Table des matières

1 Présentation de GNU/Linux

1.1 Qu'est-ce que le mouvement GNU ?

En 1984, **Richard Matthew Stallman**, chercheur en informatique du MIT[1] quitte son poste et se consacre à l'écriture d'un système d'exploitation Libre du nom de **GNU**[2].

Il annonce l'année suivante la création de la **FSF**[3] afin de supporter ce projet. C'est durant ces années qu'il écrit ce qui deviendra les préceptes du Logiciel Libre.

La concrétisation en est la publication en 1989 de la première version de la **licence GPL**[4] qui sera alors le fondement éthique, juridique et politique du mouvement du Libre.

1.2 Qu'est-ce qu'un logiciel libre ?

L'expression **Logiciel Libre** fait référence à la liberté pour les utilisateurs d'exécuter, de copier, de distribuer, d'étudier, de modifier et d'améliorer le logiciel. Plus précisément, elle fait référence à quatre types de liberté pour l'utilisateur du logiciel

- **Liberté 0** : La liberté d'exécuter le programme, pour tous les usages.
- **Liberté 1** : La liberté d'étudier le fonctionnement du programme, et de l'adapter à vos besoins.
- **Liberté 2** : La liberté de redistribuer des copies, donc d'aider votre voisin.
- **Liberté 3** : La liberté d'améliorer le programme et de publier vos améliorations, pour en faire profiter toute la communauté.

> Pour répondre aux libertés 1 et 3, l'accès au code source est une condition requise.

Un programme est un Logiciel Libre si les utilisateurs ont **toutes** ces libertés.

1.3 Linux

Linux est le nom couramment donné à tout **système d'exploitation libre** fonctionnant avec le noyau Linux. C'est une implémentation libre du système **UNIX**[5] respectant les spécifications POSIX[6]. Ce système est né de la rencontre entre le mouvement du logiciel libre et le modèle de développement collaboratif et décentralisé via Internet. Son nom vient du créateur du noyau Linux, **Linus Torvalds**.

Le système avec toutes ses applications est distribué sous la forme de distributions Linux comme *Ubuntu*, *Debian* ou *CentOS*.

[1] Le Massachusetts Institute of Technology ou MIT, en français Institut de technologie du Massachusetts, est une institution de recherche et une université américaine, spécialisée dans les domaines de la science et de la technologie.
[2] Son nom est un acronyme récursif qui signifie en anglais « GNU's Not UNIX » (littéralement, « GNU n'est pas UNIX »).
[3] Free Software Foundation
[4] La licence publique générale GNU
[5] Unix est un système d'exploitation multitâche et multi-utilisateur créé en 1969. Il est fondé sur une approche par laquelle il offre de nombreux petits outils, chacun doté d'une mission spécifique.
[6] POSIX est une famille de standards techniques définie depuis 1988 par l'Institute of Electrical and Electronics Engineers (IEEE), et formellement désignée par IEEE 1003. Ces standards ont émergé d'un projet de standardisation des interfaces de programmation des logiciels destinés à fonctionner sur les variantes du système d'exploitation UNIX.

1.4 Philosophie d'Unix

Douglas McIlroy, l'inventeur des tuyaux Unix (Unix pipes en anglais) et l'un des fondateurs de la tradition d'Unix, résume la philosophie comme suit :

Voici la philosophie d'Unix :

- Écrivez des programmes qui effectuent une seule chose et qui le font bien.
- Écrivez des programmes qui collaborent.
- Écrivez des programmes pour gérer des flux de texte [en pratique des flux d'octets], car c'est une interface universelle.

> Ce qui est souvent résumé par : **Ne faire qu'une seule chose, et la faire bien.**

1.5 Historique et genèse de Linux

Linus B.Torvalds est à l'origine de ce système d'exploitation entièrement libre. Au début des années 90, il voulait mettre au point son propre système d'exploitation pour son projet de fin d'études. Linus Torvalds avait pour intention de développer une version d'UNIX pouvant être utilisé sur une architecture de type 80386.

Le premier clone d'UNIX fonctionnant sur PC a été *Minix*, écrit par Andrew Tanenbaum, un système d'exploitation minimal pouvant être utilisé sur PC.

Linus Torvalds décida donc d'étendre les possibilités de Minix, en créant ce qui allait devenir Linux. Amusées par cette initiative, de nombreuses personnes ont contribué à aider Linus Torvalds à réaliser ce système, si bien qu'**en 1991 une première version du système a vu le jour**. C'est en mars 1992 qu'a été diffusée la première version ne comportant quasiment aucun bug.

1.6 Noyau Linux

Le noyau Linux est un noyau de système d'exploitation de type UNIX. Le noyau Linux est un logiciel libre développé essentiellement en langage C par des centaines de bénévoles et salariés communiquant par Internet.

Ses caractéristiques principales sont d'être **multitâche et multi-utilisateur**. Il respecte les normes *POSIX* ce qui en fait un digne héritier des systèmes *UNIX*.

Si au début de son histoire le développement du noyau Linux était assuré par des développeurs bénévoles, les principaux contributeurs sont aujourd'hui un ensemble d'entreprises, souvent concurrentes, comme RedHat, Novell, IBM ou Intel.

La licence du noyau Linux est la licence publique générale GNU dans sa version 2.

1.7 Définition

GNU/Linux est un système d'exploitation complètement Libre et performant. Il est hautement configurable. Il ne dépend pas d'une multinationale. Il est supporté par une grande communauté d'utilisateurs souvent prêts à vous aider.

Quelque soit votre domaine de compétence, vous pouvez participer à l'amélioration de *GNU/Linux* pour que ce dernier évolue dans votre intérêt.

Ce n'est pas un simple logiciel gratuit, mais un Logiciel Libre. Ce qui garantit qu'il restera accessible et gratuit pour tous, sans discrimination.

1.8 Qu'est-ce qu'une distribution ?

En réalité, si on vous livrait le noyau Linux seul, accompagné des outils GNU de base, vous seriez bien avancé : pas d'interface graphique, juste quelques commandes, bref, votre système d'exploitation serait inexploitable.

C'est pour cela qu'existe des distributions Linux qui contiennent le noyau Linux, les outils GNU, plus un ensemble de logiciels qu'elles ont choisi de supporter. Ceux-ci sont testés et compilés pour vous. La plupart d'entre elles contiennent un système d'installation de logiciel simplifié qui leur est propre.

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, Mageia, Red Hat/Fedora, Slackware, SUSE Linux Enterprise/openSUSE, Ubuntu). Le mainteneur de la distribution peut être une entreprise (comme dans le cas de Fedora, RedHat et Ubuntu, Canonical) ou une communauté (comme Debian, Gentoo ou Slackware).

1.9 Les caractéristiques du système

Linux est un système d'exploitation proche des systèmes UNIX pouvant être exécuté sur différentes plates-formes matérielles : x86 (c'est-à-dire des plates-formes à base de processeurs Intel, AMD, etc.), Sparc, PowerPC, Alpha, ARM, etc. Ainsi le système Linux peut fonctionner aussi bien sur des ordinateurs personnels que des consoles de jeu ou des assistants personnels !

Linux est ainsi un système **multi plate-forme**. Il est également **multi-utilisateurs** (plusieurs personnes peuvent en même temps travailler sur le même ordinateur), mais aussi **multi-tâches** (plusieurs applications peuvent être lancées en même temps sans qu'aucune n'affecte les autres) et **multi-processeurs**.

> Linux est considéré comme un système fiable, robuste et puissant. Il est d'ailleurs capable de fonctionner avec très peu de ressources sur des ordinateurs bas de gamme très peu puissants.

2 Installation

2.1 Debian ? Qu'est-ce que c'est ?

Debian est une des plus anciennes distributions de Linux et fut créé en **1993**. Elle utilise Linux en tant que noyau de manière similaire aux autres distributions.

Debian est aujourd'hui la seule distribution majeure non commerciale.

Debian est une organisation à but non lucratif constituée d'un millier de développeurs bénévoles répartis sur toute la planète. Elle est dirigée par un projet leader élu par les développeurs. Les décisions se prennent au consensus ou par vote.

Les autres distributions GNU/Linux majeures sont des sociétés commerciales, ce qui ne les empêche pas de produire des logiciels libres !

