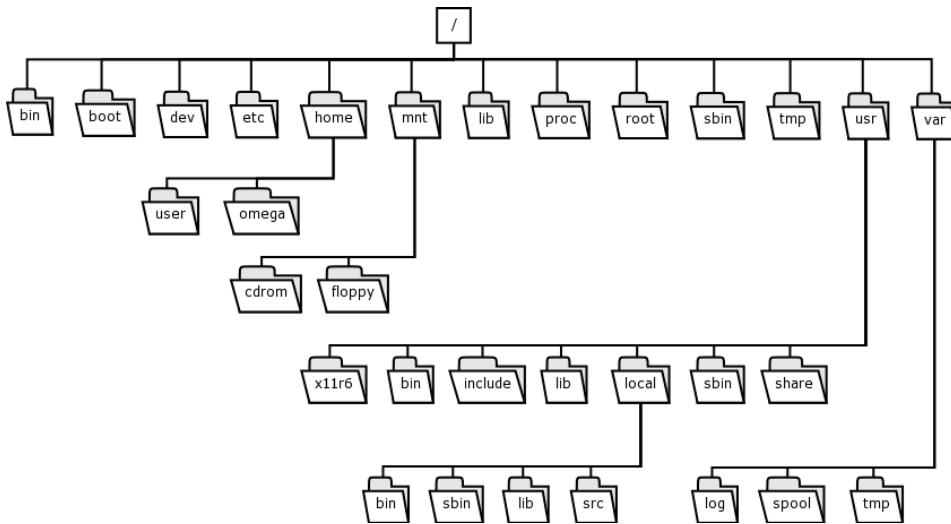
Pour éviter de dupliquer certains efforts, une zone informelle de collaboration entre ces projets du nom de Freedesktop.org a été mis en place.

On peut noter également la montée en puissance d'un troisième environnement de bureau appelé Xfce, qui vise à fournir un environnement complet fondé sur GTK+ comme GNOME, tout en restant plus léger que ce dernier ou KDE.

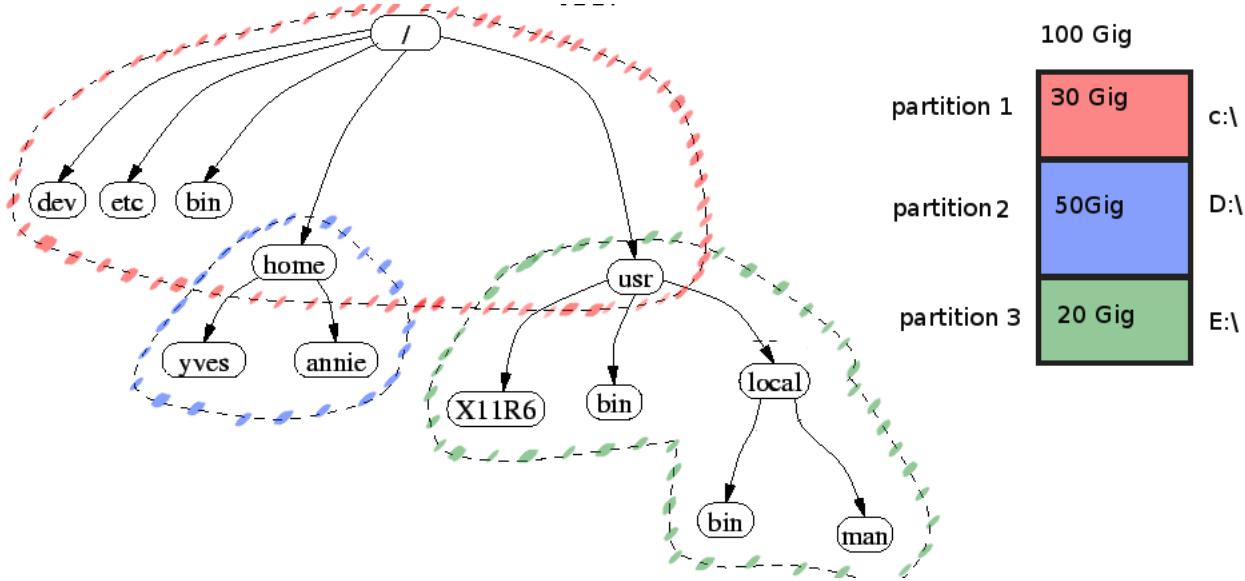
## STRUCTURE DU SYSTÈME DE FICHIERS

Comparez le stockage des répertoires et fichiers de votre ordinateur à un arbre. En partant de la racine d'un arbre, en déplaçant votre doigt tout le long de l'arbre, en suivant le tronc puis les branches, vous pouvez toucher n'importe quelle feuille de cet arbre.

Dans les systèmes de type GNU/Linux, toute l'information stockée dans vos médias de stockage (disques durs, clé USB, cartes SD, CD-ROM, etc.) est nécessairement accessible en suivant un chemin depuis un emplacement logique appelée la racine / [*notée par une barre oblique*]). La racine est une partition que vous définissez comme étant la base du stockage de vos fichiers. Puis, cette base se sépare logiquement en répertoires (*comme des branches d'un arbre*), eux-mêmes séparés en sous-répertoires et sous-sous-répertoires, et ainsi de suite, dans lesquels sont enregistrés vos fichiers (*les feuilles de l'arbre*).



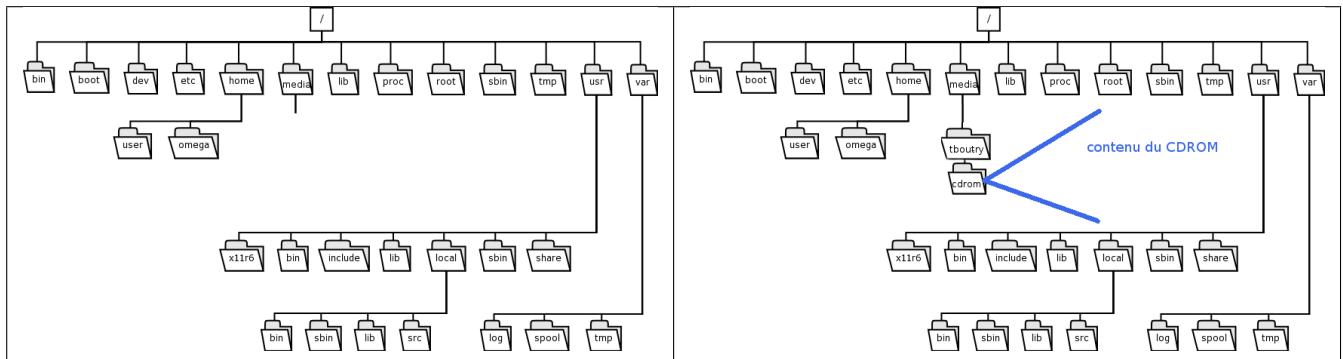
L'ensemble des partitions va se greffer dans l'arborescence (le terme consacré est « monter » « mount en anglais ». Contrairement à d'autres systèmes d'exploitation qui nomment chaque partition avec une lettre, sous UNIX et GNU/Linux, la structure du système de fichiers est toujours le même. Un logiciel ou un fichier installé dans un répertoire, par exemple "/usr/bin/Votre\_Logiciel", sera toujours accessible par le même chemin d'accès, qu'il soit installé sur la première partition ou la troisième. Ce pourrait même être sur un autre disque dur. Voici une représentation graphique de mon propos :



## INCORPORATION DES MÉDIAS MOBILES

Les supports amovibles se "montent" dans des répertoires, en général sous le répertoire /media. Lorsqu'on insère un support, un répertoire au nom variable (souvent dérivé du nom du modèle) y est créé automatiquement et le contenu du support y est monté. Ce répertoire peut mettre quelques secondes à apparaître, il faut afficher répétitivement le contenu du répertoire jusqu'à ce qu'il apparaisse.

Quand je parle de support amovible, ceci inclut les CD-ROM, clef USB, etc... Si nous reprenons la représentation graphique après l'insertion d'un CD , nous aurons le résultat suivant :



Aujourd'hui le système "monte" automatiquement les périphériques lorsqu'ils sont détectés, peu importe le type, s'il est en mesure de l'interpréter, le système vous offrira l'accès. Linux est en mesure de lire un grand nombre de partitions, au moins en lecture. Parfois l'écriture est jugée "expérimental", mais reste disponible. Ceci est le cas pour les partitions de type Mac.

Une fois que l'on a terminé, avant de retirer le support, il faut le démonter. **Attention, c'est indispensable**. Si vous oubliez de le faire, votre système de fichiers risque d'être endommagé, car le système d'exploitation est peut-être encore en train d'écrire dessus. Si le CD refuse de sortir lorsque vous appuyez sur le bouton, c'est que vous ne l'avez pas démonté.

Avec le gestionnaire de fichiers, il faut aller dans le répertoire /media (aller à la racine dans l'arbre du panneau de gauche) choisir le bon répertoire, cliquer sur le bouton droit de la souris dessus, et choisir « umount » (démonter) dans le menu.

Si le système d'exploitation refuse de démonter le support, il faut s'assurer qu'aucun programme n'a de fichier ouvert sur le périphérique ou que votre shell n'est pas le répertoire de ce dernier. En effet le système refuse de démonter le support s'il est en cours d'accès.

Nous verrons plus tard ce qui se cache derrière cette automatisation. Nous réaliserons manuellement le montage et le démontage du périphérique.

## DESCRIPTION DES RÉPERTOIRES

Une des premières difficultés pour les débutants UNIX est de se repérer dans l'arborescence. Contrairement à un environnement Windows qui regroupe toutes les ressources nécessaires à un programme dans un même répertoire, les ressources dans un système UNIX (et dans une moindre mesure MacOS) sont réparties dans une hiérarchie générale. Ainsi, si le code exécutable d'un programme se trouve dans le répertoire /bin, ses bibliothèques se trouveront dans le répertoire /lib et ses fichiers de configuration, dans le répertoire /etc.

La hiérarchie des systèmes UNIX est normalisée. La spécification est accessible sur :

<http://www.pathname.com/fhs/>

/ la racine, elle contient les répertoires principaux.

**/bin** (bin=binaire) contient des exécutables essentiels au système et employés par tous les utilisateurs (par exemple, les commandes ls, rm, cp, chmod, mount, ...) ce sont les commandes de base du système.

**/dev** (dev=devices) contient les points d'entrées des périphériques, tous les périphériques sont pratiquement tous identifiés par un fichier ; la carte réseau n'est pas identifiée sous ce répertoire, elle est directement gérée par le noyau.

**/etc** (editable text configuration) contient les commandes et les fichiers nécessaires à l'administrateur du système (fichiers passwd, group, initab, ld.so.conf, lilo.conf, ...), les fichiers de configuration ainsi que les fichiers pour initialiser les services (réseau, serveur web , ssh , ...).

**/home** est le répertoire personnel des utilisateurs.

**/lib** (lib=library) contient des bibliothèques partagées essentielles au système lors du démarrage, l'équivalent des DLL.

**/media** contient les points de montages des périphériques amovibles (cd-rom, disquette, clef usb ...).

**/opt** (opt=option) contient des packages d'applications supplémentaires.

**/root** est le répertoire de l'administrateur root

**/sbin** (abréviation de system binaries, soit systèmes des binaires en français) contient les binaires systèmes essentiels, principalement à l'usage de l'administrateur root (super utilisateur) par exemple la commande adduser.

**/tmp** contient les fichiers temporaires.

**/usr** Hiérarchie secondaire (usr comme **UNiX** system **r**esources)

**/usr/bin** contient la majorité des fichiers binaires et commandes utilisateurs **/usr/include** contient les fichiers d'en-tête pour les programmes C et C++.

**/usr/lib** contient la plupart des bibliothèques partagées du système **/usr/local**, contient les données relatives aux programmes installés sur la machine locale par le root.

**/usr/local/bin** binaires des programmes locaux, programmes installés à la main.

**/usr/local/include** fichiers d'en-tête C et C++ locaux.

**/usr/local/lib** Bibliothèques partagées locales.

**/usr/local/sbin** binaires système locaux.

**/usr/local/share** hiérarchie indépendante.

**/usr/sbin** contient les fichiers binaires non essentiels au système, réservés à l'administrateur système.

**/usr/share** réservé aux données non dépendantes de l'architecture.

**/usr/src** contient des fichiers de code source.

**/var** contient des données variables.

## PARTITION ESPACE D'ÉCHANGE (SWAP)

L'**espace d'échange**, généralement appelée par son terme anglais « **swap space** » ou simplement **swap**, est une zone d'un disque dur faisant partie de la mémoire virtuelle) de votre ordinateur. Il est utilisé pour décharger la mémoire vive physique (RAM) de votre ordinateur lorsque celle-ci arrive à saturation. L'espace d'échange se trouve généralement sous une forme de partition de disque dur – on parle alors de partition d'échange. Il peut aussi se présenter sous forme de fichier – on parle alors de fichier d'échange.

Par défaut, Ubuntu calcule et s'attribue automatiquement un espace d'échange suffisant ou recommandé lors de son installation. Il n'est pas nécessaire d'effectuer des tâches supplémentaires pour assigner un espace d'échange minimum à Ubuntu. Cependant, évaluer ses besoins en espace d'échange peut mieux rationaliser vos ressources.

## QU'EST-CE QU'UN ESPACE D'ÉCHANGE ?

La mémoire vive physique (RAM) de l'ordinateur est utilisée pour stocker des données en cours de traitement. Si celle-ci se remplit presque entièrement mais votre ordinateur a tout de même besoin de ressources pour procéder à des traitements, votre système d'exploitation peut déplacer temporairement des pages mémoires vers l'espace d'échange défini dans votre disque dur afin de libérer des ressources mémoires. L'espace d'échange agit ainsi en tant qu'extension de votre mémoire vive physique : il récupère, au besoin, des blocs mémoire en excès de votre mémoire vive physique.

Notez cependant que l'espace d'échange se situe dans votre disque dur ; l'accès aux données contenues dans celui-ci est plus lent que celui aux données contenues directement dans la RAM. De plus, l'utilisation de l'espace d'échange ralentit significativement le système et entraîne une activité permanente du disque dur (bruits de "grattements"), provoquant une usure prématuée du matériel. De ce fait, l'espace d'échange ne doit pas être considéré comme un remplacement de votre mémoire vive physique, mais plutôt comme un mécanisme d'appoint.

La plupart des systèmes d'exploitation – sinon tous – gèrent un espace d'échange. Avec les systèmes d'exploitation de type GNU/Linux, dont fait partie Ubuntu, cette zone d'échange se présente généralement sous la forme d'une partition de disque dur dédiée à cet effet. À titre comparatif, Microsoft® Windows® utilise un ou des fichiers dans chacun de ses volumes afin de jouer ce rôle. Il est aussi possible, dans Ubuntu, de créer et utiliser des fichiers assurant un espace d'échange.

Linux peut utiliser une partition dite "swap" qui peut être employée en plus de la RAM.

## POURQUOI AI-JE BESOIN D'UN ESPACE D'ÉCHANGE ?

**Utilisation de programmes exigeants en ressources** : Parfois, des programmes exigent beaucoup de ressources dans votre ordinateur (comme la suite bureautique LibreOffice, des jeux tels AlienArena ou des logiciels de montage vidéo tels Cinelerra), ce qui amène votre ordinateur à nécessiter davantage de ressources que disponibles. Souvent, une partie de la mémoire vive utilisée au chargement de ces programme n'est réellement exploitée que lors de leur initialisation et n'est

plus utilisée par la suite. Le système peut libérer des ressources en déplaçant de telles pages mémoire vers un espace d'échange dans votre disque dur.

**Hibernation (mise en veille prolongée)** : La fonction d'hibernation conserve votre système en l'état actuel lors de la mise hors-tension de votre ordinateur, ce qui vous permet de reprendre plus rapidement votre travail là où vous l'avez laissé tout en ne consommant pas l'énergie de votre batterie ou de votre alimentation secteur. Techniquement, il s'agit d'une copie de l'ensemble des informations contenues dans la mémoire vive de votre ordinateur vers l'espace d'échange de votre disque dur. Pour cette raison, une partition d'échange au moins aussi grande que la quantité de votre mémoire vive est requise, pour profiter de cette fonctionnalité.

**Circonstances imprévues** : Des événements imprévisibles peuvent survenir lors de l'utilisation de votre système (un programme ayant une fuite de mémoire, une action qui a besoin davantage de mémoire pour un court laps de temps ou une combinaison de ces événements). Disposer d'un espace d'échange vous offre un sursis pour localiser le problème ou, à tout le moins, terminer votre travail en cours.

**Cache mémoire** : Puisque l'accès aux fichiers de votre disque dur est plus lent que ceux placés en mémoire vive, le noyau Linux place en cache dans la mémoire vive les fichiers ouverts. Ceci accélère grandement les traitements. Afin de conserver un maximum d'espace pour ce cache, une partie de la RAM réservée par les programmes mais non utilisée est déplacée dans l'espace d'échange.

## VIRTUALISATION

Linux ouvre également la possibilité d'obtenir une parfaite séparation entre plusieurs environnements virtuels tournant sur un seul ordinateur physique, en prenant en compte les modules de virtualisation présents dans les processeurs récents comme AMD-V sur AMD et Intel-VT (ou IVT) sur Intel. Ces environnements de virtualisation permettent d'exécuter des environnements différents ou plusieurs environnements similaires sur une même machine, tout en assurant une certaine sécurité dans la séparation des accès. Ce système est utilisé depuis longtemps par les mainframes d'IBM. IBM a d'ailleurs porté Linux sur celles-ci afin de permettre à ses clients de continuer à les utiliser avec un système plus moderne.

KVM est, depuis la version 2.6.20 du noyau Linux, le système officiel de virtualisation de ce dernier. Couplé aux outils de QEMU, il permet de créer des ordinateurs virtuels tournant directement sur un noyau Linux non modifié, et est capable d'accueillir un grand nombre de systèmes d'exploitation, tels que Windows, Solaris, BSD, etc... KVM est au cœur de la stratégie de virtualisation de RedHat.





# Chapitre 3

## INSTALLATION DE GNU/LINUX

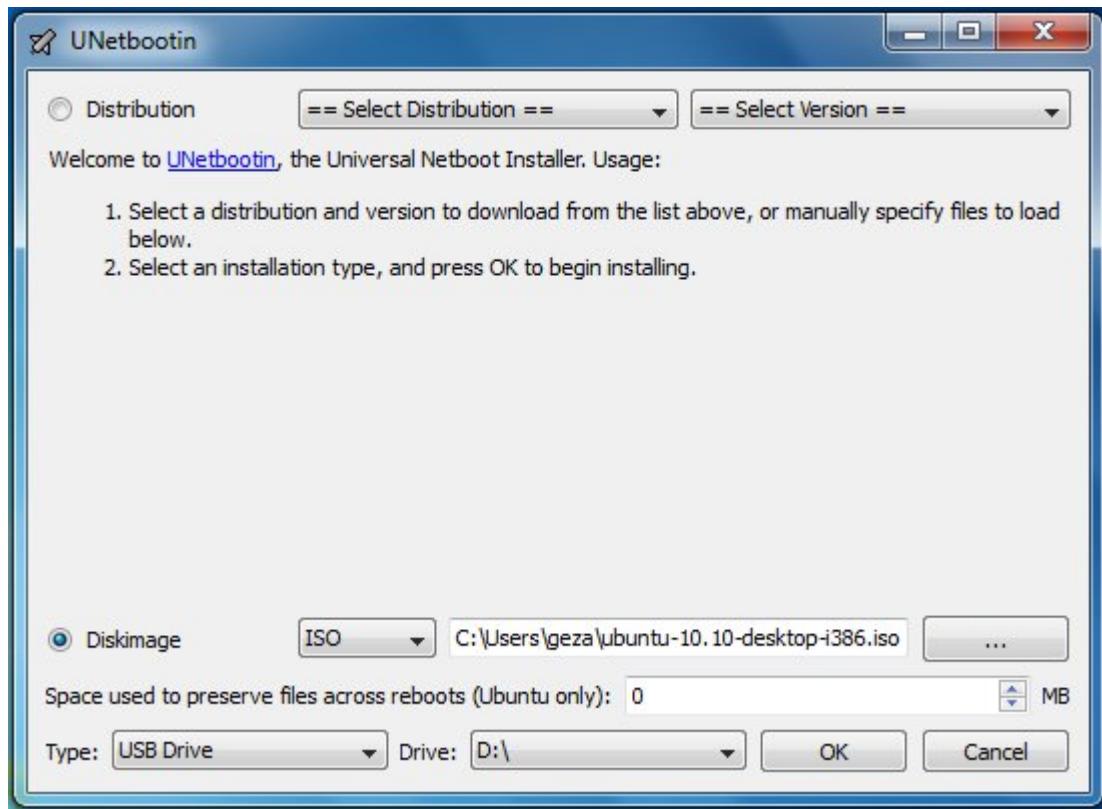
### 3.1 PRÉPARATION

**PREMIÈRE OPTION** : la réalisation d'un CD d'installation

1. Téléchargez la dernière version Desktop d'Ubuntu sur le site officiel :  
<http://www.ubuntu.com/download/desktop>, téléchargez le fichier iso.
2. Gravez le CD avec le logiciel de votre choix.

**DEUXIÈME OPTION** : créez une clef USB pour réaliser l'installation

1. Téléchargez et installez l'application UnetBootIn disponible à l'URL :  
<http://unetbootin.sourceforge.net/>
2. Téléchargez la dernière version Desktop d'Ubuntu sur le site officiel :  
<http://www.ubuntu.com/download/desktop>, téléchargez le fichier iso.
3. Insérez votre clef USB dans la machine
4. Démarrez l'application UnetBootIn
5. Sélectionnez le fichier CD préalablement téléchargé



6. Attention en bas complètement vous avez le lieu où sera installé l'installateur (si vous avez 2 clefs USB ou un disque externe assurez-vous que c'est le bon choix).

## 3.2 RÉALISATION DE L'INSTALLATION

L'ensemble des instructions sont disponibles sur la page suivante :

[http://doc.ubuntu-fr.org/tutoriel/installer\\_ubuntu\\_avec\\_le\\_live\\_cd](http://doc.ubuntu-fr.org/tutoriel/installer_ubuntu_avec_le_live_cd)

# Chapitre 4

## APPRIVOISER LA LIGNE DE COMMANDE

Ce chapitre présentera la ligne de commande. Bien qu'il ne soit pas indispensable pour un utilisateur de connaître la ligne de commande, il reste que c'est un plus. Plusieurs distributions ont vu le jour afin de simplifier l'utilisation de GNU/Linux ; l'utilisation du terminal n'est plus obligatoire. Cependant, la maîtrise de ce mode de communication avec la machine à de nombreux avantages :

- Automatisation d'instructions
- La ligne de commande est présente dans toutes les distributions
- Il est possible de programmer avec ce système
- Administration système de serveurs à distance

### L'objectif du chapitre :

- Vous offrir les bases pour utiliser le shell / terminal / ligne de commande
- Comprendre la structure d'une commande
- Être en mesure de trouver l'aide appropriée

### 4.1 PRÉSENTATION DE LA LIGNE DE COMMANDE

- Introduction à la ligne de commande sous l'angle de l'utilisateur.
- Description des interpréteurs de commandes disponibles.
- Présentation de la structure de l'interface

### DESCRIPTION D'UNE INTERFACE EN LIGNE DE COMMANDE

Une interface en ligne de commande (couramment abrégé CLI en anglais, **Common Line Interface**) est une interface homme-machine dans laquelle la communication entre l'utilisateur et l'ordinateur s'effectue en mode texte :

- l'utilisateur tape une ligne de commande, c'est-à-dire du texte au clavier, pour demander à l'ordinateur d'effectuer une opération ;

- l'ordinateur affiche du texte correspondant au résultat de l'exécution des commandes tapées ou à des questions qu'un logiciel pose à l'utilisateur.

Une interface en ligne de commandes peut servir aussi bien pour lancer l'exécution de divers logiciels au moyen d'un interpréteur de commandes, que pour les dialogues avec l'utilisateur de ces logiciels. C'est l'interaction fondamentale entre un homme et un ordinateur (ou tout autre équipement informatique).

Lorsqu'une interface est prête à recevoir une commande, elle l'indique par **une invite de commande**. Celle-ci, parfois désignée par l'anglicisme « prompt », consiste en quelques caractères, en début de ligne (généralement, le nom de compte de l'utilisateur, et/ou l'unité logique par défaut, et/ou le chemin par défaut, et/ou date, ...), se terminant par un caractère bien connu (souvent « ] », « # », « \$ » ou « > »), invitant l'utilisateur à taper une commande.

## LE SHELL BASH ET AUTRES INTERPRÉTEURS

Sous GNU/Linux, l'interpréteur de commandes par défaut ce nomme **Bash**, il existe beaucoup d'autre au même titre que les interfaces graphiques, cependant une majorité de personnes l'utilise.

Voici le nom d'autre interpréteur de commande disponible :

- Korn shell
- C shell
- Tcsh
- Z Shell (zsh)

Comme tous les interpréteurs en ligne de commandes de type script, **Bash** exécute quatre opérations fondamentales :

1. Il fournit une liste de commandes permettant d'opérer sur l'ordinateur (lancement de programmes, copie de fichiers, etc.) ;
2. Il permet de regrouper ces commandes dans un fichier unique, appelé script ;
3. Il vérifie la ligne de commande, lors de son exécution ou lors d'une éventuelle procédure de vérification, et renvoie un message d'erreur en cas d'erreur de syntaxe ;
4. En cas de validation, chaque ligne de commande est interprétée ; c'est-à-dire traduite dans un langage compréhensible par le système d'exploitation, qui l'exécute alors.

## INVITE DE COMMANDE (PROMPT)

Si vous démarrez un terminal, vous aurez tout de suite l'invite de commande. Beaucoup de distributions présentent l'invite comme suit :

UserName@Hostname :Répertoire\_courant\$

Si vous êtes connecté avec l'utilisateur root, ce devrait ressembler à :

root@nom\_de\_la\_machine :Répertoire\_courant#

- UserName : représente évidemment le nom de l'utilisateur en cours d'utilisation
- Hostname : Le nom de la machine qui attend les instructions à la ligne de commande.  
Exemple : alex@bureau :/home/alex\$. Ici la machine porte le nom de bureau
- Répertoire \_ courant : le répertoire où vous vous trouvez. Si vous avez le caractère ~ , ceci équivaut au répertoire personnel de l'utilisateur. Donc si l'utilisateur se nomme thomas, ~ est équivalent à /home/thomas
- \$ ou # : \$ indique que vous êtes connecté comme un utilisateur « normal ». # indique que vous êtes connecté avec le compte administrateur.

Exemple de l'utilisateur alex connecté sur la machine bureau se trouvant dans le repertoire de mary : alex@bureau :/home/mary\$

Bien entendu cette invite est personnalisable ; nous verrons plus tard comment réaliser cette opération.

## UTILISATION DE LA LIGNE DE COMMANDE (SHELL / TERMINAL)

**SYNTAXE GÉNÉRALE DES COMMANDES** Les commandes élémentaires sous UNIX sont de la forme :

commande [options] [argument(s)] suivi de la pression sur la touche « Entrée ». Les arguments peuvent être des [fichiers\_ou\_données].

La commande apparaissant en début de ligne est presque toujours le nom d'un logiciel. Ce logiciel peut être une commande du système d'exploitation, un logiciel écrit par un utilisateur.

**Une option en ligne de commande** (que l'on appelle aussi parfois un flag) modifie le fonctionnement d'une commande. L'effet de l'option dépend de la commande. Généralement, les options suivent immédiatement le nom de la commande et sont séparées entre elles par des espaces.

Il est impossible de savoir à priori quelles options un programme reconnaît, à moins de consulter sa documentation. Cependant, il est d'usage courant (mais pas systématique) qu'un programme affiche un résumé très sommaire de ses options lorsqu'il est lancé en ligne de commande avec l'une des options suivantes : ? ; -? ; -h ; /? ; /h ; -help ou –help.

Exemples :

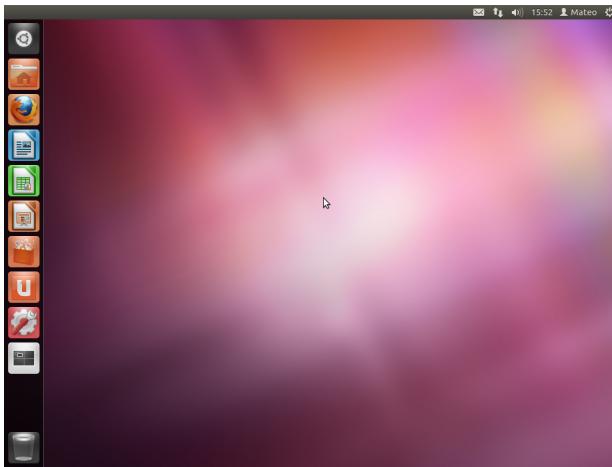
```
pat@amd-a10 :~$ man -h
```

```
pat@amd-a10 :~$ ls -help
```

Toutes les informations séparées par des espaces à droite du nom de la commande sont appelées **arguments de la ligne de commandes**.

## 4.2 ATELIER LIGNE DE COMMANDE

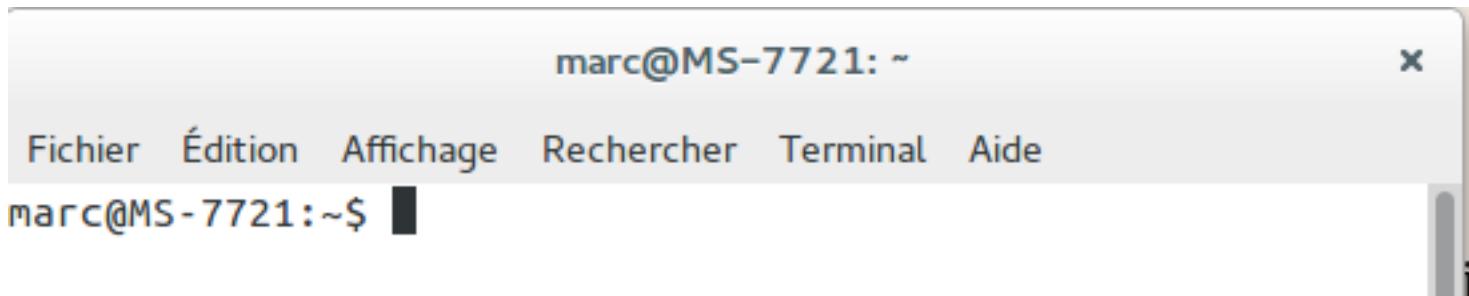
Je vais supposer que vous vous êtes connectés, c'est-à-dire que vous avez entré votre login et votre mot de passe. Vous êtes donc sur votre gestionnaire de bureau, ici Unity (figure suivante).



Nous allons commencer par la solution la plus « basique » que vous utiliserez probablement le plus souvent. Elle vous permet d'accéder à une console virtuelle, semblable à une vraie console. « Terminal » est un autre nom pour « Console ».

Sur le clavier, presser la combinaison de touches ALT + F2.

on obtient :



Pour fermer cette fenêtre il suffit de taper « exit » ou la combinaison de touches CTRL + D. (touche en bas à gauche du clavier).

Nous pouvons aussi commencer par la solution la plus « basique » que vous utiliserez probablement le moins souvent. Elle vous permet d'accéder à la vraie console (si tant est qu'il y ait une « vraie » console) en pressant une combinaison de touches.

Ctrl + Alt + F1 : terminal 1 (tty1) ;

Ctrl + Alt + F2 : terminal 2 (tty2) ;

Ctrl + Alt + F3 : terminal 3 (tty3) ;

Ctrl + Alt + F4 : terminal 4 (tty4) ;

Ctrl + Alt + F5 : terminal 5 (tty5) ;

Ctrl + Alt + F6 : terminal 6 (tty6) ;

Ctrl + Alt + F7 : retour au mode graphique !

## LES DIFFÉRENTES CONSOLES

Sous toute machine Linux, il y a donc non pas une mais six consoles qui fonctionnent en simultanées (d'où les six raccourcis différents de Ctrl + Alt + F1 à Ctrl + Alt + F6).

Vous pouvez savoir dans quel terminal vous êtes lors du chargement : il est en effet marqué « tty1 » si vous êtes sur le terminal n°1. Regardez la figure suivante.

```
Debian GNU/Linux 4.0 debian tty1
debian login:
```

Pour tester, tapez Ctrl + Alt + F1. Votre écran va peut-être clignoter quelques instants ; ne paniquez pas. Vous allez ensuite entrer en mode console plein écran.

Attention : pensez bien, si vous testez, que vous serez alors en mode console. Vous devrez donc utiliser Ctrl + Alt + F7 pour revenir en mode graphique. N'oubliez pas !

Cette dernière section a été inspirée du site :

<https://openclassrooms.com/courses/reprenez-le-controle-a-laide-de-linux/la-console-ca-se-mange>





# Chapitre 5

## COMMANDES DE BASE

Quelques commandes de base essentielles :

- déplacement dans l’arborescence /manipulation de fichier
- visualisation de fichier
- édition simple d’un fichier.

### 5.1 COMMANDES DE BASE

#### Évoluer dans l’arborescence et manipulation de fichier

- Affichage du répertoire courant (**pwd**)
- indique les opérateurs connectés au système (**who**, **who am i**, **whoami**)
- Afficher le nom de fichier du terminal associé à l’entrée standard (**tty**)
- indique la date du système (**date**)
- Affichage d’un calendrier (**cal**)
- Changement du répertoire (**cd**)
  - Chemin relatif et absolu
- Liste des fichiers d’un répertoire (**ls**)
- Affichage du contenu d’un fichier
  - Afficher tout le fichier (**cat**)
  - Afficher et naviguer dans un fichier (**less** / **more**)
- Afficher le début et la fin d’un fichier (**head** / **tail**)
- Création de répertoire (**mkdir**)
- Supprimer des répertoires vides (**rmdir**)
- Créer un fichier vide (**touch**)
- Copie de fichier ou répertoire (**cp**)
- Suppression de fichier ou répertoire (**rm**)
- Déplacement de fichier ou renommage (**mv**)
- Éditeur minimalist (**pico/nano**)
- Effacer l’écran ( **clear** )

## 5.2 ÉVOLUER DANS L'ARBORESCENCE ET MANIPULATION DE FICHIER

### Afficher le répertoire courant (pwd)

Pour savoir dans quel répertoire vous vous trouvez, vous pouvez utiliser la commande **pwd** (print working directory). C'est le répertoire par défaut lors de la connexion :

```
$ pwd
```

```
/home/Bob
```

### Indiquer les opérateurs connectés au système (who)

**who** fournit des informations sur l'ensemble des utilisateurs qui sont actuellement connectés sur la station.

**who am i** renvoie uniquement les informations relatives à l'utilisateur courant.

**whoami** renvoie seulement l'identificateur de l'utilisateur courant.

```
marc@marc-MS-7721:~$ who
marc      :0          2015-12-15 14:08 (:0)
marc      pts/3        2015-12-15 14:25 (:0)
marc@marc-MS-7721:~$ who am i
marc      pts/3        2015-12-15 14:25 (:0)
marc@marc-MS-7721:~$ whoami
marc
```

Introduisons une option pour modifier le comportement de base de la commande **who**. **-b**, ou **--boot** permet d'afficher l'heure du dernier démarrage.

```
marc@marc-MS-7721 :~$ who -b
```

```
démarrage système 2015-12-15 10 :19
```

```
marc@marc-MS-7721:~$ who --help
Utilisation : who [OPTION]... [ FILE | ARG1 ARG2 ]
Afficher des informations sur les utilisateurs connectés.

-a, --all      identique à -b -d --login -p -r -t -T -u
-b, --boot     afficher l'heure du dernier démarrage système
-d, --dead     afficher la liste des processus morts
-H, --heading   afficher les en-têtes de colonne des lignes
--ips          print ips instead of hostnames. with --lookup,
              canonicalizes based on stored IP, if available,
              rather than stored hostname
-l, --login    afficher le processus de connexion du système
```

[[ affichage partiel ]]

## Afficher le nom de fichier du terminal associé à l'entrée standard (tty)

```
marc@marc-MS-7721 :~$ tty
```

```
/dev/pts/4
```

## Indiquer la date du système (date)

```
marc@marc-MS-7721 :~$ date
```

```
mardi 15 décembre 2015, 11 :14 :29 (UTC+0100)
```

## Affichage d'un calendrier (cal)

```
marc@marc-MS-7721:~$ cal 02 2005
  Février 2005
di lu ma me je ve sa
      1  2  3  4  5
  6  7  8  9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28
```

Cette dernière commande a permis d'obtenir un mois et une année précise.

La commande « **cal** » seule fournit le mois en cours, **cal 2016** fournit un calendrier complet pour l'année 2016 !

## Changer de répertoire (cd)

La commande **cd** (change directory) vous permet de changer de répertoire :

**\$ cd .** ⇒ . désigne le répertoire courant

**\$ cd ..** ⇒ .. désigne le répertoire parent

**\$ cd /** ⇒ / désigne le répertoire racine

**\$ cd /tmp** ⇒ désigne le répertoire tmp appartenant à la racine

**\$ cd tmp** ⇒ désigne le répertoire tmp du répertoire courant

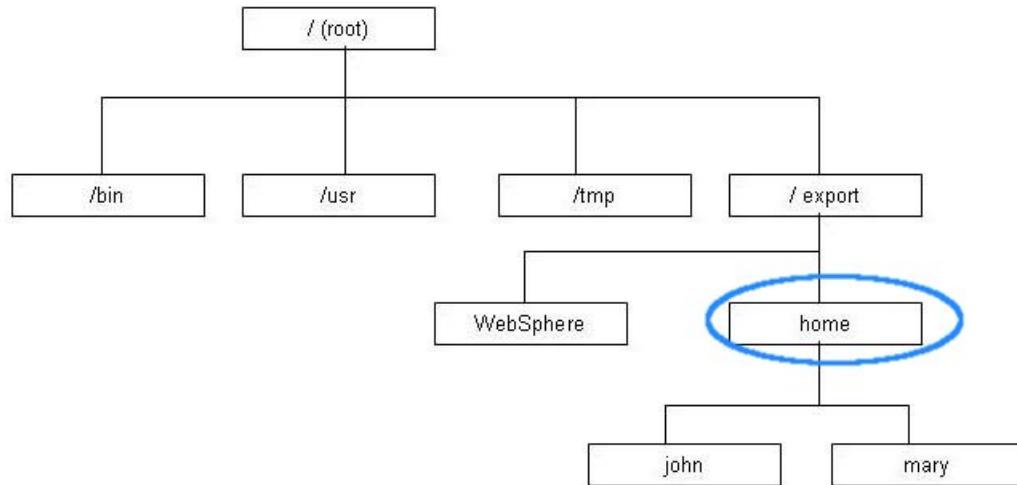
**\$ cd ./tmp** ⇒ désigne le répertoire tmp du répertoire parent du répertoire courant

**\$ cd ~** ⇒ permet de revenir dans son répertoire de travail (home directory)

**\$ cd** ⇒ idem

## Chemin relatif et absolu

- Le **chemin absolu** fait toujours référence à la racine (représenté par le caractère / slash) du système sur lequel est situé l’arborescence des dossiers avec lesquels on travaille.
- Le **chemin relatif**, comme son nom l’indique, fait référence à la position relative au dossier où vous vous trouvez.



Ici il y a le répertoire personnel de l’utilisateur john et de l’utilisatrice mary.

Par exemple si je suis dans le répertoire home/, représenté en bleu ci-dessus, l’invite de commande sera représenté comme suit :

pat@amd-a10 :/export/home\$

Si je désire aller dans le répertoire mary, je peux utiliser la notation absolue ou relative. L’ensemble des commandes suivantes sont équivalentes :

Notation absolue

pat@amd-a10 :/export/home\$ cd /export/home/mary

Notation relative

pat@amd-a10 :/export/home\$ cd mary

pat@amd-a10 :/export/home\$ cd ./mary

pat@amd-a10 :/export/home\$ cd ../../export/home/mary

## Liste les fichiers d’un répertoire (ls)

la commande **ls** (list) liste le contenu du répertoire. Un grand nombre d’options existent pour cette commande, afin de faciliter l’affichage du résultat. Par défaut, « ls » liste le contenu du répertoire courant, mais il est aussi possible de lui fournir le répertoire ou fichier à lister.

```
# ls : liste en colonnes le contenu du répertoire courant /home/alex
```

```
pat@amd-a10 :/home/alex$ ls
```

```
Desktop Downloads Music Public Videos Documents fichiers Pictures Templates
```

```
# Liste le contenu du répertoire /tmp
```

```
pat@amd-a10 :/home/alex$ ls /tmp
```

```
pulse-2L9K88eMlGn7 tmpksCRJ5 ssh-BkXkFipk2242 fichiers_temporaire un_autre_exemple
```

Option intéressante :

**-l** (long) : liste détaillée des fichiers. Affiche les permissions , le propriétaire , la taille et la date du fichier

```
pat@amd-a10 :/home/alex$ ls -l /etc/
```

```
total 1636
```

```
drwxr-xr-x 3 root root 4096 Mar 12 2013 acpi
```

```
-rw-r-r- 1 root root 2981 Apr 25 2011 adduser.conf
```

```
drwxr-xr-x 2 root root 32768 Nov 7 16:06 alternatives
```

```
-rw-r-r- 1 root root 395 Jun 20 2010 anacrontab
```

```
drwxr-xr-x 7 root root 4096 Jul 26 16:01 apache2
```

```
-rw-r-r- 1 root root 112 Jun 22 2007 apg.conf
```

```
drwxr-xr-x 6 root root 4096 Apr 25 2011 aptm
```

```
drwxr-xr-x 3 root root 4096 Sep 20 08:19 apparmor
```

```
drwxr-xr-x 8 root root 4096 Nov 5 09:47 apparmor.d
```

```
drwxr-xr-x 5 root root 4096 Nov 5 09:46 apport
```

```
drwxr-xr-x 6 root root 4096 Oct 23 13:03 apt
```

```
-rw-r-- 1 root daemon 144 Jun 27 2010 at.deny
```

```
drwxr-xr-x 2 root root 4096 Aug 24 2012 at-spi2
```

```
drwxr-xr-x 3 root root 4096 Aug 24 2012 avahi
```

..... [texte tronqué] .....

**-a** (all) : liste tous les fichiers, y compris les fichiers "cachés". Un fichier "caché" est un fichier qui débute par un ".".

```
pat@amd-a10:~$ ls -a
.               .bashrc          certification.lyx-  dma.lyx-  .gimp-2.8      .hardinfo    Images   .local   Musique
..              Bureau           .cinnamon       .dmrc     .gksu.lock     .ICEAuthority  IRQ APIC.lyx- .lyx      ports.lyx-
.bash_history  .cache          .config         Documents .gnome2       .icedove     .kde       Modèles  pratiques
.bash_logout   certification.lyx config materiel.lyx- .gconf     .gnome2_private images   .lessht   .mozilla .profile
```

**-color= auto / none** : Permet d'afficher ou non les couleurs avec ls , la majorité des distributions active la coloration par défaut

Options combinées, il est possible de combiner des options :

**-l -t -r ou -ltr** : affiche le résultat en format long ( **-l** ), trie le résultat par la date de modification du fichier contrairement au nom par défaut ( **-t** ), le fichier le plus récent s'affichera en premier le plus vieux en dernier, inverse le résultat ( **-r** ) ceci afin d'avoir le fichier le plus récent affiché en dernier.

Sans le **reverse**

```
pat@amd-a10 :/home/alex$ ls -lt /etc
..... [texte tronqué] ...
drwxr-xr-x 3 root root 4096 May 9 2010 insserv
-rw-r-r- 1 root root 645 Mar 7 2010 ts.conf
-rw-r-r- 1 root root 15752 Jul 25 2009 ltrace.conf
-rw-r-r- 1 root root 112 Jun 22 2007 apg.conf
-rw-r-r- 1 root root 10852 Apr 27 2007 gnome-vfs-mime-magic
-rw-r-r- 1 root root 1343 Jan 9 2007 wodim.conf
-rw-r-r- 1 root root 2064 Nov 23 2006 netscsid.conf
utilisateur@hotname :/home/alex$
```

Avec l'option reverse « **r** »

```
pat@amd-a10 :/home/alex$ ls -ltr /etc
..... [texte tronqué] ...
-rw—— 1 root root 1822 Nov 29 15 :51 shadow-
-rw—— 1 root root 1164 Nov 29 15 :51 group-
-rw—— 1 root root 959 Nov 29 15 :51 gshadow-
-rw-r-r- 1 root root 1176 Nov 29 15 :51 group
-rw—— 1 root root 2650 Nov 29 15 :51 passwd-
-rw-r—— 1 root shadow 967 Nov 29 15 :51 gshadow
-rw-r—— 1 root shadow 1944 Nov 29 15 :51 shadow
-rw-r-r- 1 root root 2653 Nov 29 15 :51 passw
```

**-l -block-size=M ou -h** : affiche le résultat en format long ( **-l** ) et fournit la taille des fichiers en méga-octets , beaucoup plus facile à lire. pat@amd-a10 :/home/alex\$ **ls -l -h** ou mieux **ls -lh**

```
drwxr-xr-x 6 xerus xerus 4.0K Jul 22 21 :19 gits
-rw-r-r- 1 xerus xerus 1.7K Mar 23 2013 GNUstepDefaults
-rw-r-r- 1 xerus xerus 33M Sep 17 12 :31 go.tar.gz
```

```
-rw-rw-r-- 1 xerus xerus 87K Nov 21 12 :53 hand_spacewalk.png
```

Si on rajoute à **ls** l'option **-F**, chaque nom de répertoire contenu dans le répertoire actif est suivi d'un slash.

```
pat@amd-a10 :~/pratiques$ ls -F
doc/ essai/ exo1/ fich1 fich3 fiche2
```

## 5.3 AFFICHER LE CONTENU D'UN FICHIER

Nous allons d'abord voir comment afficher le contenu d'un fichier. Il y a en gros deux commandes basiques sous Linux **cat** et **less** qui permettent de faire cela :

Aucune de ces commandes ne permet d'édition un fichier, elles permettent juste de le voir.

Afficher tout le fichier (**cat**)

La commande **cat** (concatenate) affiche le contenu d'un fichier ou de plusieurs fichiers concaténés à l'écran.

```
pat@amd-a10 :~$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
..... [texte tronqué] .....
statd:x:114:65534::/var/lib/nfs:/bin/false postfix:x:115:124::/var/spool/postfix:/bin/false
```

Option intéressante :

**-n** ou **--number** (number) : permet d'afficher le numéro de ligne

```
pat@amd-a10 :/home/alex$ cat -n /etc/passwd
1 root:x:0:0:root:/root:/bin/bash
2 daemon:x:1:1:daemon:/usr/sbin:/bin/sh
3 bin:x:2:2:bin:/bin:/bin/sh
4 sys:x:3:3:sys:/dev:/bin/sh
5 sync:x:4:65534:sync:/bin:/bin/sync
..... [texte tronqué] .....
```

```
35 statd :x :114 :65534 ::/var/lib/nfs :/bin/false
36 postfix :x :115 :124 ::/var/spool/postfix :/bin/false
```

Il est fréquent de voir des fichiers sans extension sous Linux. Notre fichier s'appelle `passwd` tout court, et non pas `passwd.txt` ou `passwd.log` comme on pourrait avoir l'habitude de le voir sous Windows. Un fichier sans extension peut être ouvert et lu sans aucun problème comme n'importe quel autre fichier.

## AFFICHER LE FICHIER PAR PAGES

Le problème de `cat` est qu'il peut être compliqué de visualiser un fichier s'il est très grand. En effet, `cat` fera dérouler le texte dans le terminal du début à la fin ; il peut donc être difficile à lire. Pour solutionner ce problème, nous avons les commandes `less` et `more`. Personnellement, j'utilise beaucoup plus `less` que `more`, car le premier a plus d'options que le deuxième.

```
# utilisation de less
pat@amd-a10 :/home/alex$ less /etc/passwd

# utilisation de less , affiche les numéros de ligne
pat@amd-a10 :/home/alex$ less -N /etc/passwd
```

Ce qui est intéressant pour nous ici, la commande `less` a arrêté la lecture du fichier au bout de quelques lignes (la taille d'un écran de console). Cela vous laisse le temps de lire le début du fichier.

On n'a lu pour le moment que les toutes premières lignes du fichier. Et comment lire la suite ? Il y a quelques raccourcis clavier à connaître.

Les raccourcis basiques indispensables :

Espace : affiche la suite du fichier. La touche Espace fait défiler le fichier vers le bas d'un « écran » de console. C'est celle que j'utilise le plus souvent.

Entrée : affiche la ligne suivante. Cela permet donc de faire défiler le fichier vers le bas ligne par ligne.

Vous pouvez aussi utiliser la touche Flèche vers le bas.

- `d` : affiche les onze lignes suivantes (soit une moitié d'écran). C'est un peu l'intermédiaire entre Espace (tout un écran) et Entrée (une seule ligne).
- `b` : retourne en arrière d'un écran.
- Vous pouvez aussi appuyer sur la touche Page Up.
- `y` : retourne d'une ligne en arrière.
- Vous pouvez aussi appuyer sur la touche Flèche vers le haut.
- `u` : retourne en arrière d'une moitié d'écran (onze lignes).
- `q` : arrête la lecture du fichier. Cela met fin à la commande `less`.

La casse des caractères est importante. Ainsi, si je vous dis qu'il faut appuyer sur la touche « d », ce n'est pas un « D » majuscule (si vous essayez, vous verrez que ça ne fonctionne pas). Sous Linux, on fait souvent la différence entre majuscules et minuscules : souvenez-vous-en !

```
# utilisation de more utilisateur@hostname :~$ more /etc/passwd
```

## AFFICHER LE DÉBUT ET LA FIN D'UN FICHIER

Nous ne désirons pas toujours avoir l'intégralité d'un fichier. Par moment, afficher un nombre X de lignes au début (**head**) ou à la fin (**tail**) d'un fichier peut être suffisant. La commande **head** et **tail**, avec l'option **-n** (number), nous permet d'avoir ce comportement. Par défaut, si le nombre de ligne n'est pas spécifié, le système affiche 10 lignes.

### # HEAD

```
pat@amd-a10 :/home/alex$ head -n 4 /etc/passwd
root :x :0 :0 .root :/bin/bash
daemon :x :1 :1 :daemon :/usr/sbin :/bin/sh
bin :x :2 :2 :bin :/bin :/bin/sh
sys :x :3 :3 :sys :/dev :/bin/sh
```

### # TAIL

```
pat@amd-a10 :/home/alex$ tail /etc/passwd
rtkit :x :106 :116 :RealtimeKit,,,:/proc :/bin/false sshd :x :107 :65534 ::/var/run/sshd :/usr/sbin/nologin
[[output coupé]].

statd :x :114 :65534 ::/var/lib/nfs :/bin/false postfix :x :115 :124 ::/var/spool/postfix :/bin/false
```

Option intéressante de **tail** :

- On peut là encore utiliser **-n** suivi d'un nombre pour afficher les dernières lignes :
 

```
pat@amd-a10 :/home/alex$ tail -n 2 /etc/passwd
```

 pour afficher les 2 dernières lignes
- **-f** ou **-follow** : permet d'afficher à l'écran les lignes qui se rajoutent au fichier de manière continue. Ceci est **TRÈS intéressant**, principalement pour la visualisation des logs.
 

```
pat@amd-a10 :/home/alex$ tail -f /var/log/syslog
```

Faites **Ctrl + C** (Ctrl et C en même temps) pour arrêter la commande **tail**.

La combinaison de touches Ctrl + C est utilisable en mode console pour arrêter les programmes en cours de fonctionnement.