Debian se distingue aussi par son attachement très fort à la philosophie du logiciel libre.

2.2 La stabilité

Debian GNU/Linux est réputé pour être un système d'exploitation très stable. Avant chaque nouvelle version, le système est longuement testé et il ne sort qu'une fois que tous les bogues connus ont été corrigés. Debian s'est doté d'un **Bug Tracking System (BTS)** très performant et très pratique qui permet aux développeurs d'avoir un retour d'expérience instructif des utilisateurs, ce qui les aide à corriger les bogues rapidement.

2.3 Les architectures

Debian GNU/Linux est disponible sous 12 architectures, dont Intel 32 et 64 bits, PowerPC (les anciens Macintosh) et Sparc (les stations Sun).

2.4 Les différentes versions de Debian

La première version de Debian, la 0.01 est sortie en 1993. Puis les versions s'enchaînent, avec des noms inspirés du film Toy Story.

Debian est toujours disponible en trois versions (trois branches) qui sont :

1. **stable** : version figée où les seules mises à jour sont des correctifs de sécurité.
2. **testing** : future version stable où seuls les paquets suffisamment matures peuvent rentrer.
3. **unstable** : il s'agit d'une version en constante évolution, alimentée sans fin par de nouveaux paquets ou de mises à jour de paquets déjà existants (on parle de rolling release).

2.5 Installation de Debian

Je vous propose de commencer par installer une version stable. Il faut savoir que vous pouvez passer facilement d'une version donnée à une version supérieure, mais l'inverse est plus difficile. Donc si vous installez une stable, vous pourrez passer facilement en testing ou en Sid (unstable) ; mais vous ne pourrez que difficilement revenir en stable ensuite.

> **Remarque :** Veuillez remplacer la variable entre crochet par le contenu idoine. (ex. *[Module]*-debian)

2.6 Préparation de Vmware

1. Téléchargez sur votre disque dur l'image du CD d'installation stable de DEBIAN [7]
2. Utilisez VMware pour créer une nouvelle machine virtuelle

- File → New → Virtual Machine
- Virtual Machine Configuration → Custom (advanced)
- Hardware compatibility → Workstation *[dernière version]*
- Install operating system from → I will install the operating system later
- Guest Operating System → 2. Linux → Debian *[dernière version]* 64-bit
- Name → *[Module]*-Debian
- Number of processors → 1
- Memory → 512 MB
- Network Connection → Use network address translation (NAT)
- I/O Controller Types → LSI Logic (Recommended)
- Virtual Disk Type → SCSI (Recommended)
- Disk → Create a new virtual disk
- Disk Size → Maximum disk size (in GB) → 20.0
- Disk File → *[Module]*-Debian.vmdk
- Customize Hardware
  - Enlever → USB Controller
  - Enlever → Sound Card
  - Enlever → Printer
  - New CD/DVD (IDE) → Use ISO image → *[image ISO d'installation de DEBIAN]*

2.7 Boot sur le support

Configurez votre ordinateur pour qu'il démarre sur le support d'installation. Au moment du boot, vous avez accès à une ligne de commande permettant de lancer l'installation en sélectionnant **Install** et en appuyant sur **Entrée**.

[7] https://www.debian.org/releases/stable/

2.8 Choix des langues et pays

Trois écrans vous permettent de choisir :

- La langue utilisée par le processus d'installation. Naviguez avec les flèches, sélectionnez **Français** et appuyez sur **Entrée** pour continuer. Dans la suite, c'est le français qui sera utilisé.
- Selon la langue initiale choisie, Debian vous demande ensuite dans quel pays vous vous situez. C'est utile, car c'est ainsi que sont positionnées les variables locales : format de date, d'heure, encodage des caractères, formats numériques et monétaires, etc. Sélectionnez **Suisse** et appuyez sur **Entrée** pour continuer.
- Enfin, choisissez votre type de clavier. Pour la Suisse, c'est **Suisse romand**

2.9 Paramètres du réseau

Les trois étapes suivantes concernent les informations réseaux de base. Si l'installateur n'a pas réussi à configurer la carte réseau par DHCP, il vous demandera de saisir les informations de base :

- Adresse IP
- Masque de sous réseau
- Passerelle par défaut
- DNS

Puis vous devez saisir un nom d'hôte (le nom de la machine sur le réseau) et le nom du domaine.

- Saisissez le nom d'hôte **[Module]-debian** et appuyez sur **Entrée** pour continuer.
- Comme votre machine n'appartient à auccn domaine, laissez le champ **Domaine** vide et appuyez sur **Entrée** pour continuer.

2.10 Comptes root et utilisateurs

Vous devez maintenant saisir le mot de passe de l'administrateur root de la machine. Celui-ci vous sera demandé deux fois pour confirmation. Ne le perdez pas ! Il n'y a aucun moyen de retrouver le mot de passe d'origine. Debian vous prévient et peut même refuser un mot de passe s'il est trop simple.

Vous devez ensuite créer au moins un utilisateur simple. Vous devrez saisir le nom complet de la personne et Debian vous propose un login. Saisissez ensuite les mots de passe associés. Vous pouvez créer plusieurs utilisateurs, mais rien ne vous empêche de faire ceci après l'installation.

- Créez l'utilisateur **cpnv** avec comme mot de passe **cpnv**

2.11 Partitionner les disques

De manière simpliste, vous avez le choix entre trois principales méthodes pour partitionner vos disques

- Une méthode manuelle
- Une méthode assistée (voire automatique) en utilisant le partitionnement classique

- Une méthode assistée proposant le LVM[8]

La méthode assistée classique dans le cas d'une nouvelle installation donne des bons résultats. Si vous réinstallez une machine, ou que vous installez Debian sur une machine disposant déjà de partitions contenant les dossiers personnels par exemple, passez par un partitionnement personnalisé.

Le LVM consiste à regrouper des disques physiques ou partitions (appelés volumes physiques) en un seul grand espace (appelé groupe de volumes) dans lequel vous pouvez découper des espaces logiques à volonté (appelés volumes logiques), les agrandir, les réduire, etc.

> *Cependant vous devriez envisager la solution LVM dans le cadre d'un serveur d'entreprise ou si vous pensez rajouter à terme des disques dans votre machine pour rajouter de l'espace de stockage. Le LVM apporte une très grande souplesse.*

- Sélectionnez **Manuel** et appuyez sur **Entrée** pour continuer.
- Sélectionnez le disque **(sda)** et appuyez sur **Entrée** pour continuer.
- Fait-il créer une nouvelle table des partitions sur ce disque ? Sélectionnez **OUI** et appuyez sur **Entrée** pour continuer.

Linux utilise deux types de systèmes de fichiers :

- **Swap** qui sert de mémoire virtuelle, qui est utilisée quand la mémoire vive est pleine
- **Ext4** qui sert à stocker les fichiers et les répertoires (il existe de nombreuses alternatives à Ext4 : Ext3, Ext2, ReiserFS, xfs, jfs…).

2.12 Explication de l'arborescence des fichiers

Voici pour information quelques explications sur les différents dossiers indispensables dans / :

- **/bin** : Contient les programmes systèmes importants
- **/boot** : Les fichiers utiles au démarrage du système
- **/dev** : Contient des fichiers factices permettant de communiquer avec vos périphériques
- **/etc** : Ici se trouve la plupart des fichiers de configuration du système
- **/home** : Contient les dossiers personnels des utilisateurs. Chacun y possède un dossier à son nom avec ses fichiers personnels
- **/lib** : Contient les librairies –bibliothèques – utiles au système
- **/media** : Les dossiers contenus correspondent aux accès de montage des périphériques de stockage
- **/opt** : A un peu la même fonction que /usr, sauf que certains l'utilisent pour les programmes qu'ils compilent eux-même et qui ne sont logiquement pas aussi intégrés qu'un logiciel disponible dans les sources de mises à jour
- **/proc** : Ce dossier contient des fichiers et dossiers virtuels qui correspondent à l'état du système en temps réel : programmes (processus) lancés, occupation mémoire, RAM disponible, etc..
- **/root** : C'est le /home de l'administrateur ! Ce dernier est séparé pour des questions de sécurité.
- **/sbin** : A un peu la même fonction que /bin, sauf que tous les programmes issus ne sont accessibles qu'à l'administrateur.

[8] LVM (Logical Volume Manager, ou gestionnaire de volumes logiques en français)

- **/tmp** : Comme son nom l'indique, ici sont stockés les fichiers temporaires utiles aux programmes en cours d'exécution. Ce dossier est vidé à chaque redémarrage.
- **/usr** : Dossier important, contenant tous les programmes et les bibliothèques installés.
- **/var** : Dossier contenant tout ce qui est variable au système. Par exemple, les fameux fichiers « log » enregistrant ce qui se passe sur votre système, utiles quand quelque chose ne fonctionne plus par exemple – contenus dans /var/log/ .

2.13 Création de la partition SWAP

Traditionnellement, on crée une partition avec un système de fichiers de type Swap de taille double ou triple de la taille de la mémoire vive quand celle-ci est inférieure à 256 Mo ou égale à la taille de la mémoire vive quand celle-ci est supérieure ou égale à 256 Mo.

Cette partition est appelée partition de swap ou d'échange.

- Sélectionnez **Espace libre** et appuyez sur **Entrée** pour continuer.
- Sélectionnez **Créer une nouvelle partition** et appuyez sur **Entrée** pour continuer.
- Entrez la valeur **512M** et appuyez sur **Entrée** pour continuer.
- Sélectionnez **Primaire** et appuyez sur **Entrée** pour continuer.
- Sélectionnez **Début** et appuyez sur **Entrée** pour continuer.
- Sélectionnez **Utiliser comme** et appuyez sur **Entrée** pour continuer.
- Sélectionnez **espace d'échange (swap)** et appuyez sur **Entrée** pour continuer.
- Sélectionnez **Fin du paramétrage de cette partition** et appuyez sur **Entrée** pour continuer.

2.14 Création de la partition / (racine)

La partition racine / est la partition qui contiendra le système et tous ses composants (programmes, paramètres systèmes, etc.)

- Sélectionnez **Espace libre** et appuyez sur **Entrée** pour continuer.
- Sélectionnez **Créer une nouvelle partition** et appuyez sur **Entrée** pour continuer.
- Entrez la valeur **10 GB** et appuyez sur **Entrée** pour continuer.
- Sélectionnez **Primaire** et appuyez sur **Entrée** pour continuer.
- Sélectionnez **Début** et appuyez sur **Entrée** pour continuer.
- Sélectionnez **Utiliser comme** et appuyez sur **Entrée** pour continuer.
- Sélectionnez **système de fichiers journalisés ext4** et appuyez sur **Entrée** pour continuer.
- Contrôlez la valeur **Point de montage** soit égale à **/**
- Sélectionnez **Indicateur d'amorçage** et appuyez sur **Entrée**. La valeur va passer à présent.
- Sélectionnez **Fin du paramétrage de cette partition** et appuyez sur **Entrée** pour continuer.

2.15 Création de la partition /home

La partition /home est la partition qui va contenir les données des utilisateurs.

Créer une partition pour le répertoire des utilisateurs est la méthode la plus pertinente pour un poste de travail. Elle permet de réinstaller facilement un autre système (mise à jour ou réinstallation complète) sans casser les données personnelles. La nouvelle distribution ainsi installée pourra réutiliser la partition montée sur /home et ainsi récupérer les données.

- Sélectionnez **Espace libre** et appuyez sur **Entrée** pour continuer.
- Sélectionnez **Créer une nouvelle partition** et appuyez sur **Entrée** pour continuer.
- Laissez la valeur proposée et appuyez sur **Entrée** pour continuer.
- Sélectionnez **Primaire** et appuyez sur **Entrée** pour continuer.
- Sélectionnez **Utiliser comme** et appuyez sur **Entrée** pour continuer.
- Sélectionnez **système de fichiers journalisés ext4** et appuyez sur **Entrée** pour continuer.
- Contrôlez la valeur **Point de montage** soit égale à **/home**
- Sélectionnez **Fin du paramétrage de cette partition** et appuyez sur **Entrée** pour continuer.

Ne soyez pas surpris par les numéros de partitions. Très souvent la seule partition primaire est la racine / tandis que toutes les autres sont des partitions logiques au sein d'une partition étendue, et Linux numérote les primaires de 1 à 4 (une partition étendue est une partition primaire) tandis que la numérotation des partitions logiques débute à 5.

Si ce schéma de partitionnement vous convient, validez. L'écran suivant vous donne un récapitulatif que vous devez de nouveau valider.

Attention ! Le partitionnement est suivi de l'écriture des systèmes de fichiers sur les partitions concernées. Cette opération est est destructive.

- Sélectionnez **Terminer le partitionnement et appliquer les changements** et appuyez sur **Entrée** pour continuer.
- Faut-il appliquer les changements sur les disques ? Sélectionnez **Oui** et appuyez sur **Entrée** pour continuer.

Une fois les changements validés, une barre de progression vous informe de l'état du partitionnement et de l'écriture des nouveaux systèmes de fichiers. Debian va ensuite monter ces systèmes pour l'installation.

2.16 Installation

L'installation se déroule en plusieurs étapes

Debian procède tout d'abord à l'installation du système de base : c'est l'ensemble des logiciels communs à toute installation de Debian tels que la bibliothèque standard, les utilitaires GNU et les outils Debian indispensables. Cette étape ne nécessite aucune intervention de votre part et prend quelques minutes.

Installation des éléments de base. Soit téléchargés ou contenus sur le support d'installation, les éléments de base sont recopiés sur votre disque. Ils se composent des packages essentiels.

Vous choisissez ensuite un miroir : c'est le lieu (sur Internet) où sont présents les dépôts de logiciels Debian. Si vous n'utilisez pas de dépôts, Debian se base uniquement sur le contenu du support d'installation. S'il s'agit d'un DVD il n'y a pas de problèmes, mais si vous utilisez comme ici un CD d'installation réseau, seule la base est présente et rien d'autre.

- Faut-il analyser un autre CD ou DVD? Sélectionnez **Non** et appuyez sur **Entrée** pour continuer.
- Faut-il continuer sans miroir sur le réseau ? Sélectionnez **Oui** et appuyez sur **Entrée** pour continuer. Nous le configurons plus tard.

L'installateur charge les listes de paquets, puis vous demande si vous souhaitez participer aux statistiques d'utilisation des paquets.

> Souhaitez-vous participer à l'étude statistique ? Sélectionnez **Non** et appuyez sur **Entrée** pour continuer. Nous le configurons plus tard.

Sélection des logiciels : À l'aide de la **barre d'espace** désélectionnez. Appuyez sur **Entrée** pour continuer

Ensuite, il procède à l'installation de nombreux paquets de base. Vous n'avez rien à faire pendant le déroulement de cette étape, qui prend quelques bonnes minutes.

2.17 Fin d'installation et redémarrage

Pour préparer le premier démarrage sous Linux, il faut rendre votre nouveau système d'exploitation démarrable directement depuis le disque dur. Pour cela, le programme GRUB va être installé dans le MBR (master boot record) de votre disque dur. C'est ce programe qui va vous proposer de choisir un des multiples systèmes d'exploitation installés sur votre ordinateur (et par la suite il vous permettra aussi de choisir la version du noyau Linux avec laquelle vous allez démarrer votre système Debian).

C'est très simple ici, car c'est le seul système installé. Sachez que le chargeur de démarrage écrase celui précédemment installé, mais que vous pouvez parfaitement utiliser grub pour démarrer n'importe quel système y compris Windows.

> Installer le programme de démarrage GRUB ? Sélectionnez **Oui** et sélectionnez **/dev/sda**

Il n'y a plus qu'à rebooter.

Voici ce que les ingénieurs en électronique appellent le test de la fumée : démarrer un système pour la première fois.

Après une installation standard, le premier écran que vous verrez au démarrage du système est le menu du programme d'amorçage GRUB .

Le premier choix est votre nouveau système Debian.

2.18 Site miroir de Debian

Les différents paquets proposés à l'installation sont placés sur des serveurs appelés les **dépôts**, et c'est en définissant à quels dépôts votre système aura accès que vous délimiterez la diversité et les versions des programmes disponibles à l'installation sur votre Debian.