## CRÉATION DE RÉPERTOIRE

La création d'un répertoire peut être réalisée en utilisant la commande **mkdir** (make directory), en donnant le nom du répertoire à créer en argument, comme suit :

Notation absolue :

```
utilisateur@hostname :/home/utilisateur$ mkdir /tmp/Nom_du_répertoire
```

Notation relative :

```
utilisateur@hostname :/home/utilisateur$ mkdir Nom_du_répertoire
```

```
utilisateur@hostname :/home/utilisateur$ mkdir ..../tmp/Nom_du_répertoire
```

Option intéressante :

**-p** : permet de créer une arborescence de répertoire plus rapidement ; mkdir, par défaut, ne crée qu'un répertoire à la fois. Par exemple, si je suis dans le répertoire /home/alex et que je désire créer la hiérarchie de répertoires /home/alex/images/2013/06/18, voici comment faire : mkdir option parent

Commande en ERREUR :

```
utilisateur@hostname :/home/alex$ mkdir images/2013/06/18
```

mkdir : impossible de créer le répertoire « images/2013/06/18 » : Aucun fichier ou dossier de ce type

Solution avec l'option **-p** les répertoires successifs sont créés :

```
utilisateur@hostname :/home/pat$ mkdir -p images/2013/06/18
```

## SUPPRIMER DES RÉPERTOIRES VIDES

Tant qu'un répertoire ne contient aucun fichier, on peut le supprimer avec la commande rmdir (remove directory).

```
marc@marc-MS-7721 :~$ rmdir lesson
```

Il ne faut pas se trouver dans le répertoire à supprimer sinon on obtient une erreur :

```
marc@marc-MS-7721 :~/lesson$ rmdir lesson
```

rmdir : échec de suppression de «lesson» : Aucun fichier ou dossier de ce type

## CRÉER UN FICHIER VIDE

Bien que cette commande « touch » à l'origine a été créée pour modifier l'horodatage d'un fichier, on l'emploie de manière indirecte pour créer un fichier vide. L'appel de « touch » suivi d'un nom de fichier crée un fichier vide à condition qu'un fichier existant ne comporte déjà le même nom.

```
marc@marc-MS-7721 :~/chap5$ touch voiture
```

création de cinq fichiers vides :

```
marc@marc-MS-7721 :~/chap5$ touch autos citroen fiat peugeot voiture
```

Une autre manière de créer des fichiers vides et d'utiliser le caractère « > » suivi du nom du fichier vide à créer. ( voir le chapitre 17 redirection et flux de données pour l'utilisation approfondie de > ). Un seul fichier peut être créé à la fois contrairement à **touch**.

```
marc@marc-MS-7721 :~/chap5$ > poire
```

## COPIE DE FICHIER OU RÉPERTOIRE

La commande **cp** (copie) nous permet de réaliser la copie d'un ou plusieurs fichiers ou répertoires.

```
# Copie du fichier passwd dans /tmp
utilisateur@hostname :/home/alex$ cp /etc/passwd /tmp
```

```
# Copie du fichier passwd et changement de nom du fichier pour copie_pass
utilisateur@hostname :/home/alex$ cp /etc/passwd /tmp/copie_pass
```

```
# Copie de plusieurs fichiers dans un répertoire, quand l'on copie plusieurs fichiers la destination doit obligatoirement être un répertoire.
```

```
utilisateur@hostname :/home/alex$ cp /etc/passwd /etc/fstab /etc/group /tmp/
```

```
# Copie d'un répertoire avec l'option -r récursive
```

```
utilisateur@hostname :/home/alex$ cp -r /etc/network /tmp
```

Options intéressantes :

**-r** (récurse) : telle que présenté plus tôt, nous permet de faire une copie récursivement des fichiers

**-l** ou **-link** (link) : ne réalise pas la copie du fichier à proprement parler, mais crée un Hard Link ( lien ) du fichier vers l'inode correspondant sur le disque.

**-parents** : cette option nous permet d'indiquer à la commande **cp** de recréer la structure de répertoire lors de la copie ; économisant l'utilisation de la commande de création de répertoire **mkdir**

```
utilisateur@hostname :/home/alex$ ls -R /tmp/testing
```

```
pat@amd-a10 :/home/alex$ cp -parent /usr/share/debianutils/shells /usr/share/gimp/2.0/images/gimp-
splash.png /etc/network/interfaces /tmp/testing/
```

```
pat@amd-a10 :/home/alex$ ls -R /tmp/testing
```

```
/tmp/testing/ :
```

```
etc usr
```

```
/tmp/testing/etc :
```

```
network
```

```
/tmp/testing/etc/network :
```

```
interfaces /tmp/testing/usr :
```

```
share
```

```
/tmp/testing/usr/share :
```

```
debianutils gimp
```

```
/tmp/testing/usr/share/debianutils :
```

```
shells
```

```
/tmp/testing/usr/share/gimp :
```

```
2.0
```

```
/tmp/testing/usr/share/gimp/2.0 :
```

```
images
```

```
/tmp/testing/usr/share/gimp/2.0/images : gimp-splash.png
```

## SUPPRESSION DE FICHIER OU RÉPERTOIRE

La commande **rm** (remove) permet la suppression de chaque fichier listé. Par défaut, il n'efface pas les répertoires.

### OPTIONS

- -f, --force : ignorer les fichiers et paramètres inexistantes, ne pas demander de confirmation
- -i : demander une confirmation avant chaque effacement, c'est le mode interactif
- -r, -R, --recursive : enlever le contenu des répertoires récursivement, commande très dangereuse avec ces options car elle peut tout effacer !
- -v, --verbose : expliquer ce qui est fait

```
Suppression du fichier photo1.png utilisateur@hostname :/home/alex$ rm Lefichier
```

```
# pour la suppression de répertoire il faut ajouter l'option -r pour récursive
```

```
# ici la suppression est réalisée sur le répertoire photos_noel ls -F
```

```
alex@amd-a10 :/home/alex$ rm -r photos_noel
```

## DÉPLACEMENT DE FICHIER OU RENOMMAGE

La commande **mv** (move) permet de déplacer les fichiers d'un répertoire à l'autre ou de renommer le fichier

```
# Déplace un fichier d'un répertoire à l'autre
```

```
alex@amd-a10 :/home/alex$ mv /home/alex/Documents/billetterie.png Projets/GrosProblem/
```

```
# renommer un fichier
```

```
alex@amd-a10 :/home/alex$ mv Projets/GrosProblem/billetterie.png Projets/GrosProblem/loterie.png
```

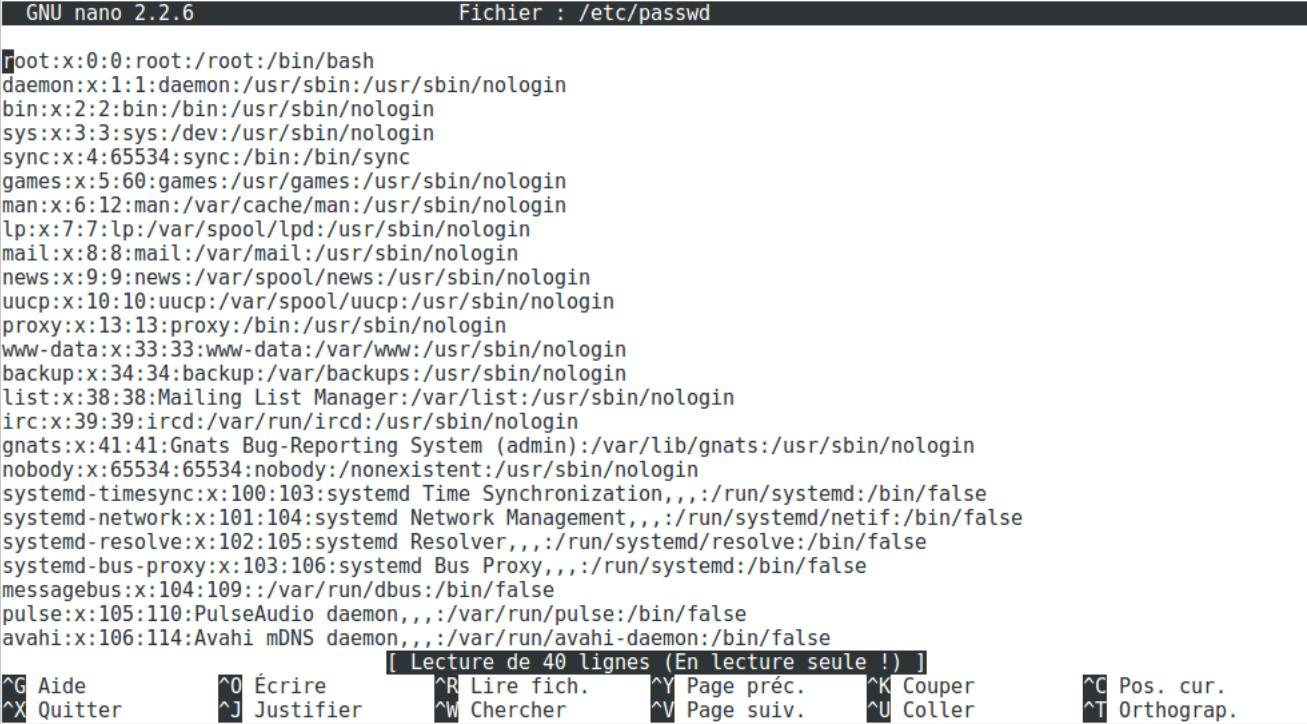
## ÉDITEUR MINIMALISTE

En attendant de présenter un "vrai" éditeur de texte du moins l'un des plus puissants disponibles sur GNU/Linux et UNIX, c'est à dire VI. Vous pouvez utiliser pico ou nano les deux font appel au même binaire lors de l'installation sous Ubuntu. **nano** vous offre l'équivalent de notepad sous windows, un éditeur de texte simple pour modifier des fichiers.

Lorsque vous appelez **nano** dans la ligne de commandes, vous pouvez spécifier plusieurs paramètres. Le plus courant est d'indiquer en paramètre le nom du fichier à ouvrir. Ainsi : nano /etc/passwd ouvre automatiquement le fichier concerné.

Si le fichier n'existe pas, il sera automatiquement créé par nano lors du premier enregistrement.

utilisateur@hostname :~\$ **nano** /etc/passwd



The screenshot shows the nano text editor interface. At the top, it says "GNU nano 2.2.6" and "Fichier : /etc/passwd". The main area displays the contents of the /etc/passwd file, which lists various system users and their details. At the bottom, there is a status bar with the message "[ Lecture de 40 lignes (En lecture seule !) ]". Below the status bar, there is a legend of keyboard shortcuts:

<b>^G</b> Aide	<b>^O</b> Écrire	<b>^R</b> Lire fich.	<b>^Y</b> Page préc.	<b>^K</b> Couper	<b>^C</b> Pos. cur.
<b>^X</b> Quitter	<b>^J</b> Justifier	<b>^W</b> Chercher	<b>^V</b> Page suiv.	<b>^U</b> Coller	<b>^T</b> Orthograp.

En bas de votre écran, vous pouvez voir un espace d'aide (figure ci-dessus). Que signifie-t-il exactement ? Il s'agit d'un aide-mémoire pour vous rappeler à tout moment les commandes principales que vous pouvez lancer sous Nano.

Le symbole ^ signifie Ctrl (la touche Contrôle de votre clavier). Ainsi, pour quitter Nano, il suffit de taper Ctrl + X.

Si l'aide-mémoire vous encombre, vous pouvez gagner de la place en appuyant sur Échap puis sur x. Vous pouvez l'afficher de nouveau avec la même suite de touches.

Pour ouvrir l'éditeur **nano** soit on tape **nano**, soit on utilise le raccourci clavier **Ctrl + x** suivi de **Ctrl + e** dans une console.

## EFFACER L'ÉCRAN

La commande « **clear** » nettoie votre fenêtre du terminal en reléguant tout le texte au dessus (donc accessible avec un scroll) et vous laissant donc face à une fenêtre propre. Bien utile de temps à autre pour y voir plus clair. Le raccourci clavier **Ctrl + L** fait la même chose.

## ÉDITER UN FICHIER DE CONFIGURATION UNIX

Sous UNIX, et en particulier sous Linux, la configuration du système et des programmes se fait très souvent en éditant des fichiers textes qui contiennent des paramètres de configuration. Ces paramètres de configuration suivent une certaine syntaxe, différente pour chaque programme, et que l'utilisateur doit connaître. Généralement, il y a une instruction de configuration par ligne de texte. Le système ou le programme va alors lire son ou ses fichier(s) de configuration et s'adapter à la configuration demandée.

Presque tous les programmes et systèmes UNIX sont conçus avec une règle qui ne tient pas compte des lignes du fichier de configuration qui commencent par un certain caractère (souvent **#**). L'utilisateur peut alors mettre des lignes de commentaires dans le fichier de configuration en commençant ces lignes par le caractère particulier. Il peut aussi facilement activer ou désactiver une ligne du fichier de configuration en enlevant ou en ajoutant le caractère particulier au début de la ligne. Le fait de désactiver ainsi une ligne de configuration se dit « commenter une ligne » et le fait d'activer ainsi une ligne de configuration se dit « dé-commenter une ligne ». Ces expressions seront régulièrement utilisées dans la suite de cette formation.

## 5.4 LIER DES COMMANDES

On peut taper plusieurs commandes sur la même ligne à condition de les séparer par un point virgule. Les commandes sont exécutées les unes à la suite des autres dans l'ordre où elles figurent dans la ligne. Les résultats apparaissent dans le même ordre, les uns en dessous des autres.

```
marc@marc-MS-7721:~/lesson$ pwd ; who ; tty ; date ; cd ; mkdir chap5 ; cd chap5 ; touch autos ;  
rmdir chap5 ; ls -l  
/home/marc/lesson  
marc      :0          2015-12-15 10:20 (:0)  
marc      pts/4        2015-12-15 11:12 (:0)  
/dev/pts/4  
mardi 15 décembre 2015, 12:01:51 (UTC+0100)  
rmdir: échec de suppression de «chap5»: Aucun fichier ou dossier de ce type  
total 0  
-rw-rw-r-- 1 marc marc 0 déc. 15 12:01 autos
```

## 5.5 INTERRUPTION D'UNE COMMANDE

Une saisie incorrecte d'une option ou d'un argument peut amener le déroulement du programme dans une situation non désirée. Il faut arrêter ce programme par l'appui simultané des touches Ctrl et C notée Ctrl+C ou encore ^C.

C'est la commande **stty -a** qui nous a indiqué cette séquence de touches que l'on trouve après le mot clé **intr=**

```
marc@marc-MS-7721:~/chap5$ stty -a
speed 38400 baud; rows 33; columns 96; line = 0;
intr = ^C; quit = ^\; erase = ^?; kill = ^U; eof = ^D; eol = M-^?; eol2 = M-^?; swtch = M-^?;
start = ^Q; stop = ^S; susp = ^Z; rprnt = ^R; werase = ^W; lnext = ^V; flush = ^O;
min = 1; time = 0;
```

## 5.6 VERSION VIDÉO

télécharger le fichier : commandes\_de\_base.ogv

## 5.7 AIDE-MÉMOIRE DES COMMANDES LINUX

<http://cli.handylinux.org/>

# Chapitre 6

## ATELIER D'UTILISATION DES COMMANDES DE BASE

L'objectif de l'atelier est d'être confortable avec l'utilisation des commandes de base pour se déplacer dans le système et d'y réaliser des modifications .

### 6.1 UTILISATION DES COMMANDES DE BASE

1. Listez les fichiers dans le répertoire /etc
2. Créez le répertoire **pratiques** dans votre répertoire personnel ( il est possible de faire les 2 répertoires en 1 commande)  
Créez le répertoire **exo1** dans le répertoire **pratiques**
3. Changez de répertoire afin que vous soyez dans le répertoire `~/pratiques/exo1` (`=` Votre répertoire personnel)
  - Copiez le fichier `/etc/passwd` dans le répertoire `~/pratiques/exo1`, utilisez le chemin absolu pour réaliser l'opération
  - Copiez le fichier `/etc/group` dans le répertoire `~/pratiques/exo1`; utilisez le chemin relatif pour réaliser l'opération.
4. Listez les fichiers cachés de votre répertoire personnel
5. Affichez le contenu du fichier `/etc/resolv.conf`. Ce fichier contient l'information DNS ; quel est le search domain ?
6. Affichez les 5 premières lignes du contenu du fichier `/var/log/boot.log`
7. Affichez les 5 dernières lignes de `/etc/shadow` ; est-ce que ceci fonctionne ? Avez-vous une idée de la solution ?
8. Affichez le fichier `/etc/passwd` avec le numéro de ligne. Sur quel numéro de ligne se trouve le mot **games** ?
9. Copiez le répertoire `/etc/NetworkManager` dans `~/pratiques/exo1`. ATTENTION ! Faîtes en sorte que la structure des répertoires parent soit recréée.
10. Supprimez le répertoire `~/pratiques/exo1/etc/NetworkManager/system-connections`

11. Créez un nouveau fichier dans le répertoire `~/pratiques/exo1/` avec le nom **mon\_fichier**. Écrivez quelques lignes à l'intérieur ; copiez/collez le texte de l'exercice, puis sauvegardez-le.
12. Copiez le fichier `~/.bash_history` dans le répertoire `~/pratiques/exo1/`. Nommez le fichier **bash\_history-partie1**

# Chapitre 7

## COMMANDES DE BASE 2

Quelques commandes essentielles (suite) :

- Exécuter de commandes sous l'identité de l'administrateur.
- Trouver un fichier.
- Compresser / décompresser et archiver.
- Modifier le mot de passe (`passwd`).
- Obtenir de l'aide directement depuis la ligne de commande

Il est possible d'avoir beaucoup d'informations directement du système. Nous verrons :

- comment obtenir de l'information sur une commande, ainsi que les options disponibles
- rechercher dans les manuels disponibles
- les sites web de ressources à votre service.

### 7.1 OBJECTIFS

- Exécution de commandes sous l'identité de l'administrateur ( `sudo` / `su` )
- Localisation d'un fichier Recherche dans l'arborescence ( `find` )  
Recherche avec la base de donnée `updatedb` ( `locate` )
- Archivage et extraction de données ( `tar` / `gzip` / `bz2` )  
Tar pour l'archivage  
gzip et bz2 pour la compression
- Modification de mot de passe ( `passwd` )

### EXÉCUTION DE COMMANDES SUDO / SU

Suite à la dernière séance, vous avez peut-être voulu éditer un fichier dans le répertoire `/etc/`, pour y modifier un paramètre, ou désirer expérimenter des commandes trouvées sur internet. Malheureusement, toutes furent des échecs, car le système indique un problème de permission.

Deux méthodes existent pour exécuter des commandes sous l'utilisateur administrateur (`root`); il y a la bonne et la mauvaise! Commençons par la bonne; si vous venez d'installer votre système **Ubuntu**, votre premier utilisateur a le droit d'utiliser la commande `sudo` (`SwitchUserDO`). En

plus d'avoir les permissions d'exécuter cette commande, il peut tout faire avec cette dernière ; **sudo** permet d'attribuer des droits de manière granulaire à un utilisateur. Nous verrons en détail cet aspect dans la section administration du système.

Pour visualiser les permissions que nous avons :

```
# affiche les permissions de l'utilisateur Alex.
```

```
alex@hostname :~$ sudo -l
```

- Matching Defaults entries for alex on this host :
  - env\_reset,
  - secure\_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin
- User alex may run the following commands on this host :
  - (ALL : ALL) ALL

# Alex dans la situation peut exécuter n'importe quelle commande sur le système sous l'utilisateur root ou autre.

Donc, si nous désirons modifier un fichier dans le répertoire **/etc/** avec la commande **nano**, nous utiliserons la commande suivante :

```
alex@hostname :~$ sudo nano /etc/Le_fichier
```

C'est une bonne pratique d'utiliser la commande **sudo**, car :

- l'ensemble des commandes sont cataloguées ; il est donc possible de savoir quelle commande a été exécutée, à quelle moment et par qui.
- permet de limiter les commandes disponibles pour une personne, afin de protéger l'utilisateur et le système d'une erreur.

L'autre méthode est de Switcher d'utilisateur, de passer complètement en **root**. Le problème avec ceci est que nous perdons toutes traces des opérations réalisées et il n'y a plus de suivi si une erreur de manipulation est réalisée. C'est l'équivalent à se connecter comme Administrateur sur une machine plutôt que d'utiliser la fonction « Exécuter en tant que ... » ("RunAs") sous Microsoft Windows. Bien entendu, dans certaines situations, nous n'avons pas le choix. C'est parfois le cas lors de l'installation de logiciels en dehors des packages de la distribution.

Voici la **mauvaise méthode** : de passer sous l'utilisateur **root** :

```
alex@hotname :~$ su
```

## LOCALISER UN FICHIER

Bon voilà, vous êtes content, votre système fonctionne bien. Vous voulez aller plus loin dans votre recherche ou vous avez un problème ; vous recherchez donc sur internet ou demandez à un ami comment réaliser une opération. Ce dernier, content de vous répondre, rétorque simplement quelque chose du genre : " Ha, c'est simple ! ou bien « change le paramètre Xy » dans le fichier smb.conf pour la valeur « true ». Bon super vous notez le tout et, arrivé devant la machine, vous constatez que vous ne savez pas où se trouve le fichier smb.conf. Trop habitué, votre collègue a oublié de fournir le chemin absolu. Voici comment trouver votre fichier :

## RECHERCHE DANS L'ARBORESCENCE (FIND)

La commande **find**, comme son nom, l'indique permet de rechercher dans le système un fichier. Bien entendu, une recherche peut être longue s'il y a beaucoup de fichiers à traiter. **find** nous permet de rechercher selon plusieurs critères : le nom du fichier, sa date de modification, son propriétaire, ses permissions, sa taille, etc.

La structure de la commande est : **find** [Répertoire\_de\_recherche] [Pattern]. Si aucun répertoire de recherche n'est fourni, le système commence la recherche depuis le répertoire courant .

Si nous reprenons mon exemple mentionné plus tôt, nous aurons la commande suivante :

# comme le fichier est un .conf, je fais une recherche dans le répertoire /etc contenant les fichiers de configuration du système

```
utilisateur@hostname :~$ find /etc/ -name "smb.conf"
```

# Si je ne trouve pas, je vais faire une recherche plus générale, à partir de la racine.

```
utilisateur@hostname :~$ find / -name "smb.conf"
```

# Si je ne trouve toujours pas, je vais élargir ma recherche , en présumant que le fichier commence par smb et fini par .conf

```
utilisateur@hostname :~$ find / -name "smb*.conf"
```

# voici un exemple du résultat dans la vie réelle

```
utilisateur@hostname :~$ find /etc -name "smb.conf"
```

find : '/etc/lvm/backup' : Permission denied

find : '/etc/lvm/cache' : Permission denied

find : '/etc/rsnapshot/keys' : Permission denied

find : '/etc/xdg/menus/applications-merged' : Permission denied

find : '/etc/openvpn/ytriaV2' : Permission denied

find : '/etc/ssl/private' : Permission denied

/etc/samba/smb.conf

La dernière commande retourne des erreurs, car je ne suis pas administrateur du système. Cependant, à la fin, j'ai trouvé le fichier, qui se trouve dans /etc/samba .

## PRINCIPALES OPTIONS DE LA COMMANDE **find**.

Option	Signification
-name	Recherche par nom de fichier.
-type	Recherche par type de fichier.
-user	Recherche par propriétaire.
-group	Recherche par appartenance à un groupe.
-size	Recherche par taille de fichier.
-atime	Recherche par date de dernier accès.
-mtime	Recherche par date de dernière modification.
-ctime	Recherche par date de création.
-perm	Recherche par autorisations d'accès.

## RECHERCHE AVEC LA BASE DE DONNÉES

Attention, le système que je vais décrire n'est pas disponible sur toutes les distributions ! Bon, **find** c'est bien, mais si j'ai un répertoire avec énormément de fichiers, la recherche peut être très longue... Heureusement, plusieurs distributions installent le système d'**updatedb** par défaut. Celui-ci exécute, toutes les nuits, la commande **find** à partir du Root ( / ) et stocke l'information dans une base de données (BD) pour un usage ultérieur.

Donc, si nous reprenons la recherche du fichier **user.conf**, ceci donne :

```
pat@amd-a10 :~$ locate user.conf
```

```
/etc/adduser.conf
/etc/deluser.conf
/etc/fonts/conf.d/50-user.conf
/etc/systemd/user.conf
/usr/share/adduser/adduser.conf
/usr/share/doc/adduser/examples/adduser.local.conf.examples/adduser.conf
/usr/share/fontconfig/conf.avail/50-user.conf
/usr/share/man/da/man5/adduser.conf.5.gz
/usr/share/man/da/man5/deluser.conf.5.gz
/usr/share/man/de/man5/adduser.conf.5.gz
/usr/share/man/de/man5/deluser.conf.5.gz
/usr/share/man/es/man5/adduser.conf.5.gz
```

...

Comme vous pouvez le constater, le résultat partiel fournit l'information pour le répertoire **/etc**, mais aussi pour **/usr** et **/var**. Ceci dans un temps record ! Attention, **locate** a quelques limitations :

Aucun résultat ne sera affiché pour les fichiers créés depuis la dernière mise à jour de la base de données. Recherche uniquement par nom de fichier. La mise à jour de la BD augmente la charge de travail de la machine. Le résultat peut être erroné si le système n'a pas mis à jour sa BD. Par exemple, si le système était éteint pendant la planification de synchronisation.

## 7.2 ARCHIVAGE ET EXTRACTION DE DONNÉES

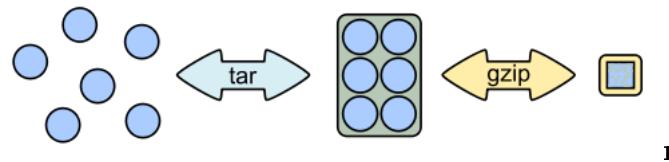
### TAR POUR L'ARCHIVAGE

Le système d'archivage de prédilection sous les systèmes Unix et les dérivés dont Linux est **tar** ( Tape ARchiver). tar est un outil très puissant, il ne compresse pas les fichiers, mais les concatène au sein d'un seul et même fichier ( archive ). La majorité des programmes linux utilisent ce système d'archivage.

On appelle parfois le fichier d'archivage créé un « **tarball** ». Tar préserve les droits, le propriétaire et le groupe des fichiers et des répertoires. Il permet également de sauvegarder les liens symboliques et les fichiers spéciaux orientés bloc ou caractère.

Pour l'essentiel, le format employé consiste en une concaténation du contenu des fichiers. Chaque fichier est précédé d'une en-tête de 512 octets. Cette taille correspond à la taille d'un bloc dans la version 7 du système de fichiers UNIX.

Pour améliorer l'efficacité de l'écriture sur bande magnétique, les blocs de 512 octets sont groupés par 20 par défaut, produisant des blocs de 10 Ko. Le bloc final est complété par des zéros binaires.



Souvent, un fichier créé par tar est ensuite compressé par un outil de compression de données. Les formats les plus courants sont :

Utilitaire de compression	extension Unix	extension MS-DOS
compress	.tar.Z	.taz
gzip	.tar.gz	.tgz
bzip2	.tar.bz2	.tbz
lzma	.tar.lz	.tlz
lzma2	.tar.xz	.txz
7zip	.tar.7z	

### COMMANDES COURANTES :

```
# Création de l'archive.tar dans le répertoire courant, à partir des fichiers de MonRepertoire1 et MonRepertoire2. Ne pas oublier de mettre l'extension tar à fichier afin de pouvoir identifier l'archive sinon le fichier est quand même créé mais par la suite, nous aurions des difficultés à reconnaître un fichier archive !
```

```
$ tar -cvf fichier.tar MonRepertoire1 [MonRepertoire2... ]  

# Affichage de la liste détaillée des fichiers de archive.tar.  

$ tar -tvf archive.tar  

# Extrait tous les fichiers contenus dans archive.tar.  

$ tar -xvf archive.tar  

# Il est possible d'extraire uniquement des fichiers d'une archive.
```

**\$ tar -xvf archive.tar fichier1 fichier 2**

##### Tar avec compression à l'aide de gzip #####

Pour utiliser **gzip**, on rajoute la lettre **z** après le **<->**

Ne pas oublier de mettre l'extension **tar.gz** à fichier afin de pouvoir identifier l'archive compressée.

# Réalisation d'une archive compressée avec gzip

**\$ tar -zcvf fichier.tar.gz MonRepertoire1 [MonRepertoire2... ]**

# Afficher l'archive compressée avec gzip \$

**tar -ztvf fichier.tar.gz**

# Extraire l'archive compressée avec gzip

**\$ tar -zxvf fichier.tar.gz**

##### Tar avec compression à l'aide de bzip2 #####

Pour utiliser **gzip**, on rajoute la lettre **j** après le **<->**

Ne pas oublier de mettre l'extension **tar.gz2** à fichier afin de pouvoir identifier l'archive compressée.

# Réalisation d'une archive compressée avec bzip2

**\$ tar -jcvf fichier.tar.gz2 MonRepertoire1 [MonRepertoire2... ]**

# Afficher l'archive compressée avec bzip2

**\$ tar -jtvf fichier.tar.gz2**

# Extraire l'archive compressée avec bzip2

**\$ tar -jxvf fichier.tar.bz2**

## GZIP ET BZ2 POUR LA COMPRESSION

Deux commandes sont disponibles pour la compression des fichiers : **gzip** et **bz2**.

**bz2** est plus performant et compresse environ 33% plus que **gzip**. Il est cependant plus gourmand en ressources. Si vous avez un très gros fichier, il peut être plus long de le décompresser avec **bz2**. Nous avons vu plus tôt qu'il est possible de le combiner avec la commande **tar**. Il est également utilisable seul. Voici quelques exemples :

```
# compression avec gzip :
```

```
utilisateur@hostname :~$ gzip le_fichier.txt
```

Résultat le fichier `le_fichier.txt` n'est plus présent mais il est remplacé par `le_fichier.txt.gz`

```
# décompression avec gzip :
```

```
utilisateur@hostname :~$ gunzip le_fichier.txt.gz
```

Résultat le fichier archive `le_fichier.txt.gz` n'est plus présent mais il est remplacé par `le_fichier.txt`

```
# compression avec bz2
```

```
utilisateur@hostname :~$ bzip2 le_fichier.txt
```

Résultat le fichier `le_fichier.txt` n'est plus présent mais il est remplacé par `le_fichier.txt.bz2`

```
# décompression avec bzip2 :
```

```
utilisateur@hostname :~$ bunzip2 le_fichier.txt.bz2
```

Résultat le fichier archive `le_fichier.txt.bz2` n'est plus présent mais il est remplacé par `le_fichier.txt`

## TRUCS ET ASTUICES

Si vous avez un fichier texte qui est compressé (un fichier `.gz`) et que vous désirez le visualiser sans le décompresser (pour une raison que j'ignore), il existe les commandes qui décompressent "live".

Les commandes `zcat` `zmore` ou `zless` qui offrent les mêmes interfaces de lecture/recherche que leurs équivalents `cat`, `more` ou `less`.

- `$ zcat fichier.txt.gz`
- `$ zmore fichier.txt.gz`
- `$ zless fichier.txt.gz`

C'est particulièrement intéressant quand nous le combinons avec `less` / `more` ou un `grep`; en d'autres mots, avec un Pipe (`|`).

```
# Visualisation du fichier gzip et bz2
```

```
utilisateur@hostname :~$ bzcat le_fichier.txt.bz2
```

```
utilisateur@hostname :~$ zcat le_fichier.txt.gz
```

```
# telle que mentionnée, utilisée avec un pipe
```

```
utilisateur@hostname :~$ zcat le_fichier.txt.gz | grep "mot"
```

## CHANGEMENT DE MOT DE PASSE

La commande **passwd** modifie le mot de passe du compte des utilisateurs. Un utilisateur normal peut uniquement modifier le mot de passe de son compte. Le Super-utilisateur a le pouvoir de changer le mot de passe de chaque compte.

**passwd** permet aussi d'associer une période de validité aux comptes et mots de passe, et de les modifier. En plus de permettre le changement de mot de passe, **passwd** permet de verrouiller un compte, de définir une période d'expiration ...

```
# l'utilisateur peut changer son propre password
utilisateur@hostname :~$ passwd
enter new UNIX password :
retype new UNIX password :
passwd : password updated successfully

# L'administrateur (root) peut manipuler les comptes des autres utilisateurs

# ici changement du mot de passe de thomas
root@hostname :~# passwd thomas

# verrouillage du compte de joe
root@hostname :~# passwd -l joe
```

### OPTIONS INTÉRESSANTES :

- **-l, --lock** : verrouille le compte spécifié. Cette option désactive complètement le mot de passe en le préfixant d'un '!' (aucune valeur cryptée selon le mécanisme utilisé par passwd ne peut donc y correspondre).
- **-u, --unlock** : déverrouille le mot de passe du compte indiqué. Cette option réactive le mot de passe en le positionnant à la valeur qui existait avant l'utilisation de l'option -l. -w,
- **-n, --mindays MIN\_JOURS** : définit le nombre minimum de jours entre chaque changement de mot de passe à MIN\_JOURS. Une valeur égale à zéro dans ce champ indique que l'utilisateur peut changer son mot de passe lorsqu'il le souhaite.
- **--warndays DUREE\_AVERTISSEMENT** : fixe le nombre de jours avant que le changement de mot de passe ne soit obligatoire. DUREE\_AVERTISSEMENT est le nombre de jours précédant la fin de la validité du mot de passe et durant lesquels l'utilisateur sera averti que son mot de passe est sur le point d'arriver en fin de validité.

## 7.3 VERSION VIDÉO

Télécharger le fichier : commandes\_de\_base2.ogv





# Chapitre 8

## AIDE DEPUIS LA LIGNE DE COMMANDE

Il est possible de récolter beaucoup d'informations directement du système. Nous verrons :

- comment obtenir de l'information sur une commande, ainsi que les options disponibles recherchées dans les manuels disponibles, les sites web de ressources à votre service.

### 8.1 AIDE DEPUIS LA LIGNE DE COMMANDE

- La Commande MAN (Manuel)
- Trouver une commande avec MAN
- Site web de documentation

#### LA COMMANDE MAN (MANUEL)

Chaque commande dispose d'une page de manuel en ligne (appelée manpages). Le manuel est une source d'informations importante sous GNU/Linux. Par défaut, les manuels sont en anglais. Mais on peut aussi les avoir en français : il suffit d'installer les paquets correspondants avec la commande (sous root) :

**apt-get install manpages-fr manpages-fr-extra**

manpages-fr : version française des pages de manuel sur l'utilisation de GNU/Linux

manpages-fr-extra : version française des pages de manuel des programmes

Cette aide en ligne est très utile pour savoir comment utiliser les commandes et connaître la liste exhaustive de toutes les options disponibles.

Pour accéder à cette aide en ligne, il suffit de taper dans un terminal man <la commande>. Exemple : man nom-de-la page

pat@amd-a10 :~\$ **man ls**

Par exemple, pour obtenir le manuel de la commande man, on lance la commande :

```
pat@amd-a10 :~$ man man
```

Les pages de manuel sont réparties en sections :

1. Programmes exécutables ou commandes de l'interpréteur de commandes (shell)
2. Appels système (Fonctions fournies par le noyau)
3. Appels de bibliothèque (fonctions fournies par les bibliothèques des programmes)
4. Fichiers spéciaux (situés généralement dans /dev)
5. Formats des fichiers et conventions. Par exemple /etc/passwd
6. Jeux
7. Divers (y compris les macro paquets et les conventions). Par exemple, man(7), groff(7)
8. Commandes de gestion du système (généralement réservées au super utilisateur)
9. Sous-programmes du noyau [hors standard]

Il arrive (rarement) que deux pages de manuel aient le même nom mais soient dans des sections différentes ; c'est le cas de man (1) et man(7) ou de printf(1) et printf(3) par exemple. Il est donc possible de spécifier dans quelle section chercher la page de manuel, en indiquant son numéro juste avant le nom de la page ou en spécifiant le paramètre **-s** (voir ci-dessous). Par exemple, pour obtenir la page de manuel de man(7) (qui parle de la syntaxe des pages de manuel), on tapera :

**man 7 man**

Chaque section possède de plus une page appelée **intro** qui présente la section, accessible comme les autres pages de manuel. Pour lire l'introduction de la section 3, il suffit donc de saisir :

**man 3 intro**

Certaines commandes sont à la fois des commandes systèmes, des appels systèmes ou des fichiers de configuration (exemple : passwd). Il est possible d'indiquer la section que l'on désire consulter :

```
pat@amd-a10 :~$ man passwd
```

```
pat@amd-a10 :~$ man 5 passwd
```

Les pages de man sont découpées en différentes rubriques (extrait de man 7 man)

Les rubriques des pages de man

RUBRIQUES	Descriptif
SYNOPSIS	Indique brièvement l'interface de la commande ou de la fonction. Pour les commandes, ce paragraphe montre sa syntaxe et ses arguments. Les caractères gras marquent le texte invariable et l'italique indique les arguments remplaçables. Les crochets encadrent les arguments optionnels, les barres verticales (caractère pipe) séparent les alternatives, et les ellipses ... signalent les répétitions. Pour les fonctions, on trouve toutes les déclarations et directives #include, suivies de la déclaration de fonction.
DESCRIPTION	Fournit une explication sur ce que la commande, la fonction ou le format représenté. Décrit les interactions avec les fichiers et l'entrée standard, ou ce qui est produit sur la sortie standard ou d'erreur. Ne contient pas les détails d'implémentation interne, sauf s'ils sont critiques pour comprendre l'interface. Décrit le cas principal, pour les détails sur les options, on utilise le paragraphe OPTIONS. S'il y a une sorte de grammaire d'entrée, ou un jeu de sous-commandes, on peut les placer dans une section UTILISATION supplémentaire (et placer un bref aperçu dans la section DESCRIPTION)
RETURN VALUE (VALEUR RENVOYÉE)	Donne une liste des valeurs qu'une routine de bibliothèque renverra à l'appelant et les conditions qui provoquent ces retours.
EXIT STATUS (CODE DE RETOUR)	Indique les codes de retour d'un programme et les conditions associées.
OPTIONS	Décrit les options acceptées par le programme et leur influence sur son comportement.
USAGE (UTILISATION)	Décrit la grammaire de tout sous-langage implémenté.
EXAMPLES (EXEMPLES)	Donne un ou plusieurs exemples d'utilisation de la fonction, du fichier ou de la commande.
FILES (FICHIERS)	Liste les fichiers utilisés par le programme ou la fonction, tels que les fichiers de configuration, de démarrage, et les fichiers manipulés directement par le programme. Il faut donner le chemin d'accès complet des fichiers et utiliser le mécanisme d'installation pour modifier le préfixe. Pour la plupart des programmes, l'installation par défaut se fait dans /usr/local, aussi, votre page de manuel de base devrait utiliser /usr/local comme base.
ENVIRONMENT (ENVIRONNEMENT)	Décrit toutes les variables d'environnement qui affectent le programme ou la fonction, ainsi que leurs effets.
DIAGNOSTICS (DIAGNOSTIQUE)	Fournit un survol des messages d'erreurs usuels et comment les considérer. Il n'est pas nécessaire d'indiquer les messages d'erreur système ou les signaux fatals qui peuvent apparaître durant l'exécution du programme, sauf s'ils sont traités spécialement.
SECURITY (SECURITÉ)	Décrit les problèmes de sécurité et leurs implications. Doit contenir les avertissements à propos des configurations ou des environnements à éviter, les commandes ayant des répercussions au niveau sécurité, etc. surtout s'ils ne sont pas évidents. Il n'est pas obligatoire de faire un paragraphe spécifique sur la sécurité. Si l'intelligibilité est améliorée, on peut placer ces informations dans les autres sections (telles que DESCRIPTION ou USAGE (UTILISATION)). Néanmoins, il est important de placer les informations de sécurité quelque part.
CONFORMING TO (CONFORMITÉ)	Décrit les standards ou les conventions suivis par l'implémentation.
NOTES	Contient des notes diverses.
BUGS (BOGUES)	Liste les limitations ou les défauts recensés, ainsi que les sujets à débat.
AUTHOR (AUTEUR)	Liste les auteurs de la documentation ou du programme afin de pouvoir leur envoyer les rapports de bogues.
SEE ALSO (VOIR AUSSI)	Fournit une liste des pages de manuel ayant un rapport, dans l'ordre alphabétique, suivie des autres documents éventuels. Il s'agit d'habitude de la dernière section.

## 8.2 TROUVER UNE COMMANDE AVEC MAN

Vous allez peut-être me dire : "c'est bien beau la commande **man** pour avoir la documentation, mais si je ne connais pas la commande à utiliser, je fais comment ?" L'option **-k** vous permet de faire une recherche dans l'ensemble des pages disponibles. Voici un exemple de résultat pour le mot **password** :

```
utilisateur@hostname :~$ man -k password
```

1. chage (1) - change user password expiry information
2. chgpasswd (8) - update group passwords in batch mode
3. chpasswd (8) - update passwords in batch mode
4. cpgr (8) - copy with locking the given file to the password or gr...
5. cppw (8) - copy with locking the given file to the password or gr...
6. crypt (3) - password and data encryption
7. crypt\_r (3) - password and data encryption
8. endpwent (3) - get password file entry
9. endspent (3) - get shadow password file entry
10. expiry (1) - check and enforce password expiration policy
11. fgetpwent (3) - get password file entry
12. fgetspent (3) - get shadow password file entry
13. fgetspent\_r (3) - get shadow password file entry

Comme vous pouvez le constater, le numéro entre parenthèses est le numéro du manuel , par exemple (1) équivaut au manuel "Programmes exécutables".

Suite à cette liste, on pourrait rechercher par exemple l'information en exécutant la commande :

```
pat@amd-a10 :~$ man 1 chage
```

A lire **man intro**

## 8.3 WHATIS - APROPOS

### ALTERNATIVES POUR RECHERCHER UNE PAGE DE MANUEL

Au lieu d'utiliser l'option **-k** de **man**, nous pouvons utiliser deux autres commandes qui s'appuient sur **man**.

- la commande **whatis** recherche sur les noms de page
- la commande **apropos** recherche sur les noms et les descriptions,

Ces utilitaires fournis avec **man** interviennent pour permettre d'effectuer rapidement une recherche à l'aide d'un mot clé, avec ou sans joker, ou bien à l'aide d'une expression rationnelle. Le comportement par défaut de **whatis** est d'utiliser la recherche par mot-clé sans joker, et celui de **apropos** est d'utiliser les expressions rationnelles.

Exemples d'utilisation :

```
vous@machine :~$ whatis whatis
```

whatis (1) - Afficher une ligne de description des pages de manuel, en fait c'est la rubrique nom de la page **man** associée.

```
vous@machine :~$ apropos apropos
```

apropos (1) - cherche le nom et la description des pages de manuel, la commande s'appuie sur la rubrique nom et description de la page man associée.

Vous pouvez consulter les pages de manuel de **whatis** et d' **apropos** pour de plus amples explications sur leurs options.

## OPTIONS UTILES

**-L** locale : permet de spécifier la **locale** pour laquelle afficher la page de manuel. Cette **locale** est par défaut celle du système. Par exemple, si les pages de **man** en français ont été préalablement installés par défaut elles s'affichent mais si l'on veut consulter la page de manuel de **man** en anglais :

**man -L en man**

**INTERACTIVITÉ** Lorsque une page de manuel est affichée, diverses actions sont accessibles via des raccourcis claviers dont voici un court extrait

Raccourcis	Action
Actions flèches directionnelles	Navigation dans la page de manuel
q	Quitte
h	Affiche l'aide
/	Rechercher en avant
?	? Comme /, mais recherche en arrière
n	Va à l'occurrence suivante de la recherche
N	Va à l'occurrence précédente de la recherche

## MAN EN COULEUR

Par défaut, **man** utilise le programme **less** pour afficher les pages. **less** ne gère pas la couleur, on peut y palier en utilisant un autre "pager", **most**. Pour cela :

Installer most :

```
$sudo apt-get install most
```

puis

```
sudo update-alternatives --config pager
```

et choisir le numéro de la ligne contenant **most**

## SITE WEB DE DOCUMENTATION

Site en Français :

- <http://doc.ubuntu-fr.org>
- <http://www.man-linux-magique.net/>
- <http://jp.barralis.com/linux-man/>

Site en Anglais :

- <https://help.ubuntu.com/13.10/ubuntu-help/index.html>
- <http://stackoverflow.com/>
- <http://www.omgubuntu.co.uk/> : site de news
- <http://www.webupd8.org/> : site de news

## 8.4 INFO

Dans le projet GNU, la documentation n'est pas au format du man, mais dans un format plus récent permettant de produire indifféremment la documentation en ligne et des manuels papiers, et comportant des renvois d'une section vers une autre : le format textinfo.

La commande **/usr/bin/info** permet de consulter et d'imprimer cette documentation qui se trouve dans le répertoire `/usr/share/info`. Une grande partie des logiciels disponibles dans une distribution Linux étant d'origine GNU, la commande info sera de fait préférée à la commande man, d'autant que la commande info renvoie automatiquement sur le « man » au cas où la documentation n'existe qu'au format man.

Vous pouvez accéder à cette documentation de 2 manières :

- en démarrant au sommaire général de la documentation
- `$ info`
- en consultant la documentation sur un point particulier
- `$ info bash`

Un tutoriel sur la commande **info** est obtenu par la commande : **info info**.

Le programme info est comme un grand livre comportant :

- un ensemble de pages hiérarchisées en hypertexte
- réparties en plusieurs niveaux ou « noeuds »
- rubriques
- du plus général au plus particulier
- liens marqués d'un astérisque (\*)

exemple :

La commande pat@amd-a10 :~\$ **info**

nous amène à la racine des pages, nous disposons alors d'un dispositif comme un sommaire où nous pouvons naviguer. Grâce à la tabulation (ou aux flèches du clavier) nous pouvons choisir la page voulue, et en validant par la touche « Entrée » nous suivons le lien.

Pour naviguer dans cette aide nous disposons de différents raccourcis clavier :

<b>q</b>	<b>Quitter</b>
<b>espace</b>	Défilement vers le bas
<b>return</b>	Défilement vers le haut
<b>b</b>	Début du noeud ( <b>beginning</b> )
<b>e</b>	Fin du noeud ( <b>end</b> )
<b>Tab</b>	Aller au lien suivant
<b>Entrée</b>	Suivre le lien
<b>n</b>	Nœud suivant ( <b>next</b> )
<b>p</b>	Nœud précédent
<b>u</b>	Nœud de niveau supérieur ( <b>up</b> )
<b>I (L min)</b>	Retour à la page précédemment affichée ( <b>last</b> )

si l'on veut accéder à une page particulière on peut la demander directement à l'aide de la commande info suivi du nom de la page

```
pat@amd-a10:~$ info cp
```

pour afficher la page « **cp** » et utiliser les raccourcis clavier pour naviguer. Si on appuie une 1ère fois sur **u**, on voit que **cp** dépend de Basic operations qui dépend lui même de coreutils.info (appui une 2eme fois sur **u**).

## 8.5 HELP

Dans la console, après l'invite de commande tapez **help** pour obtenir une aide sommaire des commandes du shell. Voici un fragment de la sortie de cette commande :

```
pat@amd-a10:~$ help
GNU bash, version 4.3.30(1)-release (x86_64-pc-linux-gnu)
Ces commandes de shell sont définies de manière interne. Saisissez « help » pour
voir cette liste.
Tapez « help nom » pour en savoir plus sur la fonction qui s'appelle « nom ».
Utilisez « info bash » pour en savoir plus sur le shell en général.
Utilisez « man -k » ou « info » pour en savoir plus sur les commandes qui
ne font pas partie de cette liste.

Une astérisque (*) à côté d'un nom signifie que la commande est désactivée.

job_spec [&]
(( expression ))
. nom_fichier [arguments]
:
[ arg... ]
[[ expression ]]
alias [-p] [nom[=valeur] ... ]
bg [job_spec ...]
bind [-lpsvPSVX] [-m keymap] [-f file]
break [n]
builtin [shell-builtin [arg ...]]
caller [expr]
case MOT in [MOTIF [| MOTIF]...) COMM>
history [-c] [-d décalage] [n] ou hi>
if COMMANDES; then COMMANDES; [ elif>
jobs [-lnprs] [jobspec ...] ou jobs >
kill [-s sigspec | -n signum | -sigs>
let arg [arg ...]
local [option] nom[=valeur] ...
logout [n]
mapfile [-n nombre] [-0 origine] [-s>
popd [-n] [+N | -N]
printf [-v var] format [arguments]
pushd [-n] [+N | -N | rép]
pwd [-LP]
read [-ers] [-a tableau] [-d delim] >
```

## 8.6 RÉSUMÉ

**help** est une commande intégrée dans le shell bash qui documente certaines des commandes et des mots clés de bash.

**man** est un système de documentation à l'échelle du système qui fournit de courts manuels de référence (pages) pour les commandes individuelles, les fonctions de l'API, les concepts, la syntaxe du fichier de configuration, formats de fichiers organisés en sections (1 pour les commandes de l'utilisateur, 2 pour les appels système ...). Voilà le système traditionnel de la documentation UNIX.

**info** est un autre système de documentation émanant du projet GNU. Il est **hypertexte** avec des liens (antérieur à web). Un manuel d'informations est comme un livre numérique avec un concept de table des matières et (consultable) indicé qui permet de trouver l'information.

Il y a chevauchement entre les 3 systèmes d'informations. Par exemple, bash faisant partie du projet GNU a à la fois une page de manuel et un manuel d'informations. La taille du manuel **man**

n'est pas très approprié pour bash. Toutefois, la structure du manuel d'informations n'est pas très bonne dans **bash** ce qui le rend pas aussi facile pour rechercher l'information.

Il convient de noter que le manuel info est généré à partir d'un format texinfo qui est également utilisé pour générer HTML et imprimables (PDF / PS).

## EXERCICES :

Exécuter la commande **man hier** et reportez vous au chapitre 2 Description des répertoires.

Utiliser la commande man pour comprendre la syntaxe des différentes commandes GNU.

Utiliser la commande **info** et les raccourcis pour se familiariser avec cette aide.

\*RTFM : Read The Fucking Manual (Lisez le putain de manuel) - couramment utilisé pour conseiller à un débutant de chercher la réponse à ses questions avant de solliciter de l'aide sur un forum ou un canal IRC.



# Chapitre 9

## ATELIER UTILISATION COMMANDES

### 9.1 UTILISATION DE LA COMMANDE FIND

- Utiliser la commande **find** recherche le fichier **update-notifier** dans le répertoire /etc, fournir le chemin absolu .
- Combien de fichiers, dans le répertoire /etc, contiennent la chaîne de caractère theme ?
- Lister les fichiers dans le répertoire /tmp qui appartiennent à l'utilisateur root ? (il peut y avoir des erreurs "permission denied" ceci est "normal")
- Lister les fichiers dans le répertoire /tmp qui appartiennent à votre utilisateur ?
- À l'aide de la commande **locate** rechercher les fichiers portant le nom "linux.h"
- Faire une recherche à l'aide de **find** dans le répertoire /usr pour les fichiers portant le nom "linux.h"

### 9.2 ARCHIVAGE ET DÉCOMPRESSION

- Créer le répertoire exo2 dans le répertoire ~/pratiques/ (répertoire pratique créé lors de l'exercice précédent)
- Réaliser l'archivage du répertoire /etc dans le répertoire ~/pratiques/exo2/. Donner le nom de fichier etc-bak.tar.gz (vous aurez des erreurs de permission car vous n'êtes pas root)
- Réaliser presque la même opération soit l'archivage du répertoire /etc dans le répertoire ~/pratiques/exo2/. Nommer le fichier etc-bak-root.tar.gz , mais réalisez la commande comme administrateur/root , l'équivalent du « run as »
- Télécharger le fichier **exo2.tar.gz**, copier le fichier dans le répertoire ~/pratiques/exo2/
- Désarchiver le fichier **exo2.tar.gz** dans le répertoire
- Télécharger le fichier **unix-hoax.txt.gz**, copier le dans le répertoire ~/pratiques/exo2/
- Lire le fichier **unix-hoax.txt.gz**, 2 possibilités : soit vous décompressez le fichier ou le lisez dans le décompressé...

## 9.3 MANUEL

- Utiliser la commande **man** pour rechercher les commandes qui permettent de modifier un fichier ( utiliser la string "change file" pour la recherche), lister uniquement les commandes.
- Quelle est l'option à la commande **tar** pour ajouter les fichiers à la fin d'une archive ?
- Grâce à la commande **man** prendre connaissance de la commande script.

## 9.4 TRAVAUX PRATIQUES

**ls** /etc cd # pour être dans mon répertoire personnel

**mkdir -p** pratiques/exo1

**cp** /etc/passwd /home/pat/pratiques/exo1/

**cp** /etc/group ./pratiques/exo1

**ls -a**

**cat** /etc/resolv.conf

**head -5** /var/log/boot.log

# pas de boot.log sur ma machine, ce fichier de toutes façons appartient à root ! Permission non accordée mais **sudo head -5 /var/log/daemon.log** fonctionne.

**tail -5 /etc/shadow** #impossible d'ouvrir « /etc/shadow » en lecture : Permission non accordée, ce fichier de toutes façons appartient à root !!

**sudo tail -5 /etc/shadow** fonctionne

**cat -n /etc/passwd**

**cat -n /etc/passwd | grep games**

# Chapitre 10

## TRUCS ET ASTUCES AVEC BASH

### 10.1 TRUCS ET ASTUCES AVEC BASH

Utiliser Bash de manière optimale est primordial, car nous tapons énormément de texte et il est toujours ennuyeux de retaper toujours la même chose ou de perdre du temps.

- Raccourcis clavier
- Ceinture jaune de bash
- Pour les ninjas

### 10.2 RACCOUCRIS CLAVIER

- TABulation : permet la complétion de nom de fichier et de commande, réalisée en appuyant sur la touche TABULATION. S'il y a plusieurs possibilités, bash vous les signalera.
- CTRL+ a : permet de se déplacer au début de la ligne
- CTRL+ e : permet de se déplacer à la fin de la ligne
- CTRL+ c : annule la ligne et retourne à une ligne vierge (annulée de l'historique)
- CTRL+ l : permet d'effacer l'écran (L minuscule)
- ALT+ . ou ALT+ \_ : permet d'ajouter à la commande courante l'argument de la commande précédente. Si l'on réutilise cette combinaison de touches, le système remontera l'historique des commandes.
- CTRL+R : permet de faire une recherche dans l'historique de commandes. Faire CTRL+R, commencer à taper la commande ; si elle est dans l'historique, elle s'affichera dans la console. En réutilisant la tâche CTRL+R, nous pouvons remonter dans l'historique.
- # : en début de ligne, annule l'exécution de la ligne de commande (contrairement à CTRL+C, la commande sera disponible dans l'historique)

### 10.3 CEINTURE JAUNE DE BASH :

- La commande « **history** » : les commandes que vous verrez seront alors uniquement celles de l'utilisateur avec lequel vous êtes connecté, cette commande affiche les dernières commandes effectuées, précédées d'un nombre.

pour rappeler une commande taper :!suivi du nombre

par exemple : john@machine~\$ !8

pour effacer l'historique des commandes :taper : john@machine~\$ **history -c**

**!!** : permet de rappeler une commande complète en utilisant **!!**. Exemple :

# exécute la commande **ls /tmp**

utilisateur@hostname :~\$ **ls /tmp**

Fichier1 fichier\_tmp pulse-PKdhtXMmr18n

# Ré-exécute la commande mais avec **!!**

utilisateur@hostname :~ \$ **!!**

Fichier1 fichier\_tmp pulse-PKdhtXMmr18n

**!c** : ré-exécute la première commande de l'historique qui commence par c

**!\$** : équivaut à ALT+. ; récupère le dernier argument de la commande précédente

#### Pour les « ninjas » :

Ici on rentre dans une autre catégorie de personnes, les « aficionados ». C'est surtout pour le plaisir que le paragraphe est écrit.

**^foo^bar** : permet de faire du **search and replace** de la dernière commande. Par exemple, pour être plus clair :

# liste le fichier /etc/passwd mais je réalise une erreur, je tape password au lieu de passwd.

ninja@hostname :~\$ ls /etc/password

ls : cannot access /etc/password : No such file or directory

# Correctif de l'erreur sur la ligne suivante :

ninja@hostname :~\$ ^word^wd

ls /etc/passwd

/etc/passwd

{,s} : liste des arguments optionnels ou multiples choix possibles. Voici un exemple pour illustrer :

# liste le répertoire /usr/bin ET /usr/sbin :

utilisateur@hostname :~\$ ls -ld /usr/{,s}bin/

```
drwxr-xr-x 2 root root 53248 Dec 6 07 :58 /usr/bin  
drwxr-xr-x 2 root root 12288 Dec 6 06 :00 /usr/sbin
```

# Permet aussi de fournir une liste :

```
utilisateur@hostname :~$ du -hsc /usr/{games,lib}/  
392K /usr/games/  
1.6G /usr/lib/  
1.6G total
```

## 10.4 VERSION VIDÉO

télécharger le fichier : truc\_astuce\_bash-installation-logiciel.ogv et le lire avec vlc



# Chapitre 11

## PRÉSENTATION DU SYSTÈME PACKAGES

- Concept de l'installation de logiciel sous GNU/Linux
- Définition d'un package
  - Gestion des dépendances
  - Script d'installation et de désinstallation
  - Les packages et Internet

### 11.1 INSTALLATION DE LOGICIEL

Voici comment le concept d'installation de logiciel sous GNU/Linux fonctionne : premièrement, un grand nombre de logiciels sont fournis / disponibles par la distribution GNU/Linux que vous avez choisie. La majorité des distributions offrent une gamme de logiciels pour votre usage et, 95% du temps, vous trouverez le logiciel dans la liste disponible avec le gestionnaire de packages.

Si vous ne trouvez pas le logiciel ou que la version disponible n'est pas celle désirée, il existe des dépôts (repository) qui vous permettent d'étendre le choix de logiciels disponibles. Pour se faire, vous configurerez votre gestionnaire de packages afin qu'il cherche les packages à ce nouvel endroit, en plus de la distribution. Encore une fois, vous avez avantage à avoir vos logiciels gérés par un gestionnaire de packages, vous assurant que les dépendances sont bien respectées et, à priori, les packages fonctionneront pour votre version de GNU/Linux. Bien entendu, il est important de faire attention aux « repository » ajoutés ; il faut s'assurer qu'ils ont une bonne réputation afin de ne pas installer des logiciels malveillants sur son Linux .

Dans la situation où personne ne fournit le logiciel, mais que ce dernier est libre, vous pouvez télécharger le logiciel et en faire l'installation manuellement en compilant le programme depuis les sources. Dans ce cas, il n'y a aucune gestion des dépendances ; il peut être ardu de faire l'installation de cette manière, surtout quand on débute...

Dans le cas où le logiciel est privatif, il faut utiliser l'installeur fourni par le programme et, bien entendu, télécharger le logiciel depuis le site officiel du produit.

J'aimerai refaire mention que 95% des logiciels sont installables avec les packages fournis par la distribution. Alors, avant de vous casser la tête, regardez s'il est disponible !

## 11.2 DÉFINITION D'UN PACKAGE

Un package est une archive contenant des données et/ou des programmes ainsi que les informations nécessaires à une installation correcte de ceux-ci sur le système. Un package est constitué d'un seul fichier, qui n'est pas un exécutable. Il est pris en charge par un programme dédié : le gestionnaire de package.

Il existe plusieurs types différents de package :

- [RPM] : pour Redhat Package Manager. C'est le type de package le plus utilisé sous Linux. Ces packages gèrent les dépendances et les scripts d'installation/désinstallation.
- [DEB] : le type de package utilisé par la distribution Debian et ses dérivés. Il gère les scripts et les dépendances de façon plus fine que RPM.
- [tar.gz] : utilisé par les distributions du type Slackware. Il s'agit d'une simple archive contenant également les scripts d'installation.
- [pkg] : cette extension peut désigner plusieurs types de package différents utilisés par des UNIX propriétaires, comme solaris ou QNX.

Il faut noter que les différentes saveurs libres de BSD utilisent un système différent, appelé ports, où les programmes sont systématiquement recompilés.

## 11.3 GESTION DES DÉPENDANCES

La gestion des dépendances est un des gros avantages des systèmes de packages.

Chaque package contient une liste des fonctionnalités qu'il fournit. Cela commence par le nom du package lui-même, cela peut également être un programme particulier, une bibliothèque générique ou dans une version plus particulière ou encore une fonctionnalité plus "diffuse", telle que "serveur ftp" pour apache, ou "gestionnaire de téléchargement" pour **wget**.

Ensuite, les packages vont contenir une liste des fonctionnalités qui leurs sont nécessaires pour être fonctionnels.

Lors de l'installation, le gestionnaire de package va vérifier que toutes les dépendances sont vérifiées et que l'installation du nouveau package ne va pas écraser des fichiers d'autres packages. Sinon, il refusera d'installer. Il reste possible de forcer l'installation à ses risques et périls.

De même, à la désinstallation d'un package, le gestionnaire de package vérifie que la suppression de ce(s) package(s) ne gène pas le fonctionnement d'autres packages. Une fois encore, on peut ne pas tenir compte des dépendances, mais cela peut s'avérer assez grave (forcer la désinstallation de la **glibc** est le parfait exemple de ce qu'il faut faire si on veut foutre en l'air son système).

## 11.4 SCRIPT D'INSTALLATION ET DE DÉSINSTALLATION

Si, pour installer une documentation, il est suffisant d'en décompresser les fichiers, il n'en va pas forcément de même pour des programmes, et encore moins pour des bibliothèques, dont l'installation nécessitera d'effectuer des opérations sur le système pour que le contenu du package soit pleinement fonctionnel.

C'est à cela que servent les scripts d'installation/désinstallation, qui se présentent sous la forme de simples scripts shell (au standard sh). Ils sont au nombre de quatre :

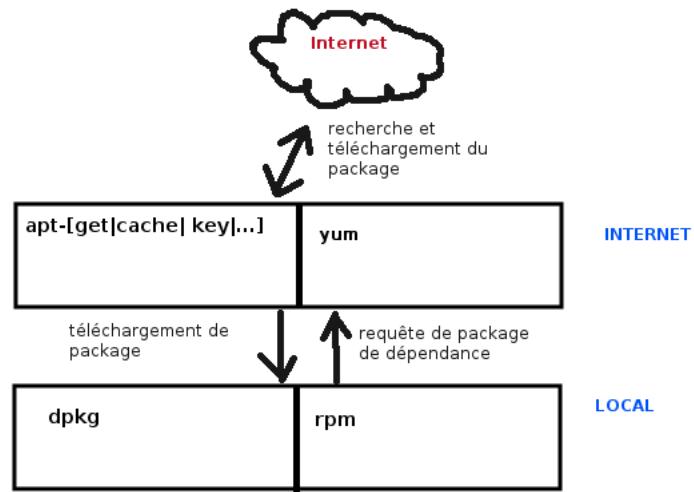
- [le script de pré-installation] : il effectue toutes les opérations préparatoires à l'installation.
- [le script de post-installation] : s'occupe de toute la configuration post-installation, par exemple enregistrer de nouvelles bibliothèques, ou générer une configuration en accord avec la machine sur laquelle il est installé.
- [le script de pré-désinstallation] : s'occupe des opérations à effectuer avant de désinstaller le package. Par exemple, s'il s'agit d'un serveur quelconque, il faut arrêter celui-ci avant de le désinstaller.
- [le script de post-désinstallation] : est chargé des opérations à effectuer après désinstallation.

## 11.5 LES PACKAGES ET INTERNET

Le système de package est indépendant d'Internet, comme chaque package contient la liste des dépendances dont il a besoin pour fonctionner, il n'a pas besoin d'une source externe. Chaque système de package conserve localement une Base de données, avec l'information des packages installés sur le système. Cette dernière est interrogée s'il y a besoin de valider des dépendances lors de l'installation.

- deb : /var/lib/dpkg : contient l'ensemble de l'information sur les packages présents sur le système et le logiciel, et gère des packages sous debian. Le système de package porte le nom de DPKG.
- rpm : /var/lib/rpm , contient l'ensemble de l'information sur les packages présent sur le système et le logiciel, et gère des packages sous RedHat. Le système de package porte le nom de RPM.

Le système de package indique la liste des dépendances à résoudre pour pouvoir installer le package. De nos jours, des outils sont nés qui permettent de rechercher directement sur Internet ces packages, de les télécharger et de fournir le fichier au gestionnaire de package.



Dans le prochain chapitre, nous allons voir l'utilisation de ses deux gestionnaires de packages.

## 11.6 VERSION VIDÉO

Télécharger le fichier : `truc_ astuce_ bash-installation-logiciel.ogv` et le lire avec vlc

# Chapitre 12

## SYSTÈME DE PACKAGE DEBIAN/UBUNTU

- Système de package Debian
  - Outils de gestion des packages
- Présentation de dpkg
  - Commande disponible
- Présentation des outils apt
  - Dépôt APT
  - Outil apt-cache
  - Outil apt-get
  - Outil aptitude

### 12.1 SYSTÈME DE PACKAGE DEBIAN

Les paquets contiennent généralement tous les fichiers nécessaires pour implémenter un ensemble de commandes ou de fonctionnalités. Il y a deux sortes de paquets Debian :

- Les paquets binaires contenant les exécutables, les fichiers de configuration, les pages de manuel ou d'info, les informations de copyright et d'autres documentations. Ces paquets sont distribués sous un format d'archives spécifiques à Debian. Ils sont habituellement reconnaissables par l'extension « .deb ». Ils peuvent être installés en utilisant l'utilitaire **dpkg** (éventuellement avec une interface comme **aptitude**) ; vous trouverez plus de détails dans les pages du manuel.
- Les paquets sources sont constitués d'un fichier .dsc décrivant le paquet source (incluant le nom des fichiers suivants), un fichier .orig.tar.gz contenant les sources originales non modifiées, au format tar compressé, et, habituellement, un fichier.diff.gz contenant les modifications spécifiques à Debian par rapport la source originale. L'utilitaire dpkg-source permet l'archivage et le désarchivage des sources Debian ; vous trouverez plus de détails dans les pages du manuel. (Le programme **apt-get** peut être utilisé comme une interface pour dpkg-source ).

Les outils de gestion de paquets Debian peuvent être utilisés pour :

- manipuler ou administrer les paquets ou une partie des paquets,
- administrer les modifications locales (« overrides ») des fichiers d'un paquet,
- aider les développeurs dans la construction de paquets et
- aider les utilisateurs dans l'installation de paquets résidant sur un serveur FTP distant.

Le nom des paquets binaires Debian se conforme à la convention suivante :

**<foo>\_<NuméroVersion>-<NuméroRévisionDebian>\_<DebianArchitecture>.deb**  
exemple : **coreutils \_8.23-4 \_amd64.deb**

## 12.2 OUTILS DE GESTION DES PACKAGES

"**dpkg**" permet de gérer l'installation, la vérification, la suppression de paquets déjà téléchargés mais ne peut pas télécharger lui même les paquets désirés. Il faut donc préalablement télécharger un paquet au format **.deb** puis l'installer avec **dpkg**.

**dpkg** – installation locale de paquets Debian

**apt-get** – frontal pour APT en ligne de commande

**aptitude** – frontal avancé pour APT en mode texte et ligne de commande

**synaptic** – frontal pour APT en mode graphique GTK

**dselect** – gestion des paquets à l'aide de menus

**tasksel** – installation de tâches

Ces outils ne sont pas tous des alternatives. Par exemple **dselect** utilise à la fois **apt** et **dpkg**.

**APT** utilise `/var/lib/apt/lists/*` pour suivre les paquets disponibles tandis que **dpkg** utilise `/var/lib/dpkg/available`.

Si vous avez installé des paquets directement en utilisant **aptitude** ou un autre frontal pour APT et que vous voulez utiliser **dselect** pour installer des paquets, assurez-vous de mettre à jour le fichier `/var/lib/dpkg/available` en sélectionnant [M]ise à jour dans le menu de **dselect** (ou en exécutant `dselect update`).

**apt-get** récupère automatiquement les paquets dont un paquet demandé dépend. Il n'installe pas les paquets recommandés ou suggérés par le paquet demandé.

**aptitude** au contraire peut être configurée pour installer les paquets recommandés ou suggérés.

**dselect** présente à l'utilisateur une liste de paquets qu'un paquet sélectionné, recommande ou suggère et permet de les sélectionner ou pas.

L'outil **dselect** est un frontend (comme APT) pour **dpkg**, qui gère les dépendances et les conflits. Historiquement, il est le premier. Cependant son remplaçant APT dispose d'une bien meilleure qualité. Le manuel de **dselect** indique clairement que l'outil est aujourd'hui tombé en désuétude. **dselect** ne devrait plus être utilisé.

## 12.3 PRÉSENTATION DE DPKG

**dpkg** (pour debian package) est un outil logiciel en ligne de commande chargé de l'installation, la création, la suppression et la gestion des paquets Debian (**.deb**), le type de paquets traités par Ubuntu et Debian. **Dpkg** fonctionne localement, c'est à dire qu'il **ne fait aucune recherche sur Internet**. Pour l'installation de paquets récupérés depuis internet utilisez l'outil **apt**. **dpkg** n'est pas destiné à être utilisé par l'utilisateur comme installateur de paquet sauf si ces derniers ont été récupérés manuellement . **Dpkg** dispose d'une interface graphique, **GDebi**, que vous pouvez utiliser si vous préférez éviter la ligne de commande.

À la différence de la commande **apt-get**, de la Logithèque, ou de l'interface graphique **GDebi**, **dpkg** est un outil qui ne gère pas les dépendances. Ainsi en cas de conflit ou bien lorsque seuls certains paquets impliquant trop de dépendances font défaut, l'utilisation de cet outil devient presque indispensable. **Synaptic** et d'autres gestionnaires de paquets utilisent justement cet outil pour résoudre certains problèmes caractéristiques. Il permet donc de « jouer » sur un seul paquet (installation, suppression, reconfiguration ) sans bouleverser les dépendances. Parmi ses autres fonctions **dpkg** permet aussi d'avoir des informations précises telles que l'état ou la description détaillée des paquets disponibles (installés ou non).

## COMMANDES DISPONIBLES

- L'option **-i**, ou **-install**, installe le ou les packages **préalablement téléchargés** passés comme argument  
 — utilisateur@hostname :~\$ **sudo dpkg -i** le\_package.deb ou **sudo dpkg -install** le\_package.deb
- Suppression de packages, suppression du binaire et suppression des fichiers de configuration  
 #suppression simple du package, du binaire (**remove**)  
 utilisateur@hostname :~\$ **sudo dpkg -r** le\_package
- # Suppression complète , binaire + fichier de configuration (**Purge**)  
 utilisateur@hostname :~\$ **sudo dpkg -P** le\_package
- Recherche **-S** (**-search**) le **paquet correspondant à un fichier** dans le système de packages.  
 utilisateur@hostname :~\$ **dpkg -S** /bin/cat  
**coreutils** : /bin/cat
- **-s** (**-status**) donne le status du paquet passé en argument  
 utilisateur@hostname :~\$ **dpkg -s** coreutils
- # Afficher la **Liste** (**-listfiles**) des fichiers installés contenus dans le paquet (pas besoin d'être administrateur)  
 utilisateur@hostname :~\$ **dpkg -L** le\_package
- # **-liste** (**-list**) un package particulier préalablement installé  
 utilisateur@hostname :~\$ **dpkg -l** le\_package  
 dpkg -l \*office\* : liste tous les paquets liés à libreoffice et openoffice

- lister l'ensemble des packages installés  
utilisateur@hostname :~\$ **dpkg -l**

Les deux premiers caractères à gauche vous donnent des indications sur l'état du paquet

**Première colonne** : souhait

- i : Install (à installer)
- r : Remove (à supprimer)
- u : Unknown (inconnu)
- p : Purge (à supprimer avec les fichiers de configuration)
- h : Hold (à conserver)

**Seconde colonne** : état

- i : Installed (installé)
- c : Config-files (fichier(s) de configuration existant)
- u : Unpacked (décompressé)
- n : Not Installed (non installé)
- f : Failed-config (problème de configuration)
- h : Half-installed (installé partiellement)

marc@marc-MS-7721 :~\$ dpkg -l | less

```
Souhaite=Inconnu/Installé/supprimé/Purgé/H=à garder
| État=Non/Installé/fichier-Config/dépaqueté/échec-confg/H=semi-installé/W=attend-traitement-déclenchements
|| Err=(aucune)/besoin Réinstallation (État,Err: majuscule=mauvais)
||/ Non Version Architecture Description
====-======
ii accountservice          0.6.35-0ubuntu7.2      amd64   query and manipulate user account information
ii acl                      2.2.52-1           amd64   Access control list utilities
ii acpi-support             0.142              amd64   scripts for handling many ACPI events
ii acpid                   1:2.0.21-1ubuntu2     amd64   Advanced Configuration and Power Interface event daemon
ii adduser                 3.113+nmu3ubuntu3    all     add and remove users and groups
ii aisleriot                1:3.10.2-1         amd64   GNOME solitaire card game collection
ii alacarte                 3.10.8-1ubuntu2     all     easy GNOME menu editing tool
ii alsabase                 1.0.25+dfsg-0ubuntu4  all     ALSA driver configuration files
ii alsu-utils               1.0.27.2-1ubuntu2     amd64   Utilities for configuring and using ALSA
ii anacron                  2.3-20ubuntu1       amd64   cron-like program that doesn't go by time
ii apg                      2.2.3.dfsg.1-2ubuntu1  amd64   Automated Password Generator - Standalone version
ii app-install-data          14.04.1            all     Ubuntu applications (data files)
```

[[output coupé]]

...

## 12.4 PRÉSENTATION DES OUTILS APT

Advanced Packaging Tool est un système complet et avancé de gestion de paquets, permettant une recherche facile et efficace, une installation simple et une désinstallation propre de logiciels et utilitaires. Il permet aussi de facilement tenir à jour votre distribution Ubuntu/Debian. A la différence de **dpkg**, **apt** résout automatiquement les dépendances entre paquets.

APT est un ensemble d'utilitaires utilisables en ligne de commande. Il dispose aussi de nombreuses interfaces graphiques (**Synaptic...**), et d'interfaces en ligne de commande, comme **apt-get** et Aptitude, afin d'en rendre l'utilisation plus aisée.

### DÉPÔT APT

Les dépôts APT sont des "sources de logiciels", concrètement des serveurs contiennent un ensemble de paquets. À l'aide d'un outil appelé gestionnaire de paquets, vous pouvez accéder à ces dépôts et, en quelques clics de souris, vous trouvez, téléchargez et installez les logiciels de votre choix.

Ubuntu, Debian... intègre aussi de base un outil nommé Gestionnaire de mises à jour, qui vérifie périodiquement dans les dépôts auxquels vous avez accès que vous disposez des dernières versions de vos logiciels et bibliothèques ; dans le cas contraire, il vous permet de les mettre à jour automatiquement.

Les dépôts auxquels Ubuntu accède par défaut, afin de vérifier les mises à jour logicielles et rechercher les logiciels à installer, sont les dépôts maintenus par la Fondation Ubuntu (le groupe s'occupant du développement d' Ubuntu) et de votre CD d'installation. Vous pouvez étendre (ou réduire) la liste des dépôts accessibles par votre système en ajoutant ou retirant des dépôts d'autres distributeurs. (voir : modifier les dépôts)

Sous Ubuntu, la grande majorité des applications sont disponibles dans les dépôts officiels et sont directement installables à l'aide d'outils graphiques comme La Logithèque Ubuntu.

Rien ne vous empêche d'installer des logiciels en provenance d'autres dépôts ou d'autres sites Web, mais soyez vigilants, car ces programmes ne sont pas testés par l'équipe de développement d' Ubuntu et peuvent donc être dangereux pour votre système, ou simplement mal s'intégrer à votre environnement, comporter des bogues...

### DÉPÔT OFFICIEL

L'accès aux dépôts officiels est configuré automatiquement. Ils regroupent des dépôts de base, des dépôts de mises à jour et de sécurité. Toutes les branches des dépôts principaux sont divisées en quatre sections :

- Sections Main et Restricted, maintenues par les développeurs d' Ubuntu. Les sections main (paquets tout à fait libres) et restricted (paquets non-libres) contiennent des paquets maintenus par les développeurs d' Ubuntu pour toute la durée de vie de la version d' Ubuntu que vous utilisez.
- Sections Universe et Multiverse, maintenues par la communauté. Les sections universe et multiverse des dépôts officiels contiennent des paquets maintenus par la communauté. La

Fondation Ubuntu ne contrôle pas ces paquets ; ils sont analysés par un comité d'utilisateurs. La section universe contient uniquement des paquets libres et la section multiverse, des paquets non-libres. L'accès à ces deux sections est paramétré par défaut.

## LA LISTE DES SOURCES

Dans ses fonctions, Apt utilise un fichier qui liste les « sources » à partir desquelles les paquets peuvent être obtenus. Ce fichier est **/etc/apt/sources.list**.

Les entrées de ce fichier suivent généralement ce format (les entrées de l'exemple sont fictives et ne doivent pas être utilisées) :

```
deb http://site.example.com/debian distribution component1 component2 component3 deb-src
http://site.example.com/debian distribution component1 component2 component3
```

### TYPE D'ARCHIVE

Le premier mot sur chaque ligne, deb ou deb-src, indique le type d'archive. Deb indique que l'archive contient des paquets binaires (deb) qui sont les paquets pré-compilés que nous utilisons généralement. Deb-src indique les paquets sources qui sont les programmes Linux originaux sources plus le fichier de contrôle de Debian (.dsc) et le diff.gz contenant les changements nécessaires pour l'empaquetage du programme.

### URL DES DÉPÔTS

L'entrée suivante sur la ligne est une URL vers le dépôt à partir duquel vous voulez télécharger les paquets. La principale liste de miroirs des dépôts Debian se trouve sur cette page.

### DISTRIBUTION DEBIAN

La « distribution » peut être soit le nom de code (c'est-à-dire wheezy, jessie, stretch, sid) ou le nom d'une catégorie de version (oldstable, stable, testing, unstable). Si vous voulez suivre une catégorie alors utilisez le nom de catégorie, si vous voulez suivre à la trace une version de Debian, utilisez le nom de code.

Par exemple, si vous avez un système fonctionnant avec Debian « Jessie » et ne voulez pas le mettre à jour quand Debian « Stretch » sortira, utilisez jessie à la place de stable. Si vous voulez sans cesse aider à tester la distribution « testing », utilisez testing. Si vous voulez suivre la version Debian « Stretch » et que vous voulez rester avec elle de sa phase « testing », puis comme version « stable » jusqu'à la fin de son cycle de vie, utilisez stretch.

### COMPOSANTS

La section main comprend l'ensemble des paquets qui se conforment aux DFSG - Directives Debian pour le logiciel libre et qui n'ont pas besoin de programmes en dehors de ce périmètre pour fonctionner. Ce sont les seuls paquets considérés comme faisant partie de la distribution Debian.

La section contrib comprend l'ensemble des paquets qui se conforment aux DFSG, mais qui ont des dépendances en dehors de main (qui peuvent être empaquetées pour Debian dans non-free).

La section non-free contient des logiciels qui ne se conforment pas aux DFSG.

Exemple : fichier sources.list de Debian 8 « Jessie »

```
deb http://httpredir.debian.org/debian jessie main deb-src http://httpredir.debian.org/debian
jessie main

deb http://httpredir.debian.org/debian jessie-updates main deb-src http://httpredir.debian.org/debian
jessie page sources.list(5) du manuelie-updates main

deb http://security.debian.org/jessie/updates main deb-src http://security.debian.org/jessie/updates
main
```

Si vous désirez disposer aussi des composants contrib et non-free, ajoutez contrib non-free après main.

Autrement, on peut utiliser un outil GNOME pour modifier le fichier sources.list. (Menu Système>Administrat Logiciel).

```
gksu --desktop /usr/share/applications/software-properties.desktop /usr/bin/software-properties-gtk
```

## CD-ROM

Si vous préférez utiliser un CD-ROM pour installer des paquets ou mettre à jour votre système automatiquement avec APT, vous pouvez le mettre dans votre /etc/apt/sources.list. Pour le faire, vous pouvez utiliser le programme apt-cdrom ainsi :

```
# apt-cdrom add
```

avec le CD-ROM Debian dans le lecteur.

Pour plus d'informations, consultez : man 5 sources.list

## OUTIL APT-CACHE

**apt-cache** est une interface permettant d'effectuer quelques manipulations basiques sur les paquets, installés ou non, disponibles dans la liste mise en cache des paquets des dépôts APT configurés. Il ne nécessite pas les droits d'administration.

### COMMANDES

- Recherche de package dans le cache  
utilisateur@hostname :~\$ **apt-cache search le\_logiciel**
- Affiche l'information détaillée du package  
utilisateur@hostname :~\$ **apt-cache showpkg le\_package**
- Affiche, liste les dépendances du package  
utilisateur@hostname :~\$ **apt-cache depends le\_package**

## OUTIL APT-GET

**Advanced Packaging Tool** est un système complet et avancé de gestion de paquets, permettant une recherche facile et efficace sur internet, une installation simple et une désinstallation propre de

logiciels et utilitaires. Il permet aussi de facilement tenir à jour votre distribution Ubuntu avec les paquets en versions les plus récentes et de passer à une nouvelle version de Ubuntu, debian lorsque celle-ci est disponible.

## COMMANDÉ

- Mise à jour du **cache local** contenant la liste des packages. Le système **apt-get** ne communique pas continuellement sur Internet , le système réalise un cache local de la liste des packages disponibles. Ceci a l'avantage de ne pas perdre du temps à chaque installation, le mauvais côté est qu'il faut mettre à jour cette liste manuellement, sinon lorsque vous essayerez d'installer un package le **apt-get** vous donnera l'erreur : "404 not found" du package désiré. Par exemple, sur une distribution Debian/Jessie, pour **mettre à jour la liste**, taper la commande suivante : **sudo apt-get update** met à jour le cache.

```
pat@amd-a10:~$ sudo apt-get update
Ign cdrom://[Debian GNU/Linux 8 _Jessie_ - Official Snapshot amd64 LIVE/INSTALL Binary 20150908-22:01] jessie InRelease
Ign cdrom://[Debian GNU/Linux 8 _Jessie_ - Official Snapshot amd64 LIVE/INSTALL Binary 20150908-22:01] jessie Release.gpg
Ign cdrom://[Debian GNU/Linux 8 _Jessie_ - Official Snapshot amd64 LIVE/INSTALL Binary 20150908-22:01] jessie Release
Ign cdrom://[Debian GNU/Linux 8 _Jessie_ - Official Snapshot amd64 LIVE/INSTALL Binary 20150908-22:01] jessie/main amd64 Packages/DiffIndex
Ign cdrom://[Debian GNU/Linux 8 _Jessie_ - Official Snapshot amd64 LIVE/INSTALL Binary 20150908-22:01] jessie/main Translation-fr_FR
Ign cdrom://[Debian GNU/Linux 8 _Jessie_ - Official Snapshot amd64 LIVE/INSTALL Binary 20150908-22:01] jessie/main Translation-fr
Ign cdrom://[Debian GNU/Linux 8 _Jessie_ - Official Snapshot amd64 LIVE/INSTALL Binary 20150908-22:01] jessie/main Translation-en
Réception de : 1 http://security.debian.org jessie/updates InRelease [63,1 kB]
Réception de : 2 http://security.debian.org jessie/updates/main Sources [101 kB]
Réception de : 3 http://security.debian.org jessie/updates/main amd64 Packages [148 kB]
Réception de : 4 http://security.debian.org jessie/updates/main Translation-en [82,5 kB]
Ign http://ftp.nerim.net jessie InRelease
Réception de : 5 http://ftp.nerim.net jessie-updates InRelease [135 kB]
Atteint http://ftp.nerim.net jessie Release.gpg
Réception de : 6 http://ftp.nerim.net jessie-updates/main Sources [2 296 B]
Réception de : 7 http://ftp.nerim.net jessie-updates/main amd64 Packages/DiffIndex [367 B]
Réception de : 8 http://ftp.nerim.net jessie-updates/main Translation-en [2 506 B]
Atteint http://ftp.nerim.net jessie Release
Atteint http://ftp.nerim.net jessie/main Sources
Atteint http://ftp.nerim.net jessie/main amd64 Packages
Atteint http://ftp.nerim.net jessie/contrib amd64 Packages
Atteint http://ftp.nerim.net jessie/non-free amd64 Packages
Atteint http://ftp.nerim.net jessie/contrib Translation-en
Atteint http://ftp.nerim.net jessie/main Translation-fr
Atteint http://ftp.nerim.net jessie/main Translation-en
Atteint http://ftp.nerim.net jessie/non-free Translation-en
535 ko réceptionnés en 7s (67,6 ko/s)
Lecture des listes de paquets... Fait
```

### — INSTALLATION DE PACKAGES

- Une fois le nom du package connu, tapez la commande, ceci installera la dernière version disponible :

```
utilisateur@hostname :~$ sudo apt-get install package_name
```

- Si vous désirez installer une version plus ancienne :

```
utilisateur@hostname :~$ sudo apt-get install package_name=version -V
```

### — SUPPRESSION DE PACKAGES

Nous avions vu qu'il est possible d'utiliser **dpkg** avec l'option **-r** (remove) ou **-P** (purge), il est aussi possible de réaliser l'opération via apt-get :

```
utilisateur@hostname :~$ sudo apt-get remove package_name ou
```

```
utilisateur@hostname :~$ sudo apt-get purge package_name
```

### — MISE À JOUR

- 2 arguments sont disponibles pour réaliser la mise à jour, **upgrade** et **dist-upgrade**. L'option **upgrade** met à jour tous les paquets installés sur le système vers les dernières versions (couramment utilisé).

L'option **dist-upgrade** met à jour tous les paquets installés vers les dernières versions en installant de nouveaux paquets si nécessaire, par opposition à l'**upgrade** simple qui n'ajoute pas de nouveaux paquets. Attention dist-upgrade ne réalisera PAS de mise à jour de la version ubuntu 12.04 à 12.10. Cependant il mettra à jour la dernière version du kernel et autre nouveau package non-présent.

```
utilisateur@hostname :~$ sudo apt-get upgrade
```

```
utilisateur@hostname :~$ sudo apt-get dist-upgrade
```

### — OBTENTION DES SOURCES

Si vous désirez récupérer les sources d'un logiciel que vous avez installé :

```
utilisateur@hostname :~$ sudo apt-get source package_name
```

### — SUPPRESSION DES PACKAGES MIS EN CACHE

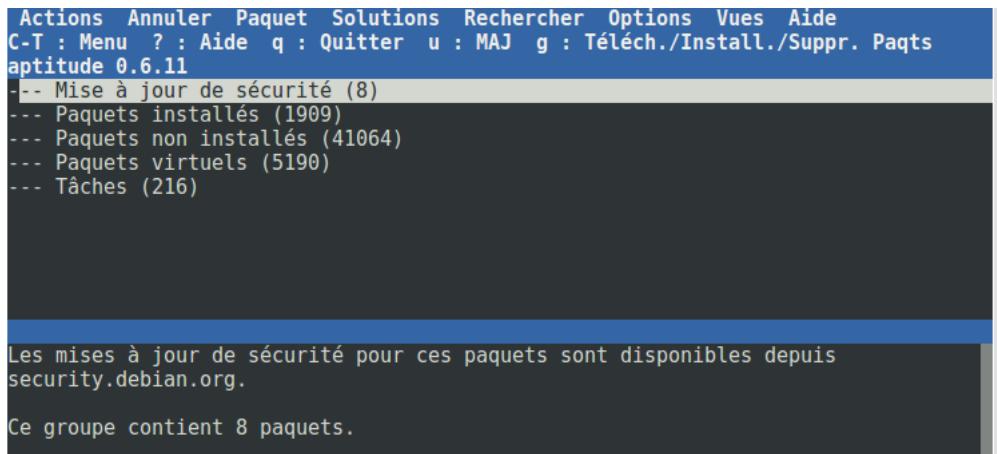
- Le système **apt-get** utilise un répertoire de cache (/var/cache/apt/archives) pour télécharger ses packages, ceci peut prendre beaucoup d'espace au fil du temps, il est possible de faire du ménage :

```
utilisateur@hostname :~$ sudo apt-get clean all
```

## OUTIL APTITUDE

**Aptitude** est un gestionnaire de paquets basé sur l'infrastructure APT, c'est-à-dire que vous pourrez installer, supprimer et mettre à jour les logiciels (paquets) avec **Aptitude**. Il présente des fonctionnalités équivalentes à **dselect** ou **apt-get**. Il y a deux façons d'utiliser Aptitude :

- d'une façon semblable à **apt-get**, "**aptitude**" fonctionne en ligne de commande (**aptitude search nom\_du\_paquet**, **aptitude install nom\_du\_paquet**, ...)
- avec une interface interactive « ncurses » (interface graphique qui s'affiche dans votre terminal et qui n'a donc pas besoin de serveur X ni de Gnome ou autres environnements de bureau), ceci en tapant la commande : **sudo aptitude**



"**apt-get**" et "**aptitude**" sont capables de chercher des paquets par mots clef, puis de les télécharger et de les installer puis de les mettre à jour et de les supprimer si besoin.

"**aptitude**" semble mieux gérer les dépendances et notamment, lorsqu'on désinstalle un paquet, aptitude pourra supprimer les dépendances non utilisées par d'autres programmes. De plus il peut être configuré pour récupérer automatiquement les paquets recommandés ou suggérés lors de l'installation.

Petites infos pour paramétrier l'installation des paquets recommandés ou suggérés avec aptitude :

Aptitude : :Recommends-Important "false" ;

Aptitude : :Recommends-Important "true" ;

Aptitude : :Suggests-Important "true" ;

## INSTALLATION

Pour installer un paquet, vous devrez faire comme avec **Synaptic** : le rechercher, le sélectionner pour installation, puis appliquer.

Pour rechercher un paquet, appuyez sur « / ». Vous serez alors face à une boîte de recherche. Entrez le nom du paquet et la recherche se fera automatiquement. Une fois que le nom est écrit au complet, appuyez sur « Entrée ». Si ce n'est pas le paquet correspondant, appuyer sur « n »

pour rechercher le paquet suivant qui contient les termes recherchés, jusqu'à ce que vous trouviez le paquet à installer.

Lorsque le paquet est trouvé, appuyez sur la touche « + » pour le sélectionner pour installation. Les dépendances seront automatiquement sélectionnées aussi.

Pour confirmer les changements appuyez sur « g » appuyez encore sur « g » pour confirmer ou sur « q » pour revenir à l'écran précédent.

En résumé :

- « / » pour la recherche
- « n » pour poursuivre la recherche ( next )
- « + » pour sélectionner pour installation
- « = » pour maintenir le paquet dans sa version actuelle
- « g » (première fois) confirmer les changements (go)
- « g » (deuxième fois) pour appliquer les changements (go)

## SUPPRESSION

Pour supprimer un paquet, il faut suivre sensiblement la même démarche, donc rechercher avec « / », puis sélectionner pour suppression avec « - » ou encore, pour supprimer les fichiers de configuration aussi, « \_ » et enfin confirmer avec « g » et appliquer avec un autre « g ». Vous remarquerez que les paquets qui avaient été installés automatiquement par Aptitude pour satisfaire les dépendances seront automatiquement supprimés s'ils ne sont plus utilisés.

En résumé :

- « - » pour une suppression simple (**apt-get remove**)
- « \_ » pour une suppression du paquet et de ses fichiers de configuration ( **apt-get remove -purge** )
- « = » pour maintenir le paquet dans sa version actuelle
- « g » (première fois) confirmer les changements
- « g » (deuxième fois) pour appliquer les changements

## MISE À JOUR

Pour une mise à jour de la liste des paquets disponibles, il suffit d'appuyer sur « u ». Pour mettre à jour les paquets qui peuvent être mis à jour, appuyez sur « U », puis sur « g » pour confirmer et une autre fois pour appliquer. Pour mettre à jour seulement un paquet parmi tous ceux qui peuvent être mis à jour, faites comme si vous vouliez l'installer, recherchez-le puis appuyez sur « + », « g » et encore « g ».

En résumé :

- « u » mise à jour de la liste des paquets ( **apt-get update** )
- « U » mise à jour des paquets ( **apt-get upgrade** )
- « g » (première fois) confirmer les changements
- « g » (deuxième fois) pour appliquer les changements

## 12.5 VERSION VIDÉO

Télécharger le fichier : truc\_astuce\_bash-installation-logiciel.ogv et le lire avec vlc

# Chapitre 13

## INSTALLATION DE LOGICIEL SOUS DEBIAN

### 13.1 MANIPULATION DE LOGICIEL SOUS UBUNTU

Gestion de packages :

1. Rechercher le package VLC
2. Afficher les informations du package VLC . Quelle est la version du logiciel ?
3. Afficher les dépendances du package VLC. Quels packages sont recommandés ?
4. Installer le logiciel VLC
5. installer le logiciel openssh-server
6. Lister les fichiers contenu dans le package openssh-server, après l'avoir installé.
7. Dans quel package le fichier /bin/ls se trouve ?
8. Télécharger le package x3rus-formation.deb
9. Installer le package x3rus-formation.deb
10. Exécuter le binaire installé avec /usr/bin/x3

## 13.2 SOLUTION

1. Rechercher le package VLC

Effacer le cache : sudo apt-get clean

Mettre à jour le cache : sudo apt-get update

rechercher le package : sudo apt-cache search vlc

2. Afficher les informations du package VLC . Quelle est la version du logiciel ?

rechercher les infos : sudo apt-cache showpkg vlc

la version est 2.2.0~rc2-2+deb8u1

3. Afficher les dépendances du package VLC. Quels packages sont recommandés ?

sudo apt-cache depends vlc

sudo apt-cache depends vlc | grep Dépend | wc -l : 58

sudo apt-cache depends vlc | grep Recommande | wc -l : 3 packages recommandés

4. Installer le logiciel VLC

sudo apt-get install vlc

5. installer le logiciel openssh-server

sudo apt-get install openssh-server

openssh-server est déjà la plus récente version disponible.

dpkg -l openssh-server

ii openssh-server 1 :6.7p1-5 amd64 secure shell (SSH) server, for secu

6. Lister les fichiers contenu dans le package openssh-server, après l'avoir installé.

dpkg -L openssh-server

7. Dans quel package le fichier /bin/ls se trouve ?

dpkg -S /bin/ls : coreutils

8. Télécharger le package x3rus-formation.deb

Télécharger à l'aide du navigateur, enregistrer la cible sous ~/Téléchargements

9. Installer le package x3rus-formation.deb

cd ~/Téléchargements/

sudo dpkg -i x3rus-formation.deb

10. Exécuter le binaire installé avec /usr/bin/x3

x3 ou /usr/bin/x3

COOL ça fonctionne!!!!

# Chapitre 14

## SYSTÈME DE PACKAGE REDHAT

- Système de package RedHat
  - Outils de gestion des packages
- Présentation rpm
  - Commande disponible
- Présentation yum
  - Commandes disponibles

### 14.1 SYSTÈME DE PACKAGE REDHAT

Les paquets contiennent généralement tous les fichiers nécessaires pour implémenter un ensemble de commandes ou de fonctionnalités. Il y a deux sortes de paquets RedHat :

- Les paquets binaires contenant les exécutables, les fichiers de configuration, les pages de manuel ou d'info, les informations de copyright et d'autres documentations. Ces paquets sont distribués sous un format d'archive **RPM**. Ils sont habituellement reconnaissables par l'extension « **.rpm** ». Ils peuvent être installés en utilisant l'utilitaire **RPM**; vous trouverez plus de détails dans les pages de manuel.
- Les paquets sources sont constitués d'un fichier **.srpm** décrivant le paquet source. Le package contient un **tar.gz** avec le code source de l'application , ainsi que des fichiers **.patch** incluant le patch appliqué au logiciel. L'ensemble des arguments de compilation et scripts d'installation sont définis dans le fichiers **.spec** .

Les outils de gestion de paquets RedHat peuvent être utilisés pour :

- manipuler ou administrer les paquets ou une partie des paquets,
- administrer les modifications locales (« overrides ») des fichiers d'un paquets,
- aider les développeurs dans la construction de paquets

Le nom des paquets binaires RedHat se conforme à la convention suivante : package\_name-version-release.architecture.**rpm**

## 14.2 OUTILS DE GESTION DES PACKAGES

- **rpm** – Installation de package et validation des dépendances
- **yum** – Permet l'obtention des packages depuis Internet (le réseau)

**rpm** utilise le répertoire `/var/lib/rpm` pour stocker sa base de données d'informations sur les packages. **yum** utilise le répertoire `/var/cache/yum` pour stocker les fichiers téléchargés depuis internet.

## 14.3 PRÉSENTATION DE RPM

**rpm** (pour RedHat Package Management) est un outil logiciel en ligne de commande chargé de l'installation, la création, la suppression et la gestion des paquets rpm (**.rpm**), le type de paquets traités par RedHat / CentOS / Fedora. Pour l'installation de paquets, **rpm** dispose d'une interface graphique, Kpackage, GnoRPM, Midnight Commander, ... que vous pouvez utiliser si vous préférez éviter la ligne de commande.

À la différence de la commande **yum**, **rpm** est un outil qui ne gère pas les dépendances. Ainsi en cas de conflit ou bien lorsque seuls certains paquets impliquant trop de dépendances font défaut, l'utilisation de cet outil devient presque indispensable. **yum** utilise justement cet outil pour résoudre certains problèmes caractéristiques. Il permet donc de 'jouer' sur un seul paquet (installation, suppression, reconfiguration) sans bouleverser les dépendances. Parmi ses autres fonctions **rpm** permet aussi d'avoir des informations précises telles que l'état ou la description détaillée des paquets disponibles.

### COMMANDES DISPONIBLES

Installation de packages, préalablement téléchargés

utilisateur@hostname :~\$ **sudo rpm -i le\_package.rpm**

Suppression de packages :

utilisateur@hostname :~\$ **sudo rpm -i le\_package**

Lister les packages installés :

utilisateur@hostname :~\$ **rpm -qa**

Recherche de fichier dans le système de packages :

utilisateur@hostname :~\$ **rpm -qF /bin/ls**

## 14.4 PRÉSENTATION DE YUM

**Yum**, pour **Yellowdog Updater Modified**, est un gestionnaire de paquets pour des distributions Linux telles que Fedora et Red Hat Enterprise Linux, créé par Yellow Dog Linux.

Il permet de gérer l'installation et la mise à jour des logiciels installés sur une distribution. C'est une surcouche de **RPM** gérant les téléchargements et les dépendances, de la même manière que **APT** de **Debian** ou **Urpmi** de **Mageïa**. Il existe, plusieurs types de paramètres qui peuvent suivre la commande **YUM**. Certains concernent l'installation (comme **install**), la suppression (comme **remove**), la recherche (**search**) ou la mise à jour du système (**update**). Lorsqu'on exécute YUM en ligne de commande, cet utilitaire va d'abord interroger un certain nombre de dépôts activés qui sont définis dans le répertoire `/etc/yum.repos.d/` ou consulter son cache. En fonction des informations obtenues, il pourra traiter le paramètre qui lui a été ajouté.

### COMMANDES DISPONIBLES

Recherche du package :

```
utilisateur@hostname :~$ yum search le_package_name
```

Installation de packages :

```
utilisateur@hostname :~$ sudo yum install le_package
```

Suppression de packages :

```
utilisateur@hostname :~$ sudo yum remove le_packge
```

Identification s'il y a des mises à jour et réalisation de mises à jour :

```
utilisateur@hostname :~$ sudo yum check-update et
```

```
utilisateur@hostname :~$ sudo yum upgrade
```

Suppression des packages mis en cache :

```
utilisateur@hostname :~$ sudo yum clean all
```



# Chapitre 15

## PRÉSENTATION DE VIM

Vim se différencie de la plupart des autres éditeurs par son fonctionnement modal, hérité de **vi**. En effet, il possède trois modes : le mode normal (dans lequel vous êtes lorsque Vim démarre), le mode commande, et le mode édition.

Contrairement à **vi**, **Vim** est un logiciel libre. Son code source a été publié pour la première fois en 1991 par Bram Moolenaar, son principal développeur. Depuis, ce dernier a continué de l'améliorer, avec l'aide de nombreux contributeurs.

- Concept de VIM
- Utilisation de VIM
  - Déplacement (mode Commande)
  - Du mode Commande vers le mode Insertion
  - Les commandes Exécutables en mode Commande
    - Recherche dans le fichier
    - Manipulation de fichier
    - Couper/copier & coller (commande) , chercher & remplacer (exec. commande)
  - Site de référence VIM

### 15.1 CONCEPT DE VIM

L'éditeur de texte est un outil de base sous Linux. Il sert notamment à modifier les fichiers de configuration du système. Les deux éditeurs de texte les plus connus et les plus utilisés sont Vim et Emacs. Et comme je ne connais pas Emacs, je vais vous expliquer comment fonctionne Vim !

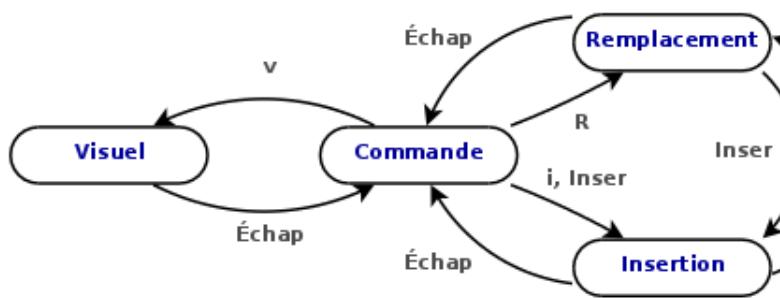
**vim** signifie **Vi IMproved** : il s'agit d'une version améliorée du classique **vi**. Il est très complet, peu gourmand en ressources, et fait très bien la coloration syntaxique. Il n'est pas facile à maîtriser au début, mais **vous serez rapidement conquis !**

L'utilisation de **VIM** n'est pas réputée comme simple, cependant bien souvent l'apparence de complexité est surtout due à un problème de compréhension du fonctionnement du logiciel. **VI** est structuré en **mode**, chaque mode a un rôle / but dans le système, l'important ici est de comprendre cette logique.

## VOICI LA LISTE DES MODES DISPONIBLES :

- Le mode Commande, dans lequel vous vous trouvez quand vous ouvrez **Vim**. Dans ce mode, vous tapez des commandes... que nous verrons plus en détail ! Si vous êtes dans un autre mode et que vous voulez revenir au mode commande, tapez **Échap**. Nous pouvons avec ce mode : sauvegarder le document, réalisé des **search and replace**, ouvrir un autre document, se déplacer dans le document, ...
- Le mode Insertion auquel on accède par la touche **Inser**. L'indicateur – **INSERTION** – apparaît alors en bas de l'écran. Dans ce mode, vous insérez du texte classiquement.
- Le mode Remplacement auquel on accède en appuyant une deuxième fois sur **Inser**. L'indicateur – **REEMPLACEMENT** – apparaît alors en bas de l'écran. Dans ce mode, le texte entré remplace le texte présent sous le curseur.
- Le mode Visuel auquel on accède par la touche **v** depuis le mode Commande. L'indicateur – **VISUEL** – apparaît alors en bas de l'écran. Ce mode permet de sélectionner du texte pour y appliquer globalement des commandes.