Cette liste de dépôts auquel votre système a accès est contenue dans un fichier essentiel à toute Debian : le fichier sources.list (/etc/apt/sources.list), ou dans le dossier /etc/apt/sources.list.d/ sous forme de plusieurs fichiers.

Debian est distribuée (copiée) sur des centaines de serveurs sur la Toile. L'utilisation d'un serveur proche devrait augmenter la vitesse de téléchargement et également réduire la charge de nos serveurs centraux et de la Toile tout entière.

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

Commentez toutes les lignes du fichier en rajoutant le caractère **#** au début de la ligne et rajouter la ligne suivant à la fin du fichier.

```
deb http://debian.ethz.ch/debian stable main contrib non-free
```

Exécutez la commande suivante afin de mettre à jour votre nouveau dépôt ainsi que votre distribution.

```
# apt-get update && apt-get upgrade
```

2.19 Gestion des paquets

Un paquet est un logiciel ou une partie d'un logiciel, qui a été préparé dans un format spécial, afin de faciliter sa recherche, la consultation d'informations à son sujet, ainsi que son installation et sa désinstallation.

Ce paquet prend la forme d'un fichier avec un nom particulier : nom-du-logiciel_numéro-de-version_nom-de-l'architecture.deb (par exemple le fichier apache_1.3.24_i386.deb contient la version 1.3.24 du programme Apache pour processeurs Intel). Ce fichier contient les binaires du programme ainsi qu'un certain nombre d'en-têtes.

Le système de gestion des paquets de Debian est très performant et très facile à utiliser. Grâce à lui, les logiciels s'installent, se retirent et peuvent être mis à jour très facilement.

APT est le *Advanced Package Tool* et fournit le programmme **apt-get**. Apt-get fournit un moyen simple pour installer des paquets depuis la ligne de commande. Apt-get travaille avec le nom du paquet et peut seulement installer les archives .deb depuis une source indiqué dans */etc/apt/sources.list*.

Les options les plus courantes de apt-get :

- Pour mettre à jour la liste des paquets connus par votre système

  ```
  # apt-get update
  ```

- Pour mettre à jour tous les paquets de votre système, sans installer de paquets supplémentaires ou en supprimer :

  ```
  # apt-get upgrade
  ```

- Pour installer le paquet lynx et toutes ses dépendances

  ```
  # apt-get install lynx
  ```

- Pour supprimer le paquet lynx de votre système

```
# apt-get remove lynx
```

- Pour supprimer le paquet lynx et ses fichiers de configuration de votre système

```
# apt-get --purge remove lynx
```

- Pour mettre à jour votre système entier, en permettant si nécessaire l'installation de paquets supplémentaires ou la suppression de paquets

```
# apt-get dist-upgrade
```

Notez que vous devez être authentifié en tant que root pour exécuter toutes commandes qui modifient le système de paquets.

Notez que **apt-get** installe par défaut les paquets recommandés et constitue le programme de référence pour la gestion des paquets en console, leur installation mais aussi la mise à jour du système.

La suite d'outils **apt** inclut aussi le programme **apt-cache** pour questionner les listes de paquet.

2.20 Installation du SSH

SSH signifie **Secure SHell**. C'est un protocole qui permet de faire des connexions sécurisées (i.e. chiffrées) entre un serveur et un client SSH.

Installer un serveur SSH permet aux utilisateurs d'accéder au système à distance, en rentrant leur login et leur mot de passe (ou avec un mécanisme de clefs). Cela signifie aussi qu'un pirate peut essayer d'avoir un compte sur le système (pour accéder à des fichiers sur le système ou pour utiliser le système comme une passerelle pour attaquer d'autres systèmes) en essayant plein de mots de passes différents pour un même login (il peut le faire de manière automatique en s'aidant d'un dictionnaire électronique). On appelle ça une attaque en force brute.

Il y a donc trois contraintes majeures pour garder un système sécurisé après avoir installé un serveur SSH :

- avoir un serveur SSH à jour au niveau de la sécurité, ce qui doit être le cas si vous faites consciencieusement les mises à jour de sécurité en suivant la procédure Debian
- que les mots de passes de TOUS les utilisateurs soient suffisamment complexes pour résister à une attaque en force brute
- surveiller les connexions en lisant régulièrement le fichier de log */var/log/auth.log*

Pour pouvoir vous connecter à distance, vous pouvez maintenant installer le serveur SSH :

```
# apt-get install openssh-server
```

L'installation comporte une étape de génération des clefs de chiffrement. Finalement, le serveur SSH se lance.

Notez l'adresse IP de votre serveur à l'aide de la commande

```
# ip addr
```

Vous pouvez à présent utiliser un client SSH depuis votre poste de travail (Ex. PuTTY[9], Cygwin[10]) pour accéder à votre serveur et ouvrir une session à distance.

[9] http://www.putty.org/
[10] https://cygwin.com/

3 Mode console

3.1 Notions de base

Une fois que la procédure d'installation est terminée, vous arrivez à l'invite de connexion :

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

Pour vous connecter, vous avez le choix entre :

a. Vous connecter en tant que root

Tapez **root**, appuyez sur **Entrée**, ensuite tapez le mot de passe root que vous avez défini pendant la procédure d'installation et appuyez sur **Entrée**. Quand vous êtes ainsi connecté en tant que root, vous avez *tous les droits sur le système*

```
    debian:~#
```

b. Vous connecter en tant que simple utilisateur

Tapez le **nom d'utilisateur** que vous avez défini pendant la procédure d'installation, appuyez sur **Entrée**, ensuite tapez le mot de passe associé à cet utilisateur et appuyez sur **Entrée**. Quand vous êtes ainsi connecté en tant que simple utilisateur, vous n'avez que des droits limités sur le système.

```
    cpnv@debian:~$
```

> **Attention** L'utilisation du compte root est réservée à la modification de la configuration du système, à l'installation de paquets et aux rares tâches qui nécessitent les droits de root. Pour toutes les autres tâches, il faut utiliser un compte utilisateur. En effet, l'utilisation du compte root est dangereuse : une fausse manipulation peut détruire le système, ce qui est impossible en tant que simple utilisateur !

3.2 Convention

Dans toute la suite de cette formation, nous adopterons la convention suivante

les commandes qui devront être exécutées en tant que root auront un prompt **#**

```
    # commande_à_exécuter
```

les commandes qui devront être exécutées en tant que simple utilisateur auront un prompt **$**

```
    $ commande_à_exécuter
```

3.3 Se déconnecter

Pour vous déconnecter, et retourner à l'invite de connexion, vous pouvez saisir la commande **logout** puis valider avec **Entrée**, ou bien utiliser la combinaison de touches **Ctrl+d**.

3.4 su

Cette commande sert à changer d'utilisateur, après avoir rentré le bon mot de passe, bien sûr !

- **su** permet de devenir root.
- **su cpnv** permet de devenir l'utilisateur *cpnv*.

Note Le passage de root à un simple utilisateur par la commande **su cpnv** se fait sans rentrer le mot de passe de l'utilisateur *cpnv*.

3.5 man

Depuis la version 9 "Strech" de Debian, nous devons installer le "man" avec la commande suivante :

```
# apt-get install man
```

man est une commande disponible sur les systèmes d'exploitation de type Unix. Elle permet de visionner le manuel d'une commande.

3.6 Utilisation

Elle doit être utilisée sous la forme suivante :

```
man [-s<section>] <nom_de_commande>
```

Par exemple, pour voir le manuel de la commande *ls*, il faut taper

```
$ man ls
```

Pour naviguer dans une page, il faut généralement utiliser les flèches vers le haut et vers le bas ou les touches page down et page up, ou encore la touche d'espacement, selon le pager utilisé. On sort généralement d'une page avec la touche Q, mais, avec certaines versions, il faut appuyer sur la touche d'espacement jusqu'à ce qu'on arrive à la fin de la page de manuel.