Ci-dessous une représentation schématique du passage d'un mode à l'autre :



Comme vous pouvez le constater la touche **Échap** / **Esc** vous permet, peu importe le mode dans lequel vous êtes, de revenir au mode commande.

Rappel pour réaliser l'installation de **VIM**

utilisateur@hostname :~\$ sudo apt-get install vim

## 15.2 UTILISATION DE VIM

Pour appeler le logiciel, rien de nouveau. Nous utiliserons le nom de la commande avec en paramètre le nom du fichier, bien que ce dernier est optionnel.

```
utilisateur@hostname :~$ vim notre_fichier
```

Le premier mode dans lequel nous sommes est le mode **commande**, ceci est logique, car ce mode est utilisé pour le déplacement dans le fichier. Nous allons donc débuter par ce mode.

### DÉPLACEMENT DANS LE TEXTE

Tel que mentionné plus tôt, le mode commande nous permet de nous déplacer dans le fichier. Voici les différentes touches pour se déplacer. Prendre note que ceci est une courte liste et qu'il y a d'autres touches disponibles.

<b>Touche</b>	<b>Description</b>
<b>← ou h</b>	Déplacement d'un caractère vers la gauche
<b>↓ ou j</b>	Déplacement d'un caractère vers le bas
<b>↑ ou k</b>	Déplacement d'un caractère vers le haut
<b>→ ou l</b>	Déplacement d'un caractère vers la droite
<b>e</b>	Va à la fin du mot courant
<b>w</b>	Saute au début du prochain mot (donc saute les caractères d'espaces , tabulation ,... )
<b>0 (zéro)</b> ou <b>^</b>	Va au début de la ligne
<b>\$</b>	Va à la fin de la ligne
<b>G</b>	Va à la fin du fichier
<b>gg</b>	Retour au début du fichier.

L'utilisation des touches **h**, **j**, **k**, **l** est peu utilisée de nos jours cependant il est important de les connaître ou de savoir qu'elles existent, car si un jour vous êtes sur un système qui n'a qu'un **VI** original que vous devez déboguer, ce sera votre seul méthode de déplacement.

## DU MODE COMMANDE VERS LE MODE INSERTION

Maintenant que nous sommes en mesure de nous déplacer dans le fichier nous allons voir comment passer en mode **insertion**.

L'indicateur – **INSERTION** – apparaît alors en bas de l'écran.

Touche	Signification
a	Ajout derrière le caractère actif
A	Ajout à la fin de la ligne
i ou la touche Insér	Insertion devant le caractère actif
I	Insertion en début de ligne
o	Insertion d'une nouvelle ligne, sous la ligne active
O	Insertion d'une nouvelle ligne, au dessus de la ligne active

En mode d'insertion, vous verrez la chaîne –**INSERT**– apparaître en bas de l'écran (de cette façon vous savez dans quel mode vous êtes). C'est dans ce mode et uniquement dans celui-ci que vous pouvez insérer du texte. Pour revenir en mode **commande**, utiliser la touche **ESC**.

En mode d'insertion, vous disposez des touches **Backspace** et **Suppr** pour effacer du texte à la volée. Pour vous déplacer dans le texte, aussi bien en mode commande qu'en mode insertion, vous disposez des touches fléchées. En mode commande, il existe également d'autres combinaisons de touches, voir le paragraphe précédent.

## LES COMMANDES EXÉCUTABLES

Le mode **ex** est disponible, à partir du mode commande, en tapant le caractère « ` : ». Ce même : apparaîtra en bas de l'écran, le curseur s'y positionnera également et tout ce que vous tapez à la suite, suivi d'une pression sur Entrée, sera considéré par **Vi** comme une commande **ex**. Si vous effacez la commande jusqu'à « effacer » le « ` : », vous revenez alors en mode commande et le curseur retrouvera sa place d'origine.

Pour enregistrer les modifications faites dans un fichier vous taperez :w en mode commande. Si vous voulez enregistrer le contenu du tampon dans un autre fichier, tapez la séquence :w <nom\_fichier>.

Les instructions réalisées en mode **commande** qui débutent par les caractères " / , ? , : " permettent l'exécution d'instructions diverses. Lors de l'utilisation de ces caractères vous verrez en bas de l'écran l'instruction que vous tapez.

/ et ? permettent de réaliser des recherches (**search**) dans le fichier alors que l'utilisation de « ` : » permet d'appeler des instructions de **VIM**, quelles soient build-in ou d'appeler des modules. Voici quelques **commandes exec** disponibles :

## RECHERCHE DANS LE FICHIER

Touche	Description
/mot_clef	pour rechercher un mot vers le bas du texte ( <b>n</b> pour passer à l'occurrence suivante, <b>N</b> pour passer à la précédente)
?mot_clef	pour rechercher un mot vers le haut du texte ( <b>n</b> pour passer à l'occurrence suivante, <b>N</b> pour passer à la précédente (donc vers le bas))

## MANIPULATION DE FICHIER

Touche	Description
:w	pour enregistrer
:w nom_du_fichier	pour faire <i>enregistrer-sous nom_du_fichier</i> ;
:q	pour quitter
:wq ou :x	pour enregistrer et quitter
:q!	pour quitter SANS enregistrer les modifications
:r	pour inclure le contenu d'un autre fichier
:e	Pour ouvrir un autre fichier

## COUPER/COPIER & COLLER, CHERCHER & REMPLACER

Nous utilisons tous allègrement le copier-coller dans la vie de tous les jours et j'en suis sûr vous mourrez d'envie de pouvoir utiliser cette fonctionnalité avec votre nouvel éditeur préféré. Nous allons constater que ces commandes se réalisent dans le mode **commande** donc avec des instructions à l'aide de caractères.

Commençons avec le « **couper** », ceci aura l'avantage de permettre la visualisation de l'opération.

Touche	Description
x	Coupe le caractère courant
X	Coupe le caractère précédent
dd	Coupe toute la ligne courante
d\$	Coupe la ligne depuis le caractère courant jusqu'à la fin de la ligne
d0	Coupe la ligne depuis le caractère courant jusqu'au début de la ligne
dw	Coupe le mot courant
[nombre]x	Coupe N caractère par exemple :3x coupe les 3 caractères suivants , il est possible de faire la même opération avec le X majuscule
[nombre]dd	Coupe N ligne par exemple : 20dd coupe les 20 prochaines lignes
[nombre]dw	Coupe N mot

Continuons avec le « copier », vous constaterez que le concept est très similaire.

Touche	Description
y	Copie le caractère courant
Y	Copie le caractère précédent
yy	Copie toute la ligne courante
y\$	Copie la ligne depuis le caractère courant jusqu'au fin de la ligne
y0	Copie la ligne depuis le caractère courant jusqu'au début de la ligne
yw	Copie le mot courant
[nombre]y	Copie N caractère par exemple :3x coupe les 3 caractère suivant , il est possible de faire la même opération avec le X majuscule
[nombre]yy	Copie N ligne par exemple : 20dd coupe les 20 prochaines lignes
:w [nombre]yw	Copie N mot

Bon c'est bien beau tout ça mais il ne reste plus qu'à coller.

Touche	Description
p	Colle ce qui est dans le buffer
[nombre] p	Colle plusieurs fois ce qui est dans le buffer

Bien entendu faut l'utiliser régulièrement, sinon à chaque fois on cherche la touche appropriée.

Pour le « chercher » et « remplacer » (**search and replace**), ceci est en mode **exécution** à partir du mode **commande**, donc ceci commence avec le caractère « : ».

Voici le mode d'utilisation :

Touche	Description
<code>:s/mot_original /mot_nouveau</code>	Ceci remplace <b>mot_original</b> par <b>mot_nouveau</b> sur la ligne courante uniquement, de plus s'il y a plusieurs occurrence du mot <b>mot_original</b> uniquement la première sera modifier.
<code>:s/mot_original /mot_nouveau/g</code>	Ceci remplace <b>mot_original</b> par <b>mot_nouveau</b> sur la ligne courante uniquement, de plus s'il y a plusieurs occurrence du mot <b>mot_original</b> elle seront toutes modifier , ceci grâce à l'option <b>/g</b>
<code>:3,7s/mot_original /mot_nouveau/g</code>	Ceci remplace <b>mot_original</b> par <b>mot_nouveau</b> pour les lignes de 3 à 7 inclusivement .
<code>:%s/mot_original /mot_nouveau/g</code>	Ceci remplace <b>mot_original</b> par <b>mot_nouveau</b> pour l'ensemble du document

## TUTORIEL VIDÉO

Télécharger **vim.ogv** et le lire avec vlc

### 15.3 ASTUCES

- Sous vi, vim, ou ex et ses dérivés, pour afficher les caractères invisibles (caractères de contrôle, retour chariot, nouvelle ligne, tabulation, etc.) passez en mode commande en pressant le caractère deux points ( : ) puis entrez **:set list** Pour faire disparaître ces caractères invisibles **:set nolist** ou **:set list !**
- Lorsque vous ouvrez plusieurs fichiers, pour passer de l'un à l'autre, utiliser **:previous** et **:next** . Ces commandes peuvent être appelées par leur raccourcis, respectivement **:prev** et **:n**
- Sous vi, vim, ou ex et ses dérivés, pour afficher les numéros de lignes, passez en mode commande en pressant le caractère deux points ( : ) puis entrez **set nu** pour faire disparaître ces numéros de lignes **:set nonu** ou **:set nu !**

## 15.4 AVOIR L'AIDE DE VIM EN FRANÇAIS

Rubrique issue de [http://vim-fr.org/index.php/Avoir\\_l'aide\\_de\\_vim\\_en\\_français](http://vim-fr.org/index.php/Avoir_l'aide_de_vim_en_français)

Avoir l'aide de vim en français c'est possible. Précisons toutefois que la totalité de l'aide n'est pas encore traduite, mais qu'il y en a déjà une bonne partie et que si vous souhaitez contribuer à la traduction, toute aide serait bienvenue.

Le principe : lorsque nous faisons un :help <commande>, une page d'aide apparaît généralement en **split** dans notre terminal. En fait il ne s'agit que de la lecture d'un fichier texte. Donc il faut changer ces fichiers par les fichiers correspondants en français.

Remplacer l'aide pour l'utilisateur courant : au lieu de faire le remplacement dans /usr/ vous le faites dans votre **home** et cela n'affectera que vous.

Pour ce faire, vous devez récupérer l'archive suivante [http://cfennajoui.net/vim/archive\\_traduit.tar.gz](http://cfennajoui.net/vim/archive_traduit.tar.gz) et extraire ses fichiers dans le répertoire ~/.vim/doc/. Supprimer le fichier hebrew.txt qui pose un problème d'encodage. Ensuite ouvrez vim et lancez la commande suivante :

```
:helptags $HOME/.vim/doc/
```

Et c'est gagné. Pour revenir à l'aide en anglais, supprimez simplement les fichiers que vous avez extraits dans ~/.vim/doc/.

## 15.5 SITE DE RÉFÉRENCE VIM

<http://formation-debian.via.ecp.fr/vim.html> : Documentation

<http://michael.peopleofhonoronly.com/vim/>

[http://www.vim-fr.org/index.php/Commandes\\_de\\_bases](http://www.vim-fr.org/index.php/Commandes_de_bases) : les commandes VIM de base

<https://openclassrooms.com/courses/reprenez-le-controle-a-l-aide-de-linux/vim-l-editeur-de-texte-du-programmeur>

<http://yannesposito.com/Scratch/fr/blog/Learn-Vim-Progressively/>

<http://cfennajoui.net/vim/>

Pour devenir un geek de vim :

<https://vimebook.com/fr/download/d0924dc2-c4ea-41ee-bae1-67b766e8c1ce/vim-pour-les-humains.pdf>



# Chapitre 16

## TUTORIAL VIM

### 16.1 VIMTUTOR

- La présentation sur **VIM** a pour but principal de vous fournir l'information sur VIM , cependant afin de bien comprendre le système, il est important de pratiquer et d'expérimenter l'éditeur.
- La commande **vimtutor** nous offre un petit tutoriel afin de s'exercer.

### 16.2 ATELIER PRATIQUE

Lancez le tutoriel Vim. Il copie d'abord le fichier du tutoriel, afin que vous puissiez le modifier sans altérer le fichier original.

**Vimtutor** est utile pour les personnes souhaitant apprendre leurs premières commandes Vim. Il s'exécute à l'aide de la commande :

```
pat@amd-a10 :~$ vimtutor
```

Lire et suivre les instructions.



# Chapitre 17

## REDIRECTION ET FLUX DE DONNÉES

### ENTRÉES / SORTIES DES COMMANDES (FLUX DE DONNÉES)

Nous allons voir ici un point fort du système d'exploitation UNIX. Ce chapitre présente la redirection des flux de données. Ça ne dit pas grand chose, cependant nous verrons que ceci nous permettra d'imbriquer plusieurs commandes les unes à la suite des les autres.

#### PRÉSENTATION DES REDIRECTIONS ET DU FLUX DE DONNÉES DES PROCESSUS.

Le tout est agrémenté d'exemples.

Nous allons découvrir qu'il est possible de rediriger le résultat d'une commande ailleurs que dans la console. Où ? 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 ! Comment ? À l'aide de petits symboles spéciaux, appelés flux de redirection, que vous allez découvrir dans ce chapitre.

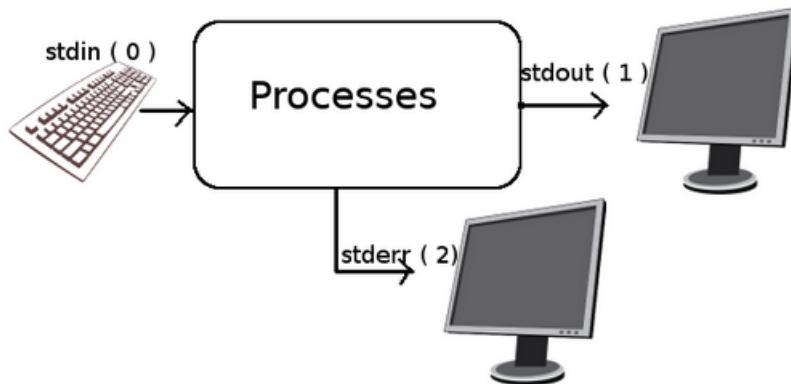
Dans ce chapitre, nous verrons comment combiner plusieurs commandes.

- le système Entrée / Sorties
- Redirection du flux standard (stdout) et du flux d'erreur (stderr)
- Redirection du flux d'Entrée (stdin)

### 17.1 LE SYSTÈME ENTRÉES / SORTIES

Chaque processus (Programme) possède 3 flux standards, qu'il utilise pour communiquer en général avec l'utilisateur :

- l'entrée standard nommée **stdin** (identifiant **0**) : il s'agit par défaut du clavier,
- la sortie standard nommée **stdout** (identifiant **1**) : il s'agit par défaut de l'écran,
- la sortie d'erreur standard nommée **stderr** (identifiant **2**) : il s'agit par défaut de l'écran.



Ces flux peuvent être redirigés afin que le processus interagisse avec un autre au lieu d'interagir avec l'utilisateur. Nous touchons un point majeur du système UNIX et du fonctionnement de la console. Il est donc important d'être attentif pour cette leçon !

## 17.2 REDIRECTION DU FLUX STANDARD ET DU FLUX D'ERREUR

Pour commencer nous allons voir comment rediriger le flux de sortie vers un fichier. Pour rediriger le résultat d'un commande, vous pouvez utiliser l'opérateur `>` ou l'opérateur `>>`.

- `>` : redirige. Permet de rediriger le résultat de la commande dans le fichier de votre choix. Ceci va créer un nouveau fichier. Si le fichier existe déjà, ce dernier sera écrasé !
- `>>` : sert aussi à rediriger le résultat dans un fichier, mais, cette fois, à la fin de ce fichier. Ceci réalise donc un ajout à la fin du fichier. Si le fichier n'existe pas, il sera créé.

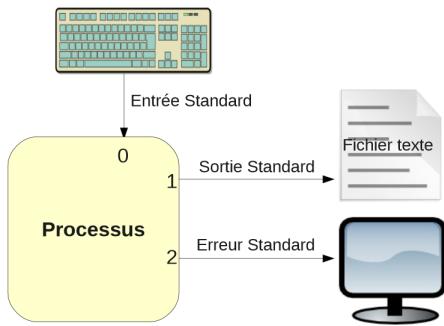
Voici un exemple de l'utilisation de l'opérateur `>`. Le répertoire de travail est `/tmp`. Affichons son contenu à l'écran :

```
pat@amd-a10 :/tmp$ ls
```

```
fic1 fic2 fic3
```

Redirection de la sortie standard dans le fichier `resultat_cmd_ls`

```
pat@amd-a10 :/tmp$ ls /tmp/ > resultat_cmd_ls
```



Le fichier vide `resultat_cmd_ls` est créé en premier puis le résultat de la commande `ls` est ajouté dans le fichier vide : `resultat_cmd_ls`

```
pat@amd-a10 :/tmp$ cat resultat_cmd_ls
fic1
fic2
fic3
resultat_cmd_ls
```

Redirection de la sortie standard dans le fichier `resultat_cmd_ls`

```
username@hostname :/tmp$ ls /etc/ > resultat_cmd_ls
```

Le fichier existant est écrasé avec le nouveau contenu.

```
username@hostname :/tmp$ head /tmp/resultat_cmd_ls
adduser.conf
adjtime
aliases
alternatives
anacrontab
apache2
apg.conf
apm
apparmor.d apt
```

Reprenons la même opération, mais avec l'opérateur double `>>` :

Redirection de la sortie standard dans le fichier « `resultat_cmd_ls_double` » dans le répertoire `/tmp`

```
pat@amd-a10 :/tmp$ ls /tmp/ >> resultat_cmd_ls_double
```

Résultat du fichier obtenu :

```
pat@amd-a10 :/tmp$ cat resultat_cmd_ls_double
fic1
fic2
fic3
resultat_cmd_ls
resultat_cmd_ls_double
```

Redirection du contenu de /etc dans le fichier resultat\_cmd\_ls\_double

```
pat@amd-a10 :/tmp$ ls /etc/ >> resultat_cmd_ls_double
```

Le résultat de la command **ls /tmp** est toujours présent au début du fichier, le résultat de la command **ls /etc** est à la fin du fichier.

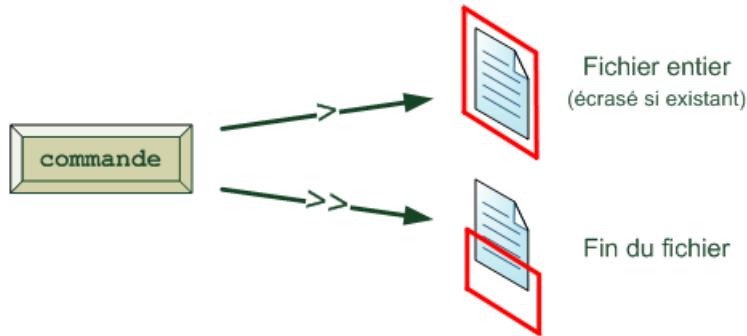
```
username@hostname :~$ head resultat_cmd_ls_double
fic1
fic2
fic3
resultat_cmd_ls
resultat_cmd_ls_double
adduser.conf
adjtime
aliases
alternatives
anacrontab
```

Le contenu de la command **ls /etc** à la fin du fichier

```
pat@amd-a10 :/tmp$ tail resultat_cmd_ls_double
wgetrc
wildmidi
wodim.conf
wpa_supplicant
X11
x3_formation.conf
xboard
```

```
xdg
xfce4
xml
```

En gros ceci se résume comme suit :



Nous avons vu qu'au lieu d'utiliser l'écran, qui est la sortie par défaut, nous avions redirigé le résultat vers un fichier. Nous avions utilisé la commande **ls**, mais ceci fonctionne aussi bien avec n'importe quelle commande (ex. : **cat** , **tail** , **head** , ...).

Dans l'exemple ci-dessus, nous avons principalement redirigé le **stdout** (1). Voici un exemple, encore avec **ls**, d'une commande sans erreur et d'une avec erreur :

#### COMMANDÉ **ls** SANS ERREURS

```
pat@amd-a10 :/tmp$ ls -l /etc/passwd /etc/group > /tmp/show_files
```

Résultat pat@amd-a10 :/tmp\$ **cat** /tmp/show\_files

```
-rw-r--r-- 1 root root 983 nov. 24 17:58 /etc/group
-rw-r--r-- 1 root root 2229 oct. 12 16:46 /etc/passwd
```

#### COMMANDÉ **AVEC ERREURS** J'ajoute le listing du fichier /etc/networking qui n'existe pas.

```
pat@amd-a10 :/tmp$ ls -l /etc/passwd /etc/group /etc/networking > /tmp/show_files
ls : impossible d'accéder à /etc/networking : Aucun fichier ou dossier de ce type
```

Résultat le fichier ne contient **que** le résultat SANS erreur :

```
pat@amd-a10 :/tmp$ cat /tmp/show_files
-rw-r--r-- 1 root root 983 nov. 24 17:58 /etc/group
-rw-r--r-- 1 root root 2229 oct. 12 16:46 /etc/passwd
```

Comme vous pouvez le constater, le résultat affiché et celui redirigé dans le fichier /tmp/show\_files sont équivalents. Cependant, lors de l'utilisation de la commande **ls** avec l'erreur, nous avions en

plus un message d'erreur à l'écran : "ls : impossible d'accéder à /etc/networking : Aucun fichier ou dossier de ce type" .

Pourquoi ce message s'affiche t'il alors que nous avions indiqué de réaliser la redirection du fichier ?

La raison est que, lors de l'utilisation de `>` ou `>>`, nous ne redirigons que l'identifiant **1**, soit **stdout**. Si nous voulons rediriger les messages d'erreur, **stderr** (**2**), nous devrons spécifiquement l'identifier. Voici comment faire :

Voyons la commande **ls** avec redirection du standard output ET du standard d'erreur, création de 2 fichiers.

```
pat@amd-a10 :~$ ls -l /etc/passwd /etc/group /etc/networking > /tmp/show_files
2>/tmp/show_files_err
```

VISUALISATION DU RÉSULTAT :

```
pat@amd-a10 :~$ cat /tmp/show_files_err
```

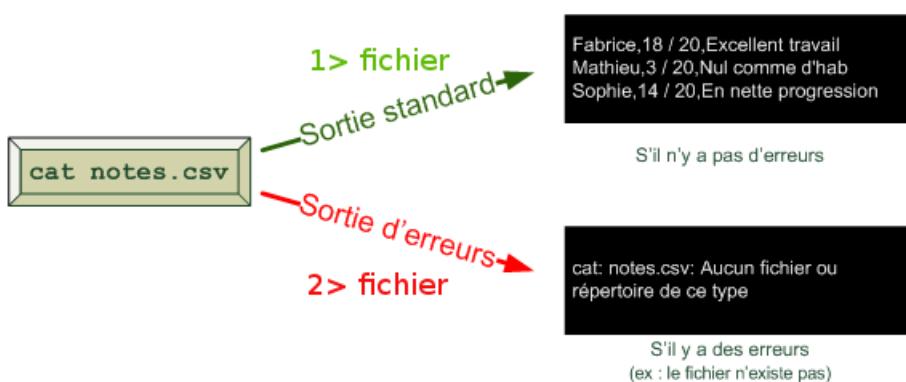
ls : impossible d'accéder à /etc/networking : Aucun fichier ou dossier de ce type

```
pat@amd-a10 :/tmp$ cat show_files
```

```
-rw-r--r-- 1 root root 983 nov. 24 17:58 /etc/group
-rw-r--r-- 1 root root 2229 oct. 12 16:46 /etc/passwd
```

Grâce à l'identifiant **2>**, nous avons redirigé les messages du **stderr** vers le fichier `/tmp/show_files_err`. En d'autres mots, quand vous utilisez l'opérateur `>`, c'est équivalent à `1>`.

Ici je ne fais mention que de l'opérateur `>`; c'est équivalent avec l'opérateur double `>>`.



C'est bien beau tout ça, mais si vous désirez avoir l'ensemble des messages ( les bons comme les mauvais ) dans un fichier ? Vous seriez peut-être tenté de faire ceci :

Tentative de redirection stdout et stderr dans le même fichier, mauvaise méthode :

```
pat@amd-a10 :/tmp$ ls -l /etc/passwd /etc/group /etc/networking  
1>/tmp/show_files 2>/tmp/show_files
```

# Malheureusement ceci **ne fonctionne pas !**

```
pat@amd-a10 :/tmp$ cat show_files  
-rw-r-r- 1 root root 983 nov. 24 17 :58 /etc/group  
-rw-r-r- 1 root root 2229 oct. 12 16 :46 /etc/passwd
```

Pour réaliser cette configuration il y a 2 méthodes, rediriger le flux d'erreurs **stderr ( 2 )** vers la sortie standard **stdout ( 1 )**, ou vous pouvez utiliser l'opérateur **&**.

- **2>&1** : Avec cette notation nous redirigeons **stderr ( 2 )** dans **stdout ( 1 )**
- **&>** : Nous redirigeons l'ensemble des flux **stdout ( 1 )** ET **stderr ( 2 )**

# redirection de l'ensemble des flux, redirection de stderr vers stdout.

```
pat@amd-a10 :/tmp$ ls -l /etc/passwd /etc/group /etc/networking 1>/tmp/show_files  
2>&1
```

# Visualisation du contenu du fichier

```
pat@amd-a10 :/tmp$ cat /tmp/show_files
```

```
pat@amd-a10 :/tmp$ cat show_files
```

ls : impossible d'accéder à /etc/networking : Aucun fichier ou dossier de ce type

```
-rw-r-r- 1 root root 983 nov. 24 17 :58 /etc/group
```

```
-rw-r-r- 1 root root 2229 oct. 12 16 :46 /etc/passwd
```

```
# Commande avec &>
```

```
pat@amd-a10 :/tmp$ ls -l /etc/passwd /etc/group /etc/networking &>/tmp/show_files
```

```
# Visualisation du contenu du fichier
```

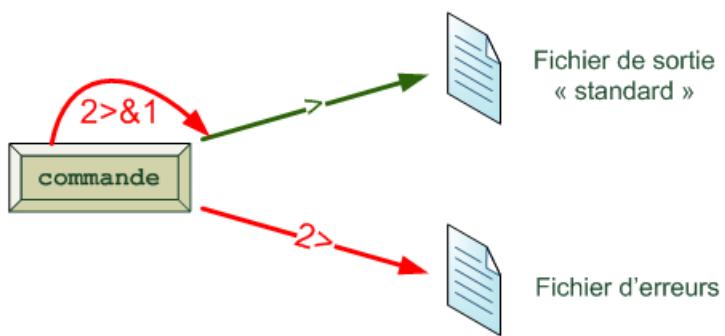
```
pat@amd-a10 :/tmp$ cat show_files
```

```
ls : impossible d'accéder à /etc/networking : Aucun fichier ou dossier de ce type
```

```
-rw-r--r-- 1 root root 983 nov. 24 17 :58 /etc/group
```

```
-rw-r--r-- 1 root root 2229 oct. 12 16 :46 /etc/passwd
```

Une représentation graphique :



## 17.3 REDIRECTION DU FLUX D'ENTRÉE (STDIN)

Nous avons donc vu, dans la section précédente, la gestion des 2 flux **stdout(1)** et **stderr(2)**, qui sont par défaut redirigés vers l'écran. Nous les avons redirigés vers des fichiers. Nous n'avons cependant pas encore touché à l'autre flux, qui est le **stdin(0)** : le flux d'entrée.

Nous allons voir un exemple avec la commande **cat**.

```
# Utilisation classic de cat
pat@amd-a10 :~/pratiques$ 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

# Utilisation de cat avec redirection stdin
pat@amd-a10 :~/pratiques$ 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
```

Écrire **cat < notes.csv** semble identique au fait d'exécuter **cat notes.csv...** du moins en apparence. Le résultat produit est le même, mais ce qui se passe derrière est très différent.

- Si vous écrivez **cat notes.csv**, la commande **cat** reçoit en argument le nom du fichier notes.csv, qu'elle doit ensuite se charger d'ouvrir pour afficher son contenu.
- Si vous écrivez **cat < notes.csv**, la commande **cat** reçoit le contenu du fichier notes.csv qu'elle se contente simplement d'afficher dans la console. C'est le shell (le programme qui gère la console) qui se charge d'envoyer le contenu de notes.csv à la commande **cat**.

Bref, ce sont deux façons de faire la même chose mais de manière très différente.

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.

Nous allons utiliser la commande **sort** pour réaliser la démonstration. Cette commande nous permet de trier le contenu d'un fichier, ou n'importe quel contenu que la commande **sort** reçoit, que ce soit à l'aide d'un fichier ou depuis le **standard input (stdin 0)**.

Voici un exemple stdin avec << en utilisant la commande **sort**.

# on va justement écrire des nombres, un par ligne. Lorsque vous avez fini, tapez FIN pour arrêter la saisie. Après visualisation du > tapez votre nombre suivi de la touche Entrée à chaque fois.

```
pat@amd-a10 :~/pratiques$ sort -n << FIN
```

```
> 13
```

```
> 132
```

```
> 10
```

```
> 131
```

```
> 34
```

```
> 87
```

```
> 66
```

```
> 68
```

```
> 65
```

```
> FIN
```

```
10
```

```
13
```

```
34
```

```
65
```

```
66
```

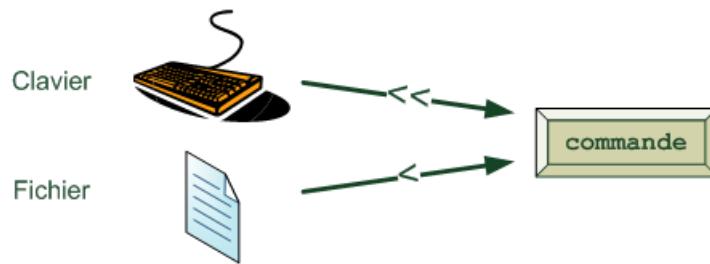
```
68
```

```
87
```

```
131
```

```
132
```

Donc << 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.



Il est bien entendu possible de combiner le tout.

Exemple utilisation stdin, stdout, stderr le tout combiné

Les nombres saisis au clavier seront envoyés au fichier « nombres\_tries.txt », de même que les erreurs éventuelles.

```
pat@amd-a10 :~/pratiques$ sort -n << FIN > nombres_tries.txt 2>&1
```

ou

```
pat@amd-a10 :~/pratiques$ sort -n << FIN &> nombres_tries.txt
```

```
> 18
```

```
> 27
```

```
> 1
```

```
> FIN
```

Cette article n'aurait pas été possible sans la réalisation de ce cours offert à l'URL :

<http://fr.openclassrooms.com/informatique/cours/reprenez-le-controle-a-l'aide-de-linux/et-lire-depuis-un-fichier-ou-le-clavier>

J'ai beaucoup copié ce contenu ; ceci reste sous licence libre mais il est important de le souligner.

## 17.4 TUTORIEL VIDÉO

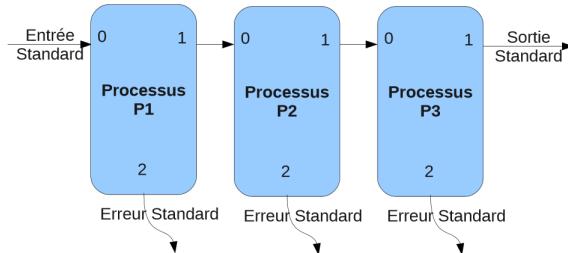
Télécharger le fichier **redirection\_flux-1.ogv** et le lire avec **vlc**.



# Chapitre 18

## CHAÎNAGE DE COMMANDES

Redirige la sortie d'une commande vers l'entrée d'une autre commande. Il s'agit donc d'une chaîne de redirection entre deux processus qui ne passe pas par un fichier, mais par une zone mémoire du système.



Cette fonctionnalité est vraiment une des plus importantes et décuple les possibilités offertes par la console.

Un filtre est une commande qui lit les données sur l'entrée standard, effectue des traitements sur les lignes reçues et écrit le résultat sur la sortie standard. Bien sûr les entrées/sorties peuvent être redirigées, et enchaînées avec des tubes.

**Remarque :** le caractère d'indirection < en entrée n'est pas obligatoire pour les filtres.

### LES COMMANDES FILTRES

- **cut** : pour supprimer une partie de chaque ligne d'un fichier
- **tr** : Convertir ou éliminer des caractères
- **grep** : afficher les lignes correspondant à un motif donné
- **wc** : afficher le nombre de lignes, de mots et d'octets d'un fichier
- **sort** : Trier les lignes de fichiers texte
- **uniq** : Signaler ou éliminer les lignes répétées
- **tee** : Lire depuis l'entrée standard et écrire sur la sortie standard et dans des fichiers
- **sed** : Éditeur de flux pour le filtrage et la transformation de texte
- **head** : Afficher le début des fichiers

- **tail** : Afficher la dernière partie de fichiers
- **xargs** : Construire et exécuter des lignes de commandes à partir de l'entrée standard

## 18.1 DESCRIPTION

Nous avons vu dans la section précédente que nous pouvions rediriger le résultat d'un processus vers un fichier . Nous redirigions donc la sortie (**stdout** et **stderr**), qui pointe par défaut vers l'écran, dans un fichier. Nous avons aussi vu qu'il était possible d'envoyer à un processus des données via le standard d'entrée **stdin**.

Dans ce chapitre, nous verrons que nous réaliserons ce même jeu de redirection, mais au lieu de réaliser une redirection vers un fichier, ce sera vers d'autre processus .

Voici une schématisation :



En gros, **tout ce qui sort de la commande1 est immédiatement envoyé à la commande2**. Et vous pouvez chaîner des commandes comme cela indéfiniment !

Parfois, l'utilité de certaines commandes seules peuvent paraître limitées, mais celles-ci prennent en général tout leur sens lorsqu'on les combine à d'autres commandes.

Pour rediriger la sortie d'une commande vers une autre commande, nous n'utilisons pas le symbole **>**, mais le symbole **|** (Alt Gr 6) appelé pipe. Quand le symbole "pipe" **|** est utilisé, seul le **stdout** (1) est transmis à la prochaine commande. Si vous désirez que les erreurs soient elles aussi envoyées au **stdin** (0) de la seconde commande, il faut rediriger les **stderr** (2) vers le **stdout** (1) avec l'instruction **2>&1** ou **&>**.

## 18.2 LES COMMANDES FILTRES

**CUT** permet d'extraire des colonnes depuis une chaîne de caractères en tenant compte d'un « séparateur (délimiter) » connu. Un exemple de type de fichier classique que **cut** gère très bien est les fichiers de type csv (Comma Separate Value) chaque champ étant séparé par le séparateur « , » ou « ; » ou autre délimiteur connu.

Il est donc possible avec **cut** d'extraire uniquement les colonnes 1 et 3. Voici un exemple avec le fichier **/etc/passwd** qui utilise le délimiteur « : » (deux points) pour séparer chaque colonne.

Présentation du fichier **/etc/passwd** qui nous servira d'exercices avec la commande « **cut** ».

```
pat@amd-a10 :~$ head /etc/passwd
root:x:0:0:root:/root:/bin/bash
```

```

daemon :x :1 :1 :daemon :/usr/sbin :/bin/sh
bin :x :2 :2 :bin :/bin :/bin/sh
sys :x :3 :3 :sys :/dev :/bin/sh
sync :x :4 :65534 :sync :/bin/:/bin/sync
games :x :5 :60 :games :/usr/games :/bin/sh
man :x :6 :12 :man :/var/cache/man :/bin/sh
lp :x :7 :7 :lp :/var/spool/lpd :/bin/sh
mail :x :8 :8 :mail :/var/mail :/bin/sh
news :x :9 :9 :news :/var/spool/news :/bin/sh

```

commande : `cut -d( séparateur ) -f( sélection_champs ) [fichiers]`

utilisation pour extraire la colonne 1 ET 3 en utilisant le séparateur : du fichier **passwd**

```
pat@amd-a10 :~$ cut -d ":" -f 1,3 /etc/passwd
```

```
root :0
```

```
daemon :1
```

```
bin :2
```

```
sys :3
```

```
sync :4
```

```
games :5
```

```
man :6
```

[[ OUTPUT COUPÉ ]]

Comme vous pouvez le constater, nous pouvons utiliser **cut** en lui passant en argument le fichier à manipuler. **Cut** est aussi en mesure de lire la chaîne de caractères depuis le **stdin (0)**.

Utilisation de **cut** avec une chaîne de caractères depuis le **Stdin**. Même commande mais avec imbrication de commande.

```
pat@amd-a10 :~$ cat /etc/passwd | cut -d ":" -f 1,3
```

```
root :0
```

```
daemon :1
```

```
bin :2
```

```
sys :3
```

```
sync :4
```

```
games :5
```

```
man :6
```

lp :7

[[OUTPUT COUPE]]

Vous constaterez à l'avenir que **cut** est essentiel dans la vie de tous les jours, un peu à l'image d'un couteau suisse. Nous le retrouverons plus tard....

## TR

**tr** nous permet de faire de la manipulation sur un type de caractère. Par exemple, il est possible de supprimer les duplicates de caractères qui se suivent. Un autre exemple courant d'utilisation est l'option **-s**, qui permet de supprimer ("squeezes") les caractères en répétition tels que les espaces ou autres. Voici un fichier formaté pour l'humain, mais qui peut être compliqué à manipuler avec **cut** par exemple, dû au nombre variable d'espaces :

Fichier des buts au foot ball : les\_buts.txt

```
Ribery      99 Munich,all fr
Giroud      87 Arsenal,ang fr
Hamouma     61 St-etienne,fr fr
Balotelli   89 Milan,it it
Neur        0 Munich,all de
Loic        29 Newcastle,ang fr
Aubameyang 98 Dortmund,all ga
Pogba       99 Juventus,it fr
kaka        53 Milan,it br
Riviere     15 Monaco,fr fr
Bastos      26 Roma,it br
Lukaku      31 Everton,ang be
Toti        17 Juventus,it it
Hazard      78 Chelsea,ang be
Ibrahimovic 99 Paris,fr su
Kruse       43 Gladbach,all de
Tabanou     82 St-etienne,fr fr
Ozil        55 Arsenal,ang de
Donati      8 Leverkusen,all it
Lyoris      0 Tottenham,ang fr
Weelbeck    2 ManU,ang en
Robben      72 Munich,all nd
Gignac      87 Marseille,fr fr
Griezmann   87 Sociedad,esp fr
Benzema     1 Madrid,esp fr
Cabaye      65 Newcastle,ang fr
Reus        78 Dortmund,all de
Neymar      45 Barcelone,esp br
Rossi       38 Florentina,it it
Lewandowski 87 Dortmund,all pl
```

Si nous tentons d'extraire la deuxième colonne, qui est le nombre de buts marqués par les joueurs, avec la commande **cut** nous aurons un problème. **tr** est présent pour "supprimer" les espaces répétitifs :

La deuxième colonne est un espace !

```
pat@amd-a10 :~/pratiques$ cat les_buts.txt | cut -d " " -f 2
```

[[ OUTPUT COUPÉ]]

Utilisation de **tr** pour couper le nombre d'espaces superflus.

```
pat@amd-a10 :~/pratiques$ cat les_buts.txt | tr -s " "
```

Ribery 99 Munich,all fr

Giroud 87 Arsenal,ang fr

Hamouma 61 St-etienne,fr fr

Balotelli 89 Milan,it it

Neur 0 Munich,all de

Loic 29 Newcastle,ang fr

Aubameyang 98 Dortmund,all ga

|| OUTPUT COUPÉ||

Bon nous avons "supprimer" les espaces en trop. Moins simple à lire pour l'humain, mais plus efficace pour le système. Il est important de noter que c'est uniquement le résultat (output) qui est en format "machine". Le fichier original n'a pas été modifié.

C'est bien beau tout ça, mais moi, je voulais un fichier csv, avec comme délimiteur « ; » .

**tr** nous permet aussi de remplacer un caractère par un autre. Attention ! Uniquement 1 caractère peut être remplacé à la fois ! Dans notre cas, je vais remplacer l'espace par le caractère « ; » . Voici le résultat :

**tr** avec remplacement de caractères :

```
pat@amd-a10 :~/pratiques$ cat les_buts.txt | tr -s " " | tr " " ";"
```

Ribery ;99 ;Munich,all ;fr

Giroud ;87 ;Arsenal,ang ;fr

Hamouma ;61 ;St-etienne,fr ;fr

Balotelli ;89 ;Milan,it ;it

Neur ;0 ;Munich,all ;de

Loic ;29 ;Newcastle,ang ;fr

Aubameyang ;98 ;Dortmund,all ;ga

Pogba ;99 ;Juventus,it ;fr

kaka ;53 ;Milan,it ;br

|| OUTPUT COUPÉ||

QUESTION :

- Maintenant, comment fais-je pour créer un fichier csv ?
- Est il possible de récupérer uniquement la deuxième colonne ?

## GREP

Dans cette section nous allons voir grep, mais le **grep** "simple" pour commencer. **Grep** est un utilitaire très complet, qui, je pense, est essentiel de maîtriser. La question n'est pas "est-ce que je l'utilise tous les jours", mais plutôt "est-ce qu'il se passe une heure au travail sans que je ne l'utilise" ?

**Grep** nous permet de faire l'extraction d'une ligne depuis une chaîne de caractères, l'extraction est réalisée si un motif (modèle) donné est trouvé (pattern match)! En d'autres termes **grep** nous permet de réaliser des recherches dans un fichier! Il renvoie l'information au standard de sortie (**stdout**).

**syntaxe :** grep [options] modèle ce critère [fichier1...]

Utilisation simple de grep, recherche d'un motif avec grep dans passwd

```
pat@amd-a10 :~/pratiques$ grep -color "bash" /etc/passwd
root :x :0 :0 :root :/bin/bash
pat :x :1000 :1000 :pat,,,:/home/pat :/bin/bash
```

Comme vous le constatez, il est possible de fournir en argument un fichier à **grep**, mais il est en mesure de lire depuis le **stdin(0)**.

Utilisation simple de **grep** depuis le **stdin** :

```
pat@amd-a10 :~/pratiques$ cat /etc/passwd | grep -color "bash"
root :x :0 :0 :root :/bin/bash
pat :x :1000 :1000 :pat,,,:/home/pat :/bin/bash
```

Bien que, dans le cas présent, il n'y ait que peu de différences. Il est important de savoir que **grep** est optimisé pour interpréter de gros fichier. Il peut être plus optimal par moment de fournir en paramètre le fichier, au lieu de transmettre l'information par le canal d'entrée standard (**stdin**).

Bon voilà pour la présentation simple de **grep**! Pour revenir à notre cas :

Je désire manipuler mon fichier afin d'extraire les joueurs qui jouent dans des club anglais. Je reprends donc ma commande et ajoute le **grep** :

## COMMANDÉ GREP AVEC LES AUTRES FILTRES

Recherche des joueurs évoluant en Angleterre :

```
pat@amd-a10 :~/pratiques$ cat les_buts.txt | tr -s " " | tr "\n" ";" | grep ",ang"
Giroud ;87 ;Arsenal,ang ;fr
Loic ;29 ;Newcastle,ang ;fr
```

```
Lukaku ;31 ;Everton,ang ;be
Hazard ;78 ;Chelsea,ang ;be
Ozil ;55 ;Arsenal,ang ;de
Lyoris ;0 ;Tottenham,ang ;fr
Weelbeck ;2 ;ManU,ang ;en
Cabaye ;65 ;Newcastle,ang ;fr
```

Est-ce optimal comme notation ? Le fichier que nous manipulons n'est pas trop volumineux, l'impact est donc minime. Je vous conseille cependant de filtrer vos données avant de les manipuler avec d'autres filtres. Voici une commande "optimisée" dont la différence ne s'applique que sur des fichiers volumineux.

commande **grep** pour les joueurs en Angleterre :

```
pat@amd-a10 :~/pratiques$ cat les_buts.txt | grep ",ang" | tr -s " " | tr "\n" ";"
```

```
Giroud ;87 ;Arsenal,ang ;fr
Loic ;29 ;Newcastle,ang ;fr
Lukaku ;31 ;Everton,ang ;be
Hazard ;78 ;Chelsea,ang ;be
Ozil ;55 ;Arsenal,ang ;de
Lyoris ;0 ;Tottenham,ang ;fr
Weelbeck ;2 ;ManU,ang ;en
Cabaye ;65 ;Newcastle,ang ;fr
```

OPTIONS INTÉRESSANTES :

**-v** : il est possible de faire une recherche inversée. Ici, je recherche les lignes qui ne contiennent pas la chaîne (string) bash :

**-n** pour indiquer le numéro de ligne concernée. grep recherche inversée :

```
pat@amd-a10 :~/pratiques$ cat /etc/passwd | grep -v bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
```

```
mail :x :8 :8 :mail :/var/mail :/bin/sh
news :x :9 :9 :news :/var/spool/news :/bin/sh
uucp :x :10 :10 :uucp :/var/spool/uucp :/bin/sh
proxy :x :13 :13 :proxy :/bin :/bin/sh
www-data :x :33 :33 :www-data :/var/www :/bin/sh
backup :x :34 :34 :backup :/var/backups :/bin/sh
list :x :38 :38 :Mailing List Manager :/var/list :/bin/sh
[[ OUTPUT COUPÉ ]]
```

Nous avons vu comment extraire la liste des joueurs qui évoluent dans un club anglais en utilisant l'utilitaire filtre **grep**. Au fait, combien sont-ils ? Il est simple de compter 5, 10 lignes, mais après 20 lignes, le risque d'erreur grandit. Ce qui nous amène à la commande **wc**.

## WC

**wc** nous permet de compter les lignes, caractères, mots, etc. Encore une petite commande qui ne fait pas grand chose mais qui le fait bien !! C'est le concept sous GNU/Linux : pas besoin d'avoir une application qui fait tout plus ou moins bien si on peut avoir plusieurs petites applications qui s'imbriquent et qui réalisent bien leur travail ! L'utilisation de **wc** : ici, je ne ferai la démonstration que du compteur de lignes. Pour le reste, vous connaissez la commande man. Je vous invite donc à y découvrir les autres options.

**SYNOPSIS** **wc [OPTION] ... [FICHIER] ... wc [OPTION] ... -files0-from=FICHIER**

Les options ci-dessous permettent de choisir quels décomptes sont affichés. Ils le sont toujours dans l'ordre suivant : lignes, mots, caractères, octets, longueur maximale des lignes.

- l, --lines afficher le nombre de lignes
- w, --words afficher le nombre de mots
- c, --bytes afficher le nombre d'octets
- m, --chars afficher le nombre de caractères

Exemple utilisation de wc en compteur de lignes :

```
pat@amd-a10 :~/pratiques$ cat les_buts.txt | grep ",ang" | wc -l
```

## SORT

La commande **sort**, comme son nom l'indique, nous permet de trier du contenu. Cela peut être un fichier transmis en paramètre ou en provenance du standard d'entrée (**stdin**). Nous avons déjà vu rapidement cette commande dans la section précédente, mais prenons quelques minutes pour l'explorer un peu plus. Si nous reprenons mon fichier avec la liste de joueurs, nous allons encore manipuler les joueurs évoluant en Angleterre, avec le filtre **grep**.

**SYNOPSIS** **sort** [OPTION] ... [FICHIER] ...

Options principales de la commande sort

- **k** permet de spécifier les colonnes de tri (la numérotation commence à 1).
- **t** permet de spécifier le séparateur des champs (utile avec les fichiers csv ou le fichier /etc/passwd)
- **f** –ignore-case convertir les caractères minuscules en majuscules
- **n** –numeric-sort comparer selon la valeur numérique de la chaîne
- **r** –reverse inverser le résultat des comparaisons

AFFICHAGE DES BUTS ANGLAIS SANS TRI

```
pat@amd-a10 :~/pratiques$ cat les_buts.txt | grep ",ang"
```

Giroud	87 Arsenal,ang fr
Loic	29 Newcastle,ang fr
Lukaku	31 Everton,ang be
Hazard	78 Chelsea,ang be
Ozil	55 Arsenal,ang de
Lyoris	0 Tottenham,ang fr
Weelbeck	2 ManU,ang en
Cabaye	65 Newcastle,ang fr

Comme nous pouvons le constater, le résultat n'est pas classé. Nous allons commencer par mettre un peu d'ordre en affichant le résultat avec, comme critère de tri, le nom des joueurs. Classement simple avec « **sort** » (trie par défaut par ordre alphabétique sur la 1ère colonne)

```
pat@amd-a10 :~/pratiques$ cat les_buts.txt | grep ",ang" | sort
```

Cabaye	65 Newcastle,ang fr
Giroud	87 Arsenal,ang fr
Hazard	78 Chelsea,ang be
Loic	29 Newcastle,ang fr
Lukaku	31 Everton,ang be
Lyoris	0 Tottenham,ang fr
Ozil	55 Arsenal,ang de
Weelbeck	2 ManU,ang en

C'est mieux, mais, finalement, ce serait mieux par but inscrit. Le problème pour réaliser cette opération est que le nombre de but est défini au milieu de la chaîne de caractères. Nous devons donc indiquer à la commande **sort** que le tri ne doit pas se faire sur le premier caractère, mais sur un autre. **Sort** a l'option **-k nombre**, qui permet de définir le numéro de la clé, de la colonne à

utiliser pour le tri. Dans notre cas, c'est la **colonne 2**. Si nous réalisons la même commande **sort** avec l'option **-k2**, que se passe t'il ?

Tri sur colonne 2

```
pat@amd-a10 :~/pratiques$ cat les_buts.txt | grep ",ang" | sort -k2
```

Lyoris	0	Tottenham,ang	fr
Loic	29	Newcastle,ang	fr
Weelbeck	2	ManU,ang	en
Lukaku	31	Everton,ang	be
Ozil	55	Arsenal,ang	de
Cabaye	65	Newcastle,ang	fr
Hazard	78	Chelsea,ang	be
Giroud	87	Arsenal,ang	fr

Wowww ça marche ! Enfin, presque. Si vous regardez le résultat, le classement par but est un peu bizarre. Actuellement, **sort** ne sait pas que ce sont des chiffres, donc il a fait un classement basé sur les caractères. Pour que **sort** interprète les nombres, il faut lui passer l'option **-n**

Classement sort avec interprétation numérique ( **sort -k2 -n** ou **sort -nk2** )

```
pat@amd-a10 :~/pratiques$ cat les_buts.txt | grep ",ang" | sort -k2 -n
```

Lyoris	0	Tottenham,ang	fr
Weelbeck	2	ManU,ang	en
Loic	29	Newcastle,ang	fr
Lukaku	31	Everton,ang	be
Ozil	55	Arsenal,ang	de
Cabaye	65	Newcastle,ang	fr
Hazard	78	Chelsea,ang	be
Giroud	87	Arsenal,ang	fr

Yeahh là c'est bon ! Je vous laisse regarder le **man** afin de trouver l'option pour **inverser** (**reverse**) le résultat....

Bon, je vous aide un peu : **cat les\_buts.txt | grep ",ang" | sort -nk2r**

## DÉMONSTRATION AVEC UNE COMMANDE

Nous avons vu l'utilisation de **sort** avec un fichier, mais tout comme l'ensemble des commandes, ceci passe par le flux **stdout > stdin**. Si nous utilisons la commande **ps**, qui nous permet de lister les processus qui sont en exécution sur le système (nous la verrons en détail plus tard), il est aussi possible de trier ou de compter le nombre de lignes.