3.7 Sections

Les pages du manuel Unix sont divisées en plusieurs sections. Par exemple, sous Linux, on retrouve les sections suivantes :

- Commandes utilisateur (Section 1)
- Appels système (Section 2)
- Fonctions de bibliothèque (Section 3)
- Fichiers spéciaux (Section 4)
- Formats de fichier (Section 5)
- Jeux (Section 6)
- Divers (Section 7)
- Administration système (Section 8)
- Interface du noyau Linux (Section 9)

Chaque section possède une page d'introduction qui présente la section, disponible en tapant

```
$ man <section> intro
```

Pour plus d'information sur man, il est possible de regarder la page man de man, avec la commande

```
$ man man
```

4 Arborescence

4.1 cd (change directory)

cd (abréviation de l'anglais change directory), est une commande informatique disponible dans les interfaces en ligne de commande pour changer de répertoire courant.

cd est un appel système qui modifie l'attribut **current working directory** du processus qui l'invoque, cette fonction n'est pas un programme à part, et elle est intégrée à l'interpréteur de commandes.

On spécifie comme premier paramètre le répertoire où l'on désire aller :

```
$ cd <répertoire>
```

Lorsque le répertoire n'est pas spécifié, le répertoire personnel de l'utilisateur qui a lancé la commande est choisi.

En pratique, en utilisant **-** (moins) comme répertoire, cela a pour effet d'aller dans le répertoire où l'on se trouvait avant (un peu comme le bouton précédent d'un gestionnaire de fichiers). Le shell utilise en fait de façon transparente les variables d'environnement **PWD** (contenant le répertoire courant) et **OLDPWD** (contenant le répertoire précédent). Ainsi, lorsque l'on fait un **cd -**, le contenu de ces deux variables est échangé.

Lorsque l'on désire aller dans le répertoire parent, on spécifie le répertoire fictif .. (deux points).

```
$ cd ..
```

4.2 mkdir (make directory)

mkdir est une commande Unix permettant de créer des répertoires. **mkdir** est l'abréviation de *make directory* (termes anglais signifiant « créer répertoire »).

Voici par exemple comment créer le dossier "Linux" à l'emplacement courant:

```
$ mkdir Linux
```

Une option particulièrement utile de **mkdir** est *-p* (parent). Cette option permet de créer un ensemble de dossier plutôt qu'un dossier seul.

Par exemple, si on veut créer la hiérarchie de dossier "dir1/dir2/dir3", il faut normalement utiliser mkdir trois fois:

```
$ mkdir dir1
$ mkdir dir1/dir2
$ mkdir dir1/dir2/dir3
```

L'option -p permet de créer les dossiers intermédiaires:

```
$ mkdir -p dir1/dir2/dir3
```

4.3 ls (list)

ls est une commande *POSIX* (abréviation de list en anglais), qui permet d'afficher le contenu d'un répertoire.

Il est possible d'indiquer des arguments permettant d'obtenir plus ou moins d'informations. Par exemple l'argument *-a* permet d'afficher les fichiers dont le nom commence par "." (fichiers « cachés »).

Ces arguments peuvent varier d'une implémentation à l'autre, aussi il convient de vérifier la liste de ceux qui sont acceptés sur un système en regardant la page de manuel associée.

```
$ man ls
```

5 Liste de fichiers

5.1 touch

En fait, il n'existe aucune commande spécialement faite pour créer un fichier vide sous Linux (ce n'est pas très utile). En général, on se contente d'ouvrir un éditeur de texte et d'enregistrer, ce qui provoque la création d'un fichier.

La commande **touch** est à la base faite pour modifier la date de dernière modification d'un fichier. D'où son nom : on « touche » le fichier pour faire croire à l'ordinateur qu'on vient de le modifier alors que l'on n'a rien changé. Ça peut se révéler utile dans certains cas précis qu'on ne verra pas ici.

L'intérêt de touch pour nous, c'est que si le fichier n'existe pas, il sera créé ! On peut donc aussi utiliser touch pour créer des fichiers, même s'il n'a pas vraiment été fait pour ça à la base.

Pour créer un fichier vide, il vous suffit de taper la commande suivantes'il n'existe pas déjà. Dans le cas contraire, elle modifiera la date d'accès du fichier. :

```
$ touch fichier
```

Vous pouvez également utiliser la commande :

```
$ > fichier
```

5.2 cp (copy)

La commande **cp** sert à copier des fichiers (et eventuellement des répertoires). On peut aussi bien copier un fichier donné vers une destination précise que copier un ensemble de fichiers dans un répertoire.

Si le dernier argument correspond à un nom de répertoire, **cp** copie dans ce répertoire chaque fichier indiqué en conservant le même nom.

Sinon, s'il n'y a que deux fichiers indiqués, il copie le premier sur le second.

5.3 mv (move)

La commande **mv** est un raccourci pour "move" (déplacer). Elle est utilisée pour déplacer/renommer un fichier d'un répertoire à un autre. La commande **mv** est différente de la commande **cp** puisqu'elle retire le fichier entièrement de la source et le déplace au répertoire spécifique, tandis que la commande **cp** copie simplement le contenu d'un fichier à un autre.

- Pour renommer/déplacer un fichier

  ```
  $ mv file1.txt file2.txt
  ```

- Pour déplacer un répertoire

```
$ mv dir tmp
```

- Pour déplacer de multiples fichiers/Plus de fichiers dans un autre répertoire

```
$ mv file1.txt tmp/file2.txt newdir
```

5.4 more

La commande **more** permet d'afficher le contenu d'un fichier texte page par page dans la console.

6 Les liens

Les **liens** sont des fichiers spéciaux permettant d'associer plusieurs noms (liens) à un seul et même fichier. Ce dispositif permet d'avoir plusieurs instances d'un même fichier en plusieurs endroits de l'arborescence sans nécessiter de copie, ce qui permet notamment d'assurer un maximum de cohérence et d'économiser de l'espace disque.

6.1 Les liens symboliques

Les **liens symboliques** représentant des pointeurs virtuels (raccourcis) vers des fichiers réels. En cas de suppression du lien symbolique, le fichier pointé n'est pas supprimé. Les liens symboliques sont créés à l'aide de la commande **ln -s** selon la syntaxe suivante :

```
$ ln -s nom-du-fichier-reel nom-du-lien-symbolique
```

Le principe du **lien symbolique** est que l'on crée un lien vers un autre nom de fichier. On pointe vers le nom de fichier et non vers l'inode directement.

```
Fichier2 --> Fichier1 --> Inode
```

l'avantage des **liens symboliques** est qu'ils fonctionnent aussi sur des répertoires, contrairement aux **liens physiques**.

6.2 Les liens physiques

Les **liens physiques** (aussi appelées liens durs ou en anglais hardlinks) représentent un nom alternatif pour un fichier. Ainsi, lorsqu'un fichier possède deux liens physiques, la suppression de l'un ou l'autre de ces liens n'entraine pas la suppression du fichier. Plus exactement, tant qu'il subsiste au minimum un lien physique, le fichier n'est pas effacé. En contrepartie lorsque l'ensemble des liens physiques d'un même fichier est supprimé le fichier l'est aussi. Il faut noter toutefois qu'il n'est possible de créer des liens physiques qu'au sein d'un seul et même système de fichiers. Les liens physiques sont créés à l'aide de la commande **ln** (sans l'option -s) selon la syntaxe suivante :

```
$ ln nom-du-fichier-reel nom-du-lien-physique
```

Un **lien physique** permet d'avoir deux noms de fichiers qui partagent exactement le même contenu, c'est-à-dire le même **inode**

```
Fichier1 --> Inode <-- Fichier2
```

6.3 Exemple de liens

```
    $ ln -s fichier3.txt fichier
    $ ln fichier3.txt fichier2
    $ ls -i

1610496 lrwxrwxrwx  1 user user    14 fév 28 14:52 fichier -> fichier3.txt
1606777 -rw-r--r--  2 user user  3407 fév 27 12:57 fichier2
1606777 -rw-r--r--  2 user user  3407 fév 27 12:57 fichier3.txt
```

- Le lien en dur possède le même numéro d'inode que le fichier original (no d'identification du fichier)
- Le tiret en tête des permissions pour le lien en dur, un «l» pour le lien symbolique
- Le même volume (3407) pour le lien en dur, 14 pour le lien symbolique
- Le même horodatage (date de création du fichier) pour le lien en dur
- Le lien symbolique montre, à la fin de la ligne, où celui-ci pointe
- Le numéro 2 (avant user indique le nombre de noms que le fichier possède)

Si on:

- Supprime le fichier original (charlotte3.txt):
  - le lien symbolique tombe dans le vide (inutilisable)
  - le lien en dur est toujours fonctionnel

- Déplace l'original:
  - le lien symbolique tombe dans le vide (inutilisable)
  - le lien en dur fonctionne toujours

- Met l'original dans un dossier avec des droits interdits:
  - le lien symbolique ne pourra pas rediriger vers l'original
  - le lien en dur sera accessible

7 Les permissions

Les **permissions UNIX** constituent un système simple de définition des droits d'accès aux ressources, représentées par des fichiers disponibles sur un système informatique.