Résultat avec une commande simple : affichage de la commande **ps**

```
pat@amd-a10 :~/pratiques$ ps aux
[[ OUTPUT COUPÉ ]]
tboutry 29061 2.5 6.1 1405196 501416 ? Sl Jan14 292 :19 /usr/lib/firefox/firefox
tboutry 29092 0.0 0.0 35188 3016 ? Sl Jan14 0 :00 /usr/lib/at-spi2-core/at-spi-bus-launcher
tboutry 29171 0.0 0.1 178520 13864 ? Sl Jan14 0 :12 update-notifier
root 29196 0.0 0.1 17980 9040 ? S Jan14 0 :00 /usr/bin/python /usr/lib/system-service/system-
service-d
tboutry 29202 0.0 0.4 250312 35512 ? SNl Jan14 0 :43 /usr/bin/python /usr/bin/update-manager
-no-focus-on-map
tboutry 29247 0.0 0.0 43944 3552 ? Sl Jan14 0 :13 /usr/lib/deja-dup/deja-dup/deja-dup-monitor
[[ OUTPUT COUPÉ ]]
```

#### CLASSEMENT PAR UTILISATEUR

```
pat@amd-a10 :~/pratiques$ ps aux | sort
[[ OUTPUT COUPÉ ]]
tboutry 9029 0.0 0.0 12384 7412 pts/8 Ss Jan21 0 :00 /bin/bash
whoopsie 1180 0.0 0.0 25936 4364 ? Ssl Jan13 0 :24 whoopsie
www-data 12765 0.0 0.0 12276 2592 ? S Jan19 0 :00 /usr/sbin/apache2 -k start
www-data 12792 0.0 0.0 235124 3240 ? Sl Jan19 0 :00 /usr/sbin/apache2 -k start
```

#### UNIQUEMENT L'UTILISATEUR ROOT

```
pat@amd-a10 :~/pratiques$ ps aux | grep "root"
[[ OUTPUT COUPÉ ]]
root 28837 0.0 0.0 25580 3880 ? Sl Jan14 0 :13 /usr/lib/udisks/udisks-daemon
root 28840 0.0 0.0 6556 716 ? S Jan14 0 :00 udisks-daemon : not polling any devices
root 29196 0.0 0.1 17980 9040 ? S Jan14 0 :00 /usr/bin/python /usr/lib/system-service/system-
service-d
```

#### COMBIEN DE PROCESS SONT EXÉCUTÉS PAR ROOT ?

```
pat@amd-a10 :~/pratiques$ ps aux | grep "root" | wc -l
```

## UNIQ

Voyons une autre commande de filtrage **uniq**, qui nous permet de ne conserver qu'une instance de la valeur extraite. Reprenons nos joueurs de foot. Si je désire avoir les joueurs évoluant en championnat anglais, mais uniquement leur nationalité.

### EXTRACTION LE LA LISTE DES JOUEURS DE FOOT

```
pat@amd-a10 :~/pratiques$ cat les_buts.txt | grep ",ang"
```

Giroud	87	Arsenal	,ang	fr
Loic	29	Newcastle	,ang	fr
Lukaku	31	Everton	,ang	be
Hazard	78	Chelsea	,ang	be
Ozil	55	Arsenal	,ang	de
Lyoris	0	Tottenham	,ang	fr
Weelbeck	2	ManU	,ang	en
Cabaye	65	Newcastle	,ang	fr

Je n'extrait que la liste des nationalités donc la dernière colonne : pat@amd-a10 :~/pratiques\$ **cat les\_buts.txt | grep ",ang" | tr -s " " | cut -d " " -f 4**

```
fr
fr
be
be
de
fr
en
fr
```

Avec la commande **uniq**, nous allons pouvoir enlever tous ces doublons pour ne conserver que les valeurs uniques. Démonstration **uniq** :

```
pat@amd-a10 :~/pratiques$ cat les_buts.txt | grep ",ang" | tr -s " " | cut -d " " -f 4 | uniq
```

```
fr
be
de
fr
en
fr
```

Comme on le voit, il y a plusieurs sorties fr, pourquoi ?

**uniq** ne traite que les lignes identiques qui se suivent. Il faut donc faire un **sort** avant le **uniq**.

```
pat@amd-a10 :~/pratiques$ cat les_buts.txt | grep ",ang" | tr -s " " | cut -d " " -f 4 | sort | uniq
```

be

de

en

fr

AVEC L'OPTION **-c** IL PEUT COMPTER DES ENTRÉES

```
pat@amd-a10 :~/pratiques$ cat les_buts.txt | grep ",ang" | tr -s " " | cut -d " " -f 4 | sort | uniq -c
```

2 be

1 de

1 en

4 fr

## TEE

La commande **tee** nous permet de copier dans un fichier le résultat et de continuer de le transmettre au standard output (**stdout**). Ceci peut être intéressant d'avoir un état pendant le traitement.

VOICI UN EXEMPLE DE LA COMMANDE **tee**

```
utilisateur@hostname :~$ cat les_buts.txt | grep ",ang" | tee buteurs_ang.txt | tr -s " " | cut -d " " -f 4 | sort | uniq -c
```

2 be

1 de

1 en

4 fr

Le fichier **buteurs\_ang.txt** comprend la section intermédiaire du résultat.

```
pat@amd-a10 :~/pratiques$ cat buteurs_ang.txt
```

```

Giroud      87 Arsenal,ang fr
Loic        29 Newcastle,ang fr
Lukaku      31 Everton,ang be
Hazard       78 Chelsea,ang be
Ozil         55 Arsenal,ang de
Lyoris       0 Tottenham,ang fr
Weelbeck    2 ManU,ang en
Cabaye      65 Newcastle,ang fr
pat@amd-a10:~/pratiques$ ls -ltr

```

## SED

Sed, une commande Unix permettant de manipuler du texte. Sed et cut sont des outils très puissants de manipulation de chaînes de caractères permettant de modifier/supprimer des occurrences dans une chaîne. Ils permettent de réaliser des actions comme remplacer un caractère par un autre dans un fichier, supprimer des chaînes de caractères inutiles et moultes autres actions que nous allons à présent aborder.

Je vais montrer une utilisation de la commande **sed**. Ce n'est qu'une utilisation simple ; nous pourrions parler pendant plusieurs jours des possibilités de **sed**. Dans le cadre de cette session, nous verrons comment remplacer une chaîne de caractère.

Je vais simplement changé le mot "ang" par angleterre : exemple de sed

```

pat@amd-a10 :~/pratiques$ cat buteurs_ang.txt | grep ",ang" | sed 's/ang/angleterre/g'
Giroud      87 Arsenal,angleterre fr
Loic        29 Newcastle,angleterre fr
Lukaku      31 Everton,angleterre be
Hazard       78 Chelsea,angleterre be
Ozil         55 Arsenal,angleterre de
Lyoris       0 Tottenham,angleterre fr
Weelbeck    2 ManU,angleterre en
Cabaye      65 Newcastle,angleterre fr

```

## HEAD / TAIL

Rappelez-vous de **head** et **tail** qui sont aussi des commandes de **filtres** qui permettent de visualiser le début ou la fin d'un fichier...

## XARGS

Nous sommes maintenant en mesure d'extraire de l'information et de l'afficher à l'écran ou de la rediriger dans un fichier ! Nous serons, car il faut un peu de pratique pour être à l'aise, en mesure de manipuler n'importe quelles données. De les manipuler pour répondre à nos besoins. Il est possible que vous ayez besoin d'autres commandes « filtre » non couvertes ici, mais sachez qu'elles existent !

Mais que faire si je désire réaliser une opération avec le résultat de la commande ? Restons dans le simple, car je manque d'imagination. Je désire créer un répertoire pour chaque joueur évoluant dans le championnat anglais, portant le nom de leur nationalité. Si je reprends la commande :

```
pat@amd-a10 :~/pratiques$ cat les_buts.txt | grep ",ang" | tr -s " " | cut -d " " -f 4 |
sort | uniq
be
de
en
fr
```

Il faudrait créer les répertoires : be, de, en et fr .

Naturellement, l'idée serait d'ajouter **mkdir** à la fin de la commande, mais nous aurions le résultat suivant :

ERREUR

```
pat@amd-a10 :~/pratiques$ cat les_buts.txt | grep ",ang" | tr -s " " | cut -d " " -f 4 |
sort | uniq | mkdir
```

**mkdir** : opérande manquant Saisissez « **mkdir -help** » pour plus d'informations.

Pourquoi ? Parce que **mkdir** reçoit via le standard d'entrée (**stdin**) la liste des répertoires, mais la liste des répertoires n'est PAS transmise en paramètre à **mkdir**.

Pour réaliser cette opération, nous devons utiliser **xargs**, comme ceci : Exemple commande valide

**xargs**

```
pat@amd-a10 :~/pratiques$ cat les_buts.txt | grep ",ang" | tr -s " " | cut -d " " -f 4 |
sort | uniq | xargs mkdir
```

```
pat@amd-a10 :~/pratiques$ ls -l
```

total 16

```
drwxr-xr-x 2 xerus xerus 4096 22 Jan 16 :50 be
drwxr-xr-x 2 xerus xerus 4096 22 Jan 16 :50 de
drwxr-xr-x 2 xerus xerus 4096 22 Jan 16 :50 en
drwxr-xr-x 2 xerus xerus 4096 22 Jan 16 :50 fr
```

Avec **xargs**, la commande filtre a réécrit **mkdir** comme suit : **mkdir be de en fr**. Donc elle a passé la liste du résultat, reçu par le **stdin** à la fin de la commande, en argument.

## 18.3 COMPLÉMENTS

Nous avons vu que **grep** est un filtre, ce qui lui permet d'être combiné avec d'autres commandes, sous la forme d'un pipeline.

Trois variantes du programme sont disponibles : egrep, fgrep et rgrep ; egrep est identique à grep -E, fgrep est identique à grep -F et rgrep est identique à grep -r. L'appel direct à egrep ou fgrep est déconseillé depuis 2001, mais est toujours possible pour permettre à d'anciennes applications qui les utilisent de fonctionner sans modification.

Ici nous allons seulement nous intéresser à **egrep** afin de pouvoir faire des recherches sur plusieurs critères en même temps.

Au lieu de faire deux recherches : cat mon\_fichier | grep "chaine1" puis cat mon\_fichier | grep "chaine2"

Il est possible de faire un grep sur les deux valeurs chaine1 et chaine2 en même temps en utilisant la commande suivante :

```
cat mon_fichier | egrep 'chaine1|chaine2'
```

Pour rechercher les mots anglais ou allemands :

```
cat les_buts.txt | egrep "ang|all"
```

ou

```
cat les_buts.txt | grep -E "ang|all"
```

## 18.4 TUTORIEL VIDÉO

Télécharger le fichier **redirection\_flux-1.ogv** et le lire avec **vlc** à partir 31'

Télécharger le fichier **redirection\_flux-2.ogv** et le lire avec **vlc**





# Chapitre 19

## PRATIQUE DES PIPES ET DES DE FILTRES

Nous allons mettre en pratique ce que nous avons vu. Quelques exercices afin de valider la compréhension de la matière.

Certains exercices sont plus simples que d'autres, et il y a plus d'une réponse possible. Libre à vous !

### 19.1 TRAVAIL SIMPLE

1. /etc/passwd
  - Combien d'utilisateurs sur la machine ont le shell "**sh**", attention uniquement **sh** et non **bash**
  - Extraire un seul utilisateur **random**, uniquement le nom de l'utilisateur depuis le fichier.
2. /sbin/ifconfig
  - Depuis la commande **ifconfig**, faire en sorte d'extraire les adresses **IP** de la machine
3. Logs Apache
  - télécharger le fichier de logs apache **www.x3rus.com\_access.log.gz** et décompresser le fichier
  - Extraire la liste des adresses ip (Unique) ayant communiquées avec le serveur en Janvier.
  - Ajouter le nombre de fois que l' ip apparaît.
  - Trier le résultat afin que l' ip qui accède le plus souvent soit en bas.

### 19.2 TRAVAIL INTERMÉDIAIRE

1. xargs
  - Grâce à la commande **find**, lister la liste des fichiers .h dans le répertoire **/usr/include** (récursivement !) et passer ces fichiers à grep afin d'y rechercher le mot "zombies". Vous devriez trouver 2 fichiers .

## 2. Addition

- Reprendre le fichier de joueurs et faire l'addition des buts marqués en championnat anglais.

### 19.3 TRAVAIL EXPERT

## 1. dig, whois

- avec la commande dig -x 204.19.176.100, extraire l'information afin que le résultat ne soit QUE le nom de domaine : acceo.com : voici le résultat de la commande :
  - acceo.com
- Depuis ce résultat, passer l'information à whois et extraire uniquement les informations sur le contact technique, voici le résultat :
  - Tech Name : Thierry Bonte
  - Tech Organization : ACCEO Solutions Inc.
  - Tech Street : 75 Queen St
  - Tech Street : suite 6100
  - Tech City : Montreal
  - Tech State/Province : Quebec
  - Tech Postal Code : H3C2N6
  - Tech Country : Canada
  - Tech Phone : +1.5148680333 Tech Phone Ext :
  - Tech Fax : Tech Fax Ext :
  - Tech Email : registrar-manager@acceo.com

# Chapitre 20

## PERMISSIONS

### 20.1 PERMISSIONS ET PROPRIÉTAIRE DES FICHIERS

Dans cette section nous verrons comment lire les permissions d'un fichier, et comment réaliser la modification des ces dernières. Nous couvrirons les permissions **posix**, pour les ACL (**acces control list**) étendues ceci fera l'objet d'une autre formation.

Les systèmes d'exploitation inspirés d'Unix (dont Linux fait partie) possèdent la capacité de définir de façon poussée la gestion de droits d'accès aux divers fichiers de votre OS.

Les **droits d'accès** définissent la possession d'un fichier ou d'un répertoire à un utilisateur et à un groupe d'utilisateurs. Ils gèrent aussi quelles actions les utilisateurs ont le droit d'effectuer sur les fichiers, selon qu'ils sont propriétaires du fichier, membre du groupe propriétaire du fichier ou ni l'un ni l'autre. La possession et la gestion des permissions associées s'effectuent individuellement avec chaque fichier.

- Le présent article est un document d'explications à propos des droits d'accès. Les sections "Les propriétaires" et "Les permissions" exposent de façon générale ce que sont ces attributs auxquels vous devrez faire face dans votre vie linuxienne.
- Les manipulations des droits d'accès des fichiers et dossiers sont abordées dans l'article « Permissions ».

GNU/Linux soit un système multi-utilisateurs : chaque fichier est la propriété exclusive d'un utilisateur et d'un groupe. Chaque utilisateur dispose de son propre répertoire (appelé son répertoire personnel, à savoir son **home** directory en anglais). Il est le propriétaire de ce répertoire, ainsi que de tous les fichiers qu'il y créera par la suite. Lui, et personne d'autre.

Cependant, la notion de propriété d'un fichier, prise seule, ne servirait pas à grand-chose. Mais il y a plus : en tant que propriétaire d'un fichier, un utilisateur peut établir des droits sur ce fichier. Ces droits distinguent trois catégories d'utilisateurs.

## 20.2 LES PROPRIÉTAIRES

Par la propriété d'un fichier, on désigne à quel utilisateur appartient le fichier qui le possède. À partir de cette possession (ou non), il sera ensuite possible de définir des permissions d'accès sur le fichier.

La possession d'un fichier se définit sur trois catégories :

1. **l'utilisateur propriétaire** du fichier (**u**). Il s'agit généralement du créateur du fichier. (Prenez note qu'un fichier créé par une commande exécutée à l'aide de sudo appartiendra à l'utilisateur root. Vous serez potentiellement amené à devoir changer le propriétaire de ce fichier pour pouvoir vous en servir avec votre propre compte utilisateur).
2. le **groupe propriétaire** du fichier (**g**). Si un utilisateur est membre d'un certain groupe qui possède la propriété d'un fichier, l'utilisateur aura aussi certaines permissions particulières sur ce fichier.
3. les autres, **other**, le reste du monde (**o**). Bref, tout un chacun n'étant ni propriétaire du fichier, ni membre du groupe propriétaire du fichier.

Faisons une analogie avec les voitures. Le propriétaire serait la personne au nom de laquelle la voiture est immatriculée. Le groupe propriétaire est l'ensemble des personnes qui sont inscrites en tant que conducteurs secondaires de la voiture chez l'assureur. Enfin, les autres correspondent à toutes les autres personnes n'étant ni détenteur de l'immatriculation, ni inscrites en tant que conducteurs de la voiture chez l'assureur.

## 20.3 LES PERMISSIONS

Les permissions désignent les diverses catégories d'utilisateurs (propriétaire d'un fichier, membres du groupe propriétaire d'un fichier et le reste du monde) qui ont l'autorisation d'effectuer une opération sur un fichier donné. Par exemple, une catégorie d'utilisateurs peut avoir accès en lecture et écriture à un fichier, alors qu'une autre catégorie à accès en lecture seulement.

Les permissions se définissent sur trois niveaux :

1. la **lecture** d'un fichier : cette permission est nécessaire pour pouvoir accéder au contenu d'un fichier (écouter une piste audio, visionner un film, lire un texte, naviguer à l'intérieur d'un répertoire...). Cette permission est notée **r** (pour **read**, lire).
2. l'**écriture** dans un fichier : cette permission est nécessaire pour pouvoir apporter des modifications à un fichier (corriger un texte et enregistrer les changements ; effacer les "yeux rouges" dans une photo et enregistrer la correction ; ajouter, modifier, renommer ou supprimer un fichier dans un dossier ; etc.). Cette permission est notée **w** (pour **write**, écrire).
3. l'**exécution** d'un fichier : cette permission est nécessaire particulièrement pour les logiciels, afin qu'ils puissent être exécutés. Cette permission est notée **x** (pour **execute**, exécuter).

Par exemple, l'utilisateur **toto** dispose des droits de lecture et d'exécution sur le répertoire **foo**, mais pas la permission d'écriture sur ce répertoire ; **toto** peut donc exécuter les programmes

présents dans ce répertoire et ouvrir les fichiers qu'il contient, mais ne peut pas les modifier ni en créer de nouveaux.

Pour chacune des trois catégories d'utilisateurs (propriétaire, membres du groupe propriétaire et reste du monde) sont définies ces trois permissions :

- le **propriétaire** dispose ou non de la permission de lecture, d'écriture et d'exécution sur un fichier ;
- le membre du **groupe propriétaire** dispose ou non de la permission de lecture, d'écriture et d'exécution sur un fichier ;
- tous les **autres** utilisateurs disposent ou non de la permission de lecture, d'écriture et d'exécution sur un fichier.

Les droits sont affichés par une série de 9 caractères, associé 3 par 3 (rwx rwx rwx), ils définissent les droits des 3 identités (**u**, **g** et **o**).

## 20.4 MANIPULATION AVEC LA LIGNE DE COMMANDE

### LISTER LES PERMISSIONS

Nous l'avons déjà vu mais nous n'avions pas pris le temps d'éclaircir le résultat avec la commande **ls -l** , le résultat nous montre les permissions. Voici un exemple :

```
$ ls -l
drwxrwxr-x 2 tboutry tboutry 4096 Feb 3 13 :03 Documents
-rw-rw— 1 root root 0 Feb 3 13 :00 Fichier_du_root
drwx—— 2 tboutry tboutry 4096 Feb 3 13 :04 Perso
-rw-r— 1 tboutry adm 0 Feb 3 13 :01 lefichier_des_adm
-rw-rw-r- 1 tboutry tboutry 0 Feb 3 13 :01 mon_fichier
-rw-r-r- 1 root root 2030 Feb 3 13 :00 passwd
-rw-rw-r- 1 tboutry video 233821918 Jan 21 14 :12 redirection_flux-1.ogv
```

Les droits d'accès apparaissent alors comme une liste de **10 symboles** : **drwxr-xr-x**

Le premier symbole est soit « - », « d », soit « l », nous indiquant la nature du fichier :

- - : fichier régulier
- d : répertoire
- l : lien symbolique
- c : périphérique caractère
- b : périphérique bloc
- p : tube nommé
- s : socket local

Suivent ensuite 3 groupes de 3 symboles chacun, indiquant si le fichier (ou répertoire) est autorisé en lecture, écriture ou exécution. Les 3 groupes correspondent, dans cet ordre, aux droits du propriétaire, du groupe puis du reste des utilisateurs. Dans le paragraphe introductif, vous aurez remarqué des lettres en gras dans les termes anglais. Ce sont ces lettres qui sont utilisées pour symboliser les dites permissions.

Si la permission n'est pas accordée, la lettre en question est remplacé par « - ».

Si l'on reprend les lettres données pour lecture/écriture/exécution (**r**/w/**e**xecute), nous obtenons les symboles : **rwx** et pour propriétaire/groupe/autres (**u**sers/**g**roups/**o**ther), nous obtenons : **ugo**

Reprendons l'exemple théorique suivant :

**drwxr-xr-x**

Il se traduit de la manière suivante :

- d : c'est un répertoire.
- rwx pour le **1er groupe de 3 symboles** : son propriétaire peut lire, écrire et exécuter.
- r-x pour le **2nd groupe de 3 symboles** : le groupe peut uniquement lire et exécuter le fichier, sans pouvoir le modifier.
- r-x pour le **3ème groupe de 3 symboles** : le reste du monde peut uniquement lire et exécuter le fichier, sans pouvoir le modifier.

Toutes les combinaisons de ces droits sont possibles : vous pouvez par exemple autoriser la lecture du fichier à vous seul et l'interdire à tous les autres. Vous pouvez même faire l'inverse, même si ce n'est pas très logique à première vue... En tant que propriétaire du fichier, vous pouvez en changer le groupe propriétaire (si et seulement si vous êtes aussi membre du nouveau groupe), et même vous déposséder du fichier (c'est-à-dire en changer le propriétaire). Bien entendu, si vous vous dépossédez d'un fichier, vous perdrez tous les droits sur celui-ci...

## CHOWN

La commande **chown** (*change owner, changer le propriétaire*) permet de changer le propriétaire du fichier. Seul le super-utilisateur peut utiliser **chown**. La commande s'utilise de la façon suivante :

**chown [OPTION]... [PROPRIÉTAIRE][ :GROUPE]] FICHIER...**

exemple **chown (change uniquement le propriétaire ) : \$ sudo chown Mon\_utilisateur fichier1**

**chown** change en une seule commande *le propriétaire et le groupe du fichier* :

**\$ sudo chown marc :adm fichier1**

Le document **fichier1** appartient alors à l'utilisateur marc et au groupe adm.

Il est aussi possible d'utiliser **chown** pour ne modifier *uniquement le groupe* exemple changement groupe. Exemple :

**\$ sudo chown :adm fichier2**

## CHMOD

L'outil **chmod** (**c**hange **m**ode, *changer les permissions*) permet de modifier les permissions sur un fichier. Il peut s'employer de deux façons : soit en précisant les permissions de manière octale, à l'aide de chiffres ; soit en ajoutant ou en retirant des permissions à une ou plusieurs catégories d'utilisateurs à l'aide des symboles **r** **w** et **x**, que nous avons présenté plus haut. Nous préférerons présenter cette seconde façon ("ajout ou retrait de permissions à l'aide des symboles"), car elle est probablement plus intuitive pour les néophytes. Sachez seulement que les deux méthodes sont équivalentes, c'est-à-dire qu'elles affectent toutes deux les permissions de la même manière.

De cette façon, on va choisir à l'aide des symboles :

1. À qui s'applique le changement
  - **u** (user, utilisateur) représente la catégorie "propriétaire" ;
  - **g** (group, groupe) représente la catégorie "groupe propriétaire" ;
  - **o** (others, autres) représente la catégorie "reste du monde" ;
  - **a** (all, tous) représente l'ensemble des trois catégories.
2. La modification que l'on veut faire
  - **+** : ajouter
  - **-** : supprimer
  - **=** : ne rien changer
3. Le droit que l'on veut modifier
  - **r** : `read` ⇒ lecture
  - **w** : `write` ⇒ écriture
  - **x** : `execute` ⇒ exécution
  - **X** : `eXecute` ⇒ exécution, concerne uniquement les répertoires et les fichiers qui ont déjà une autorisation d'exécution pour l'une des catégories d'utilisateurs. Nous allons voir plus bas dans la partie des traitements récursifs l'intérêt du X.

la commande : **chmod** [OPTION]... MODE[,MODE]... FICHIER...

Parmi les options de la commande chmod (qui ne sont pas nombreuses - voir man chmod) en voici deux :

- **v** pour verbose (affichage sur la sortie standard STDOUT du résultat de la commande)
- **R** traiter les répertoires de façon réursive (application de la commande à l'arborescence entière du répertoire en question)

Il y a deux modes d'utilisation de la commande chmod : de façon littérale ou de façon numérique

Par exemple : supprimer les permissions d'écriture dans le fichier3

**\$ chmod o-w fichier3**

enlèvera le droit d'écriture pour les autres.

Par exemple : ajout permission d'exécution pour tout le monde (**a=all**)

**\$ chmod a+x fichier3**

Par exemple : suppression permission de lecture pour tout le groupe (**g=groupe**)

**\$ chmod g-r fichier3**

On peut aussi combiner plusieurs actions en même temps :

- On ajoute la permission de lecture, d'écriture et d'exécution sur le document fichier3 pour le propriétaire ;
- On ajoute la permission de lecture et d'exécution au groupe propriétaire, on retire la permission d'écriture ;
- On ajoute la permission de lecture aux autres, on retire la permission d'écriture et d'exécution.

modification (ajout et suppression) de plusieurs droits avec chmod

**\$ chmod u+rwx,g+rx-w,o+r-wx fichier3**

### En octal

En octal, chaque « groupement » de droits (pour user, group et other) sera représenté par un chiffre et à chaque droit correspond une valeur :

- r = 4
- w = 2
- x = 1
- - = 0

Par exemple,

- Pour rwx, on aura :  $4+2+1 = 7$
- Pour rw-, on aura :  $4+2+0 = 6$
- Pour r-, on aura :  $4+0+0 = 4$

La commande : **chmod [OPTION]... MODE-OCTAL FICHIER**

Reprendons le répertoire **Documents**. Ses permissions sont :

drwxr-x—

En octal, on aura 750 :

rwx	r-x	—
<b>7=(4+2+1)</b>	<b>5=(4+0+1)</b>	<b>0=(0+0+0)</b>

Pour mettre ces permissions sur le répertoire, on taperait donc la commande :

**\$ chmod 750 Documents**

## 20.5 VIDÉOS

Télécharger et visualiser le fichier permission-et-lien.ogv.

# Chapitre 21

## FICHIERS DE LIENS

Nous verrons dans cette section les deux types de fichiers qui nous permet de lier ou pointer vers un autre fichier.

Le concept de lien est présent sur beaucoup d'OS mais sous \*nix (et donc linux) c'est une religion. Les liens sont partout et il suffit de taper un `ls -l /etc | grep ^l` pour s'en convaincre. Ce petit tutorial a donc pour objectif d'expliquer un peu les différents liens et leurs utilisations possibles.

### 21.1 DÉFINITIONS

Il existe deux types de liens qui permettent de rediriger un fichier vers un autre : les liens physiques (hardlink) et les liens symboliques (softlinks).

#### LIEN SYMBOLIQUE (SYMLINK)

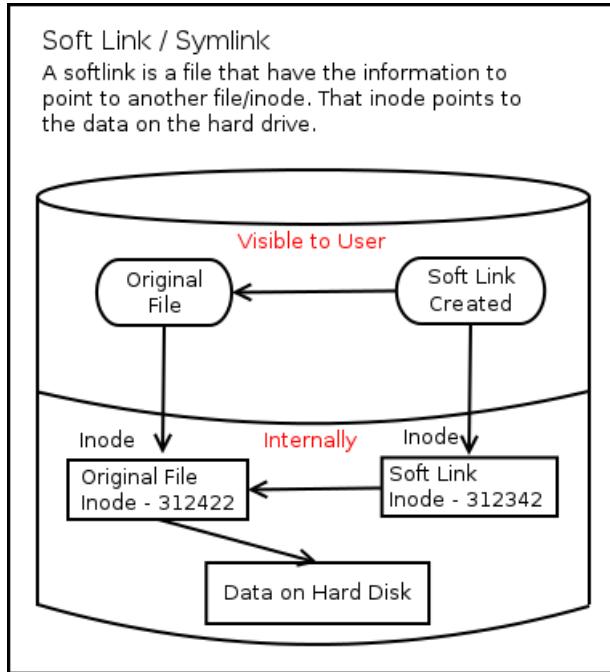
Le lien symbolique est un "faux fichier" ou un "faux dossier" qui contient le chemin d'un "vrai fichier" ou d'un "vrai dossier". Une fois un lien symbolique établi, pour la majorité des applications, le fichier ou dossier cible est bien réel. Et lorsqu'on supprime un lien symbolique, le fichier ou dossier d'origine n'en est en rien modifié. Dans l'autre sens, si l'on détruit le fichier/dossier source d'un lien symbolique, celui-ci ne disparaît pas pour autant, et lors d'un `ls`, il va apparaître clignotant et en erreur, ce sera un lien "cassé".

Une utilisation naturelle du lien symbolique est de "bouturer" un dossier/fichier venant d'un bout de l'arborescence du système de fichier vers un autre dossier, mais en le laissant "physiquement" là où il est.

L'autre aspect du lien symbolique, il peut relier des systèmes de fichiers différents. C'est à dire que l'on peut faire un lien symbolique entre n'importe lesquelles des dossiers/fichiers sans se préoccuper du type de montage du dossier parent (clé usb, cd-rom, nfs, etc..).

Maintenant, l'inconvénient des liens symboliques reste leur relative lenteur. En effet, à chaque entrée dans un dossier lié, le fichier du lien symbolique va devoir être lu par le système, interprété,

puis celui ci va devoir le fermer pour aller ouvrir le "vrai" fichier ou dossier. Il n'est donc pas très conseillé sur un serveur web par exemple où un dossier va être accédé des centaines de fois par minutes.



## LIEN DUR (HARDLINK)

Si le lien symbolique est géré par le système d'exploitation et ne dépend donc pas du système de fichier utilisé, Le lien "hard" quant à lui est directement géré par un système de fichier. Cela veut déjà dire qu'il ne fonctionnera donc pas pour tous les systèmes de fichier. Un volume monté en FAT par exemple ne passera pas, mais tous les FS sérieux disposent du hard-link (ext3, reiser, xfs, etc..).

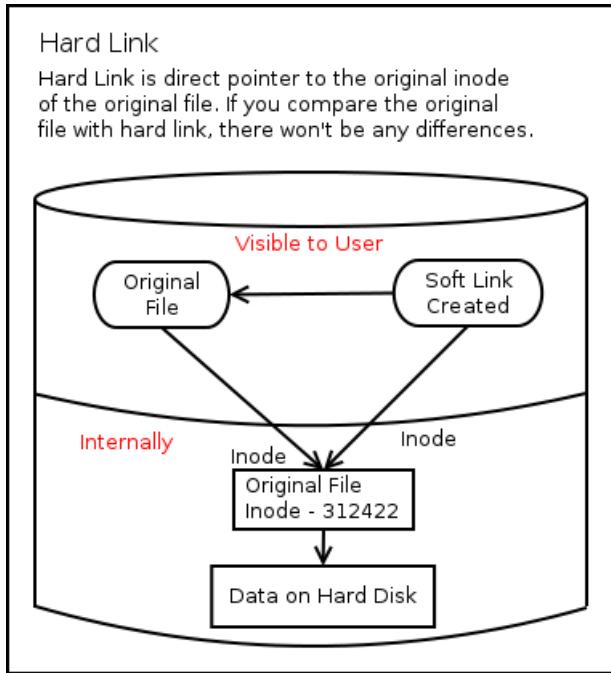
Pour bien saisir ce que sont les hard-links, il faut comprendre que, pour les systèmes de fichiers qui le supportent, tous les fichiers/dossiers sont des liens hard. En effet, il faut ici distinguer deux notions : le nom du fichier (avec son nom, son chemin, etc.) et *le bloc de données* (un paquet de 0 et de 1 localisé à une position N du disque dur). Lorsque l'on crée un fichier, le système va d'abord allouer un bloc de données sur le disque dur, puis faire un *hard-link* entre *ce bloc de données* et le nom de fichier qui est stocké ailleurs.

Ainsi, si vous avez un fichier qui s'appelle toto.txt dans le dossier /mes\_dossiers, le nom de fichier /mes\_donnees/toto.txt est en réalité un hard-link entre d'un côté le bloc de données du fichier toto.txt (les 0 et les 1) et le nom de fichier que vous lui avez donné (comprenant son chemin). Il y a donc toujours au moins 1 hard-link entre toto.txt et les données qu'il représente. Ainsi, faire un hard-link sur toto.txt, consiste en réalité à en créer un deuxième...

Cette caractéristique octroie une propriété très utile au hard-link : le bloc de données d'un fichier continue à exister (n'est pas détruit) tant qu'au moins un hard-link pointe dessus. Ainsi si j'ai, pour reprendre l'exemple précédent, mon bloc de donnée pointé par /mes\_donnees/toto.txt et

/autre\_chemin/tutu.txt. Si je détruis toto.txt, le bloc de données n'est pas détruit et pour le système, c'est comme s'il s'était toujours appelé tutu.txt et qu'il a toujours été dans le dossier /autre\_chemin. En revanche, si je fais une deuxième destruction, si je détruis tutu.txt, le bloc de données est perdu, le fichier est physiquement détruit...

En conséquence, contrairement aux liens symboliques, les hard-links ne peuvent pas être réalisés entre deux fichiers systèmes de fichiers différents, et à fortiori, entre deux volumes d'origine différente (usb, cd-rom, etc.). Autre conséquence, nous ne pouvons créer un hard-link entre deux systèmes de fichiers.



## DANS LA PRATIQUE

### CRÉATION D'UN LIEN SYMBOLIQUE (SOFT LINK)

Nous utilisons la commande ln pour réaliser un lien symbolique avec l'option -s , voici un exemple de création de lien symbolique et visualisation :

Création du fichier etc\_passwd qui pointe vers le vrai fichier /etc/passwd :

```
$ ln -s /etc/passwd etc_passwd
```

Visualisation du fichier :

```
$ ls -l etc_passwd
```

```
lrwxrwxrwx 1 user group 11 fév 5 12 :24 etc_passwd -> /etc/passwd
```

Dans l'exemple ci dessus, j'utilise la définition du chemin absolu , donc avec l'utilisation du / au début donc de la racine du système. Il est aussi possible de définir un chemin relatif , cependant

la définition du lien ne sera plus bonne si le fichier du lien symbolique est déplacé voici une démonstration.

Utilisation du chemin relatif pour la définition du fichier :

```
$ ln -s ../../etc/passwd passwd_relatif
```

Visualisation du lien :

```
$ ls -l passwd_relatif
```

```
lrwxrwxrwx 1 user group 13 fév 5 12 :27 passwd_relatif -> ../../etc/passwd
```

Création d'un répertoire :

```
$ mkdir nouveau_rep
```

Déplacement des fichiers de lien dans le répertoire :

```
$ mv passwd_relatif etc_passwd nouveau_rep
```

Visualisation

```
$ ls -l nouveau_rep
```

```
lrwxrwxrwx 1 user group 11 fév 5 12 :24 etc_passwd -> /etc/passwd
```

```
lrwxrwxrwx 1 user group 13 fév 5 12 :27 passwd_relatif -> ../../etc/passwd
```

Le (lien) fichier passwd\_relatif est cassé alors que etc\_passwd est OK

## CRÉATION D'UN LIEN DUR (HARD LINK)

Nous utilisons la commande **ln** pour réaliser un lien dur mais cette fois-ci sans argument :

Création du lien dur :

```
$ sudo ln /etc/passwd hard_link_passwd
```

Visualisation du fichier :

```
$ ls -l
```

```
lrwxrwxrwx 1 user group 11 fév 5 12 :24 etc_passwd -> /etc/passwd
```

```
-rw-r--r-- 2 root root 2653 nov 29 15 :51 hard_link_passwd
```

Nous constatons que contrairement au lien symbolique nous ne voyons plus la référence au fichier identifié avec les caractères `->`.

Cependant si nous regardons après les permissions le premier fichier, le lien symbolique à le chiffre **1** alors que le lien dur à le chiffre **2**, ceci indique que les données sur le Disque Dur ont 2 liens associés à un fichier mais qui porte deux noms différents.

Il n'est cependant pas possible de réaliser un lien symbolique entre 2 partitions ou devices :

Exemple problème de création d'un hardlink entre une clé USB (/mnt/USB\_KEY) et la partition / :

```
user@hostname :/mnt/USB_KEY $ ln /etc/passwd hd_passwd  
ln : creating hard link 'hd_passwd' => '/etc/passwd' : Invalid cross-device link
```

## RÉSUMÉ

Le lien symbolique est essentiellement un pointeur vers le fichier original et lorsque le fichier original est supprimé le lien symbolique (soft link) pointe sur rien et donc une erreur "pas de fichier ou répertoire" est signalée.

Le lien en dur (hard link) agit plus comme un miroir du fichier original, il pointe effectivement sur le même «nœud» dans le système de fichiers que le fichier d'origine, de sorte que lorsque nous supprimons le fichier original , il existe encore le même nœud du fichier d'origine dans le système de fichiers.

## 21.2 VIDEO

Télécharger permission-et-lien.ogv. Le lire avec vlc



# Chapitre 22

## PRATIQUE PERMISSIONS ET LIENS

Suite à la théorie, voici quelques exemples pratiques d'utilisation des liens symboliques et liens durs.

De plus nous reviendrons sur les permissions des fichiers

### 22.1 PERMISSIONS

#### QUELQUES FICHIERS PRÉSENTS SUR LE SYSTÈME

- -rw-r-r- 1 root root 1706 Jan 22 05 :26 /etc/passwd  
Qui peut modifier le fichier ? Qui peut lire le fichier ? Est-ce que l'utilisateur Thomas membre du group root peut modifier le fichier ?
- -rw-r— 1 root shadow 1295 Jan 22 05 :26 /etc/shadow  
Qui peut lire le fichier ? Sans être root ou utiliser sudo comment puis-je lire le fichier ?
- drwx—— 11 mysql mysql 4096 Dec 24 06 :29 /var/lib/mysql  
Est-ce un fichier ou un répertoire ? qui peut le consulter ?
- -rw-rw— 1 mysql mysql 5242880 Feb 6 2013 /var/lib/mysql/ib\_logfile0  
Si je suis membre du group mysql est-ce que je vais pouvoir lire le fichier ? attention rappelez vous les permissions de /var/lib/mysql

### 22.2 LIENS

#### LIENS SYMBOLIQUES (SOFT LINKS)

Les liens symboliques sont beaucoup plus utilisés que les liens durs (hardlink), je pense que le coté visuel du lien symbolique facilite son utilisation. De plus le lien symbolique peut être utilisé pour les répertoires ce qui n'est pas possible avec les liens durs. Voici quelques utilisations de liens symboliques :

**DÉFINITION DU BINAIRE JAVA** Il est fréquent de voir le binaire java définit avec un lien symbolique, comme ceci :

```
/usr/bin/java -> /usr/bin/java6.40
```

Il est même possible d'avoir sous Ubuntu plusieurs liens symboliques. Voici un exemple :

```
/usr/bin/java -> /etc/alternatives/java
```

mais si je regarde /etc/alternatives/java, ceci est un autre lien symbolique

```
/etc/alternatives/java -> /usr/lib/jvm/java-6-sun/jre/bin/java
```

en d'autre mot

```
/usr/bin/java -> /etc/alternatives/java -> /usr/lib/jvm/java-6-sun/jre/bin/java
```

Ceci à l'avantage de définir à l'ensemble des logiciels système d'utiliser /usr/bin/java peut importe le vrai binaire utilisé. Nous pouvons donc facilement avoir plusieurs versions de Java d'installées et basculer de l'une à l'autre sans aucune autre modification que le pointeur /usr/bin/java.

## LIBRAIRIE PARTAGÉE (SHARED LIBRAIRIE / DLL)

Un autre exemple d'utilisation de soft link est pour la définition des librairies partagées, l'équivalent des DLL. Tels que vus dans la définition des partitions, ces fichiers se trouvent dans le répertoire /lib ou /usr/lib. Si nous listons les liens symboliques dans le répertoire :

```
$ ls -l /usr/lib/ | grep ^l | head
lrwxrwxrwx 1 root root 27 oct 8 23 :51 libaccountsservice.so.0 -> libaccountsservice.so.0.0.0
lrwxrwxrwx 1 root root 17 nov 28 2011 libaften.so.0 -> libaften.so.0.0.8
lrwxrwxrwx 1 root root 20 aoû 27 16 :05 libapparmor.so.1 -> libapparmor.so.1.0.2
lrwxrwxrwx 1 root root 17 avr 3 2012 libapphb.so.2 -> libapphb.so.2.0.0
lrwxrwxrwx 1 root root 25 mar 7 2013 libappindicator3.so.1 -> libappindicator3.so.1.0.0
lrwxrwxrwx 1 root root 24 mar 7 2013 libappindicator.so.1 -> libappindicator.so.1.0.0
lrwxrwxrwx 1 root root 17 mar 20 2012 libapr-1.so.0 -> libapr-1.so.0.4.6
lrwxrwxrwx 1 root root 22 déc 18 2011 libaprutil-1.so.0 -> libaprutil-1.so.0.3.12
lrwxrwxrwx 1 root root 19 oct 17 2011 libaspell.so.15 -> libaspell.so.15.2.0
lrwxrwxrwx 1 root root 20 déc 2 2011 libasprintf.so -> libasprintf.so.0.0.0
```

Il est probable que vous n'ayez pas les mêmes fichiers mais l'idée reste identique.

Regardons les fichiers listés. L'utilisation classique est d'indiquer au logiciel d'utiliser le fichier .so avec le numéro de version "court". Si nous regardons la ligne contenant la librairie **libaften**, l'ensemble des logiciels qui ont besoin de la version 0 de la librairie libaften vont charger le fichier /usr/lib/libaften.so.0 qui est en réalité le fichier /usr/lib/libaften.so.0.0.8.

Vous constaterez qu'ils utilisent la nomenclature absolue pour nommer le fichier. Si la version **0.0.9** est disponible le fichier **libaften.so.0.0.9** sera créé et le lien symbolique sera modifié pour pointer vers ce dernier. L'avantage est que ceci est totalement transparent pour les applications qui furent compilées sur le système !

## RÉPERTOIRE DE DONNÉES VERS UN AUTRE RÉPERTOIRE

Voici le partitionnement de la 2<sup>ème</sup> partition :

```
$ df -h
```

```
/dev/md1 20G 13G 6.3G 67% /
/dev/md2 897G 6.0G 846G 1% /data
```

Si ma base de données prend trop d'espace et que par défaut elle se trouve dans le répertoire **/var/lib/db**, alors je désire la déplacer dans le répertoire **/data/db**, où j'ai plus d'espace disque dur pour stocker les informations. Idéalement il est mieux de changer la configuration du logiciel, malheureusement il n'est pas toujours possible de le faire.

Je peux créer un lien symbolique pour remplacer **/var/lib/db** afin qu'il pointe vers l'autre partition, ceci donne :

- # arrêt de la BD  
  \$ sudo /etc/init.d/DB stop (ici utiliser la bonne commande )
- # déplacement des données  
  \$ sudo mv /var/lib/db /data
- # création du lien symbolique  
  \$ sudo ln -s /data/db /var/lib/db
- # redémarrage du système  
  \$ sudo /etc/init.d/DB start (ici utiliser la bonne commande )

Et voilà le tour est joué !!!

## 22.3 VIDEO

Télécharger permission-et-lien-2.ogv. Le lire avec vlc



# Chapitre 23

## ATELIER GESTION DES PERMISSIONS ET LIENS

Quelques exercices pour être confortable (à l'aise) avec les permissions des fichiers, ainsi que sur les fichiers de liens (soft et hard).

### 23.1 QUESTIONS PERMISSIONS

- lister les permissions des fichiers /etc/passwd et /etc/shadow
- Quelle est la différence entre les 2 fichiers au niveau des permissions ( /etc/passwd et /etc/shadow ) ?
- Créer un répertoire dans votre répertoire personnel avec le nom exo-perm (`~/exo-perm`), copier les fichiers /etc/passwd et /etc/group dans le répertoire exo-perm.
- assurez-vous de ne pas être dans le répertoire, allez dans votre home (`cd ~`). À ce niveau réalisez la commande : `$ ls exo-perm`, maintenant modifiez les permissions du répertoire **exo-perm** afin que PERSONNE ne puisse exécuter le répertoire.  
Refaire la commande `$ls exo-perm` (il y aura des erreurs mais ça devrait "fonctionner").  
Essayer maintenant d'aller dans le répertoire `$ cd exo-perm`, est-ce que ça fonctionne ?
- remettre les permissions de **exo-perm** afin que ceci soit comme suit : **rwxr-xr-x**, listez les fichiers dans le répertoire. Normalement il ne devrait pas y avoir de problème.
- Modifier les permissions de **exo-perm** afin que les permissions soit comme suit : **—r-xr-x**, relister les fichiers dans le répertoire. Quelle est le comportement et pourquoi ?
- Modifier les permissions afin que vous puissiez lire et exécuter donc **r-xr-xr-x** , maintenant supprimer le fichier `~/exo-perm/passwd`. Est-ce que ceci fonctionne ?
- Afficher les groupes auxquels votre utilisateur est membre avec la commande : `id`
- Choisir un groupe dans la liste renversée et changez le propriétaire de `~/exo-perm/` avec l'un de ces groupes et modifier les permissions pour que ceci soit **r-xrwxr-x**. Refaites la suppression du fichier `~/exo-perm/passwd`.
- Modifier le propriétaire de `~/exo-perm/` afin que ce soit l'utilisateur root le propriétaire, et supprimer le fichier `~/exo-perm/passwd`.

## 23.2 QUESTIONS LIENS

- Créez le répertoire **/home/public/medias/Videos**
- Modifier le répertoire dans votre répertoire personnel **Videos** pour que ce soit un lien symbolique vers **/home/public/medias/Videos**

# Chapitre 24

## PÉRIPHÉRIQUES SOUS GNU/LINUX

### PRÉSENTATION DES PÉRIPHÉRIQUES

Nous allons faire une présentation des périphériques sous GNU/Linux, comprendre leur identification, le système de driver, comment interagir avec ces « devices ».

nous couvrirons : le répertoire /dev les types de devices, les majeurs et mineurs, lister les périphériques et informations sur les modules du noyau (drivers).

### 24.1 RÉPERTOIRE /DEV

Tel que mentionné lors de la présentation du système de fichier, le répertoire **/dev** contient la définition sous forme de fichier des périphériques disponibles sur le système. Donc si on liste les fichiers contenus dans le répertoire, nous constaterons qu'il y en a beaucoup, une petite explication s'impose afin de comprendre leur signification.

Il existe 2 types de devices :

- **blocks** : les périphériques par **blocks** transmettent ou reçoivent les informations sous forme de paquets (blocs) d'octets, d'une taille fixe : c'est par exemple le cas des supports de mémoire de masse (disquettes, disques durs...).
- **caractères** : Les périphériques de caractères ont comme caractéristiques de transmettre et recevoir les informations octet par octet : c'est par exemple le cas des ports séries ou parallèles, des modems, etc.

Pour connaître le type du device utilisez la commande **ls -l**, le premier caractère nous donne cette information, **C** pour caractère et **B** pour **block**. Voici un exemple pour le périphérique de la console et du disque dur sata :

```
utilisateur@hostname :~$ ls -l /dev/sda /dev/sda1 /dev/console
crw—— 1 root root 5, 1 Jan 27 12 :16 /dev/console
brw-rw— 1 root disk 8, 0 Jan 27 12 :16 /dev/sda
brw-rw— 1 root disk 8, 1 Jan 27 12 :16 /dev/sda1
```

Voici un tableau explicatif des noms et leurs rôles :

Fichier	Majeur	Mineur	B/C	Péphérique
/dev/mem	1	1	c	accès direct à la mémoire centrale
/dev/fd0	2	0	b	premier lecteur de disquettes
/dev/hda	3	0	b	disque maître sur le premier port IDE
/dev/hda2	3	2	b	seconde partition primaire sur ce disque
/dev/hdb	3	64	b	disque esclave sur le premier port IDE
/dev/hdb5	3	69	b	première partition logique sur ce disque
/dev/tty1	4	1	c	première console virtuelle
/dev/lp0	6	2	c	troisième port parallèle (imprimante)
/dev/sda	8	0	b	premier disque dur SCSI / SATA / USB drive , ...
/dev/sda3	8	3	b	troisième partition sur ce disque
/dev/sdb	8	16	b	deuxième disque dur SCSI / SATA / USB drive , ...
/dev/psaux	10	1	c	port PS/2 (souris)
/dev/bus/usb/001	11	0	c	Premier Device USB
/dev/scd0	11	0	b	premier CD-ROM SCSI
/dev/video0	81	0	c	Acquisition vidéo

et ainsi de suite ... la liste complète occuperait plusieurs pages !

La documentation se trouve à <https://www.kernel.org/doc/Documentation/devices.txt>.

Il existe aussi des pseudo-périphériques, qui existe que virtuellement mais qui offre de vraie fonctionnalité :

- **/dev/zero** génère des zéros.
- **/dev/random** génère de l' aléatoire.
- **/dev/null** constitue un trou noir à octets, et notamment utilisé pour se débarrasser des fichiers et des affichages.
- **/dev/loop0** permet de créer de faux périphériques de type bloc (stockage) à partir de fichiers créés avec la commande **dd**.

Le nom du fichier est indépendant du driver ou matériel utilisé. Ce sont des noms génériques, attention ceci n'est pas le cas pour tous les UNIX. FreeBSD par exemple nomme les périphériques selon le driver.

À présent que nous sommes en mesure d'identifier les périphériques grâce à leur nom, nous allons voir comment le système fait en sorte pour communiquer avec ces derniers. Prenons le cas du premier disque dur SATA sur le système, **/dev/sda**, si je liste les partitions sur le disque j'utilisera la commande **fdisk** , comme ceci :

```
utilisateur@hostname:~$ sudo fdisk -l /dev/sda
Disk /dev/sda: 60.0 GB, 60011642880 bytes
255 heads, 63 sectors/track, 7296 cylinders, total 117210240 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0xeeede9d79

      Device Boot   Start     End   Blocks  Id  System
/dev/sda1  *       2048  58593279  29295616  83  Linux
/dev/sda2          58595326 117209087 29306881   5  Extended
/dev/sda5          58595328  64475135  2939904  82  Linux swap / Solaris
/dev/sda6          64477184 117209087  26365952  83  Linux
```

J'utilise **sudo** afin d'avoir la permission de lire le périphérique.

## MAJEUR ET MINEUR

Le fichier **/dev/sda** est un fichier spécial il ne suffit pas de créer un fichier vide dans **/dev** avec le bon nom pour que ceci fonctionne. Il faut que le fichier ait les bonnes propriétés ! Il est important de définir le bon type **block** ou **caractère**, de plus il faut définir une valeur **majeur (major)** et **mineur (minor)**. Ces 2 valeurs en combinaison avec le type permet d'identifier le périphérique avec lequel nous désirons interagir. L'ensemble des communications avec le matériel est réalisé par le **Noyau (kernel)**, ce dernier conserve un tableau qui lui permet de faire la correspondance avec le périphérique.