7.1 Notion d'utilisateur (user)

Toute entité (personne physique ou programme particulier) devant interagir avec un système UNIX est authentifiée sur cet ordinateur par un utilisateur ou **user**. Ceci permet d'identifier un acteur sur un système UNIX. Un utilisateur est reconnu par un nom unique et un numéro unique (la correspondance nom/numéro est stockée dans le fichier **/etc/passwd**).

Tous les utilisateurs UNIX n'ont pas les mêmes droits d'accès à l'ordinateur (ils ne peuvent pas tous faire la même chose), et ceci simplement pour des raisons de sécurité et d'administration.

Sur tout système UNIX, il y a un **super-utilisateur**, généralement appelé **root**, qui a tous les pouvoirs. Il peut accéder librement à toutes les ressources de l'ordinateur, y compris à la place d'un autre utilisateur, c'est-à-dire sous son identité. En général, du moins sur les systèmes de production, seul l'administrateur système possède le mot de passe root. L'utilisateur root porte le numéro 0.

7.2 Groupe

Un utilisateur UNIX appartient à un ou plusieurs **groupes**. Les groupes servent à rassembler des utilisateurs afin de leur attribuer des droits communs.

7.3 Propriété

Tout fichier UNIX possède un **propriétaire**. Au départ, c'est l'utilisateur qui a créé le fichier mais "root" peut le donner à un autre utilisateur. Seul le propriétaire du fichier et le super utilisateur (root) peuvent changer les droits.

Un fichier UNIX appartient aussi à un groupe.

7.4 Droits d'accès à un fichier

À chaque fichier est associée une **liste de permissions** (ACL), qui déterminent ce que chaque utilisateur et groupe a le droit de faire du fichier.

Sous UNIX **les répertoires sont aussi des fichiers**. Les droits sur les répertoires (mais aussi les périphériques, etc.) fonctionnent exactement de la même façon que sur des fichiers ordinaires.

7.5 Permissions du système de fichiers

Les permissions du système de fichiers d'un système basé sur UNIX sont définies pour trois catégories d'utilisateurs :

- **U** l'utilisateur qui possède le fichier
- **G** les autres utilisateurs du groupe à qui appartient le fichier
- **O** tous les autres utilisateurs dont on parle aussi en tant que « monde entier » ou « tout le monde »

Pour les fichiers, chaque permission correspondante permet les actions suivantes :

- **r** la permission en lecture permet à son propriétaire de voir le contenu du fichier
- **w** la permission en écriture permet à son propriétaire de modifier le fichier
- **x** la permission d'exécution permet à son propriétaire de lancer le fichier comme une commande

Pour les répertoires, chaque permission correspondante permet les actions suivantes :

- **r** la permission en lecture permet à son propriétaire d'afficher le contenu du répertoire
- **w** la permission en écriture permet à son propriétaire d'ajouter ou supprimer des fichiers de ce répertoire
- **x** la permission d'exécution permet à son propriétaire d'accéder aux fichiers du répertoire

La permission en exécution sur un répertoire ne signifie pas uniquement l'autorisation de lire des fichiers dans ce répertoire mais aussi l'autorisation de voir leurs attributs, tels que leur taille et l'heure de modification.

ls est utilisé pour afficher les informations de permissions (et davantage) des fichiers et répertoires. Lorsque cette commande est passée avec l'option *-l*, elle affiche les informations suivantes dans l'ordre donné :

- Type de fichier (premier caractère)
- Autorisation d'accès au fichier (neuf caractères, constitués de trois caractères pour l'utilisateur, le groupe et « les autres », dans cet ordre)
- Nombre de liens physiques vers le fichier
- Nom de l'utilisateur propriétaire du fichier
- Nom du groupe à qui appartient le fichier
- Taille du fichier en caractères (octets)
- Date et heure du fichier (mtime)
- Nom du fichier

7.6 Représentation des droits

Cet ensemble de 3 droits sur 3 entités se représente généralement de la façon suivante : on écrit côte à côte les droits r, w puis x respectivement pour le propriétaire (U), le groupe (G) et les autres utilisateurs (O).

Les codes U, G et O (U comme user, G comme group et O comme others) sont utilisés par les commandes UNIX qui permettent d'attribuer les droits et l'appartenance des fichiers. Lorsqu'un flag est attribué à une entité, on écrit ce flag (r, w ou x), et lorsqu'il n'est pas attribué, on écrit un "-".

```
rwx r-x r--
\ / \ / \ /
```

```
 v   v   v
 |   |   droits des autres utilisateurs (o)
 |   droits des utilisateurs appartenant au groupe (g)
 droits du propriétaire (u)
```

signifie que le propriétaire peut lire, écrire et exécuter le fichier, mais que les utilisateurs du groupe attribué au fichier ne peuvent que le lire et l'exécuter, et enfin que les autres utilisateurs ne peuvent que lire le fichier.

Une autre manière de représenter ces droits est sous forme binaire grâce à une clef numérique fondée sur la correspondance entre un nombre décimal et son expression binaire :

```
0 = 000
1 = 001
2 = 010
3 = 011
4 = 100
5 = 101
6 = 110
7 = 111
```

À l'expression binaire en trois caractères sont associés les 3 types de droits (r w x) ; il suffit donc de déclarer pour chacune des catégories d'utilisateur (user, group, others) un chiffre entre 0 et 7 auquel correspond une séquence de droits d'accès. Par exemple :

```
777 donne 111 111 111 soit rwx rwx  rwx
605 donne 110 000 101 soit rw- ---  r-x
644 donne 110 100 100 soit rw- r--  r--
666 donne 110 110 110 soit rw- rw-  rw-
```

Une astuce permet d'associer rapidement une valeur décimale à la séquence de droits souhaitée. Il suffit d'attribuer les valeurs suivantes pour chaque type de droit :

```
lecture   (r) correspond à 4
écriture  (w) correspond à 2
exécution (x) correspond à 1
```

Puis on additionne ces valeurs selon qu'on veuille ou non attribuer le droit en correspondant.

Ainsi, rwx « vaut » 7 (4+2+1), r-x « vaut » 5 (4+1) et r– « vaut » 4. Les droits complets (rwxr-xr–) sont donc équivalent à 754. Une manière directe d'attribuer les droits est de les écrire sous cette forme et d'utiliser le code à 3 chiffres résultant avec chmod (voir ci-après).

7.7 chown (change owner)

La commande **chown** (*change owner*, changer le propriétaire) permet de changer le propriétaire du fichier. Seuls le super-utilisateur ou le propriétaire actuel d'un fichier peut utiliser chown. La commande s'utilise de la façon suivante :

```
$ chown toto mon_fichier
```

Le fichier *mon_fichier* appartient maintenant à l'utilisateur *toto*.

chown permet aussi de changer en une seule commande le propriétaire et le groupe du fichier :

```
$ chown toto:lesPotes mon_fichier
```

Le fichier *mon_fichier1* appartient alors à l'utilisateur *toto* et au groupe *lesPotes*.

7.8 chgrp (change group)

La commande **chgrp** (pour *change group*) permet de changer le groupe auquel appartient le fichier. Tous les membres de ce groupe seront concernés par les permissions du groupe de la 2ème série de rwx. Encore une fois, seuls le super-utilisateur ou le propriétaire actuel d'un fichier peut utiliser chgrp (un membre du groupe ne peut pas changer le groupe propriétaire). La commande s'utilise de la façon suivante :

```
$ chgrp mesPotes mon_fichier
```

Le fichier *mon_fichier* appartient maintenant au groupe *mesPotes*. Tous les membres du groupe *mesPotes* auront accès à ce fichier selon les permissions du groupe.