Voici un exemple pour les disques dur sata :

8	0	<b>/dev/sda</b>	Premier disque dur dans son ensemble
8	1	<b>/dev/sda1</b>	La première partition du Premier Disque dur
8	2	<b>/dev/sda2</b>	La deuxième partition du Premier Disque dur
8	16	<b>/dev/sdb</b>	Deuxième disque dur dans son ensemble
8	17	<b>/dev/sdb1</b>	Première partition Deuxième Disque dur

Donc le **Majeur 8** représente les devices sur le BUS SCSI , SATA, ... le mineur représente le détail avec lequel nous interagissons !

Quel est le processus de création de ces fichiers ? Si je branche une clef USB, le système va créer les fichiers **/dev/sdb** et **/dev/sdb1**, **/dev/sdb2**, ... Le système incrémente automatiquement de **/dev/sda** pour prendre le prochain disponible donc **/dev/sdb** .

Ceci est magnifique et encore une fois merci d'avoir modernisé le système, grâce à **udev**, par contre j'aimerai fournir l'information, sans le système automatique. Car sous le capot il y a un processus.

Le système **udev** utilise la commande **/bin/mknod** pour faire la création des fichiers contenu dans **/dev**, voici un exemple :

```
# création du device /dev/bidon
# type    : block
# majeur : 42
# mineur : 0
$ mknod /dev/bidon b 42 0

# Documentation du kernel.
# 42 block      Demo/sample use
#
#           This number is intended for use in sample code, as
#           well as a general "example" device number. It
#           should never be used for a device driver that is being
#           distributed; either obtain an official number or use
#           the local/experimental range. The sudden addition or
#           removal of a driver with this number should not cause
#           ill effects to the system (bugs excepted.)
```

Ici nous réalisons la création dans **/dev** cependant ceci pourrait être n'importe où. Ceci est à titre INFORMATIF, car si sur un système moderne vous êtes obligé de réaliser cette opération ça veut dire qu'il y a VRAIMENT un problème quelque part. Je doute que cette opération corrige le problème car l'automatisation du processus de création des **devices** fonctionne très bien et ceci à chaud . Cependant je trouve intéressant de mieux connaître le processus de création des fichiers.

## 24.2 LISTER LES PÉRIPHÉRIQUES

Il est bien de savoir où seront situés les fichiers pour communiquer avec les périphériques. Mais si je ne vois pas le fichier de mon périphérique comment peut-on le lister ? Nous allons donc voir, comment avoir la liste des périphériques du système, nous en profiterons pour voir l'association des "driver/pilote" qui gère le matériel.

## LISTER LES PÉRIPHÉRIQUES PCI

La commande **lspci** nous permet de lister les cartes PCI ou les devices intégrés. Il serait possible qu'une carte soit insérée dans le système mais que le système d'exploitation ne soit pas en mesure de communiquer avec, dans ce cas elle serait listée, mais il y aurait le mot **unknow** de présent. Voici un exemple de la commande lspci :

```

1 $ lspci
2 00:00.0 Host bridge: Intel Corporation Mobile 945GM/PM/GMS, 943/940GML and 945GT Express Memory Controller Hub (rev 03)
3 00:02.0 VGA compatible controller: Intel Corporation Mobile 945GM/GMS, 943/940GML Express Integrated Graphics Controller (rev 03)
4 00:02.1 Display controller: Intel Corporation Mobile 945GM/GMS/GME, 943/940GML Express Integrated Graphics Controller (rev 03)
5 00:1b.0 Audio device: Intel Corporation NM10/ICH7 Family High Definition Audio Controller (rev 01)
6 00:1c.0 PCI bridge: Intel Corporation NM10/ICH7 Family PCI Express Port 1 (rev 01)
7 00:1c.1 PCI bridge: Intel Corporation NM10/ICH7 Family PCI Express Port 2 (rev 01)
8 00:1c.2 PCI bridge: Intel Corporation NM10/ICH7 Family PCI Express Port 3 (rev 01)
9 00:1d.0 USB controller: Intel Corporation NM10/ICH7 Family USB UHCI Controller #1 (rev 01)
10 00:1d.1 USB controller: Intel Corporation NM10/ICH7 Family USB UHCI Controller #2 (rev 01)
11 00:1d.2 USB controller: Intel Corporation NM10/ICH7 Family USB UHCI Controller #3 (rev 01)
12 00:1d.3 USB controller: Intel Corporation NM10/ICH7 Family USB UHCI Controller #4 (rev 01)
13 00:1d.7 USB controller: Intel Corporation NM10/ICH7 Family USB2 EHCI Controller (rev 01)
14 00:1e.0 PCI bridge: Intel Corporation 82801 Mobile PCI Bridge (rev e1)
15 00:1f.0 ISA bridge: Intel Corporation 82801GBM (ICH7-M) LPC Interface Bridge (rev 01)
16 00:1f.2 IDE interface: Intel Corporation 82801GBM/GHM (ICH7-M Family) SATA Controller [IDE mode] (rev 01)
17 00:1f.3 SMBus: Intel Corporation NM10/ICH7 Family SMBus Controller (rev 01)
18 03:01.0 CardBus bridge: 02 Micro, Inc. OZ601/6912/711E0 CardBus/SmartCardBus Controller (rev 40)
19 09:00.0 Ethernet controller: Broadcom Corporation NetXtreme BCM5752 Gigabit Ethernet PCI Express (rev 02)
20 0c:00.0 Network controller: Intel Corporation PRO/Wireless 3945ABG [Golan] Network Connection (rev 02)
```

Nous constatons que j'ai 2 cartes réseaux présentes (ligne 19 et 20) :

- Ethernet controller : Broadcom Corporation (carte réseau filaire)
- Network controller : Intel Corporation (carte réseau wireless)

J'ai aussi la carte graphique et audio (ligne 3 et 5) :

- VGA compatible controller : Intel Corporation Mobile 945GM/GMS
- Audio device : Intel Corporation NM10/ICH7

La commande **lspci** peut être exécutée par le simple utilisateur, cependant si vous désirez avoir plus de détails utilisez l'option **-v**. Voir tableau ci dessous. Les chiffres indiquent le nombre de lignes renvoyées sur la sortie standard. Ces tests ont été faits sur un autre ordinateur.

Commande\user	Utilisateur normal	Administrateur root
lspci	25	25
lspci -v	150	182
lspci -vv	215	405
lspci -vvv	219	409

Je n'ai pas copié l'ensemble des lignes retournées car il y a beaucoup de lignes. Je me concentre sur les 4 périphériques mentionnés plus tôt :

```

1 # lspci avec verboce
2 $ sudo lspci -v
3 00:00.0 Host bridge: Intel Corporation Mobile 945GM/PM/GMS, 943/940GML and 945GT Express Memory Controller Hub (rev 03)
4     Subsystem: Dell Device 01c2
5     Flags: bus master, fast devsel, latency 0
6     Capabilities: [e0] Vendor Specific Information: Len=09 <?>
7     Kernel driver in use: agpgart-intel
8
9 00:02.0 VGA compatible controller: Intel Corporation Mobile 945GM/GMS, 943/940GML Express Integrated Graphics Controller (rev 03) (prog-if 00 [VGA]
10    Subsystem: Dell Device 01c2
11    Flags: bus master, fast devsel, latency 0, IRQ 16
12    Memory at eff00000 (32-bit, non-prefetchable) [size=512K]
13    I/O ports at eff8 [size=8]
14    Memory at d0000000 (32-bit, prefetchable) [size=256M]
15    Memory at efec0000 (32-bit, non-prefetchable) [size=256K]
16    Expansion ROM at <unassigned> [disabled]
17    Capabilities: [90] MSI: Enable- Count=1/1 Maskable- 64bit-
18    Capabilities: [d0] Power Management version 2
19    Kernel driver in use: i915
20
21 00:02.1 Display controller: Intel Corporation Mobile 945GM/GMS/GME, 943/940GML Express Integrated Graphics Controller (rev 03)
22    Subsystem: Dell Device 01c2
23    Flags: bus master, fast devsel, latency 0
24    Memory at eff80000 (32-bit, non-prefetchable) [size=512K]
25    Capabilities: [d0] Power Management version 2
26
27 00:1b.0 Audio device: Intel Corporation NM10/ICH7 Family High Definition Audio Controller (rev 01)
28    Subsystem: Dell Device 01c2
29    Flags: bus master, fast devsel, latency 0, IRQ 43
30    Memory at efdbc000 (64-bit, non-prefetchable) [size=16K]
31    Capabilities: [50] Power Management version 2
32    Capabilities: [60] MSI: Enable+ Count=1/1 Maskable- 64bit+
33    Capabilities: [70] Express Root Complex Integrated Endpoint, MSI 00
34    Capabilities: [100] Virtual Channel
35    Capabilities: [130] Root Complex Link
36    Kernel driver in use: snd_hda_intel
37
38 [[ OUTPUT COUPÉ ]]
39
40 09:00.0 Ethernet controller: Broadcom Corporation NetXtreme BCM5752 Gigabit Ethernet PCI Express (rev 02)
41    Subsystem: Dell Latitude D620
42    Flags: bus master, fast devsel, latency 0, IRQ 45
43    Memory at efcf0000 (64-bit, non-prefetchable) [size=64K]
44    Expansion ROM at <ignored> [disabled]
45    Capabilities: [48] Power Management version 2
46
47    Capabilities: [50] Vital Product Data
48    Capabilities: [58] MSI: Enable+ Count=1/8 Maskable- 64bit+
49    Capabilities: [d0] Express Endpoint, MSI 00
50    Capabilities: [100] Advanced Error Reporting
51    Capabilities: [13c] Virtual Channel
52    Kernel driver in use: tg3
53
54 0c:00.0 Network controller: Intel Corporation PRO/Wireless 3945ABG [Golan] Network Connection (rev 02)
55    Subsystem: Intel Corporation Device 1020
56    Flags: bus master, fast devsel, latency 0, IRQ 44
57    Memory at edff0000 (32-bit, non-prefetchable) [size=4K]
58    Capabilities: [c8] Power Management version 2
59    Capabilities: [d0] MSI: Enable+ Count=1/1 Maskable- 64bit+
60    Capabilities: [e0] Express Legacy Endpoint, MSI 00
61    Capabilities: [100] Advanced Error Reporting
62    Capabilities: [140] Device Serial Number 00-18-de-ff-ff-17-93-6f
63    Kernel driver in use: iwl3945

```

J'aimerai particulièrement porter votre attention sur les lignes contenant l'information "**Kernel driver in use**", ceci nous informe quel « driver/pilote » le système utilise pour communiquer avec le périphérique. Si nous reprenons nos **4 devices** mentionnés plus tôt :

- Ethernet controller : Broadcom Corporation (carte réseau filaire) : **tg3**
- Network controller : Intel Corporation (carte réseau wireless) : **iwl3945**
- VGA compatible controller : Intel Corporation Mobile 945GM/GMS : **i915**
- Audio device : Intel Corporation NM10/ICH7 : **snd\_hda\_intel**

Nous reviendrons sur les drivers très bientôt. J'aimerai que nous continuons à lister les périphériques du système !

## LISTER ET IDENTIFIER LES PÉRIPHÉRIQUES USB

/!\ Assurez vous que tous les périphériques sont connectés et alimentés pour pouvoir les lister.

Les périphériques sont généralement identifiés par une paire de nombres hexadécimaux, comme ceci : 045e :0779.

Les 4 premiers chiffres représentent l'ID du vendeur (045e = Microsoft).

Les 4 derniers chiffres représentent l'ID du périphérique (0779 = LifeCam HD 3000).

Avec la prolifération de périphérique USB, il est important d'être en mesure de les lister. Il est peut-être encore plus important d'être en mesure de le faire, car ces derniers peuvent être branché à « Chaud » ! Nous retrouvons la même syntaxe de commande, **lsusb** et avec l'option **-v** nous aurons plus d'informations. Mais attention **-v** donne beaucoup plus d'informations ! Le concept reste le même utilisez **sudo** avec **-v** afin d'avoir l'ensemble de l'information.

```
$ lsusb
Bus 001 Device 003: ID 0930:6545 Toshiba Corp. Kingston DataTraveler 102 Flash Drive / HEMA Flash Drive 2 GB / PNY Attache 4GB Stick
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 005 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 004 Device 002: ID 0461:4d22 Primax Electronics, Ltd
Bus 004 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 003 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 002 Device 004: ID 0b97:7762 O2 Micro, Inc. Oz776 SmartCard Reader
Bus 002 Device 003: ID 0b97:7761 O2 Micro, Inc. Oz776 1.1 Hub
Bus 002 Device 002: ID 413c:a005 Dell Computer Corp. Internal 2.0 Hub
Bus 002 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
```

Si nous regardons dans l'exemple ci-dessus j'ai :

- Une **clé USB de données branchée** : Bus 001 Device 003 : ID 0930 :6545 Toshiba Corp. Kingston DataTraveler 102 Flash Drive / HEMA Flash Drive 2 GB / PNY Attache 4GB Stick
- **une souris** ( faut le savoir ) : Bus 004 Device 002 : ID 0461 :4d22 Primax Electronics, Ltd
- **Un lecteur de carte interne** : Bus 002 Device 004 : ID 0b97 :7762 O2 Micro, Inc. Oz776 SmartCard Reader

Comment ai-je fait pour savoir que Primax Electronics est une souris ? Avec l'information transmise par **lsusb** je ne pouvais pas le savoir mais avec l'option **-v** j'ai obtenu l'information. Il est possible de transmettre en paramètre un device spécifique à interroger pour avoir l'ensemble de l'information avec l'option **-s [bus] :[numero-device]**.

Voici la commande : **sudo lspci -v -s 004 :002** pour avoir plus d'informations sur la souris.

```

1 # Affichage information sur la souris.
2 $ sudo lsusb -v -s 004:002
3
4 Bus 004 Device 002: ID 0461:4d22 Primax Electronics, Ltd
5 Device Descriptor:
6     bLength          18
7     bDescriptorType   1
8     bcdUSB           2.00
9     bDeviceClass      0 (Defined at Interface level)
10    bDeviceSubClass    0
11    bDeviceProtocol    0
12    bMaxPacketSize0    8
13
14    idVendor          0x0461 Primax Electronics, Ltd
15    idProduct          0x4d22
16    bcdDevice          2.00
17    iManufacturer      0
18    iProduct           2 USB Optical Mouse
19    iSerial            0
20    bNumConfigurations 1
21 Configuration Descriptor:
22     bLength          9
23     bDescriptorType   2
24     wTotalLength      34
25     bNumInterfaces    1
26     bConfigurationValue 1
27     iConfiguration     0
28     bmAttributes       0xa0
29         (Bus Powered)
30     Remote Wakeup
31     MaxPower          100mA
32 Interface Descriptor:
33     bLength          9
34     bDescriptorType   4
35     bInterfaceNumber  0
36     bAlternateSetting 0
37     bNumEndpoints     1
38     bInterfaceClass    3 Human Interface Device
39     bInterfaceSubClass 1 Boot Interface Subclass
40     bInterfaceProtocol 2 Mouse
41     iInterface         0
42     HID Device Descriptor:
43         bLength          9
44         bDescriptorType   33
45         bcdHID           1.11
46         bCountryCode      0 Not supported
47         bNumDescriptors   1
48         bDescriptorType   34 Report
49         wDescriptorLength 52
50
51 Report Descriptors:
52     ** UNAVAILABLE **
53 Endpoint Descriptor:
54     bLength          7
55     bDescriptorType   5
56     bEndpointAddress  0x81 EP 1 IN
57     bmAttributes       3
58         Transfer Type   Interrupt
59         Synch Type      None
60         Usage Type       Data
61     wMaxPacketSize     0x0004 1x 4 bytes
62     bInterval          10
63 Device Status: 0x0000
64     (Bus Powered)

```

Comme vous pouvez le constater à la ligne 17 nous voyons clairement que ceci est un périphérique de type **Optical Mouse**, de plus à la ligne 37 le type du périphérique est **Human Interface**. Voici un exemple d'une commande pour extraire l'information de **lsub** avec plusieurs critères :

```

1 # utilisation de egrep
2 $ sudo lsusb -v | egrep 'bInterfaceClass|iProduct|idVendor|^Bus' | tr -s " "
3 Bus 001 Device 003: ID 0930:6545 Toshiba Corp. Kingston DataTraveler 102 Flash Drive / HEMA Flash Drive 2 GB / PNY Attache 4GB Stick
4   idVendor 0x0930 Toshiba Corp.
5     iProduct 2 DataTraveler G3
6     bInterfaceClass 8 Mass Storage
7 Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
8   idVendor 0x1d6b Linux Foundation
9     iProduct 2 EHCI Host Controller
10    bInterfaceClass 9 Hub
11 Bus 005 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
12   idVendor 0x1d6b Linux Foundation
13     iProduct 2 UHCI Host Controller
14     bInterfaceClass 9 Hub
15 Bus 004 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
16   idVendor 0x1d6b Linux Foundation
17     iProduct 2 UHCI Host Controller
18     bInterfaceClass 9 Hub
19 Bus 003 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
20   idVendor 0x1d6b Linux Foundation
21     iProduct 2 UHCI Host Controller
22     bInterfaceClass 9 Hub
23 Bus 002 Device 004: ID 0b97:7762 02 Micro, Inc. 0z776 SmartCard Reader
24   idVendor 0x0b97 02 Micro, Inc.
25     iProduct 2 02Micro CCID SC Reader
26     bInterfaceClass 11 Chip/SmartCard
27 Bus 002 Device 003: ID 0b97:7761 02 Micro, Inc. 0z776 1.1 Hub
28   idVendor 0x0b97 02 Micro, Inc.
29     iProduct 0
30     bInterfaceClass 9 Hub
31 Bus 002 Device 002: ID 413c:a005 Dell Computer Corp. Internal 2.0 Hub
32   idVendor 0x413c Dell Computer Corp.
33     iProduct 0
34     bInterfaceClass 9 Hub
35 Bus 002 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
36   idVendor 0x1d6b Linux Foundation
37     iProduct 2 UHCI Host Controller
38     bInterfaceClass 9 Hub
39

```

Ici j'utilise **egrep** ou **grep -E**, avec un pipe « | » muni de plusieurs critères, pour qu'il retourne plusieurs lignes. Ceci me permet de ne conserver que l'essentiel.

```
sudo lsusb -v | grep -iE 'binterfaceclass|^Cproduct|idvendor|^Bus|iserial' |tr -s " "
```

Grâce à **lsub** je peux donc voir "live" les équipements que vous branchez. De plus nous pouvons récolter une multitude d'informations. Nous reviendrons sur cette commande quand nous traiterons **udev**. Les informations retournées par **lsub** nous permettront d'identifier un périphérique et de réaliser une action spécifique, à suivre.

## PÉRIPHÉRIQUES DE TYPE BLOCK

Finalement nous avons plusieurs possibilités.

- On peut lister le répertoire /dev :
  - **ls -l /dev | grep ^b** qui fournit les périphériques et leurs partitions.
- On peut lister les périphériques de type « block » par la commande :
- **lsscsi -d** qui fournit les périphériques et les numéros majeurs et mineurs. À comparer par la commande suivante.
- On peut aussi lister les périphériques de type « block » par la commande :
- **lsblk** qui fournit les périphériques et leurs partitions, les numéros majeurs et mineurs et les points de montage.

```
pat@amd-a10:~$ ls -l /dev | grep ^b
brw-rw---- 1 root disk      8,   0 janv. 12 15:27 sda
brw-rw---- 1 root disk      8,   1 janv. 12 15:27 sda1
brw-rw---- 1 root disk      8,   2 janv. 12 15:27 sda2
brw-rw---- 1 root disk      8,   3 janv. 12 15:27 sda3
brw-rw---- 1 root disk      8,   4 janv. 12 15:27 sda4
brw-rw---- 1 root disk      8,   5 janv. 12 15:27 sda5
brw-rw---- 1 root disk      8,   6 janv. 12 15:27 sda6
brw-rw---- 1 root disk      8,   7 janv. 12 15:27 sda7
brw-rw---- 1 root disk      8,  16 janv. 12 15:27 sdb
brw-rw---- 1 root disk      8,  18 janv. 12 17:46 sdb2
brw-rw---- 1 root disk      8,  32 janv. 12 15:27 sdc
brw-rw---- 1 root disk      8,  33 janv. 12 15:27 sdc1
brw-rw---- 1 root disk      8,  48 janv. 12 15:27 sdd
brw-rw---- 1 root disk      8,  49 janv. 12 15:27 sdd1
brw-rw----+ 1 root cdrom    11,   0 janv. 12 15:27 sr0
```

```
pat@amd-a10:~$ lsscsi -d
[0:0:0:0]    disk    ATA      WDC WD10EZEX-00B 1A01  /dev/sda [8:0]
[1:0:0:0]    disk    ATA      WDC WD10EZEX-00B 1A01  /dev/sdb [8:16]
[3:0:0:0]    cd/dvd  HL-DT-ST BD-RE BH16NS40  1.01  /dev/sr0 [11:0]
[6:0:0:0]    disk    USB DISK 2.0     PMAP  /dev/sdc [8:32]
[7:0:0:0]    disk    Generic  STORAGE DEVICE 0902  /dev/sdd [8:48]
```

---

```
pat@amd-a10:~$ lsblk
NAME  MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda    8:0    0 931,5G  0 disk
└─sda1  8:1    0 234,8G  0 part
└─sda2  8:2    0 499,7M  0 part
└─sda3  8:3    0 93,1G  0 part /boot
└─sda4  8:4    0    1K  0 part
└─sda5  8:5    0 279,4G  0 part /
└─sda6  8:6    0 14,9G  0 part [SWAP]
└─sda7  8:7    0 308,8G  0 part /data2
sdb    8:16   0 931,5G  0 disk
└─sdb2  8:18   0 473,2G  0 part /media/pat/data
sdc    8:32   1 29,9G  0 disk
└─sdc1  8:33   1 29,9G  0 part /media/pat/MA_CLE_32
sdd    8:48   1 121,3M  0 disk
└─sdd1  8:49   1 121,2M  0 part /media/pat/MA_CARTE_SD
sr0    11:0   _ 1 1024M 0 rom
```

## 24.3 LISTER LES INFORMATIONS MATÉRIELLES DU SYSTÈME

Maintenant que nous avons vu comment collecter l'information sur les périphériques "amovibles" et les cartes "d'extention" (PCI), voyons les informations systèmes. Il est possible d'extraire l'information de la carte mère, du CPU, de la mémoire, etc... Nous allons voir l'application **dmidecode** qui nous permet de réaliser cette opération, c'est la même application qui est utilisée par **OcsInventory/GLPI**.

**dmidecode** retourne les informations **SMBios** (**System Management Bios**). Voici à quoi ressemble le résultat. J'ai coupé le « output » pour ne conserver que les parties pertinentes, je vous invite à l'essayer sur votre système :

```

1 $ sudo dmidecode
2 [[ OUTPUT COUPÉ ]]
3 Handle 0x0000, DMI type 0, 24 bytes
4 BIOS Information
5     Vendor: Dell Inc.
6     Version: A09
7     Release Date: 04/03/2008
8     Address: 0xF0000
9     Runtime Size: 64 kB
10    ROM Size: 2048 kB
11    Characteristics:
12        ISA is supported
13        PCI is supported
14        PC Card (PCMCIA) is supported
15        PNP is supported
16        BIOS is upgradeable
17        BIOS shadowing is allowed
18        Boot from CD is supported
19        Selectable boot is supported
20        3.5''/720 kB floppy services are supported (int 13h)
21        Print screen service is supported (int 5h)
22        8042 keyboard services are supported (int 9h)
23        Serial services are supported (int 14h)
24        Printer services are supported (int 17h)
25        CGA/mono video services are supported (int 10h)
26        ACPI is supported
27        USB legacy is supported
28        AGP is supported
29        Smart battery is supported
30        BIOS boot specification is supported
31        Function key-initiated network boot is supported
32        Targeted content distribution is supported
33        BIOS Revision: 0.9
34        Firmware Revision: 0.9
35
36 Handle 0x0100, DMI type 1, 27 bytes
37 System Information
38     Manufacturer: Dell Inc.
39     Product Name: Latitude D630
40     Version: Not Specified
41     Serial Number: 3ZGBXF1
42     UUID: 44454C4C-5A00-1047-8042-B3C04F584631
43     Wake-up Type: Power Switch
44     SKU Number: Not Specified
45     Family:
46
47 Handle 0x0200, DMI type 2, 9 bytes
48 Base Board Information
49     Manufacturer: Dell Inc.
50     Product Name: 0KU184
51     Version:
52     Serial Number: .3ZGBXF1.CN1296182R7975.
53     Asset Tag:
54
55 Handle 0x0300, DMI type 3, 13 bytes
56 Chassis Information
57     Manufacturer: Dell Inc.
58     Type: Portable
59     Lock: Not Present
60     Version: Not Specified

```

```

61     Serial Number: 3ZGBXF1
62     Asset Tag: Not Specified
63     Boot-up State: Safe
64     Power Supply State: Safe
65     Thermal State: Safe
66     Security Status: None
67
68 Processor Information
69     Socket Designation: Microprocessor
70     Type: Central Processor
71     Family: Core 2 Duo
72     Manufacturer: Intel
73     ID: FB 06 00 00 FF FB EB BF
74     Signature: Type 0, Family 6, Model 15, Stepping 11
75     Flags:
76         FPU (Floating-point unit on-chip)
77         VME (Virtual mode extension)
78         DE (Debugging extension)
79         PSE (Page size extension)
80         TSC (Time stamp counter)
81         MSR (Model specific registers)
82         PAE (Physical address extension)
83         MCE (Machine check exception)
84         CX8 (CMPXCHG8 instruction supported)
85         APIC (On-chip APIC hardware supported)
86         SEP (Fast system call)
87         MTRR (Memory type range registers)
88         PGE (Page global enable)
89         MCA (Machine check architecture)
90         CMOV (Conditional move instruction supported)
91         PAT (Page attribute table)
92         PSE-36 (36-bit page size extension)
93         CLFSH (CLFLUSH instruction supported)
94         DS (Debug store)
95         ACPI (ACPI supported)
96         MMX (MMX technology supported)
97         FXSR (FXSAVE and FXSTOR instructions supported)
98         SSE (Streaming SIMD extensions)
99         SSE2 (Streaming SIMD extensions 2)
100        SS (Self-snoop)
101        HTT (Multi-threading)
102        TM (Thermal monitor supported)
103        PBE (Pending break enabled)
104    Version: Not Specified
105    Voltage: 3.3 V
106    External Clock: 200 MHz
107    Max Speed: 2200 MHz
108    Current Speed: 2200 MHz
109
110    Status: Populated, Enabled
111    Upgrade: None
112    L1 Cache Handle: 0x0700
113    L2 Cache Handle: 0x0701
114    L3 Cache Handle: Not Provided
115    Serial Number: Not Specified
116    Asset Tag: Not Specified
117    Part Number: Not Specified
118    Core Count: 2
119    Core Enabled: 2
120    Thread Count: 2
121    Characteristics:
122        64-bit capable
123    [[ OUTPUT COUPÉ ]]
```

Si vous l'avez exécutée sur votre système vous constatez qu'il y a beaucoup, beaucoup d'informations. Il faut avoir une idée de ce que l'on cherche quand on utilise cette commande. J'utilise cette commande quand je désire savoir :

- Le numéro de série du système ainsi que le manufacturier
- Connaître les flags du disque dur afin de savoir s'il support nativement la virtualisation
- Savoir combien de slots de mémoire sont utilisés, quelle est la taille de chaque barrette et savoir s'il y en a de libre.

## LES PÉRIPHÉRIQUES ET LE NOYAU (KERNEL)

Tel que mentionné dans la section sur le système d'exploitation, le noyau (kernel) Linux réalise le lien entre le matériel et le logiciel. Le concept du driver est un peu différent sous Linux de part la nature open-source et aussi par le fait que les fabricants de matériel n'offrent pas toujours les pilotes pour leurs périphériques.

Une particularité sous Linux est que le driver est souvent générique. Si je prends le cas des cartes réseaux, je peux avoir plusieurs différents fournisseurs. Mais ils utilisent tous le même driver. Simplement parce que lors de la détection du matériel, le kernel Linux regarde le chipset qui est utilisé et choisit le pilote associé. Que les cartes aient l'étiquette Intel, Dell, IBM ou Ziatugs si elles supportent les mêmes instructions je vois pas l'intérêt d'avoir plusieurs drivers. Ceci s'applique pour l'ensemble des périphériques ! Nous retrouvons aussi des drivers "génériques" qui offrent parfois moins de performances mais qui nous permettent de démarrer le système. Nous retrouvons ce cas particulièrement pour la carte graphique, ceci nous permet tout de même d'avoir un affichage et d'ajuster par la suite le pilote.

Une grande majorité de drivers sous GNU/Linux sont libres ! Bien entendu ils en existent de plus libre que d'autres et parfois on a même le choix. Si nous prenons le cas de Nvidia, la compagnie offre un driver propriétaire pour ces cartes graphiques. Bien entendu ce driver permet d'utiliser l'ensemble des fonctionnalités de la carte avec l'accélération 3D la plus optimale car la compagnie connaît l'ensemble des spécifications de la carte et possède la bible pour communiquer avec cette dernière. Par contre il existe aussi un driver libre, cependant moins performant en terme de 3D et autres, parce que les développeurs non pas accès à la bible de communication. Ils sont donc obligés de faire du reverse engenering pour concevoir leur pilote. Il existe d'autres cas comme les cartes réseaux wireless *broadcom* qui furent pendant un temps fermés, mais ces dernières années les compagnies ouvrent de plus en plus leurs drivers.

Avec Ubuntu, le kernel vient avec des drivers propriétaires, si vous voulez être sûr de ne pas en avoir je vous conseille d'utiliser :

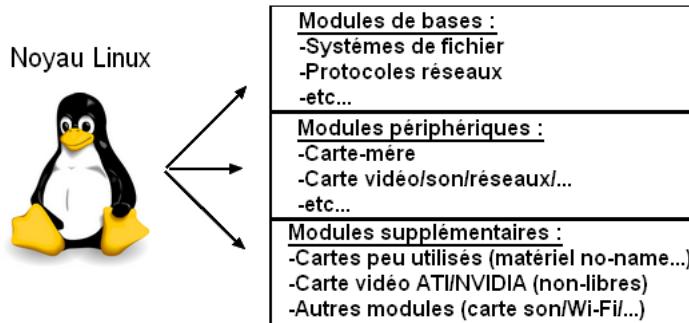
<http://www.gnewsense.org/>, mais attention faudra aussi s'assurer que votre matériel est totalement supporté, principalement la carte réseau wireless.

## 24.4 LE NOYAU ET SES MODULES

Le noyau Linux nous permet de charger à chaud des modules. Ces modules sont l'équivalent de drivers, en fait ces modules nous permettent de rajouter de nouvelles fonctionnalités au noyau afin de pouvoir :

- communiquer avec un périphérique,
- réaliser des opérations au niveau système, ...

Si nous prenons le cas de système GNU/Linux embarqué, cette fonctionnalité est enlevée et l'ensemble des modules et fonctionnalités sont directement compilés dans le noyau sans les charger. Pour les systèmes embarqués ceci est plus performant. Dans le cas de desktop nous sommes susceptibles d'ajouter et enlever du matériel plus facilement, il est donc plus pratique d'avoir ces modules qui se chargent et déchargent au besoin. Voici un petit schéma :



## LISTER LES MODULES (LSMOD)

Les distributions "modernes" offrent une panoplie de modules disponibles qui n'attendent qu'à être utilisés. Si nous regardons actuellement nous avons déjà une liste de modules utilisés. Pour lister les modules utilisés on dispose de la commande **lsmod** (liste module) :

```

1 # Liste les modules
2 $ lsmod
3 nfnetlink_acct      3772  0
4 xt_connmark         1853  2
5 iptable_nat          3366  0
6 nf_conntrack_ipv4    9206  3
7 nf_defrag_ipv4       1411  1 nf_conntrack_ipv4
8 nf_nat_ipv4          3576  1 iptable_nat
9 nf_nat               12343  2 nf_nat_ipv4,iptable_nat
10 nf_conntrack        73690  5 nf_nat,nf_nat_ipv4,xt_connmark,iptable_nat,nf_conntrack_ipv4
11 iptable_mangle       1592  1
12 iptable_filter       1496  0
13 ip_tables            17282  3 iptable_filter,iptable_mangle,iptable_nat
14 x_tables             17359  4 ip_tables,iptable_filter,xt_connmark,iptable_mangle
15 nfnetlink            4301  2 nfnetlink_acct
16 joydev                9671  0
17 snd_hda_codec_idt    37852  1
18 pcmcia              45396  0
19 coretemp             6334  0
20 psmouse              85428  0
21 yenta_socket          32385  0
22 kvm                  389093  0
23 arc4                  2008  2
24 iwl3945              55252  0
25 snd_hda_intel         36904  0
26 snd_hda_codec         149569  2 snd_hda_codec_idt,snd_hda_intel
27 snd_hwdep             6340  1 snd_hda_codec
28 snd_pcm               77709  2 snd_hda_codec,snd_hda_intel
29 gpio_ich              4536  0
30 iTCO_wdt              5407  0
31 pcmcia_rsrc           9112  1 yenta_socket
32 iTCO_vendor_support   1937  1 iTCO_wdt
33 snd_page_alloc         7242  2 snd_pcm,snd_hda_intel
34 iwlegacy              49297  1 iwl3945
35 snd_timer              18726  1 snd_pcm
36 i915                  656950  3
37 snd                   59173  6 snd_hwdep,snd_timer,snd_hda_codec_idt,snd_pcm,snd_hda_codec,snd_hda_intel
38 dell_laptop            8859  0
39 pcmcia_core            14264  3 pcmcia,pcmcia_rsrc,yenta_socket
40 dcdbas                 6463  1 dell_laptop
41 microcode              15024  0
42 dell_wmi                1493  0
43 sparse_keymap          3154  1 dell_wmi
44 evdev                  10989  11
45 mac80211              466554  2 iwl3945,iwlegacy
46 serio_raw              5049  0
47 tg3                   155658  0
48 soundcore              5450  1 snd

49 pcspkr                2035  0
50 drm_kms_helper         36286  1 i915
51 cfg80211              412854  3 iwl3945,iwlegacy,mac80211
52 i2c_i801                11277  0
53 lpc_ich                 13120  0
54 ptp                   8276  1 tg3
55 pps_core                8961  1 ptp
56 libphy                  20342  1 tg3
57 drm                    238206  4 i915,drm_kms_helper
58 i2c_algo_bit             5399  1 i915
59 i2c_core                 24164  5 drm,i915,i2c_i801,drm_kms_helper,i2c_algo_bit
60 rfkill                  15651  3 cfg80211
61 thermal                  8532  0
62 shpchp                  25465  0
63 wmi                     8419  1 dell_wmi
64 button                  4677  1 i915
65 intel_agp              10880  1 i915

```

```

65 intel_agp      10880  1 i915
66 intel_gtt      12664  3 i915,intel_agp
67 video          11196  1 i915
68 battery         6837   0
69 ac              3332   0
70 acpi_cpufreq    10779  1
71 processor       24999  3 acpi_cpufreq
72 nfs             191701 0
73 lockd          76942  1 nfs
74 sunrpc         231063  2 nfs,lockd
75 fscache        47028  1 nfs
76 ext4           474187 1
77 crc16          1367   1 ext4
78 mbcache        6082   1 ext4
79 jbd2           83504  1 ext4
80 sd_mod         30789  2
81 sr_mod         14898  0
82 cdrom          34848  1 sr_mod
83 ata_generic    3410   0
84 pata_acpi      3395   0
85 ata_piix       25088  1
86 firewire_ohci  31877  0
87 libata         170792 3 pata_acpi,ata_generic,ata_piix
88 scsi_mod       130669  3 libata,sd_mod,sr_mod
89 firewire_core  52259  1 firewire_ohci
90 crc_itu_t      1371   1 firewire_core
91 uhci_hcd       24795  0
92 ehci_pci       4000   0
93 ehci_hcd       59220  1 ehci_pci
94 usbcore        179880 3 uhci_hcd,ehci_hcd,ehci_pci
95 usb_common     1656   1 usbcore

```

Je n'ai volontairement pas coupé le résultat afin de bien visualiser l'ensemble des modules qui sont chargés. Comme nous pouvons le constater ceci est long et pourtant nous n'avons pas beaucoup de périphériques. Je vais donc mettre en lumière quelques modules afin de démontrer que lorsque l'on parle de module ceci ne correspond pas obligatoirement à un périphérique mais bien à une fonctionnalité :

coretemp : Intel Core temperature monitor

gpio\_ich : GPIO interface for Intel ICH series

pcmcia : PCMCIA Driver Services

pcspkr : PC Speaker beeper driver

battery et ac : ACPI Battery Driver et ACPI AC Adapter Driver

ext4 : Fourth Extended Filesystem , type du file système équivalent à fat32 ou ntfs

uhci\_hcd : USB Universal Host Controller Interface driver

ehci\_hcd : USB 2.0 'Enhanced' Host Controller (EHCI) Driver

Bien entendu je n'ai pas listé l'ensemble des drivers car ils sont nombreux. L'objectif ici était surtout de démontrer que les modules ne sont pas obligatoirement associés à un périphérique physique, mais peut être une extension de fonctionnalité. Vous me demanderez peut-être où puis-je trouver l'information sur les modules qui sont chargés ? Par besoin d'aller sur un site web, il existe une commande **modinfo**, voici un exemple avec le module coretemp

```

1 # exemple de modinfo pour afficher l'information sur un module.
2 $ modinfo coretemp
3 filename:      /lib/modules/3.12.1-1-ARCH/kernel/drivers/hwmon/coretemp.ko.gz
4 license:       GPL
5 description:   Intel Core temperature monitor
6 author:        Rudolf Marek <r.marek@assembler.cz>
7 alias:         x86cpu:vendor:0000:family:*:model:*:feature:*00E7*
8 depends:
9intree:        Y
10vermagic:     3.12.1-1-ARCH SMP preempt mod_unload modversions
11parm:          tmax:TjMax value in degrees Celsius (int)
12

```

Comme vous pouvez le constater, il y a la licence, la description du module, les dépendances de ce dernier (ici aucune) et les paramètres que l'on pourrait associer au module ( ici TjMax ). Reprenons le résultat de la commande **lspci**. Nous avions vu que cette commande indiquait les modules que les cartes utilisent. Nous allons prendre un module et afficher l'information sur ce dernier :

```

affiche les modules pci et l'information sur le modules
1 # j'extraie l'information de lspci avec le grep faut que la ligne commence par 0 (^0) ou que la ligne contient Kernel modules (/Kernel modules)
2 $ lspci -v | grep '^0|Kernel modules'
3 00:00.0 Host bridge: Intel Corporation Mobile PM965/GM965/GL960 Memory Controller Hub (rev 0c)
4   Kernel modules: intel_agp
5 00:02.0 VGA compatible controller: Intel Corporation Mobile GM965/GL960 Integrated Graphics Controller (primary) (rev 0c) (prog-if 00 [VGA control]
6   Kernel modules: i915
7 00:02.1 Display controller: Intel Corporation Mobile GM965/GL960 Integrated Graphics Controller (secondary) (rev 0c)
8 00:1a.0 USB controller: Intel Corporation 82801H (ICH8 Family) USB UHCI Controller #4 (rev 02) (prog-if 00 [UHCI])
9   Kernel modules: uhci_hcd
10 00:1a.1 USB controller: Intel Corporation 82801H (ICH8 Family) USB UHCI Controller #5 (rev 02) (prog-if 00 [UHCI])
11   Kernel modules: uhci_hcd
12 00:1a.7 USB controller: Intel Corporation 82801H (ICH8 Family) USB2 EHCI Controller #2 (rev 02) (prog-if 20 [EHCI])
13   Kernel modules: ehci_pci
14 00:1b.0 Audio device: Intel Corporation 82801H (ICH8 Family) HD Audio Controller (rev 02)
15   Kernel modules: snd_hda_intel
16 00:1c.0 PCI bridge: Intel Corporation 82801H (ICH8 Family) PCI Express Port 1 (rev 02) (prog-if 00 [Normal decode])
17   Kernel modules: shpchp
18 00:1c.1 PCI bridge: Intel Corporation 82801H (ICH8 Family) PCI Express Port 2 (rev 02) (prog-if 00 [Normal decode])
19   Kernel modules: shpchp
20 00:1c.5 PCI bridge: Intel Corporation 82801H (ICH8 Family) PCI Express Port 6 (rev 02) (prog-if 00 [Normal decode])
21   Kernel modules: shpchp
22 00:1d.0 USB controller: Intel Corporation 82801H (ICH8 Family) USB UHCI Controller #1 (rev 02) (prog-if 00 [UHCI])
23   Kernel modules: uhci_hcd
24 00:1d.1 USB controller: Intel Corporation 82801H (ICH8 Family) USB UHCI Controller #2 (rev 02) (prog-if 00 [UHCI])
25   Kernel modules: uhci_hcd
26 00:1d.2 USB controller: Intel Corporation 82801H (ICH8 Family) USB UHCI Controller #3 (rev 02) (prog-if 00 [UHCI])
27   Kernel modules: uhci_hcd
28 00:1d.7 USB controller: Intel Corporation 82801H (ICH8 Family) USB2 EHCI Controller #1 (rev 02) (prog-if 20 [EHCI])
29   Kernel modules: ehci_pci
30 00:1e.0 PCI bridge: Intel Corporation 82801 Mobile PCI Bridge (rev f2) (prog-if 01 [Subtractive decode])
31 00:1f.0 ISA bridge: Intel Corporation 82801HM (ICH8M) LPC Interface Controller (rev 02)
32   Kernel modules: lpc_ich
33 00:1f.1 IDE interface: Intel Corporation 82801HM/HEM (ICH8M/ICH8M-E) IDE Controller (rev 02) (prog-if 8a [Master SecP PriP])
34   Kernel modules: ata_piix, pata_acpi, ata_generic
35 00:1f.2 IDE interface: Intel Corporation 82801HM/HEM (ICH8M/ICH8M-E) SATA Controller [IDE mode] (rev 02) (prog-if 8f [Master SecP SecO PriP PriO])
36   Kernel modules: ata_piix, pata_acpi, ata_generic
37 00:1f.3 SMBus: Intel Corporation 82801H (ICH8 Family) SMBus Controller (rev 02)
38   Kernel modules: i2c_i801
39 03:01.0 CardBus bridge: O2 Micro, Inc. Cardbus bridge (rev 21)
40   Kernel modules: yenta_socket
41 03:01.4 FireWire (IEEE 1394): O2 Micro, Inc. Firewire (IEEE 1394) (rev 02) (prog-if 10 [OHCI])
42   Kernel modules: firewire_ohci
43 09:00.0 Ethernet controller: Broadcom Corporation NetXtreme BCM5755M Gigabit Ethernet PCI Express (rev 02)
44   Kernel modules: tg3
45 0c:00.0 Network controller: Intel Corporation PRO/Wireless 3945ABG [Golan] Network Connection (rev 02)
46   Kernel modules: iwl3945

```

```

47 # Affiche l'information sur la carte réseau filaire
48 $ modinfo tg3
49 filename: /lib/modules/3.12.1-1-ARCH/kernel/drivers/net/ethernet/broadcom/tg3.ko.gz
50 firmware: tigon/tg3_tso5.bin
51 firmware: tigon/tg3_tso.bin
52 firmware: tigon/tg3.bin
53 version: 3.133
54 license: GPL
55 description: Broadcom Tigon3 ethernet driver
56 author: David S. Miller (davem@redhat.com) and Jeff Garzik (jgarzik@pobox.com)
57 srcversion: 5F6693B7BE37032F61B0883
58 [[ OUTPUT COUPÉ ]]
59 depends: libphy,ptp
60 intree: Y
61 vermagic: 3.12.1-1-ARCH SMP preempt mod_unload modversions
62 parm: tg3_debug:Tigon3 bitmapped debugging message enable value (int)
63

64

65 # Affiche l'information sur la carte réseaux sans file
66 $ modinfo iwl3945
67 filename: /lib/modules/3.12.1-1-ARCH/kernel/drivers/net/wireless/iwlegacy/iwl3945.ko.gz
68 firmware: iwlwifi-3945-2.ucode
69 license: GPL
70 author: Copyright(c) 2003-2011 Intel Corporation <iwl@linux.intel.com>
71 version: in-tree:s
72 description: Intel(R) PRO/Wireless 3945ABG/BG Network Connection driver for Linux
73 srcversion: 102CBF0C44B023F828879B7
74 alias: pci:v00000086d00004227sv*sd*bc*sc*i*
75 alias: pci:v00000086d00004222sv*sd*bc*sc*i*
76 alias: pci:v00000086d00004227sv*sd00001014bc*sc*i*
77 alias: pci:v00000086d00004222sv*sd00001044bc*sc*i*
78 alias: pci:v00000086d00004222sv*sd00001034bc*sc*i*
79 alias: pci:v00000086d00004222sv*sd00001005bc*sc*i*
80 depends: iwlegacy,cfg80211,mac80211
81 intree: Y
82 vermagic: 3.12.1-1-ARCH SMP preempt mod_unload modversions
83 parm: antenna:select antenna (1=Main, 2=Aux, default 0 [both]) (int)
84 parm: swcrypto:using software crypto (default 1 [software]) (int)
85 parm: disable_hw_scan:disable hardware scanning (default 1) (int)
86 parm: fw_restart:restart firmware in case of error (int)
87

88 # affiche l'information sur le module de carte de son
89 $ modinfo snd_hda_intel
90 filename: /lib/modules/3.12.1-1-ARCH/kernel/sound/pci/hda/snd-hda-intel.ko.gz
91 description: Intel HDA driver
92 license: GPL
93 alias: pci:v00001022d*sv*sd*bc04sc03i00*
94 alias: pci:v00001002d*sv*sd*bc04sc03i00*
95 [[ OUTPUT COUPÉ ]]
96

97 depends: snd-hda-codec,snd-pcm,snd,snd-page-alloc
98 intree: Y
99 vermagic: 3.12.1-1-ARCH SMP preempt mod_unload modversions
100 parm: index:Index value for Intel HD audio interface. (array of int)
101 parm: id:ID string for Intel HD audio interface. (array of charp)
102 parm: enable:Enable Intel HD audio interface. (array of bool)
103 parm: model:Use the given board model. (array of charp)
104 parm: position_fix:DMA pointer read method. (-1 = system default, 0 = auto, 1 = LPIB, 2 = POSBUF, 3 = VIACOMBO, 4 = COMBO). (array of int)
105 parm: bdl_pos_adj:BDL position adjustment offset. (array of int)
106 parm: probe_mask:Bitmask to probe codecs (default = -1). (array of int)
107 parm: probe_only:Only probing and no codec initialization. (array of int)
108 parm: jackpoll_ms:Ms between polling for jack events (default = 0, using unsol events only) (array of int)
109 parm: single_cmd:Use single command to communicate with codecs (for debugging only). (bool)
110 parm: enable_msi:Enable Message Signaled Interrupt (MSI) (bint)
111 parm: patch:Patch file for Intel HD audio interface. (array of charp)
112 parm: beep_mode:Select HDA Beep registration mode (0=off, 1=on) (default=1). (array of bool)
113 parm: power_save:Automatic power-saving timeout (in second, 0 = disable). (xint)
114 parm: power_save_controller:Reset controller in power save mode. (bool)
115 parm: align_buffer_size:Force buffer and period sizes to be multiple of 128 bytes. (bint)
116 parm: snoop:Enable/disable snooping (bool)

```

Comme nous pouvons le constater la carte de son a beaucoup de paramètres disponibles !

## CHARGER ET DÉCHARGER UN MODULE (MODPROBE ET RMMOD)

Maintenant que nous avons vu comment lister les modules, nous allons voir comment les charger (**modprobe**) et décharger (**rmmod**), soit dit en passant il est très rare que nous soyons obligé de faire cela ! Nous verrons un peu plus tard le système **udev** qui réalise dynamiquement cette opération quand nous ajoutons un périphérique ou le supprimons. Je présente ici cette fonctionnalité pour information plus que par nécessité, cependant que nous sommes en train de couvrir le sujet pourquoi ne pas en profiter. Pour charger et décharger un module dans le kernel il faut avoir les permissions d'administrateur, nous serons donc obligés de réaliser cette opération avec **sudo** ou en étant **root**.