7.9 chmod (change mode)

L'outil **chmod** (*change mode*, changer les permissions) permet de modifier les permissions sur un fichier.

Il peut s'employer de deux façons :

- soit en ajoutant ou en retirant des permissions à une ou plusieurs catégories d'utilisateurs à l'aide des symboles *r w et x* de manière "relative"
- soit en précisant les permissions de manière octale

  Les deux méthodes sont équivalentes, elles affectent toutes deux les permissions de la même manière.

7.9.1 La manière "symbolique"

De cette façon, on va choisir :

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

La modification que l'on veut faire

- + : ajouter
- - : supprimer
- = : force la valeur

Le droit que l'on veut modifier

- **r** : read –> lecture
- **w** : write –> écriture
- **x** : execute –> exécution

Exemple :

```
$ chmod o-w mon_fichier
--> Enléve le droit d'écriture pour les autres.

$ chmod a+x mon_fichier
--> Ajoute le droit d'exécution à tout le monde.

$ chmod a=r mon_fichier
--> Alloue pour tous le monde le droit en lecture seul.
```

On peut aussi combiner plusieurs actions en même temps :

```
$ chmod u+rwx,g+rx-w,o+r-wx mon_fichier
--> Ajoute la permission de lecture, d'écriture
    et d'exécution sur le fichier pour le propriétaire
--> Ajoute la permission de lecture et d'exécution au
    groupe propriétaire, on retire la permission d'écriture
--> Ajoute la permission de lecture aux autres, on retire
    la permission d'écriture et d'exécution
```

7.9.2 La manière "octale"

En octal, chaque « groupement » de droits (pour user, group et other) sera représenté par un chiffre et à chaque droit correspond une valeur :

```
r = 2^2 --> 4
w = 2^1 --> 2
x = 2^0 --> 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
```

Reprenons le répertoire *Documents*. Ses permissions sont :

```
drwxr-x---
rwx         r-x         ---
7(4+2+1)    5(4+0+1)    0(0+0+0)
En octal, on aura 750
```

Pour mettre ces permissions sur le répertoire on taperait donc la commande :

```
$ chmod 750 Documents
```

7.9.3 Remarques importantes

- Un programme ne peut être exécuté que si le fichier exécutable correspondant possède le droit d'exécution dans la catégorie à laquelle appartient l'utilisateur.
- On ne peut accéder à un fichier que si les répertoires successifs constitutifs du chemin absolu de ce fichier possèdent le droit en exécution.
- Pour pouvoir lister les fichiers d'un répertoire, ce dernier doit être accessible en lecture.
- Le droit en exécution n'a aucune incidence sur un fichier non exécutable.
- Un script (c'est-à-dire un fichier texte contenant des commandes du Shell) doit avoir les droits en lecture et en exécution pour pouvoir être interprété et exécuté par le Shell.

7.10 Masque de protection des fichiers (umask)

Le masque de protection de fichier permet de définir les droits par défaut de tout fichier créé.

Ce masque se comporte comme un filtre et utilise la notation numérique. On parle de filtre car il ne contient pas la série des 3 chiffres octaux correspondants aux droits à allouer aux fichiers, mais celle correspondant aux droits à ne pas allouer.

Le système Unix affecte à un fichier les droits globaux résultant de la soustraction des droits maxima *777* par le masque de protection.

Exemple : si le masque de protection vaut *037* alors *740 (=777-037)* seront les droits alloués à tout nouveau fichier.

La commande permettant de définir un nouveau masque de protection est umask.

```
$ umask 037
```

Exemple de calcul

```
      777 = rwx rwx rwx = 111 111 111
    - 037 = --- -wx rwx = 000 011 111
    = 740 = rwx r-- --- = 111 100 000
```

D'après cet exemple, tout nouveau fichier aura les droits 740 (rwxr—–) car le masque de protection vaudra 037 (—-wxrwx).

Pour connaître la valeur du masque de protection, tapez *umask* sans attribut.

> **Attention** : Lors de la création d'un fichier, même si le masque de protection spécifie le droit en exécution, ce dernier ne sera pas affecté au fichier nouvellement créé mais seulement à un répertoire. Donc, si vous créez un fichier exécutable ou un script il faudra lui rajouter manuellement le droit en exécution.

8 Les entrées/sorties des processus

Chaque processus 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.

8.1 Rediriger la sortie standard

Quand on exécute une commande, le shell affiche le résultat sur la console de sortie (l'écran par défaut). On peut rediriger cette sortie vers un fichier en utilisant le signe **>**.

Exemple :

```
$ ls > resultat
--> Si le fichier existe déjà, il est écrasé.
```

8.2 Concaténation

Au lieu de créer un fichier, il est possible d'ajouter les sorties d'un processus à un fichier existant en utilisant le double signe **>>**.

Exemple :

```
$ ls >> resultat
--> Si le fichier résultat existe déjà, les affichages sont concaténés.
```

8.3 Syntaxe complète

En fait, les signes **>** peuvent être précédés de l'identifiant du flux à rediriger. Pour la sortie standard, on peut donc utiliser les syntaxes suivantes :

```
$ ls 1> resultat
$ ls 1>> resultat
--> Ce qui revient au même que les deux premiers exemples ci-dessus
```

8.4 Rediriger la sortie d'erreur standard

La redirection du flux de sortie d'erreur standard utilise les même signes, mais précédés de l'identifiant du flux : *2*.

Exemples :

```
$ ls 2> erreurs
$ ls 2>> erreurs
```

8.5 Rediriger l'entrée standard

Rediriger l'entrée standard permet d'entrer des données provenant d'un fichier au lieu du clavier.

Exemple :

```
$ cat < mon_fichier.txt
```

8.6 /dev/null

/dev/null est un fichier spécial des systèmes d'exploitation Unix.

Ce pseudo-périphérique (device, en anglais) sert à rediriger un contenu dont on n'a pas besoin, et qui ne doit pas être sauvegardé ni affiché à l'écran. Toute information envoyée vers ce « périphérique » est automatiquement détruite.

Ce périphérique est appelé null et il est situé dans le répertoire */dev* où sont répertoriés les périphériques. On envoie donc ce qu'on ne veut pas garder vers ce */dev/null*.

Exemple :

Voici un exemple qu'un utilisateur de système basé sur Unix peut utiliser. Cette ligne de commande affiche tous les dossiers et fichiers du système mais va générer des erreurs sur les dossiers ou fichiers sur lesquels il n'aura pas les droits d'accès. Ce script va alors rediriger *stderr (2)* vers le pseudo périphérique */dev/null* inhibant ainsi l'affichage de celles-ci.

```
$ find / 2> /dev/null
```

8.7 Le pipe (un tube)

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.

Exemples :

```
$ du | sort -rn
--> Afficher la taille des fichiers et répertoires
    et les trier du plus grand au plus petit :

$ du | sort -rn | more
--> Même résultat, mais affiché page par page
```

```
$ ls -1 /usr/bin | wc -l
--> Connaître le nombre de fichiers du répertoire /usr/bin
    L'option -1 de la commande ls affiche un fichier
    ou répertoire par ligne. La commande wc (word count)
    avec l'option -l (line) compte le nombre de lignes.
```

9 Les utilisateurs

Chaque utilisateur du système dispose de droits différents sur les processus et sur les fichiers. Il existe des utilisateurs humains qui ont un compte sur la machine, et des utilisateurs systèmes qui sont en général des utilisateurs utilisés par certains programmes spécifiques.

9.1 Connaitre la liste des utilisateurs du système

Il y a deux types d'utilisateurs, ceux que l'on appellera *réels* et ceux considérés comme *systèmes*.

9.2 Les utilisateurs systèmes

Les *utilisateurs systèmes* sont ceux employés par des tâches spécifiques (exécution de logiciels ou commandes) régulières qui souvent portent le même nom que l'utilisateur.

9.3 Les utilisateurs réels

Les *utilisateurs réel*s, quant à eux sont associés à des personnes qui se connectent sur la machine soit via une interface graphique ou un accès en ligne de commande (shell) particulier.

9.4 Ajout d'un utilisateur

Cette opération se fait via la commande *adduser*.