Nous avions vu dans la liste des modules celui de la carte réseau **tg3**, dans l'exemple suivant je vais décharger et recharger le module :

```

1 # liste le module contenant le pattern tg3
2 $ lsmod | grep tg3
3 tg3          155658  0
4 ptp          8276   1 tg3
5 libphy       20342   1 tg3
6
7 # décharge le module
8 $ sudo rmmod tg3
9 # reliste les modules et constate qu'il n'y a plus de module tg3
10
11 # recharge le module
12 $ sudo modprobe tg3
13
14 $ lsmod | grep tg3
15 tg3          155658  0
16 ptp          8276   1 tg3
17 libphy       20342   1 tg3

```

Tel que mentionné plus tôt, je ne préconise pas cette approche autrement que pour faire un test, il n'est pas viable pour l'usage en production ou de tous les jours pour un desktop de devoir entrer des lignes de commandes à chaque reboot du système. De plus ces commandes doivent être réalisées comme l'utilisateur **root**. Cependant c'est intéressant de pouvoir faire des tests. Nous avons vu plus tôt avec l'utilisation de la commande **modinfo** qu'il est possible de passer des paramètres au module, ceci se fait simplement en ajoutant le paramètre après l'argument du module à **modprobe**.

```

1 # liste les paramètre disponible à tg3
2 $ modinfo tg3 | grep parm
3 parm:           tg3_debug:Tigon3 bitmapped debugging message enable value (int)
4
5
6 # ajout d'une option
7 $ sudo modprobe tg3 patate=10
8 # le module ne renvoie pas d'erreur et se charge.
9
10 # ajout d'une option
11 $ sudo modprobe tg3 tg3_debug=10
12

```

Ces options peuvent être aussi définies dans le fichier de configuration

**/etc/modprobe.d/Nom\_fichier.conf**, le nom du fichier est arbitraire le système charge l'ensemble des fichiers avec l'extension .conf voici un exemple du fichier **/etc/modprobe.d/alsa-base.conf**

```
1 # Load saa7134-alsa instead of saa7134 (which gets dragged in by it anyway)
2 install saa7134 /sbin/modprobe --ignore-install saa7134 $CMDLINE_OPTS && { /sbin/modprobe --quiet --use-blacklist saa7134-alsa ; : ; }
3 # Prevent abnormal drivers from grabbing index 0
4 options bt87x index=-2
5 options cx88_alsa index=-2
6 options saa7134-alsa index=-2
7 options snd-atiixp-modem index=-2
8 options snd-intel8x0m index=-2
9 options snd-via82xx-modem index=-2
10 options snd-usb-audio index=-2
11 options snd-usb-caiaq index=-2
```

## 24.5 MESSAGE DU KERNEL

Quand le système se charge (load), le kernel va réaliser l'ensemble des détections du matériel. Quand un nouveau périphérique est branché, le système va charger le module en conséquence ou le module va générer un événement au niveau du kernel, ce dernier écrira un fichier « log ». Il est possible en utilisant la commande **dmesg** d'obtenir ces informations . Voici un exemple de **dmesg** , bien entendu il est pas toujours simple de le lire car nous allons voir l'ensemble des cartes PCI passer, etc ...

```

2 $ dmesg
3 [ 0.00000] Initializing cgroup subsys cpuset
4 [ 0.00000] Initializing cgroup subsys cpu
5 [ 0.00000] Linux version 3.2.0-57-generic-pae (builddd@lamiak) (gcc version 4.6.3 (Ubuntu/Linaro 4.6.3-1ubuntu5) ) #87-Ubuntu SMP Tue Nov 12 21
6 [ 0.00000] KERNEL supported cpus:
7 [ 0.00000]   Intel GenuineIntel
8 [ 0.00000]   AMD AuthenticAMD
9 [ 0.00000]   NSC Geode by NSC
10 [ 0.00000]   Cyrix CyrixInstead
11 [ 0.00000]   Centaur CentaurHauls
12 [ 0.00000]   Transmeta GenuineTMx86
13 [ 0.00000]   Transmeta TransmetaCPU
14 [ 0.00000]   UMC UMC UMC UMC
15 [[ OUTPUT COUPÉ ]]

16 [ 22.334400] parport_pc 00:05: reported by Plug and Play ACPI
17 [ 22.334453] parport0: PC-style at 0x378 (0x778), irq 7, dma 3 [PCSPP,TRISTATE,COMPAT,ECP,DMA]
18 [ 22.371152] device-mapper: multipath: version 1.3.1 loaded
19 [ 22.431573] ACPI: PCI Interrupt Link [LAZA] enabled at IRQ 22
20 [ 22.431582] snd_hda_intel 0000:00:05.0: PCI INT B -> Link[LAZA] -> GSI 22 (level, low) -> IRQ 22
21 [ 22.431586] hda_intel: Disabling MSI
22 [ 22.431622] snd_hda_intel 0000:00:05.0: setting latency timer to 64
23 [ 22.432242] lp0: using parport0 (interrupt-driven).
24 [ 22.444540] nvidia: module license 'NVIDIA' taints kernel.
25 [ 22.444545] Disabling lock debugging due to kernel taint
26 [ 22.781575] ACPI: PCI Interrupt Link [LNED] enabled at IRQ 19
27 [ 22.781600] nvidia 0000:02:00.0: PCI INT A -> Link[LNED] -> GSI 19 (level, low) -> IRQ 19
28 [ 22.781610] nvidia 0000:02:00.0: setting latency timer to 64
29 [[ OUTPUT COUPÉ ]]

30 [ 1.492540] scsi 2:0:0:0: Direct-Access      ATA      ST31000524AS    JC4B PQ: 0 ANSI: 5
31 [ 1.492727] sd 2:0:0:0: [sda] 1953525168 512-byte logical blocks: (1.00 TB/931 GiB)
32 [ 1.492734] sd 2:0:0:0: Attached scsi generic sg0 type 0
33 [ 1.492776] sd 2:0:0:0: [sda] Write Protect is off
34 [ 1.492780] sd 2:0:0:0: [sda] Mode Sense: 00 3a 00 00
35 [ 1.492834] sd 2:0:0:0: [sda] Write cache: enabled, read cache: enabled, doesn't support DPO or FUA
36 [ 1.500133] ata5.00: configured for UDMA/100
37 [ 1.514237] sda: sdal sda2 sda3
38 [ 1.514635] sd 2:0:0:0: [sda] Attached SCSI disk
39 [ 1.660024] usb 2-1: new low-speed USB device number 2 using ohci_hcd
40 [ 1.960054] ata4: SATA link up 1.5 Gbps (SStatus 113 SControl 300)
41 [ 1.968505] ata4.00: ATA-8: ST3500320AS, SD15, max UDMA/133
42 [ 1.968510] ata4.00: 976773168 sectors, multi 16: LBA48 NCQ (depth 0/32)
43 [ 1.984493] ata4.00: configured for UDMA/133
44 [ 1.984671] scsi 3:0:0:0: Direct-Access      ATA      ST3500320AS    SD15 PQ: 0 ANSI: 5
45 [ 1.984820] sd 3:0:0:0: [sdb] 976773168 512-byte logical blocks: (500 GB/465 GiB)
46 [ 1.984853] sd 3:0:0:0: Attached scsi generic sg1 type 0
47 [ 1.984868] sd 3:0:0:0: [sdb] Write Protect is off
48 [ 1.984871] sd 3:0:0:0: [sdb] Mode Sense: 00 3a 00 00
49 [[ OUTPUT COUPE ]]

50 [ 1.984891] sd 3:0:0:0: [sdb] Write cache: enabled, read cache: enabled, doesn't support DPO or FUA
51 [ 1.988251] scsi 4:0:0:0: CD-ROM           HL-DT-ST DVDRAM GH20NS10  EL00 PQ: 0 ANSI: 5
52 [ 1.990491] sdb: sd1 sdb2
53 [ 1.990884] sd 3:0:0:0: [sdb] Attached SCSI disk
54 [ 1.993339] sr0: scsi3-mmc drive: 48x/48x writer dvd-ram cd/rw xa/form2 cdda tray
55 [ 1.993342] cdrom: Uniform CD-ROM driver Revision: 3.20
56 [ 1.993466] sr 4:0:0:0: Attached scsi CD-ROM sr0
57 [ 1.993593] sr 4:0:0:0: Attached scsi generic sg2 type 5
58 [[ OUTPUT COUPE ]]

59 [26467.825422] tg3.c:v3.133 (Jul 29, 2013)
60 [26467.861300] tg3 0000:09:00.0: irq 46 for MSI/MSI-X
61 [26467.868040] tg3 0000:09:00.0 eth0: Tigon3 [partno(BCM95755m)] rev a002] (PCI Express) MAC address 00:1c:23:41:6d:eb
62 [26467.868046] tg3 0000:09:00.0 eth0: attached PHY is 5755 (10/100/1000Base-T Ethernet) (WireSpeed[1], EEE[0])
63 [26467.868050] tg3 0000:09:00.0 eth0: RXChecksums[1] LinkChgREG[0] Miirq[0] ASF[0] TS0cap[1]
64 [26467.868054] tg3 0000:09:00.0 eth0: dma_rwctrl[76180000] dma_mask[64-bit]
65 [26467.896710] IPv6: ADDRCONF(NETDEV_UP): eth0: link is not ready
66 [26895.116779] tg3: unknown parameter 'patate' ignored
67 [26895.116982] tg3.c:v3.133 (Jul 29, 2013)

```

```

71 [26895.161137] tg3 0000:09:00.0 eth0: Tigon3 [partno(BCM95755m) rev a002] (PCI Express) MAC address 00:1c:23:41:6d:eb
72 [26895.161142] tg3 0000:09:00.0 eth0: attached PHY is 5755 (10/100/1000Base-T Ethernet) (WireSpeed[1], EEE[0])
73 [26895.161146] tg3 0000:09:00.0 eth0: RXcsums[1] LinkChgREG[0] MIirq[0] ASF[0] TS0cap[1]
74 [26895.161148] tg3 0000:09:00.0 eth0: dma_rwctrl[76180000] dma_mask[64-bit]
75 [26895.161656] tg3 0000:09:00.0: irq 46 for MSI/MSI-X
76 [26895.197172] IPv6: ADDRCONF(NETDEV_UP): eth0: link is not ready
77 [26914.509842] tg3:c:v3.133 (Jul 29, 2013)
78 [26914.544289] tg3 0000:09:00.0: irq 46 for MSI/MSI-X
79 [26914.547560] tg3 0000:09:00.0 eth0: Tigon3 [partno(BCM95755m) rev a002] (PCI Express) MAC address 00:1c:23:41:6d:eb
80 [26914.547565] tg3 0000:09:00.0 eth0: attached PHY is 5755 (10/100/1000Base-T Ethernet) (WireSpeed[1], EEE[0])
81 [26914.547568] tg3 0000:09:00.0 eth0: RXcsums[1] LinkChgREG[0] MIirq[0] ASF[0] TS0cap[1]
82 [26914.547571] tg3 0000:09:00.0 eth0: dma_rwctrl[76180000] dma_mask[64-bit]
83 [26914.579784] IPv6: ADDRCONF(NETDEV_UP): eth0: link is not ready

```

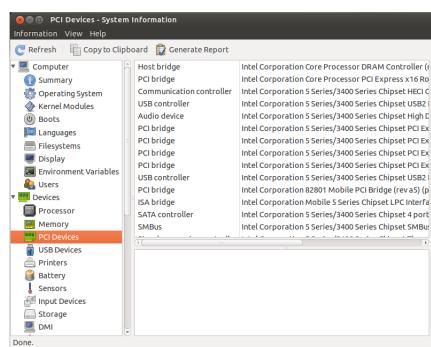
Nous voyons au début le chargement de l'ensemble du système, la détection du disque dur, du cdrom, la carte de son, la carte **nvidia**, l'ensemble des informations de la détection et des chargements de module noyaux sont présents. Nous voyons d'ailleurs à la toute fin que le module **tg3** fut chargé plusieurs fois au niveau du système. Ceci représente mes démonstrations de chargement et déchargeant réalisés plus tôt. J'aimerai porter votre attention à la **ligne 69** nous retrouvons l'erreur du paramètre **patate** que j'avais transmis au module, nous constatons que réellement le paramètre fut ignoré. Lors de l'ajout d'une clef USB de données, vous constaterez aussi que le kernel renvoie de l'information avec les partitions présentes et le nom de fichier dans **/dev** qui fut créé ainsi que les partitions qui lui sont associées.

## APPLICATION GRAPHIQUE DISPONIBLE

**Hardinfo** n'est pas vraiment un programme spécifique pour les modules, mais il propose un grand nombre d'informations sur votre ordinateur, dont les modules utilisés par les périphériques.

Pour vous le procurer, il vous suffit d'installer le paquet **hardinfo**

Vous le trouverez ensuite dans **Applications → Outils système → System Profiler and Benchmark**



## 24.6 VIDÉOS

Télécharger devices.ogv et devices-2.ogv et lire avec VLC.



# Chapitre 25

## PRATIQUE SUR PÉRIPHÉRIQUES N°1

### 25.1 LES PÉRIPHÉRIQUES D'ENTRÉES

#### VISUALISATION DES PÉRIPHÉRIQUES DÉTECTÉS DE TYPE INPUT

```
$ ls -l /dev/input/by-path/
total 0
lrwxrwxrwx 1 root root 9 20 Feb 14 :11 platform-i8042-serio-0-event-kbd -> ../event0
lrwxrwxrwx 1 root root 10 20 Feb 14 :11 platform-i8042-serio-1-event-mouse -> ../event10
lrwxrwxrwx 1 root root 9 20 Feb 14 :11 platform-i8042-serio-1-mouse -> ../mouse1
lrwxrwxrwx 1 root root 9 20 Feb 14 :11 platform-pcspkr-event-spkr -> ../event4
```

#### VISUALISATION DE L'UTILISATION DE LA SOURIS ET DU CLAVIER :

Grâce à la commande **cat** qui affiche le contenu du fichier `/dev/input/mouse0`. Pour réaliser cette commande, bien entendu il faut être **root** (`/dev` est root) pour capter l'information.

```
$ sudo cat /dev/input/mouse0
(((((((((((((^C
```

Bouger la souris tranquillement avec le touchpad et le bouton du milieu afin de voir le résultat (des caractères incompréhensibles), pour terminer taper **CTRL+C** pour arrêter la commande.

Réalisez la même commande mais avec le fichier `/dev/input/event0` qui est le clavier, comme nous l'avons vu grâce au listing de `/dev/input/by-path`.

```
$ sudo cat /dev/input/event0
DgS}DgS}DgS}EgSnK*EgSnK*EgSnKEgSV4
```

\*EgSV4

Touchez la touche majuscule, et d'autres touches et visualisez le résultat, pour terminer tapez **CTRL+C** pour arrêter la commande.

```
$ sudo cat /dev/input/event0 | strings
```

hello tout le monde

Le fait de rajouter le pipe | et la commande strings, on peut visualiser les caractères imprimables, pour terminer taper **CTRL+C** pour arrêter la commande.

## LE TERMINAL ET L'ENVOI DE MESSAGE

Pour connaître le terminal virtuel que vous utilisez pour l'échange d'information vous pouvez utiliser la commande **tty**.

```
$ tty
```

```
/dev/pts/1
```

Ouvrons un autre terminal virtuel, et avec la commande **tty** nous obtenons le device associé ; soit le nouvel terminal : `/dev/pts/5`

## TRANSMETTRE DU TEXTE VERS CE TERMINAL

```
$ echo "wow super" > /dev/pts/5
```

envoie le texte « wow super » du terminal `/dev/pts/1` vers le terminal `/dev/pts/5`.

## 25.2 UTILISATION DES PSEUDOS PÉRIPHÉRIQUES

### GÉNÉRER D'UN MOT DE PASSE ALÉATOIRE

générer un password de 10 caractères :

```
$ strings /dev/urandom | grep -o '[[ :alnum :]]' | head -n 10 | tr -d '\n'; echo
$ u7wixjD38
```

Utilisation du device **/dev/urandom** avec la commande **strings** pour ne conserver que les caractères imprimables, utilisation de **grep** pour extraire que les caractères et chiffres, **head -n** limite aux 10 premières lignes, utilisation de **tr** pour supprimer les retours de chariots, **echo** pour passer à la ligne. On aime!!!

## SIMULER QUE LE DISQUE EST PLEIN

Utilisation de /dev/full

```
$ echo "Hello world" > /dev/full
```

bash : echo : erreur d'écriture : Aucun espace disponible sur le périphérique.

je ne vois pas l'utilité sinon d'obtenir une erreur d'écriture sur un serveur en upload.

## 25.3 MANIPULATION DE PARTITION (CLEF USB)

### DÉTECTION DE LA CLEF ET VISUALISATION

Branchement de la clef USB dans le système et utilisation de la commande **dmesg** pour visualiser ce que le kernel a détecté. Nous voyons donc que le système a ajouté **sdb**, **sda** étant le disque dur interne du système.

```
branchement d'une clef usb et visualisation de la détection par le kernel
1 # Utilisation de dmesg pour voir les messages du kernel , exemple de output
2 $ dmesg
3 [[ ...OUTPUT COUPÉ... ]]
4 [446674.336029] usb 2-2: new high-speed USB device number 4 using ehci_hcd
5 [446674.474417] scsi10 : usb-storage 2-2:1.0
6 [446675.518575] scsi 10:0:0:0: Direct-Access      Kingston DataTraveler G3  PMAP PQ: 0 ANSI: 0 CCS
7 [446675.519335] sd 10:0:0:0: Attached scsi generic sg2 type 0
8 [446676.625162] sd 10:0:0:0: [sdb] 15638528 512-byte logical blocks: (8.00 GB/7.45 GiB)
9 [446676.627279] sd 10:0:0:0: [sdb] Write Protect is off
10 [446676.627282] sd 10:0:0:0: [sdb] Mode Sense: 03 41 00 00
11 [446676.629409] sd 10:0:0:0: [sdb] No Caching mode page found
12 [446676.629414] sd 10:0:0:0: [sdb] Assuming drive cache: write through
13 [446676.637155] sd 10:0:0:0: [sdb] No Caching mode page found
14 [446676.637158] sd 10:0:0:0: [sdb] Assuming drive cache: write through
15 [446676.659610] sdb: sdb1
16 [446676.666655] sd 10:0:0:0: [sdb] No Caching mode page found
17 [446676.666660] sd 10:0:0:0: [sdb] Assuming drive cache: write through
18 [446676.666663] sd 10:0:0:0: [sdb] Attached SCSI removable disk
```

Avec les versions modernes de Linux, le système monte (**mount**) automatiquement la clef USB, il est possible d'utiliser la commande **df** pour visualiser la partition montée.

```
df pour voir la partition mounté
1 # utilisation de df pour voir la partition
2 $ df -h
3 Sys. de fichiers Taille Utilisé Disp. Utile Monté sur
4 /dev/sdal    143G   118G   19G  87% /
5 udev        3,9G   4,0K  3,9G  1% /dev
6 tmpfs       1,6G   964K  1,6G  1% /run
7 none        5,0M     0  5,0M  0% /run/lock
8 none        3,9G   12M  3,9G  1% /run/shm
9 /dev/sdb1    7,5G   2,7G  4,8G  37% /media/KINGSTON
```

## MANIPULATION DE LA CLEF (PARTITION)

— Nous allons manipuler les tables de partition sur la clef USB, nous allons créer une partition !

- Éjection (démontage) de la clef USB car elle fut « mountée » automatiquement lors du branchement. Comme nous allons manipuler les partitions et les supprimer, mieux vaut pour l'intégrité des données, ne pas réaliser l'opération alors que le système accède à la clef.  
**`$ sudo umount /dev/sdb1`**

- Utilisation de la commande **fdisk** pour faire la manipulation des partitions sur la clef USB

```

Utilisation de fdisk pour les partitions
1 # fdisk pour manipuler les partition , attention bien mettre le bon disque device (/dev/sdb)
2 $ sudo fdisk /dev/sdb
```

- Visualisation des partitions avec l'espace utilisé : **`$ df -h`**
- Démonter / éjection de la clef USB :**`$ umount /dev/sdb1`**
- Création de partition : **`$ sudo fdisk /dev/sdb`**
  - d** (*pour détruire la partition*)
  - p** (*pour visualiser le table de partition - voir la destruction*)
  - n** (*pour créer une nouvelle partition*)
  - p** (*pour choisir une partition primaire*)
  - 1** (*pour choisir le numéro de la partition*)
  - [enter] (*pour accepter par défaut le premier secteur*)
  - +50M** (*pour créer une partition d'une taille de 50Meg*)
  - p** (*pour visualiser le type d'identification de la partition ici 83 linux*)
- *A ce stade de la création de partition il est intéressant d'ouvrir un autre terminal et de lancer la commande sudo fdisk -l*  
*On peut s'apercevoir que les manipulations faites auparavant n'ont pas été encore prises en compte. En fait la commande fdisk travaille dans un fichier mémoire (buffer), aucune altération n'a eu lieu sur la clé usb .*  
*Les changements interviennent avec la commande « w » de fdisk qui écrit le « buffer » sur le disque. Continuons donc notre création en écrivant sur le disque ces transformations.*  
*On pourrait à ce moment quitter par la commande « q » de fdisk, aucune modification ne serait effectuée.*
- **w** (*écriture des modifications et sortie de fdisk*)
  - \$ sudo partprobe** (*pour s'assurer de la relecture des partitions par le système*)
  - \$ sudo mkfs.ext4 /dev/sdb1** (*création du système de fichiers sur sdb1*)
- Copie de partition dans un fichier : **`$ sudo dd if=/dev/sdb1 of=~/copie-partition`**
- Monter en périphérique de masse virtuel : **`sudo mount -o loop ~/copie-partition /mnt`**

Avant de monter une partition, il faut s'assurer que la partition ait été formatée avec **mkfs** c'est à dire qu'elle supporte un système de fichiers.

**NB :** Pour lire tous les périphériques de type **block**, on peut aussi utiliser la commande **/bin/lsblk** du paquet **util-linux** que j'installe sous ubuntu avec la commande **sudo apt-get install util-linux**. Cette commande permet de voir en même temps les points de montage et le type des périphériques.

```
marc@marc-MS-7721:~$ lsblk
NAME  MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda    8:0    0 931,5G  0 disk
└─sda1  8:1    0 234,8G  0 part /boot
└─sda2  8:2    0 499,7M  0 part /boot
└─sda3  8:3    0 638,3G  0 part /
└─sda4  8:4    0     1K  0 part
└─sda5  8:5    0    50G  0 part /home
└─sda6  8:6    0   7,9G  0 part [SWAP]
sdb    8:16   0 931,5G  0 disk
└─sdb1  8:17   0     1K  0 part
└─sdb2  8:18   0 473,2G  0 part /media/marc/data
└─sdb5  8:21   0 450,4G  0 part
└─sdb6  8:22   0     8G  0 part [SWAP]
sdc    8:32   1 245,5M  0 disk
sr0    11:0   1 1024M  0 rom
```

## 25.4 EXTRA

### RESTAURATION D'UN FICHIER

```
$ grep -a -B 25 -A 100 'some string in the file' /dev/sda1 > results.txt
```

grep searches through a file and prints out all the lines that match some pattern. Here, the pattern is some string that is known to be in the deleted file. The more specific this string can be, the better. The file being searched by grep (/dev/sda1) is the partition of the hard drive the deleted file used to reside in. The « a » flag tells grep to treat the hard drive partition, which is actually a binary file, as text.

Since recovering the entire file would be nice instead of just the lines that are already known, context control is used. The flags « -B 25 -A 100 » tell grep to print out 25 lines before a match and 100 lines after a match. Be conservative with estimates on these numbers to ensure the entire file is included (when in doubt, guess bigger numbers). Excess data is easy to trim out of results, but if you find yourself with a truncated or incomplete file, you need to do this all over again. Finally, the « > results.txt » instructs the computer to store the output of grep in a file called results.txt.

Source : [http://spin.atomicobject.com/2010/08/18/undelete?utm\\_source=y-combinator&utm\\_medium=social-media&utm\\_campaign=technicalVIDÉOS](http://spin.atomicobject.com/2010/08/18/undelete?utm_source=y-combinator&utm_medium=social-media&utm_campaign=technicalVIDÉOS)

## 25.5 VIDÉOS

Télécharger exo-device.ogv et lire avec VLC.

## 25.6 COMPLÉMENTS

### MANIPULATIONS DE LA CLEF (PARTITION)

- Nous allons manipuler les tables de partitions sur la clef USB (`/dev/sdc1`), nous allons créer 3 partitions primaires :
  - la première primaire de type linux ext4 de 50 Mega-octets,
  - la deuxième primaire de type windows 95 Fat 32 de 100 Mega-octets,
  - la troisième primaire de type linux ext4 de 50 Mega-octets.
- Utilisation de la commande **fdisk** pour faire la manipulation des partitions sur la clef USB  
**sudo fdisk -l**  
 — Visualisation des partitions avec l'espace utilisé : **\$ df -h**  
 — Démonter / éjection de la clef USB :**\$ umount /dev/sdc1**  
 — Création des partitions : **\$ sudo fdisk /dev/sdc**
  - d** (pour détruire la partition)
  - p** (pour visualiser le table de partition - voir la destruction)

*Création de la 1ère partition*

  - n** (pour créer une nouvelle partition)
  - p** (pour choisir une partition primaire)
  - 1** (pour choisir le numéro de la partition)
  - [enter] (pour accepter par défaut le premier secteur)
  - +50M** (pour créer une partition d'une taille de 50Meg)
  - p** (pour visualiser le le type d'identification de la partition ici 83 linux)

*Création de la 2ème partition*

  - n** (pour créer une nouvelle partition)
  - p** (pour choisir une partition primaire)
  - 2** (pour choisir le numéro de la partition)
  - [enter] (pour accepter par défaut le premier secteur)
  - +100M** (pour créer une partition d'une taille de 100Meg)
  - p** (pour visualiser le le type d'identification de la partition ici 83 linux non conforme à notre souhait donc nous allons changer son type)
  - l** (lister les types de partitions connues : Win 95 Fat32 = type b)
  - t** (modifier le type)
  - 2** (la 2ème partition)
  - b**
  - p** (pour visualiser le le type d'identification de la partition ici Win 95 Fat32)

*Création de la 3ème partition*

  - n** (pour créer une nouvelle partition)
  - p** (pour choisir une partition primaire)
  - 3** (pour choisir le numéro de la partition)
  - [enter] (pour accepter par défaut le premier secteur)
  - +50M** (pour créer une partition d'une taille de 50Meg)

- **p** (pour visualiser le type d'identification de la partition ici 83 linux )
- **w** (écriture des modifications et sortie de fdisk)
- **\$ sudo partprobe** (pour s'assurer de la relecture des partitions par le système)
- **\$lsblk** (pour lister tous les devices de type blocks)

*Lors de la création des partitions nous leurs avons donné un type mais en fait cela n'est qu'à titre « informatif », un peu comme si nous avions collé une étiquette sur chaque partition. Maintenant nous allons vraiment leur donner un type en les formatant (création d'un système de fichiers).*

- **\$ sudo mkfs.ext4 /dev/sdc1** (création du système de fichiers ext4 sur sdc1)
- **\$ sudo mkfs.vfat /dev/sdc2** (création du système de fichiers fat32 sur sdc2)
- **\$ sudo mkfs.ext4 /dev/sdc3** (création du système de fichiers ext4 sur sdc3)

Afin de pouvoir continuer nos travaux pratiques, créons 3 répertoires dans /mnt dans lesquels nous plaçons 3 fichiers vides dans chaque répertoire.

**\$ls -ld /mnt** (pour connaître le user et le group)

**sudo touch /mnt/fic01 /mnt/fic02 /mnt/fic03** (touch crée un fichier vide)

**sudo mkdir -p /mnt/part1** (emploi de sudo car /mnt appartient à root)

**cd /mnt/part1**

**sudo touch fichier11 fichier12 fichier13**

**sudo mkdir -p /mnt/part2** (emploi de sudo car /mnt appartient à root)

**cd /mnt/part2**

**sudo touch fichier20 fichier21 fichier22**

**sudo mkdir -p /mnt/part3**

**cd /mnt/part3**

**sudo touch fichier30 fichier31 fichier32**

**ls -lR /mnt** (pour vérifier la création des répertoires et fichiers)

Montons sdc1 sur /mnt et regardons ce qui se passe :

**sudo mount /dev/sdc1 /mnt**

**ls -lR /mnt**

Les fichiers et répertoires existants de /mnt ont **disparus**, en fait ils **existent toujours mais sont cachés** par la partition **sdc1**, un peu comme si une feuille de papier avait été déposée sur un livre et le cachait. Enlevons la feuille, le livre réapparaît.

Ici de même, démontons la partition **sdc1** et listons à nouveau le répertoire /mnt, mais auparavant copions le répertoire /etc sur la clé usb (sdc1) et visualisons le résultat et l'occupation disque de la clé.

```
$ sudo cp -r /etc /mnt
```

```
$ ls -l /mnt; df -h /mnt (ou du -hs /mnt)
```

```
sudo umount /dev/sdc1
```

**ls -lR /mnt** (tous les répertoires et fichiers d'origine sont présents !)

Faisons à présent un miroir (une image) de la partition1 en utilisant la commande **dd** qui copie (octet par octet sans interpréter) un fichier en le convertissant et le formatant selon les opérandes. Nous allons faire cette image :

- 1) dans un fichier, puis
- 2) dans une autre partition de même taille ou de taille plus grande

1) Copie de partition dans un fichier : **\$ sudo dd if=/dev/sdc1 of=~/img\_part1.dsk**

```
marc@marc-MS-7721:~$ sudo dd if=/dev/sdc1 of=~/img_part1.dsk
[sudo] password for marc:
102400+0 enregistrements lus
102400+0 enregistrements écrits
52428800 octets (52 MB) copiés, 2,61893 s, 20,0 MB/s
marc@marc-MS-7721:~$ ls -l
total 51260
drwxr-xr-x 3 marc marc      4096 déc. 11 12:52 Bureau
-rw-rw-r-- 1 marc marc      6903 déc.   9 16:47 cle_usb
drwxr-xr-x 2 marc marc      4096 déc.   9 17:12 Documents
-rw-rw-r-- 1 marc marc     8222 déc.   9 14:46 entree
drwxrwxr-x 2 marc marc      4096 déc.  10 22:34 exo-device
drwxr-xr-x 2 marc marc      4096 déc.   6 13:09 Images
-rw-r--r-- 1 root root 52428800 déc.  11 14:45 img_part1.dsk
```

- Monter le fichier **img\_part1.dsk** en périphérique de masse virtuel : **sudo mount -o loop ~/img\_part1.dsk /mnt**
- **\$ df -h /mnt ; ls /mnt**

```
marc@marc-MS-7721:~$ df -h /mnt ; ls /mnt
Sys. de fichiers Taille Utilisé Dispo Utile% Monté sur
/dev/loop0       45M    8,7M   33M  22% /mnt
etc  lost+found      -
```

Le fichier **img\_part1.dsk** monté à l'aide de « loop » se manipule comme une partition normale. Si nous avions effectué ce processus à partir d'un cdrom nous aurions obtenu une image iso sans l'aide de logiciel dédié.

2) Copions la partition (effectuons une image) **sdc1** dans **sdc3**, puis montons **sdc1** dans **/mnt/part1** et **sdc2** dans **/mnt/part2**

- **sudo dd if=/dev/sdc1 of=/dev/sdc3**
- **sudo mount /dev/sdc1 /mnt/part1**
- **sudo mount /dev/sdc3 /mnt/part3**
- **ls /mnt/part\***

```
marc@marc-MS-7721:~$ ls -l /mnt/part*
/mnt/part1:
total 20
drwxr-xr-x 138 root root 7168 déc. 11 13:59 etc
drwx----- 2 root root 12288 déc. 11 12:01 lost+found

/mnt/part2:
total 0
-rw-r--r-- 1 root root 0 déc. 11 13:45 fichier20
-rw-r--r-- 1 root root 0 déc. 11 13:45 fichier21
-rw-r--r-- 1 root root 0 déc. 11 13:45 fichier22

/mnt/part3:
total 20
drwxr-xr-x 138 root root 7168 déc. 11 13:59 etc
drwx----- 2 root root 12288 déc. 11 12:01 lost+found
```

**cd /mnt/part3**

Créons un petit fichier texte avec nano : **sudo nano essai.txt**

Taper quelques lignes de texte et enregistrer. Listons le répertoire : **ls -l**

Montons la partition win95 Fat 32 sur /mnt/part2 :

— **sudo mount /dev/sdc2 /mnt/part2**

```
marc@marc-MS-7721:/mnt/part3$ df -h | grep sdc
/dev/sdc1      45M    8,7M   33M  22% /mnt/part1
/dev/sdc3      45M    8,7M   33M  22% /mnt/part3
/dev/sdc2     100M      0  100M   0% /mnt/part2
```

**sudo umount /dev/sdc2**

Montons le fichier img\_part1.dsk dans /mnt pour rajouter dans cette image un fichier vide, et démontons le :

- **sudo mount -o loop ~/img\_part1.dsk /mnt**
- **sudo touch /mnt/UN\_NOUVEAU\_FICHIER**
- **ls -l /mnt**
- **sudo umount /mnt**

Créons une image de img\_part1.dsk dans /dev/sdc2

**sudo dd if=~/img\_part1.dsk of=/dev/sdc2**

```
marc@marc-MS-7721:~$ sudo dd if=~/img_part1.dsk of=/dev/sdc2
102400+0 enregistrements lus
102400+0 enregistrements écrits
52428800 octets (52 MB) copiés, 0,577078 s, 90,9 MB/s
```

Cette copie va écraser la fat32, **sdc2** sera du type du fichier **img\_part1.dsk** soit linux ext4

**sudo mount /dev/sdc2 /mnt/part2;**

```
marc@marc-MS-7721:~$ df -hT | grep sdc2; ls /mnt/part2
/dev/sdc2      ext4      45M    8,7M   33M  22% /mnt/part2
etc  lost+found  UN_NOUVEAU_FICHIER
```

Regardons la table de partition de sdc :

```
marc@marc-MS-7721:~$ sudo fdisk -l /dev/sdc

Disque /dev/sdc : 257 Mo, 257425408 octets
8 têtes, 62 secteurs/piste, 1013 cylindres, total 502784 secteurs
Unités = secteurs de 1 * 512 = 512 octets
Taille de secteur (logique / physique) : 512 octets / 512 octets
taille d'E/S (minimale / optimale) : 512 octets / 512 octets
Identifiant de disque : 0xd244d244

Périphérique Amorçage Début Fin Blocs Id. Système
/dev/sdc1 2048 104447 51200 83 Linux
/dev/sdc2 104448 309247 102400 b W95 FAT32
/dev/sdc3 309248 411647 51200 83 Linux
-
```

Nous avons copié **img\_part1.dsk** sur la partition mais nous n'avons pas modifié la table de partition qui est indépendante du contenu, si bien que **fdisk** la voit encore en **Fat32**! Comme la copie s'est faite octet par octet nous avons recopié le filesystem ext4 de sdc1 sur **sdc2**, c'est pourquoi la commande **df -h** renvoie une taille de **45M** au lieu de **100M** mais avec le bon type de filesystem!!!

En fait il reste sur la partition **sdc2** encore de disponible **50** Mega-octets mais qui sont invisibles (avec sudo gparted, on peut visualiser la taille de la partition et l'espace occupé)! Essayons de récupérer cet espace à l'aide de la commande **resize2fs**.

```
marc@marc-MS-7721:~$ sudo resize2fs /dev/sdc2
resize2fs 1.42.9 (4-Feb-2014)
Le système de fichiers de /dev/sdc2 est monté sur /mnt/part2 ; le changement de
taille doit être effectué en ligne
old_desc_blocks = 1, new_desc_blocks = 1
Le système de fichiers /dev/sdc2 a maintenant une taille de 102400 blocs.
```

A « chaud », alors que la partition **sdc2** toujours montée la commande **resize2fs** a récupéré **50M** ce que nous confirme **sudo df -hT**.

Pour finir nous allons démonter la partition sdc3, avec fdisk la supprimer, redonner le type correct à **sdc2**, la supprimer et la recréer avec une taille supérieure de 150M.

```
sudo umount /mnt/part3
```

```
marc@marc-MS-7721:~$ sudo fdisk /dev/sdc

Commande (m pour l'aide) : d
Numéro de partition (1-4): 3

Commande (m pour l'aide) : t
Numéro de partition (1-4): 2
Code Hexa (taper L pour lister les codes): 83
Type système de partition modifié de 2 à 83 (Linux)

Commande (m pour l'aide) : p

Disque /dev/sdc : 257 Mo, 257425408 octets
8 têtes, 62 secteurs/piste, 1013 cylindres, total 502784 secteurs
Unités = secteurs de 1 * 512 = 512 octets
Taille de secteur (logique / physique) : 512 octets / 512 octets
taille d'E/S (minimale / optimale) : 512 octets / 512 octets
Identifiant de disque : 0xd244d244

Périphérique Amorçage Début Fin Blocs Id. Système
/dev/sdc1 2048 104447 51200 83 Linux
/dev/sdc2 104448 309247 102400 83 Linux

Commande (m pour l'aide) : d
Numéro de partition (1-4): 2

Commande (m pour l'aide) : n
Partition type:
  p  primary (1 primary, 0 extended, 3 free)
  e  extended
Select (default p): p
Numéro de partition (1-4, 2 par défaut): 2
Premier secteur (104448-502783, 104448 par défaut):
Utilisation de la valeur 104448 par défaut
Dernier secteur, +secteurs ou +taille[K,M,G] (104448-502783, 502783 par défaut): +150M

Commande (m pour l'aide) : p

Disque /dev/sdc : 257 Mo, 257425408 octets
8 têtes, 62 secteurs/piste, 1013 cylindres, total 502784 secteurs
Unités = secteurs de 1 * 512 = 512 octets
Taille de secteur (logique / physique) : 512 octets / 512 octets
taille d'E/S (minimale / optimale) : 512 octets / 512 octets
Identifiant de disque : 0xd244d244

Périphérique Amorçage Début Fin Blocs Id. Système
/dev/sdc1 2048 104447 51200 83 Linux
/dev/sdc2 104448 411647 153600 83 Linux

Commande (m pour l'aide) : w
La table de partitions a été altérée.
```

Appel d'ioctl() pour relire la table de partitions.

Attention : la table de partitions n'a pas pu être relue : erreur 16 : Périphérique ou ressource occupé.  
Le noyau continue à utiliser l'ancienne table. La nouvelle sera utilisée  
lors du prochain démarrage ou après avoir exécuté partprobe(8) ou kpartx(8).

Attention : si vous avez créé ou modifié une partition DOS 6.x,  
veuillez consulter les pages du manuel de fdisk pour des informations  
complémentaires.  
Synchronisation des disques.

Pour redimensionner (agrandir) la partition **sdc2**, on la supprime et on la recrée avec une taille plus grande (en fait c'est la table de la partition qui a été supprimée, pas les données).

**sudo partprobe** (*pour forcer la relecture des partitions par le kernel*)

```
marc@marc-MS-7721:~$ df -hT | grep sdc
/dev/sdc2      ext4      94M   9,1M  78M  11% /media/marc/0abdde98-5f29-462b-9a53-1ad5b958a88e
/dev/sdc1      ext4      45M   8,7M  33M  22% /media/marc/0abdde98-5f29-462b-9a53-1ad5b958a88e1
marc@marc-MS-7721:~$ # nous avons récupérer l'ancienne partition il reste plus qu'à la redimensionner
marc@marc-MS-7721:~$ sudo resize2fs /dev/sdc2
[sudo] password for marc:
resize2fs 1.42.9 (4-Feb-2014)
Le système de fichiers de /dev/sdc2 est monté sur /media/marc/0abdde98-5f29-462b-9a53-1ad5b958a88e ; le
changement de taille doit être effectué en ligne
old_desc_blocks = 1, new_desc_blocks = 1
Le système de fichiers /dev/sdc2 a maintenant une taille de 153600 blocs.

marc@marc-MS-7721:~$ df -hT | grep sdc
/dev/sdc2      ext4     142M   9,1M  125M   7% /mnt/part2
/dev/sdc1      ext4      45M   8,7M  33M  22% /media/marc/0abdde98-5f29-462b-9a53-1ad5b958a88e1
marc@marc-MS-7721:~$ ls -l /mnt/part2
total 20
drwxr-xr-x 138 root root 7168 déc. 11 13:59 etc
drwx-----  2 root root 12288 déc. 11 12:01 lost+found
-rw-r--r--  1 root root    0 déc. 11 19:14 UN_NOUVEAU_FICHIER
```

Si nous voulons supprimer réellement la partition, c'est à dire vider son contenu, il faudrait écrire un motif quelconque dans chaque octet :

**sudo dd if=/dev/urandom of=/dev/sdc2** (*pour un caractère quelconque*)

**sudo dd if=/dev/zero of=/dev/sdc2** (*pour un zéro*)

Le mieux serait de créer un fichier (efface\_partition) dans un éditeur de texte comme nano :

```
#!/bin/bash
for n in `seq 7` ; do dd if=/dev/urandom of=/dev/sdc bs=8b conv=noTrunc ; done
```

Maintenant vous avez un script shell qui exécute sept passes d'inscriptions aléatoires de caractères sur tout le disque. Il faut rendre le script exécutable avec la commande :

**marc@marc-MS-7721 :~\$ chmod a+x efface\_partition**

et lancer le script :

**marc@marc-MS-7721 :~\$ sudo ./efface\_partition**

## 25.7 VIDÉO

Télécharger exo-device-2.ogv et lire avec VLC.

# Chapitre 26

## PRATIQUE SUR PÉRIPHÉRIQUES N°2

Un peu plus de pratique afin de mieux saisir l'utilisation des périphériques.

### 26.1 CRÉATION D'UN FICHIER SIMULANT UNE PARTITION

Nous allons créer un fichier qui pourra être utilisé comme simulacre de partition, pour ce faire nous allons utiliser la commande **dd**.

- Création d'un fichier de 100Meg composé de zéro
  - # Création du fichier de 100Meg

```
$ dd if=/dev/zero of=fac_hd.dsk bs=1 count=0 seek=100M
```
  - # Validation de la taille

```
$ du -hs fac_hd.dsk
```
  - 101M fac\_hd.dsk
    - # Visualisation du type de fichier
    - \$ file fac\_hd.dsk fac\_hd.dsk : data
- Formatage du fichier
  - \$ mkfs.ext4 ./fac\_hd.dsk
  - mke2fs 1.42.9 (28-Dec-2013) ./fac\_hd.dsk n'est pas un périphérique spécial en mode bloc.  
Procéder malgré tout ? (o,n) o  
Rejet des blocs de périphérique : complété
  - # validation du type de fichier
  - \$ file fac\_hd.dsk fac\_hd.dsk : Linux rev 1.0 ext4 filesystem data, UUID=5c1ad91b-f5c4-4171-a74d-04e672ce468e (extents) (huge files)

- Monte le fichier
  - # monter le fichier
 

```
$ sudo mount -o loop fac_hd.dsk /mnt/
# visualisation
$ df -h | grep mnt
/dev/loop0 93M 1.6M 85M 2% /mnt
```
- Écrire un fichier texte avec du contenu, peu importe le contenu
  - # Création du fichier wikipedia à l'aide du clavier
 

```
$ cat > wikipedia << EOF
Tornade - L alerte
Tornade - L alerte
Données clés Titre original Tornado – Der Zorn des Himmels
Réalisation Andreas Linke
Scénario Don Bohlinger
Stephan Lacant
EOF
```
- Faire une recherche sur le périphérique **/dev/loop0** avec le texte
  - \$ sudo -i grep -binary-files=text "Tornade" /dev/loop0
- Démontez et recommencez
  - # démonter et refaire la commande
  - \$ sudo umount /dev/loop0
  - # refaire la recherche
 

```
$ sudo -i grep -binary-files=text "Tornade" /dev/loop0
```
  - # remonter le fichier
 

```
$ sudo mount -o loop fac_hd.dsk /mnt
```
  - # refaire la recherche
 

```
$ sudo -i grep -binary-files=text "Tornade" /dev/loop0
```

# Chapitre 27

## RÉVISION REDIRECTION ET FILTRES

Un atelier de plus afin de faire une révision sur certains aspects couverts jusqu'à présent.

Petite révision pour le plaisir, principalement sur les redirections et filtres

### 27.1 THÉORIE

- Redirection et flux de données
- Chaînage de commande

### 27.2 DÉMONSTRATION

- Téléchargement et extraction en deux temps d'un fichier tar.gz
  - wget http://x3rus.com/demo.tar.gz (télécharge le fichier dans le répertoire courant)
  - tar -ztf demo.tar.gz (liste l'archive)
- Téléchargement et extraction direct d'un fichier tar.gz
  - wget http://x3rus.com/demo.tar.gz -O - (télécharge le fichier et l'envoie vers le standard output (ici l'écran))
  - wget http://x3rus.com/demo.tar.gz -O - | tar -ztv (télécharge le fichier et l'envoie vers tar pour décompression à l'écran)
  - wget http://x3rus.com/demo.tar.gz -O - | tar -zxv (télécharge le fichier et le décomprime dans le répertoire courant)

wget télécharge une page et fait jouer une URL ( joue un fichier mp3 depuis un site web ).

```
wget -qO- "http://dictionary.reference.com/browse/dog" | grep mp3 | head -n 1 | sed 's/.*/\1/' | xargs mpg123
```

## 27.3 EXERCICES

- extraire de la commande history toute les commandes réalisées avec sudo
- affichez le contenu d'un fichier sans les lignes vides et sans les commentaires (ligne contenant un #). Indice ou rappel le symbole ^ représente le début d'une ligne et \$ représente la fin d'une ligne. Pour faire l'exercice vous pouvez utiliser le fichier /etc/rsyslog.conf. Voici le résultat attendu : rsyslog sans les retours de ligne

```
$ActionFileDefaultTemplate RSYSLOG_TraditionalFileFormat
$RepeatedMsgReduction on
$FileOwner syslog
$FileGroup adm
$FileCreateMode 0640
$DirCreateMode 0755
$Umask 0022
$PrivDropToUser syslog
$PrivDropToGroup syslog
$WorkDirectory /var/spool/rsyslog
$IncludeConfig /etc/rsyslog.d/*.conf
```

- grâce à la commande **lsof** lister l'ensemble des fichiers ouverts sur le système, formater l'affichage pour qu'il affiche le nombre de fichiers ouverts par application. Exemple de résultat :
- #triez le résultat de fin
 

```
306 Socket
306 Timer
306 URL
446 threaded-
568 pool
612 Analysis
612 Cache
612 JS
720 evolution
774 indicator
918 DNS
918 DOM
2536 gmain
2754 mozStorage
3802 dconf
4778 dbus
```
- extraire le fichier <http://x3rus.com/revision-music.tar.gz>, le fichier contient des mp3, à l'aide de la commande mp3info extraire le genre de musique et créer le répertoire pour chaque genre. Ceci doit être réalisé en une seule commande ; xargs est votre ami.
- Pour les pros !! Faire une deuxième commande pour copier les fichiers selon le genre dans le bon répertoire : [pas facile ]

## 27.4 VIDÉOS

Télécharger revision-pipe.webm et revision-pipe2.webm et lire avec VLC.

## 27.5 SOLUTIONS

- extraire de la commande history toutes les commandes réalisées avec sudo
  - marc@marc-MS-7721 :~\$ history | grep sudo
- afficher le contenu d'un fichier sans les lignes vides et sans les commentaires (ligne contenant un # ), indice ou rappel le symbole ^ représente le début d'une ligne et \$ représente la fin d'une ligne. Pour faire l'exercice vous pouvez utiliser le fichier /etc/rsyslog.conf
  - marc@marc-MS-7721 :~\$ cat /etc/rsyslog.conf | grep -v "#" | grep -v "^\$" ou
  - marc@marc-MS-7721 :~\$ cat /etc/rsyslog.conf | grep "^[^#]"
- grâce à la commande **lsof** lister l'ensemble des fichiers ouverts sur le système, formater l'affichage pour qu'il affiche le nombre de fichiers ouverts par application.
  - lsof |cut -d " " -f 1 | sort | uniq -c (*avec un tri sur le nom*)
  - lsof |cut -d " " -f 1 | sort | uniq -c | sort -n (*avec un tri sur le nombre*)
- extraire le fichier http://x3rus.com/revision-music.tar.gz, le fichier contient des mp3, à l'aide de la commande mp3info extraire le genre de musique et créer le répertoire pour chaque genre. Ceci doit être réalisé en une seule commande ; xargs est votre ami.
  - wget http://x3rus.com/revision-music.tar.gz
  - cd revision-music
  - sudo apt-get install mp3info
  - marc@marc-MS-7721 :~/revision-music\$ mp3info \*.mp3
  - \$ mp3info \*.mp3 | grep Comment | cut -d : -f3 | cut -d " " -f2 | sort | uniq | 'xargs' mkdir
  - ou bien si l'on exploite les informations du manpage, on établit la commande :  
mp3info -p "%g\n" \* | sort | uniq | 'xargs' mkdir

Cette page provient de <https://doc.ubuntu-fr.org/wget>

## 27.6 WGET

**wget** est un programme en ligne de commande non interactif de téléchargement de fichiers depuis le Web. Il supporte les protocoles HTTP, HTTPS et FTP ainsi que le téléchargement au travers des proxies HTTP.

**wget** peut travailler en arrière-plan et ainsi vous permettre de lancer un téléchargement et de vous déconnecter du système ! Utile car il ne requiert d'action de l'utilisateur et vous permet d'effectuer ses tâches en arrière plan, ce qui peut être très utile pour les téléchargements de données nombreuses et lourdes. Vous pouvez ainsi changer de session et laisser **wget** finir le travail !

Ce logiciel libre permet le simple téléchargement d'un fichier mais aussi la recopie en local tout ou partie d'un site qui sera par la suite consultable hors-ligne. Point fort appréciable, **wget** vous permet de reprendre un téléchargement échoué suite à divers problèmes (connexions instables ou très lentes etc...). Les nombreuses options de **wget** en font un outil de téléchargement très puissant !

## UTILISATION

`wget [options] [url]`

Par défaut, le fichier sera enregistré dans le dossier courant, ordinairement votre dossier personnel /home/vous.

## DESCRIPTIONS DES OPTIONS PRINCIPALES

**-V** -> Renseigne sur la version de Wget.

**-h** -> Affiche toutes les options disponibles.

**-i** -> Lit les adresses depuis un fichier .txt.

**-N** -> ( -timestamping ) Active l'horodatage (time-stamping). Permet d'écraser les fichiers sur la destination s'ils existent

**-t** -> Définit le nombre de tentatives, 0 ou inf pour un nombre illimité de tentatives. Par défaut 20 tentatives sont effectuées, à moins qu'une erreur fatale apparaisse.

**-c** -> Reprend un téléchargement interrompu.

**-S** -> Affiche les messages envoyés par les serveurs FTP ou HTTP.

**-Y** -> on/off active ou désactive le support proxy.

**-nd** -> Si cette option est utilisée les fichiers sauvegardés le seront dans un seul et même répertoire.

**-r** -> Active le téléchargement récursif.

-l -> Indique la profondeur à utiliser lors d'un téléchargement récursif.

-k -> Convertit les liens pour être disponible en consultation locale.

-p -> Oblige Wget à télécharger tous les fichiers requis pour une consultation convenable d'une page HTML.

-m -> Active toutes les options convenables pour faire un miroir.

-follow-ftp -> Sans cette option tous les liens FTP donnés depuis un document HTML seront ignorés.

-H -> Autorise lors d'un téléchargement récursif le changement d'hôtes.

-np -> Ne remonte pas les répertoires parents.

-A -> Permet de ne télécharger que le type de fichier choisi.

-o -> Permet d'enregistrer tous les messages de Wget dans un fichier.

-a -> Idem que -o sauf que les messages sont ajoutés à la suite du fichier empêchant ainsi d'écraser l'ancien fichier.

-user-agent=paul -> S'identifier sous le nom paul pour le serveur HTTP. Permet de falsifier la valeur de user-agent envoyé par Wget, ceci n'est pas recommandé à moins de vraiment savoir ce que vous faites

-limit-rate=30K -> Permet de limiter le débit, ici à 30 ko/s.

-q -> Évite l'affichage des messages du wget.

D'autres options sont bien sûr disponibles, elles sont disponibles sur le manpage en français. (<http://www.delafond.org/traducmanfr/man/man1/wget.1.html>)

## EXEMPLES D'UTILISATIONS COURANTES

Télécharger simplement un fichier :

```
wget http://www.site.org/rep/01/fichier.txt
```

Reprendre un téléchargement si celui-ci est incomplet ( option inutile s'il s'agit de retenter un téléchargement échoué )

```
wget -c ftp://serveur.org/rep/01/fichier.txt
```

Dans ce cas seul le répertoire /01/ sera copié ( -np ), les répertoires parents étant ignorés :

```
wget -r -np http://www.site.org/rep/01/
```

Ici aucun répertoire ne sera créé ( -nd ) :

```
wget -r -nd http://www.site.org/rep/01/
```

Télécharger sur un FTP avec authentification (ici le nom d'utilisateur est paul et son mot de passe 123) :

```
wget -r l4 ftp ://paul :123@serveur.org/
```

Spécifier un dossier de destination.

```
wget -P $HOME/dossier/de/destination http ://www.site.org/rep/01/fichier.txt
```

## EXEMPLES D'UTILISATION AVANCÉE

Télécharger les URL contenues dans un fichier :

```
wget -i fichier
```

Télécharge récursivement le site ( -r ) et enregistre les messages dans le fichier wgetlog ( -o ) en limitant le débit du téléchargement à 30 Ko/s ( --limit-rate=30k ) :

```
wget -r --limit-rate=30k http://www.site.org/ -o wgetlog
```

Téléchargement avec une profondeur de 4 ( -l4 ) et en enregistrant les messages à la suite du fichier wgetlog déjà créé ( -awgetlog ) :

```
wget -r -l4 http ://www.site.org/ -awgetlog
```

Dans ce cas seul les fichiers de type .txt seront téléchargés ( -A.txt ) :

```
wget -r -l3 -A.txt ftp ://serveur.org/
```

Ici seul les fichiers .jpg seront téléchargés ( -A.jpeg ), les messages seront inscrit à la suite du fichier wgetlog ( -awgetlog ) et Wget téléchargera à partir des adresses indiqués dans le fichier maliste.txt ( -imliste ) :

```
wget -r -A.jpg -awgetlog -imliste.txt
```

Télécharger le site récursivement avec une profondeur infinie ( -linf ), convertit les liens pour une consultation en local ( -k ), rapatrie tous les fichiers nécessaires à l'affichage convenable d'une page HTML ( -p ) et renomme toutes les pages HTML avec l'extension .html ( -E ) :

```
wget -r -linf -k -p -E http ://www.site.org/
```

# Chapitre 28

## SCRIPTING AVEC BASH

De quoi s'agit-il ? Imaginez un mini langage de programmation intégré à Linux. Ce n'est pas un langage aussi complet que peuvent l'être le C, le C++ ou le Java par exemple, mais cela permet d'automatiser la plupart de vos tâches : sauvegarde des données, surveillance de la charge de votre machine, etc.

### 28.1 INTRODUCTION AU SCRIPTING BASH

Un script shell permet d'automatiser une série d'opérations. Il se présente sous la forme d'un fichier contenant une ou plusieurs commandes qui seront exécutées de manière séquentielle.

#### DÉFINITION

Un **langage de script** est un langage de programmation qui permet de manipuler les fonctionnalités d'un système informatique configuré pour fournir à l'interpréteur de ce langage un environnement et une interface qui déterminent les possibilités de celui-ci. Le langage de script peut alors s'affranchir des contraintes de bas niveau prises en charge par l'intermédiaire de l'interface et bénéficier d'une syntaxe de haut niveau.

Le langage de script est généralement exécuté à partir de fichiers contenant le code source du programme qui sera interprété. Historiquement, ils ont été créés pour raccourcir le processus traditionnel de développement « édition, compilation, édition des liens, exécution » propre aux langages compilés. Les premiers langages étaient souvent appelés « langage de commandes » ou « langage d'enchaînement des travaux » (JCL : Job Control Language) car ils permettaient simplement d'automatiser une succession de commandes simples, à la manière d'un « script » de théâtre. Par la suite, ils furent munis d'exécutions conditionnelles implicites (IBM 1130) ou explicites (JCL), et enfin d'ordres de boucle et d'opérateurs les transformant en quasi-langages de programmation.

## 28.2 RÉALISATION DE NOTRE PREMIER SCRIPT

Tel que mentionné dans la définition ci-dessous, un script est en fait un fichier qui sera lu par l'interpréteur de commande. Nous avions vu lors de la description de GNU/Linux qu'il existe plusieurs interpréteurs de commande, par défaut l'interpréteur est **bash**. Cependant il en existe d'autre tels que **Zsh**, **tcs**, **csh**, ... Je vous invite à relire cette section si vous désirez vous rafraîchir la mémoire, cependant ici nous nous concentrerons sur **Bash**. Ce chapitre ne sera pas une formation de programmation **Bash** ceci prendrait beaucoup de temps. Cependant je vais réaliser une introduction afin de comprendre le fonctionnement d'un script .

Commençons si vous le voulez bien par écrire un petit script, ce sera le classique "hello world". Pour commencer, on va le faire dans le terminal, on utilisera la commande echo qui permet d'afficher un message, exemple : affiche « hello world » dans le terminal.

```
username@hostname :~$ echo "hello world"
```

```
hello world
```

C'est bien, mais ce que l'on veut c'est faire un script. Utilisez votre éditeur préféré et créez le fichier **hello\_world.sh**, personnellement j'utiliserais **VIM** mais libre à vous d'utiliser d'autre éditeur comme nano. Voici le contenu du fichier : hello\_world.sh

```
echo "hello world"
```

Maintenant nous allons exécuter le script. Voici comment procéder :

exécution de hello\_world.sh

```
username@hostname :~$ bash hello_world.sh
```

```
hello world
```

On donne au shell **bash** en argument le script hello\_world.sh et le script est exécuté. Bien entendu nous pourrions mettre n'importe quelle commande que nous utilisons dans la console à l'intérieur du script. Un autre exemple : je rajoute une archive du répertoire /etc.

Soit le fichier **archive\_etc.sh** contenant :

```
echo " Archivage de /etc dans /tmp "
```

```
tar -zcvf /tmp/etc.tar.gz /etc
```

même commande pour exécuter le script :

```
username@hostname :~$ bash archivage_etc.sh
```

```
Archivage de /etc dans /tmp
```

```
tar : Removing leading '/' from member names
```

```
tar : /etc/NetworkManager/system-connections/maison : Cannot open : Permission denied
```

```
/etc/
/etc/gai.conf
/etc/profile
/etc/bash.bash_logout
/etc/pulse/
/etc/pulse/client.conf
/etc/NetworkManager/
/etc/NetworkManager/NetworkManager.conf
[[ .. OUTPUT COUPÉ ... ]]
```

Afin de ne pas être obligé de nommer l'interpréteur, ici **bash**, avant d'appeler le script nous pouvons le définir dans le fichier directement. Voici la nouvelle version du script **hello\_world.sh**.

```
#!/bin/bash
# Affichage de hello world echo "hello world"
```

Le fichier en détail :

- ligne 1 : définition de l'interpréteur à utiliser
- ligne 2, 3 : commentaires, les lignes qui commencent par # sont des commentaires.
- ligne 4 : l'instruction **bash** qui sera exécutée

Toutes les commandes utilisables dans le terminal peuvent être définies dans le fichier de scripting. L'ensemble des commandes que nous avons vues peuvent être mises dans un fichier. Pour pouvoir exécuter le fichier, il va falloir modifier ce dernier pour le rendre exécutable avec la commande **chmod**.

Changement de permission :username@hostname :~\$ chmod u+x hello\_world2.sh

Exécution du script : username@hostname :~\$./hello\_world2.sh hello world

Mais pourquoi le ./ me direz vous ? Si nous tapions la commande directement hello\_world2.sh, nous aurions le message **bash : hello\_world.sh : command not found**. Quand on tape une commande, le shell bash cherche dans la liste des répertoires définis dans la variable **\$PATH**, le fichier exécutable portant le nom entré sur la ligne de commande.

Vous pouvez afficher la valeur de la variable **\$PATH** avec echo :

```
username@hostname :~$ echo $PATH
/usr/local/sbin :/usr/local/bin :/usr/sbin :/usr/bin :/sbin :/bin :/usr/games
username@hostname :~$ which ls
/bin/ls
```

Dans l'exemple ci-dessus, quand on utilise la commande **ls**, le système va rechercher le fichier exécutable **ls** dans le répertoire **/usr/local/sbin**, puis **/usr/local/bin** suivi de **/usr/sbin** par la suite **/usr/bin** et encore **/sbin** et finalement exécutera le fichier dans le répertoire **/bin**. Dans notre exemple le shell ne recherchera pas dans le répertoire **/usr/games**, car l'exécutable a été trouvé avant.

Comme le répertoire où nous nous trouvons ne fait pas partie des répertoires listés, nous devons fournir le chemin complet pour accéder au fichier.

Donc si je reprends l'exécution de **hello\_world2.sh** nous pouvons l'écrire comme suit :

```
# exécution du script
username@hostname :~$./hello_world2.sh
hello world
# OU
username@hostname :~$/home/username/hello_world2.sh
hello world
```

Les deux notations sont équivalentes, le **".."** (point) représente le répertoire courant. Pour rappel le **".."** représente le répertoire précédent. Quand j'utilise l'annotation **"./"** ceci équivaut à mettre le répertoire complet courant.

Il est aussi possible de modifier la variable **\$PATH** comme suit :

```
username@hostname :~$ export PATH=$PATH :/home/username/
username@hostname :~$ echo $PATH
/usr/local/sbin :/usr/local/bin :/usr/sbin :/usr/bin :/sbin :/bin :/usr/games :/home/username/
username@hostname :~$ hello_world2.sh hello world
```

Donc dans le cas présent le shell va parcourir l'ensemble des répertoires avant d'exécuter notre script, ce qui n'est vraiment pas un problème en terme de performance.

## 28.3 LES VARIABLES

Nous avons déjà fait un aperçu des variables avec la variable **\$PATH** présentée juste avant. Voyons les autres variables disponibles avec le système, la commande **env** peut vous fournir une liste des variables disponibles.

```
username@hostname :~$ env
TERM=xterm
SHELL=/bin/bash
XDG_SESSION_COOKIE=6a603b66850cd8643667fdca0000000a-1394754492.491257-1773303034
SSH_CLIENT=192.168.42.204 1420 22
SSH_TTY=/dev/pts/7
USER=username
MAIL=/var/mail/username
PATH=/usr/local/sbin :/usr/local/bin :/usr/sbin :/usr/bin :/sbin :/bin :/usr/games
LC_MESSAGES=fr_FR.UTF-8
LC_COLLATE=fr_FR.UTF-8
PWD=/home/username
LANG=fr_CA.UTF-8
SHLVL=1
HOME=/home/username
LANGUAGE=fr :en
LOGNAME=username
LC_CTYPE=fr_FR.UTF-8
SSH_CONNECTION=192.168.42.204 1420 192.168.42.10 22
_==/usr/bin/env
```

Comme vous constatez la variable **\$PATH** est présente, il est aussi possible de définir nos variables. Pour ce faire, simplement donner un nom, ajouter le symbole **=** (attention ce dernier doit être collé au nom de la variable sans espace) et assigner la valeur. L'ensemble des variables sont en majuscules. Ceci n'est PAS obligatoire mais ceci est une convention. Voici un exemple d'assignation de variable :

Assigne la valeur **/tmp** à **DST\_REPO** :

```
username@hostname :~$ DST_REPO=/tmp
username@hostname :~$ echo $DST_REPO
/tmp
```

Pour faire appel à la variable, on utilise le symbole \$ devant le nom de la variable. Reprenons mon script d'archivage :

```
#!/bin/bash
#
# Description : Archive un répertoire
#####
# Variable
DIR_2_ARCHIVE=/etc
# MAIN
echo " Archivage de $DIR_2_ARCHIVE dans /tmp "
tar -zcf /tmp/archive.tar.gz $DIR_2_ARCHIVE
echo " DONE "
```

Nous voyons l'utilisation de la variable **DIR\_2\_ARCHIVE** pour afficher un message mais aussi pour l'utiliser dans une commande. Le petit problème est que cette variable est une constante elle ne peut pas changer sans qu'une personne édite le fichier. L'idéal est de pouvoir passer en argument le nom du répertoire que l'on désire archive. Heureusement quand on appelle un script, des variables sont initialisées avec l'information voulue, voici une table :

\$*	contient tous les arguments passés à la fonction
\$#	contient le nombre d'argument
\$?	contient le code de retour de la dernière opération
\$0	contient le nom du script
\$1, \$2, \$3	contient le premier argument, le deuxième, etc ...

À la lumière de cette information on va reprendre le script d'archivage :

```
#!/bin/bash
#
# Description : Archive un répertoire
#####
# Variable
DIR_2_ARCHIVE=$1
# MAIN
echo " le script $0 réalise l'archivage de $DIR_2_ARCHIVE dans /tmp "
tar -zcf /tmp/archive.tar.gz $DIR_2_ARCHIVE
```

```
echo " DONE, la commande était $0 $* "
```

Si nous réalisons un test du script, nous l'appellerions comme suit :

```
username@hostname :~$ ./archivage /tmp
tar : Removing leading '/' from member names
tar : /tmp/pulse-PKdhtXMmr18n : Cannot open : Permission denied
tar : /tmp/.X11-unix/X0 : socket ignored
tar : /tmp/cvcd : Cannot open : Permission denied
tar : /tmp/keyring-uZh1m5/control : socket ignored
tar : /tmp/OSL_PIPE_1000_SingleOfficeIPC_dfc3cd856fa49aedd2b54d711f42fc : socket ignored
tar : /tmp/.lxterminal-socket\ :0.0-xerus : socket ignored
tar : /tmp/pulse-m5tkYEvRYHHT/native : socket ignored
tar : /tmp/.vbox-xerus-ipc/ipcd : socket ignored
tar : /tmp/e-xerus@0/\ :0.0-6454|0 : socket ignored
tar : /tmp/ssh-EtxCQxFm6415/agent.6415 : socket ignored
tar : /tmp/.winbinddd/pipe : socket ignored
tar : /tmp/archive.tar.gz : file changed as we read it
tar : Exiting with failure status due to previous errors DONE, commande été ./archivage.sh /tmp
```

Comme vous le constatez il y a quelques erreurs ici à cause du problème de permissions. Je vous laisse faire la redirection des messages d'erreurs dans un autre fichier grâce à la redirection du canal **stderr (2)**.

Autres éléments intéressants, il est possible d'assigner le résultat d'une commande dans une variable. Par exemple nous aimerais écrire la date dans le nom du fichier d'archive, exemple /tmp/archivage-2014-03-13.tar.gz. La commande **date** nous permet d'afficher la date du jour et avec l'argument **+%F** nous avons le format d'affichage qui nous convient voici l'exemple :

```
username@hostname :~$ date +%F
```

```
2014-03-13
```

dans le script ceci donne :

```
#!/bin/bash
# Description : Archive un répertoire
#####
# Variable
```

```

DIR_2_ARCHIVE=$1
DATE=$(date +%F)
# autre notation ancienne possible DATE='date +%F'
# MAIN
echo " le script $0 réalise l'archivage de $DIR_2_ARCHIVE dans /tmp " tar -zcf /tmp/archive-$DATE.tar.gz $DIR_2_ARCHIVE
echo " DONE, la commande était $0 $* "

```

Il est aussi possible d'acquérir de l'information lors de l'exécution du script avec la commande **read**. Reprenons le script vu plus haut. Au lieu de définir **\$DIR\_2\_ARCHIVE** avec un argument dans le script, faisons l'acquisition de l'information lors de l'exécution.

```

#!/bin/bash
#
# Description : Archive un répertoire
#####
# Variable
DIR_2_ARCHIVE=""
DATE=$(date +%F)
# autre notation possible DATE='date +%F'
echo -n " Entrez le nom du répertoire : "
read DIR_2_ARCHIVE
# MAIN
echo " le script $0 réalise l'archivage de $DIR_2_ARCHIVE dans /tmp "
tar -zcf /tmp/archive-$DATE.tar.gz $DIR_2_ARCHIVE
echo " DONE , la commande était $0 $* "

```

Dans l'exemple ci-dessus j'affiche le message au client. J'utilise l'option **-n** à **echo** afin qu'il ne réalise PAS un retour à la ligne après avoir affiché le message. La commande **read** à la ligne suivante attendra une saisie de texte du client et assignera la valeur entrée dans la variable **\$DIR\_2\_ARCHIVE**.

## 28.4 LES STRUCTURES DE CONTRÔLE

### STRUCTURE CONDITIONNELLE ( IF )

Suivant l'ordre des choses, après les variables, voyons les structures de contrôle. Mais c'est quoi une structure de contrôle, ça contrôle quoi ?

Une structure de contrôle nous permet de réaliser des validations sur une variable, testons si sa valeur est **égale** à quelque chose, **plus grande**, **plus petite** ou correspond à un état, etc ...

Continuons avec mon script d'archivage. Ici je réalise un **tar** mais ce pourrait tout aussi bien être un **cp** ou n'importe quelle commande. Nous avons modifié le script pour permettre à l'utilisateur de transmettre le répertoire à prendre en archive, en argument du script . Il est fort à parier que ceci va causer un problème et que l'utilisateur va transmettre un mauvais argument et plus rien ne va fonctionner. Afin de corriger le problème nous allons valider l'argument, nous allons vérifier que **\$DIR\_2\_ARCHIVE** est un fichier. Il existe la commande **test** qui permet de faire de la validation. Voici un exemple d'utilisation de la commande test.

```
username@hostname :~$ test -e /etc/
username@hostname :~$ echo $?
0
username@hostname :~$ test -e /etc/passwd
username@hostname :~$ echo $?
0
username@hostname :~$ test -e /etc/toto
username@hostname :~$ echo $?
1
```

La commande **test** ne retourne aucune information, cependant elle assigne (on dit qu'elle retourne) une valeur à la variable **\$?**

TOUTES les commandes enregistrent la valeur de retour dans cette variable :

- **0 == OK,**
- **>0 == problème.**

Ceci est aussi vrai pour la commande **cp** par exemple :

```
username@hostname :~$ cp /etc/passwd /tmp/
username@hostname :~$ echo $?
0
username@hostname :~$ cp /etc/shadow /tmp/
cp : cannot open '/etc/shadow' for reading : Permission denied
```

```
username@hostname :~$ echo $?
1
```

Donc on ajoute la condition dans le script d'archivage :

```
1 #!/bin/bash
2 #
3 # Description : Archive un répertoire
4 #####
5 #
6 # Variable
7 DIR_2_ARCHIVE=$1
8 DATE=$(date +%F)
9 # autre notation possible DATE=`date +%F`
10 #
11 # MAIN
12 # autre notation
13 # if [ -e $DIR_2_ARCHIVE ]
14 if test -e $DIR_2_ARCHIVE
then
    echo " le script $0 réalise l'archivage de $DIR_2_ARCHIVE dans /tmp "
    tar -zcf /tmp/archive-$DATE.tar.gz $DIR_2_ARCHIVE
    echo " DONE , commande été $0 $* "
else
    echo " $DIR_2_ARCHIVE n existe pas !! "
fi
```

Si nous regardons la ligne où nous avons la condition **if Commande**, si la commande retourne **\$?==0** alors le script réalise les lignes 16,17 et 18, sinon (**else**) ligne 19, alors le script réalise la ligne 20 et la structure de contrôle est finie par **fi** à la ligne 21. Je vous invite à consulter le man page de la commande **test**, **man test**.

Il est important de comprendre que la commande « **if commande** » interprète le résultat d'une commande. Il est donc possible de mettre n'importe quelle commande. Voici un exemple avec la commande **whoami** et **grep**.

```
username@hostname :~$ whoami
username
username@hostname :~$ whoami | grep root
username@hostname :~$ echo $?
1
username@hostname :~$ whoami | grep username
username username@hostname :~$ echo $?
0
```

Je modifie le script pour obliger l'utilisateur d'être root quand il exécute le script.

```

1 #!/bin/bash
2 #
3 # Description : Archive un répertoire
4 #####
5
6 # Variable
7 DIR_2_ARCHIVE=$1
8 DATE=$(date +%F)
9 # autre notation possible DATE=`date +%F`
10
11 # MAIN
12
13 # Validation de l'utilisateur
14 if whoami | grep root
15 then
16     # Validation de l'argument
17     if test -e $DIR_2_ARCHIVE
18     then
19         echo " le script $0 réalise l'archivage de $DIR_2_ARCHIVE dans /tmp "
20         tar -zcf /tmp/archive-$DATE.tar.gz $DIR_2_ARCHIVE
21         echo " DONE , commande été $0 $* "
22     else
23         echo "$DIR_2_ARCHIVE n existe pas !! "
24     fi # test -e $DIR_2_ARCHIVE
25 else
26     echo "Il faut être root pour executer ce script"
27 fi # whoami | grep root

```

Résultat à l'exécution :

```

username@hostname :~$ ./archivage.sh
il faut être root pour exécuter ce script
username@hostname :~$ ./archivage.sh /tmp
il faut être root pour exécuter ce script
username@hostname :~$ sudo ./archivage.sh /tmp/ 2>/dev/null
root
le script ./archivage.sh réalise l'archivage de /tmp/ dans /tmp
DONE, la commande était ./archivage.sh /tmp/

```

Vous constaterez qu'après chaque **fi** j'identifie la commande d'origine. Ceci facilite le « débogage » surtout si vous avez plusieurs **if** imbriqués.

## STRUCTURE DE RÉPÉTITION ( FOR )

La structure de contrôle **for** nous permet de réaliser une boucle pour un nombre de fois déterminé. La boucle **for** va parcourir une liste et exécutera une liste d'instructions pour chaque élément de la liste. Voici deux exemples simples :

```
# exemple simple for 1

# Exemple de boucle for avec une variable pré-définie

LST_VILLE="Paris Montreal Marseille Berlin Rennes Sydney"

for ville in $LST_VILLE

do

    echo "ville : $ville"

done # for ville
```

résultat exemple for 1

```
username@hostname :~$ bash exemple.sh

ville : Paris

ville : Montreal

ville : Marseille

ville : Berlin

ville : Rennes

ville : Sydney
```

Dans l'exemple ci-dessus nous voyons que nous avons une Variable **\$LST\_VILLE**. La boucle **for** va parcourir la liste des villes et prendre chaque élément 1 par 1 et assigner l'élément en cours de traitement dans la variable **\$ville**. Une fois la variable assignée, **for** réalise le bloc compris entre **do** et **done**, dans ce bloc il est possible d'utiliser la variable **\$ville**. Comme vous le constatez, ici j'ai uniquement affiché le nom de la ville. Vous constaterez que je n'ai pas défini cette variable en majuscule, ceci est purement personnel. Pour les variables "temporaires" de traitement dans **for** j'ai l'habitude de pas les mettre en majuscule.

Bien entendu **for** est en mesure de traiter autre chose que des variables pré-définies, il peut aussi interpréter le résultat d'une commande. Voici un autre exemple :

```

exemple for avec commande ls
1 #!/bin/bash
2 #
3 # Liste les fichiers /etc/*.conf avec un for
4
5 for fic in $(ls /etc/*.conf)
6 do
7     echo "fichier config : $fic"
8 done # fic in $(ls /etc/*.conf)

résultat commande for avec ls /etc/*.conf
1 [[ ... OUTPUT COUPÉ ... ]]
2 fichier config : /etc/pnm2ppa.conf
3 fichier config : /etc/popularity-contest.conf
4 fichier config : /etc/request-key.conf
5 fichier config : /etc/resolv.conf
6 fichier config : /etc/rsyslog.conf
7 fichier config : /etc/sensors3.conf
8 fichier config : /etc/smi.conf
9 fichier config : /etc/sword.conf
10 fichier config : /etc/sysctl.conf
11 fichier config : /etc/ts.conf
12 fichier config : /etc/ucf.conf
13 fichier config : /etc/updatedb.conf
14 fichier config : /etc/usb_modeswitch.conf
15 fichier config : /etc/wodim.conf
16 fichier config : /etc/x3_formation.conf

```

Dans le cas présent la commande **for** reçoit comme liste le résultat de la commande **ls /etc/\*.conf**. Par la suite le processus est le même, chaque entrée est assignée 1 à 1 à la variable **\$fic** qui est utilisable dans les blocs entre les lignes 6 et 8.

Bon pour le moment il est peut-être difficile d'en voir l'utilité. Donc reprenons notre script de backup / archivage, voici un exemple avec la commande **for**

```

script d'archivage / backup avec boucle for
1 #!/bin/bash
2 #
3 # Description :
4 # Réalisation d'un backup / archivage des répertoires lister par la variable $LST_REPO_2_BK
5 # chaque fichier tar.gz contient le nom du dernier répertoire et la date
6 # l'ensemble est réalisé dans le répertoire $DST_REPO
7 #
8 # Auteur : Thomas Boutry <*****@x3rus.com>
9
10 # Variables
11 LST_REPO_2_BK="/etc /var/backup /var/spool/mail"
12 DST_REPO="/tmp/Mybackup/"
13 DATE=$(date +%F)
14
15
16 # Main
17
18 for rep2bk in $LST_REPO_2_BK
19 do
20     # Extraction pour avoir que le nom court sans le /
21     rep2bk_prefix=$(basename $rep2bk)
22
23     # réalisation du tar de chaque répertoire avec le nom significatif
24     # et la date
25     tar -zcf $DST_REPO/$rep2bk_prefix-$DATE.tar.gz $rep2bk
26
27 done # rep2bk in $LST_REPO_2_BK
28

```

## 28.5 DEBUG

Il peut devenir difficile de « débugger » un script bash SURTOUT s'il y a beaucoup de variables et qu'elles sont réinitialisées pendant le script. Heureusement il est possible d'exécuter le script en mode « debug ». Voici un exemple avec le script défini plus haut.

*mode debug du script d'archivage*

```
1 username@hostname:~$ bash -x archivage_for.sh
2 + LST REP_2_BK='/etc /var/backup /var/spool/mail'
3 + DST REP=/tmp/Mybackup/
4 ++ date +%F
5 + DATE=2014-03-17
6 + for rep2bk in '$LST REP_2_BK'
7 ++ basename /etc
8 + rep2bk_prefix=etc
9 + tar -zcf /tmp/Mybackup//etc-2014-03-17.tar.gz /etc
10 + for rep2bk in '$LST REP_2_BK'
11 ++ basename /var/backup
12 + rep2bk_prefix=backup
13 + tar -zcf /tmp/Mybackup//backup-2014-03-17.tar.gz /var/backup
14 + for rep2bk in '$LST REP_2_BK'
15 ++ basename /var/spool/mail
16 + rep2bk_prefix=mail
17 + tar -zcf /tmp/Mybackup//mail-2014-03-17.tar.gz /var/spool/mail
```

Comme vous pouvez le voir, l'ensemble des commandes réalisées apparaissent ainsi que les valeurs des variables, très intéressant.

# Chapitre 29

## GESTION DE PÉRIPHÉRIQUES - suite

Suite à l'acquisition de connaissances sur le scripting langage, nous allons revenir sur le système de gestion de périphériques, particulièrement avec **udev**.

### 29.1 SYSTÈME UDEV

Contrairement au système traditionnel de gestion de périphériques sous Linux, qui utilisait un ensemble statique de nœuds de périphériques, **udev** est un système de gestion de périphériques qui fournit dynamiquement des nœuds seulement pour les périphériques réellement présents sur le système.

Le système **udev** est composé de quelques services du noyau et du démon **udevd**. Le noyau avertit le démon **udevd** que tel ou tel événement est arrivé. Le démon **udevd** est configuré pour répondre aux événements par des actions correspondantes. L'information sur l'événement vient du noyau, les actions se produisent dans l'espace utilisateur. Les réponses aux événements sont configurables dans des "règles".

### 29.2 CONCEPT

#### TERMINOLOGIE

Sur les systèmes à base de Linux, le répertoire **/dev** sert à contenir les périphériques sous forme de fichier, les "nodes", qui se rapportent aux périphériques système. Chaque "node" se réfère à un périphérique, qui existe ou non. Les applications « utilisateur » peuvent utiliser ces "nodes" pour interagir avec le périphérique. Par exemple, le serveur graphique X va "écouter" **/dev/input/mice** qui se réfère à la souris et faire bouger le pointeur à l'écran.

Le répertoire original **/dev** contenait tous les périphériques du système, c'est pourquoi il était si volumineux. **Devfs** a été créé pour simplifier cette utilisation, mais ce système a montré ses limites lorsqu'il y a des problèmes compliqués à résoudre.

**Udev** est le nouveau système pour gérer le répertoire **/dev**, conçu pour repousser les limites mises en avant par les précédentes versions de **/dev**, et fournir un lien robuste. Dans le but de créer et nommer les périphériques dans **/dev**, le système crée les "nodes", qui correspondent aux périphériques systèmes, grâce à **udev** qui fait le lien entre les informations données par **sysfs** et les règles données par l'utilisateur. Ce wiki a pour but d'expliquer comment écrire les règles **udev**.

**Sysfs** a été officialisé avec les noyaux de la série 2.6. Il est géré par le noyau pour exporter les informations basiques sur les périphériques actuellement connectés au système. **Udev** utilise ces informations pour créer les 'nodes' correspondant aux périphériques de votre ordinateur. **Sysfs** est monté sur **/sys** et vous pouvez le parcourir. Vous pouvez regarder ces fichiers avant de vous plonger dans **udev**. Dans ce wiki, j'utiliserai **/sys** et **sysfs**, qui signifient la même chose.

## NAISSANCE DE UDEV

**udev** a été créé pour répondre à des événements du type connexion à chaud (hotplug). Il y a beaucoup de documentations qui se réfèrent à la création des références de périphériques en réponse à l'apparition de nouveaux périphériques. Mais **udev** est plus général. Il peut exécuter des commandes arbitraires de l'espace utilisateur en réponse à l'apparition de nouveaux périphériques ou n'importe quel événement transmis par le noyau.

Les règles **udev** sont flexibles et très puissantes. Voici quelques exemples de ce que vous pouvez faire :

- changer le nom assigné par défaut à un périphérique ;
- donner un nom alternatif/persistant à un périphérique en créant un lien symbolique ;
- nommer un périphérique en fonction de la sortie d'un programme ;
- changer les permissions et les propriétés d'un périphérique ;
- lancer un script quand un périphérique est créé ou supprimé ( en général pour un périphérique qui se branche à chaud, comme l'USB ) ;
- renommer les interfaces réseaux

L'écriture de règles n'est pas une solution s'il n'existe pas du tout de périphérique "node" pour votre périphérique particulier. S'il n'y a pas de règle, **udev** va créer le périphérique "node" avec le nom donné par défaut par le noyau.

Il y a plusieurs avantages à avoir un nom de périphérique constant. Supposons que vous ayez deux périphériques USB : une webcam et une clé USB. Ces périphériques sont normalement assignés aux périphériques "nodes" **/dev/sda** et **/dev/sdb**, mais cela dépend de l'ordre dans lequel ils ont été connectés. C'est pourquoi il est plus pratique de nommer les périphériques à chaque fois de la même manière, par exemple **/dev/camera** et **/dev/flashdisk**.

## 29.3 PRÉSENTATION

### INTRODUCTION

Le système **udev** est composé de quelques services du noyau et du démon **udevd**. Le noyau avertit le démon **udevd** que tel ou tel événement est arrivé. Le démon **udevd** est configuré pour répondre aux événements par des actions correspondantes. L'information sur l'événement vient du noyau, les actions se produisent dans l'espace utilisateur. Les réponses aux événements sont configurables dans des "règles".

Le démon **udevd**, comme les autres démons, est lancé au démarrage grâce à un script init : `/etc/init.d/udev`.

Son fichier de configuration est `/etc/udev/udev.conf`. Les fichiers de règles (qui ne sont rien d'autre qu'une configuration supplémentaire d'`udevd`) sont dans `/etc/udev/rules.d`, par défaut seul le fichier `70-persistent-net.rules` est présent. Tous les fichiers de ce répertoire, dont les noms se terminent par `".rules"`, sont analysés par ordre alphabétique.

Quand le fichier de configuration ou les fichiers de règles sont modifiés, le démon **udevd** doit être redémarré (ou, comme on le dit plus bas dans cette page, on peut se servir du programme **udevadm**).

La plupart des fichiers `/etc/udev/rules.d` sont des liens vers des fichiers situés ailleurs. L'auteur suppose que c'est ainsi pour que lorsque les fichiers de règles sont modifiés, les fichiers de sauvegarde de l'éditeur ne restent pas dans l'espace des règles où ils pourraient être utilisés au démarrage suivant du démon. Aussi, dans la mesure où les liens peuvent avoir des noms différents des fichiers originaux, ils peuvent être ordonnés sans se soucier du nom qu'ils portent (comme c'est le cas des scripts init).

Les moments où **udevd** est actif sont :

- Au démarrage, il analyse les fichiers de configuration et les fichiers de règles et construit une base de données des règles en mémoire.
- Quand un événement se produit, il vérifie sa base de données et effectue les actions appropriées.

### UDEV CONCRÈTEMENT

Bon c'est bien cette théorie, l'explication du rôle de la gestion dynamique et des règles mais concrètement comment ceci se matérialise ? Exécutons la commande `ls` dans le répertoire `/dev` pour lister les fichiers de ce répertoire, croyez-moi avant la présence de **udev** c'était plutôt pénible ! Mais regardons plutôt le résultat de simplification que le système **udev** nous offre grâce aux règles déjà présentes.

J'ai fait mention que **udev** pouvait réaliser des liens symboliques afin de faciliter l'accès pour l'être humain. Voici un lien avec le CD-ROM.

**udev** et lien symbolique du CD-ROM ; affichage de /dev/cdrom :

```
$ ls -l /dev/cdrom
```

```
lrwxrwxrwx 1 root root 3 Feb 3 12 :29 /dev/cdrom -> sr0
```

Affiche le périphérique "réel" :

```
$ ls -l /dev/sr0 brw-rw---+ 1 root cdrom 11, 0 Feb 3 12 :29 /dev/sr0
```

Nous pouvons constater dans cette exemple que le « block » device **/dev/sr0** à les permissions 660 et que le « group » propriétaire est cdrom. Ceci a été modifié lors de sa création par **udev**. Ceci nous permet de limiter l'accès au cdrom ( lire, écrire sur un cd ). De plus afin de faciliter la communication avec ce cdrom, **udev** crée un lien symbolique **/dev/cdrom** vers **/dev/sr0**. Ceci permet au programme ou à l'utilisateur de toujours voir le même fichier **/dev/cdrom** et c'est le lien qui change vers le vrai périphérique.

Une autre utilisation du lien symbolique pour les périphériques concerne les disques dur.

**udev** gère le répertoire **/dev/disk** et va créer des répertoires. Ceci à l'avantage de pouvoir utiliser des noms de périphériques constants principalement pour les disques durs amovibles ou clef usb. Nous constatons que le nom attribué **/dev/sd[a-z]** peut varier selon le moment où nous branchons le périphérique. La première clef usb aura **/dev/sdb** et la deuxième clef **/dev/sdc**. Mais la prochaine fois, si je branche les clefs dans un ordre différent le nom sera différent. Afin de solutionner ce problème, **udev** nous permet d'utiliser plusieurs modes d'identifications : plusieurs noms pour les mêmes périphériques.

```
/dev/disk/by-id    $ ls -l by-id/
```

```
total 0
```

```
lrwxrwxrwx 1 root root 9 jan 31 11 :15 ata-SAMSUNG_HD161GJ_S1VCJ90S738645 -> ../../sda
lrwxrwxrwx 1 root root 10 jan 13 18 :48 ata-SAMSUNG_HD161GJ_S1VCJ90S738645-part1 -> ../../sda1
lrwxrwxrwx 1 root root 10 jan 13 18 :48 ata-SAMSUNG_HD161GJ_S1VCJ90S738645-part2 -> ../../sda2
lrwxrwxrwx 1 root root 10 jan 13 18 :48 ata-SAMSUNG_HD161GJ_S1VCJ90S738645-part5 -> ../../sda5
lrwxrwxrwx 1 root root 9 jan 13 18 :48 ata-TSS Tcorp_DVD+-RW_TS-H653G_R4576G_AS71306300 -> ../../sr0
lrwxrwxrwx 1 root root 9 jan 31 11 :15 scsi-SATA_SAMSUNG_HD161GJ_S1VCJ90S738645 -> ../../sda
lrwxrwxrwx 1 root root 10 jan 13 18 :48 scsi-SATA_SAMSUNG_HD161GJ_S1VCJ90S738645-part1 -> ../../sda1
lrwxrwxrwx 1 root root 10 jan 13 18 :48 scsi-SATA_SAMSUNG_HD161GJ_S1VCJ90S738645-part2 -> ../../sda2
lrwxrwxrwx 1 root root 10 jan 13 18 :48 scsi-SATA_SAMSUNG_HD161GJ_S1VCJ90S738645-part5 -> ../../sda5
lrwxrwxrwx 1 root root 9 fév 10 12 :25 usb-Kingston_DataTraveler_G3_00142238268CEB31F51D00B7-0 :0 -> ../../sdb
lrwxrwxrwx 1 root root 10 fév 10 12 :25 usb-Kingston_DataTraveler_G3_00142238268CEB31F51D00B7-0 :0-part1 -> ../../sdb1
lrwxrwxrwx 1 root root 9 jan 31 11 :15 wwn-0x50024e9200beb099 -> ../../sda
lrwxrwxrwx 1 root root 10 jan 13 18 :48 wwn-0x50024e9200beb099-part1 -> ../../sda1
lrwxrwxrwx 1 root root 10 jan 13 18 :48 wwn-0x50024e9200beb099-part2 -> ../../sda2
lrwxrwxrwx 1 root root 10 jan 13 18 :48 wwn-0x50024e9200beb099-part5 -> ../../sda5
```

Dans l'exemple ci-dessus nous pouvons constater que l'**id** est constitué du type de device ATA, SCSI, USB. Nous constatons aussi que le disque dur **/dev/sda** est représenté plusieurs fois sous différentes formes, ceci correspond au différents modes d'interprétations du disque dur :

- ata-SAMSUNG\_HD161GJ\_S1VCJ90S738645 -> ../../sda
- scsi-SATA\_SAMSUNG\_HD161GJ\_S1VCJ90S738645 -> ../../sda
- wwn-0x50024e9200beb099 -> ../../sda

Un world wide name (**wwn**) (ou world wide identifier (wwid)) est un identifiant unique dans un réseau SAN de type Fibre Channel ou Serial Attached SCSI, équivalent de l'adresse MAC d'une carte réseau « classique ».

Chaque WWN est un nombre codé sur 8 octets dont les 3 premiers sont attribués par l'IEEE et les autres par le constructeur de l'équipement (HBA ou autre).

Il existe deux formats de wwn définis par l'IEEE :

format original : adresses attribuées par le comité IEEE, et intégrées à l'équipement lors de sa fabrication, comme pour l'Adresse MAC Ethernet. Les 2 premiers octets sont soit 10 :00 (hexa) soit 2x :xx (code spécifié par le constructeur) suivis de 3 octets d'identification constructeur, et 3 octets pour le numéro de série de l'équipement ;

nouveau schéma : le début du WWN est 5 ou 6 (hexa) suivi par un code constructeur sur 3 octets, le reste étant un numéro de série attribué par le constructeur.

#### **/dev/disk/by-label :**

- \$ ls -l by-label/
  - lrwxrwxrwx 1 root root 10 fév 10 12 :25 KINGSTON -> ../../sdb1

Constatez que dans le répertoire **by-label** il n'y a qu'un disque dur vers la partition **/dev/sdb1**. Le Label fait référence au nom assigné à la partition lors de son formatage. Dans le cas présent AUCUNE partition sur le disque **/dev/sda** n'a été écrite avec un nom "Label", résultat le système ne les liste pas alors que ces partitions existent. Dans ce répertoire ne sera listées que des partitions car un disque dur n'a PAS de label. Ceci est uniquement sur le « filesystem ». Le label sera lu, peut importe le type de filesystem utilisé NTFS, ext3 , ... Ce n'est pas restreint au filesystem Unix.

#### **/dev/disk/by-path :**

- \$ ls -l by-path/
  - lrwxrwxrwx 1 root root 9 fév 10 12 :25 pci-0000 :00 :1d.7-usb-0 :2 :1.0-scsi-0 :0 :0 :0 -> ../../sdb
  - lrwxrwxrwx 1 root root 10 fév 10 12 :25 pci-0000 :00 :1d.7-usb-0 :2 :1.0-scsi-0 :0 :0 :0-part1 -> ../../sdb1
  - lrwxrwxrwx 1 root root 9 jan 31 11 :15 pci-0000 :00 :1f.2-scsi-0 :0 :0 :0 -> ../../sda
  - lrwxrwxrwx 1 root root 10 jan 13 18 :48 pci-0000 :00 :1f.2-scsi-0 :0 :0 :0-part1 -> ../../sda1
  - lrwxrwxrwx 1 root root 10 jan 13 18 :48 pci-0000 :00 :1f.2-scsi-0 :0 :0 :0-part2 -> ../../sda2
  - lrwxrwxrwx 1 root root 10 jan 13 18 :48 pci-0000 :00 :1f.2-scsi-0 :0 :0 :0-part5 -> ../../sda5
  - lrwxrwxrwx 1 root root 9 jan 13 18 :48 pci-0000 :00 :1f.2-scsi-1 :0 :0 :0 -> ../../sr0

Dans ce répertoire nous trouvons la référence du disque selon son chemin système. Personnellement je l'utilise moins. Cependant ceci peut nous indiquer dans quel port USB se trouve la clef USB par exemple, nous assurer que l'on communique bien avec le premier disque dur branché sur la nappe.

**/dev/disk/by-uuid :**

```
— $ ls -l by-uuid/
— lrwxrwxrwx 1 root root 10 jan 13 18 :48 0af1443a-ad46-4373-8534-ef1f7ee564d0 ->
  ..../sda1
— lrwxrwxrwx 1 root root 10 jan 13 18 :48 88f0c393-0aa5-4462-9c79-6bd33db98705 ->
  ..../sda5
— lrwxrwxrwx 1 root root 10 fév 10 12 :25 E79B-6B7C -> ..../sdb1
```

Encore une fois ici nous ne voyons **que** les partitions car le **UUID** est associé à une partition lors de sa création. **UUID** est une suite plus ou moins longue de caractères alpha-numériques qui permet d'identifier de façon absolument sûre chaque périphérique de stockage et partition. Le chiffre de l'**UUID** est calculé automatiquement au moyen d'un algorithme intégrant notamment certaines données de l'ordinateur hôte, au moment de la création ou du formatage de la partition ou de la table des partitions. Ce mode de calcul ne présente aucun risque de sécurité crédible.

La présentation ci-dessus n'est vraiment que la pointe de l'iceberg, car **udev** réalise énormément d'opérations en arrière plan. Ceci va nous permettre d'introduire le concept de règles de **udev**.

## 29.4 LES RÈGLES (RULES) UDEV

Pour décider comment nommer un périphérique et quelles actions faire, **udev** utilise une série de fichiers de règles. Ces fichiers se trouvent dans le répertoire **/etc/udev/rules.d**, et doivent tous avoir l'extension **.rules**. Les règles **udev** créées par défaut sont dans le fichier **/lib/udev/rules.d/50-udev-default.rules**. Il pourrait être intéressant d'y jeter un oeil, il contient quelques exemples. Certaines règles contiennent un exemple de sortie de **devfs** que vous trouverez dans **/dev** par défaut. Cependant, il est conseillé de ne pas écrire de règle directement dedans.

Les fichiers de **/lib/udev/rules.d** sont triés par ordre alphabétique, et dans certaines circonstances, l'ordre dans lequel ils sont analysés est important. En général, vous voulez que vos propres règles soient prises en compte avant les règles créées par défaut. Donc créez votre fichier comme ceci : **/etc/udev/rules.d/10-local.rules** (le nombre 10 influe sur l'ordre de prise en compte) et écrivez vos propres règles dans ce fichier.

Dans un fichier de règles, les lignes commençant par **"#"** sont traitées comme des commentaires. Toutes les autres lignes sont donc considérées comme des règles. Les règles s'écrivent sur une seule ligne ( on ne les coupe pas par un passage à la ligne avec ENTRÉE ).

Un périphérique peut être contrôlé par plusieurs règles. Ceci peut être avantageux lorsque par exemple, nous écrivons deux règles pour un périphérique, qui donnent un nom différent pour le même périphérique. Les deux règles seront appliquées même si ces règles sont dans des fichiers séparés. Il est important de comprendre que **udev** ne s'interrompt pas quand il trouve une règle, il continue sa recherche et tente d'appliquer chaque règle trouvée.

Chaque règle est faite d'un ensemble de clefs de correspondances et de clefs d'assignation, séparées par des virgules. Les clefs de correspondances sont les conditions utilisées pour identifier le périphérique sur lequel la règle agit. Quand toute la série de ces clefs de correspondance correspond bien au périphérique, alors la règle est appliquée et les actions des clefs d'assignation sont appliquées. Chaque règle doit se composer d'au moins une clef de correspondance et d'une clef d'assignation.

## LES RÈGLES BASIQUES

Dans une règle, **Udev** peut utiliser plusieurs clefs pour identifier un périphérique de manière très précise. Les clefs les plus communes sont présentées ci-dessous, les autres seront traitées plus loin. Pour la liste complète, consultez l'aide de **udev** avec la commande **man udev**.

- KERNEL – le nom du périphérique donné par le noyau ;
- SUBSYSTEM - le nom du sous système contenant le périphérique ;
- DRIVER - le nom du pilote du périphérique.

Après avoir utilisé une série de clefs pour définir précisément le périphérique, **udev** vous donne le contrôle, grâce aux clefs d'assignation. Pour la liste complète de ces clefs, consultez l'aide de **udev** avec la commande **man udev**. Les clefs d'assignation les plus fréquentes se trouvent ci-dessous, les autres seront traités plus loin.

- NAME – nom du périphérique "node" ;
- SYMLINK - liste des liens symboliques, ceux-ci étant les noms alternatifs pour le périphérique.

Voici un exemple de règles simples. Nous allons créer un lien symbolique afin que ce lien symbolique nommé **/dev/Premier\_disque** pointe vers **/dev/sda** .

Création du fichier de règles :

```
$ sudo vim /etc/udev/rules.d/99-custom.rules
KERNEL=="sda", SYMLINK+="Premier_disque"
```

Udev ne va pas relire la configuration tout de suite donc pour le moment la règle n'est pas active.

```
$ ls -l /dev/Premier_disque
ls : cannot access /dev/Premier_disque : No such file or director
```

Nous allons donc indiquer à udev de relire les fichiers :

```
$ sudo udevadm trigger
```

Validation du fonctionnement :

```
$ ls -l /dev/Premier_disque
lrwxrwxrwx 1 root root 3 Feb 17 12:41 /dev/Premier_disque -> sda
```

Il est possible de visualiser les événements **udev** comme présentés ci-dessous, très pratique bien entendu quand on « débug ».

AFFICHE LES ÉVÉNEMENTS UDEV :

```
$ udevadm monitor --udev
```

```
pat@amd-a10:~$ udevadm monitor --udev
monitor will print the received events for:
UDEV - the event which udev sends out after rule processing

UDEV [1638.189631] add      /devices/pci0000:00/0000:00:12.2/usb1/1-1 (usb)
UDEV [1638.190697] add      /devices/pci0000:00/0000:00:12.2/usb1/1-1/1-1:1.0 (usb)
UDEV [1638.191585] add      /devices/pci0000:00/0000:00:12.2/usb1/1-1/1-1:1.0/host9 (scsi)
UDEV [1638.192343] add      /devices/pci0000:00/0000:00:12.2/usb1/1-1/1-1:1.0/host9/scsi_host/host9 (scsi_host)
UDEV [1639.175824] add      /devices/pci0000:00/0000:00:12.2/usb1/1-1/1-1:1.0/host9/target9:0:0 (scsi)
UDEV [1639.176702] add      /devices/pci0000:00/0000:00:12.2/usb1/1-1/1-1:1.0/host9/target9:0:0/9:0:0:0 (scsi)
UDEV [1639.177724] add      /devices/pci0000:00/0000:00:12.2/usb1/1-1/1-1:1.0/host9/target9:0:0/9:0:0:0/scsi_disk/9:0:0:0 (scsi_disk)
UDEV [1639.178437] add      /devices/pci0000:00/0000:00:12.2/usb1/1-1/1-1:1.0/host9/target9:0:0/9:0:0:0/scsi_device/9:0:0:0 (scsi_device)
UDEV [1639.179182] add      /devices/pci0000:00/0000:00:12.2/usb1/1-1/1-1:1.0/host9/target9:0:0/9:0:0:0/bsg/9:0:0:0 (bsg)
UDEV [1640.179330] add      /devices/pci0000:00/0000:00:12.2/usb1/1-1/1-1:1.0/host9/target9:0:0/9:0:0:0/scsi_generic/sg3 (scsi_generic)
UDEV [1640.095395] add      /devices/virtual/bdi/8:32 (bdi)
UDEV [1641.072879] add      /devices/pci0000:00/0000:00:12.2/usb1/1-1/1-1:1.0/host9/target9:0:0/9:0:0:0/block/sdc (block)
UDEV [1641.145522] add      /devices/pci0000:00/0000:00:12.2/usb1/1-1/1-1:1.0/host9/target9:0:0/9:0:0:0/block/sdc/sdcl (block)
```

La première ligne indique le device concerné : **/devices/pci0000 :00/0000 :00 :12.2/usb1/1-1**

L'avant dernière ligne indique le nom du block concerné : **/block/sdc (block)**.

Il est possible de collecter de l'information sur le device avec la commande **udevadm**, il a plusieurs moyens de lui fournir le périphérique que l'on désire interroger.

Voici deux exemples : information udev pour USB device.

1 ) \$ sudo udevadm info -a -p **/devices/pci0000 :00/0000 :00 :12.2/usb1/1-1**

```
pat@amd-a10:~$ sudo udevadm info -a -p /devices/pci0000:00/0000:00:12.2/usb1/1-1
```

```
Udevadm info starts with the device specified by the devpath and then
walks up the chain of parent devices. It prints for every device
found, all possible attributes in the udev rules key format.
A rule to match, can be composed by the attributes of the device
and the attributes from one single parent device.
```

```
looking at device '/devices/pci0000:00/0000:00:12.2/usb1/1-1':
KERNEL=="1-1"
SUBSYSTEM=="usb"
DRIVER=="usb"
ATTR{bDeviceSubClass}=="00"
ATTR{bDeviceProtocol}=="00"
ATTR{devpath}=="1"
ATTR{idVendor}=="13fe"
ATTR{speed}=="480"
ATTR{bNumInterfaces}=="1"
ATTR{bConfigurationValue}=="1"
ATTR{bMaxPacketSize0}=="64"
ATTR{busnum}=="1"
ATTR{devnum}=="7"
ATTR{configuration}==""
ATTR{bMaxPower}=="200mA"
ATTR{authorized}=="1"
ATTR{bmAttributes}=="80"
ATTR{bNumConfigurations}=="1"
ATTR{maxchild}=="0"
ATTR{bcdDevice}=="0100"
ATTR{avoid_reset_quirk}=="0"
ATTR{quirks}=="0x0"
ATTR{serial}=="070156DD97FB5C65"
ATTR{version}=="2.00"
```

[[output coupé]]

2 ) autre méthode pour collecter l'information

```
$ sudo udevadm info -a -p /sys/block/sdc
```

```
looking at device '/devices/pci0000:00/0000:00:12.2/usb1/1-1/1-1:1.0/host9/target9:0:0/9:0:0:0/block/sdc':
KERNEL=="sdc"
SUBSYSTEM=="block"
DRIVER==""
ATTR{ro}=="0"
ATTR{size}=="62685184"
ATTR{stat}=="367    15282    17210    2372      1      0      1      1156      0      2820    3528"
ATTR{range}=="16"
ATTR{discard_alignment}=="0"
ATTR{events}=="media_change"
ATTR{ext_range}=="256"
ATTR{events_poll_msecs}=="2000"
ATTR{alignment_offset}=="0"
ATTR{inflight}=="0      0"
ATTR{removable}=="1"
ATTR{capability}=="51"
ATTR{events_async}=="
```

À l'aide de ces informations nous pourrons réaliser nos propres règles. Par exemple pour que le système exécute un script de backup. Voici un exemple de définition d'un script de backup.

## SCRIPT DE BACKUP

Suite à l'information collectée par la commande **udevinfo** nous sommes en mesure d'identifier convenablement le périphérique. Dans le répertoire **/etc/udev/rules.d/** il est possible de définir nos règles personnalisées pour notre système. Nous réaliserons donc le fichier **99-backup.rules**. Dans ce dernier nous aurons donc l'instruction à réaliser lors de la détection du périphérique.

```
/etc/udev/rules.d/99-backup.rules
KERNEL=="sd ?1", SUBSYSTEM=="block", ATTRS{serial}=="070156DD97FB5C65",
SYMLINK+="backup", RUN+=" /usr/local/bin/backup.sh"
```

Donc quand le kernel va détecter le périphérique, que ce dernier soit associé au kernel par `/dev/sd ?1` (donc sda1, ou sdb1 ou sdc1 ou ...), que ceci est un device de type block, que le numéro de série est 070156DD97FB5C65, et que si l'ensemble de ces conditions sont respectées, le système réalisera le lien symbolique `/dev/backup` vers `/dev/sd ?1` et exécutera le script `/usr/local/bin/backup.sh`.

Voici le contenu du script /usr/local/bin/backup

```
#!/bin/bash
# les variables
REP_2_BACKUP=/mnt/backup
DEVICE_DISK=/dev/backup
REP_2_BACKUP="/etc /var/log /home"
DATE=$(date +%F)

# monte le périphérique : mount /dev/device /répertoire
mount $DEVICE_DISK $REP_2_BACKUP

# réalisation du backup
for rep in $REP_2_BACKUP
do
    rep_clean=$(basename $rep)
    tar -zcf $REP_2_BACKUP/$rep_clean-$DATE.tar.gz $rep
done

# démonte le périphérique
umount $DEVICE_DISK $REP_2_BACKUP
```

Si vous désirez afficher des messages via une fenêtre graphique je vous invite à regarder la commande zenity. Je ne couvrirai malheureusement pas cette matière ici.

## 29.5 VIDÉOS

Télécharger [http://x3rus.com/moodle/pluginfile.php/139/mod\\_label/intro/udev.webm](http://x3rus.com/moodle/pluginfile.php/139/mod_label/intro/udev.webm) et lire avec VLC.