Pour ajouter "toto" on lance donc la commande suivante :

```
$ adduser toto
--> Aprés plusieurs questions facultatives et la demande
    d'un mot de passe, l'utilisateur "toto" est créé sur le systéme.
```

Par défaut son répertoire utilisateur sera */home/toto* Les options par défaut affiliées à un utilisateur lors de sa création sont définies dans le fichier de configuration */etc/adduser.conf* qui est éditable en tant que super-utilisateur.

9.5 Changer d'utilisateur

Losque l'on est connecté à la machine et que l'on souhaite changer d'utilisateur, il suffit de lancer la commande :

```
$ su autre_utilisateur
--> autre_utilisateur est le nom d'un utilisateur existant
    sur le systéme. Son mot de passe sera alors demandé.
```

Pour revenir à l'utilisateur précédent il suffira ensuite de lancer la commande :

```
$ exit
```

Pour devenir super-utilisateur, la commande *su* seule suffit.

9.6 Changer le mot de passe d'un utilisateur

Pour modifier le mot de passe d'un utilisateur, on utilise la commande *passwd*.

Après le lancement, le système demandera alors le nouveau mot de passe choisi et sa confirmation. Cette commande lancée seule changera le mot de passe sur compte avec lequel vous êtes connecté.

Pour changer le mot de passe d'un autre utilisateur, exécutez cette commande connecté en *root*

```
# passwd toto
```

9.7 Suppression d'un utilisateur du système

Dans cet exemple, nous allons supprimer le compte *toto* précédemment créé du système.

Avant de supprimer l'utilisateur, il est important de bloquer son compte. On utilise la commande suivante pour ce faire :

```
# passwd -l toto
```

L'utilisateur ne devrait plus pouvoir se connecter (vous pouvez à ce moment sauvegarder sainement le contenu du répertoire /home/toto/ par exemple). On ne peut supprimer un compte de quelqu'un logué sur la machine. On va donc fermer l'ensemble des programmes lancés par cet utilisateur ainsi que sa connexion.

La commande suivante liste les pid de ces programmes :

```
# pgrep -u toto
```

Si elle ne renvoie aucun nombre, cela veut dire que l'utilisateur n'est pas logué sur la machine et qu'il n'a lancé aucun programme.

Si vous souhaitez voir les programmes lancés par l'utilisateur, lancez cette commande :

```
# ps -fp $(pgrep -u toto)
```

Si la commande a retourné quelque chose, utilisez les commandes suivantes pour fermer les programmes :

```
# killall -KILL -u toto
```

Sous Debian, la commande *killall* se trouve dans le paquet *psmisc*.

On peut ensuite supprimer l'utilisateur *toto* et son répertoire *home* ainsi que sa boîte email (avec l'option -r) avec la commande userdel :

```
# userdel -r toto
```

Si l'utilisateur a des tâches *CRON* dans son fichier crontab, supprimez-les avec la commande :

```
# crontab -r -u toto
```

9.8 sudo

L'arrivée d'*Ubuntu* a contribué à l'utilisation régulière de la commande **sudo** installée par défaut sur cette distribution. Elle permet de gérer simplement la délégation de droits normalement associés au super-utilisateur (root) du système à d'autres utilisateurs d'une manière simple.

9.8.1 Installation de sudo

sudo n'est pas installé par défaut sur Debian (contrairement à Ubuntu), il est donc nécessaire de l'installer :

```
$ su
# apt-get install sudo
```

9.9 Configuration minimale de sudo

Pour éditer les autorisations de *sudo*, tout se passe dans le fichier */etc/sudoers*.

On ouvre donc ce fichier afin de l'éditer :

```
# visudo
```

On souhaite qu'un utilisateur dont l'identifiant est *cpnv* puisse avoir accès à l'ensemble des droits root sur le serveur, on ajoute donc la ligne suivante en fin de fichier :

```
cpnv      ALL=(ALL:ALL) ALL
```

Il en sera de même pour chaque utilisateur système auquel on souhaite autoriser ces fonctions. Si vous souhaitez affiner les droits d'utilisateurs, vous pouvez vous référer à cette adresse http://guide.andesi.org/html/ksudo.html.

10 Le Shell

10.1 history

La commande **history** est une commande *POSIX* (commune à toutes les distributions type Unix) qui permet de remonter dans les commandes que nous avons passées.

Il vous arrive probablement souvent de remonter vos commandes pour les réexecuter avec le **UP** (flèche du haut) de votre clavier, vous faites ainsi défiler l'ensemble des commande que vous avez exécutées. Cela est possible grâce à *history*. En effet, quand vous passez une commande, celle-ci est stockée dans un fichier du nom de *.historyb* du répertoire *home/$User*. Si l'utilisateur *cpnv* tape les commandes *ls* et *cd /*, celles-ci seront stockées dans le fichier */home/cpnv/.history* et quand *cpnv* voudra remonter dans ses commandes, c'est le processus et le fichier *.history* qui l'aideront.

Tapez la commande suivante pour voir vos dernières commandes passées:

```
$ history
```

Les commandes que vous verrez seront alors uniquement celles de l'utilisateur avec lequel vous êtes connecté. Pour voir les commandes d'un autre utilisateur, il faut aller ouvrir le fichier *.history* de son répertoire, dans une architecture de droit normale, vous n'y serez bien sur pas autorisé (mis à part si vous êtes root).

10.2 Le nombre d'entrées

Il faut savoir qu'*history* garde les *1000 commandes* (change selon les version et les distributions) les plus récentes et commence à effacer les plus anciennes une fois que le nombres maximum est dépassé.

Il est possible d'agrandir le nombre de commande stocké mais cela risque de ralentir votre shell.

Pour voir le nombre de commande qu'history peut stocker, entrez la commande:

```
$ echo $HISTSIZE
```

Pour changer ce nombre à 300 (par exemple):

```
$ export HISTSIZE=300
```

10.3 Effacer ses traces

Pour vider votre fichier *.history* afin que root ou un utilisateur avec certains privilèges ne puisse plus voir les commandes que vous avez entrées, utilisez la commande:

```
$ history -c
```

Un autre moyen est de supprimer le fichier *.history* dans votre répertoire */home/user*. Il faut savoir que sous Linux, quand vous mettez un . devant un fichier, cela le rend invisible. Cela explique pourquoi un simple *ls* n'affiche pas ce fichier, il faut utiliser *ls -a*.

10.4 Utilisations supplémentaires

history permet comme je vous l'ai dit de remonter dans les commandes que vous avez entrées. Quelques raccourcis sont disponibles en plus du simple **UP** ou **DOWN** pour naviguer dans ses précédentes commandes. Cette commande permet de remonter dans ses commandes de manière plus rapide:

```
$ !n -3
```

Par exemple avec *-3* pour retrouver l'avant avant avant dernière commande tapée. Il est ainsi possible de remonter jusqu'à $HISTSIZE commande:

```
$ !n 3
```

A l'inverse, sans le -, on retrouve la troisième commande tapée. Cela perd de son intérêt quand nous avons tapé beaucoup de commandes car elles s'effacent une fois que l'HISTSIZE est dépassée.

La commande suivante permet de trouver les commandes contenant un mot précis, par exemple pour retrouver les commandes qui contiennent le mot "doc" nous entrerons:

```
$ !?doc?
```

10.5 alias

Certaines commandes récurrentes sous Linux peuvent être longues et préter à des erreurs de saisie. Pour cela il existe la possibilité de créer des *alias de commandes* qui sont des raccourcis vers des commandes plus complexes et/ou plus grande.

10.6 Utilisation

Il est par exemple possible de remplacer la commande suivante :

```
# /etc/init.d/networking restart
```

par

```
# rn
```

Pour Restart Network par exemple, le nom de la commande alias est totalement libre. Il doit bien sur être plus cours ou plus explicite que la commande auquel il se substitue.

10.7 Création

La création d'un *alias* est assez simple. Il faut utiliser la commande *alias* comme suivant :

```
# alias rn='/etc/init.d/networking restart'
```

Il faut savoir qu'un alias s'efface lors du redémarrage. Pour parer à ce problème et faire en sorte que l'alias existe toujours après un redémarrage, il faut mettre la commande précédente dans le fichier *.bashrc* de votre utilisateur. Par exemple pour root :

```
# nano /root/.bashrc
```

Il faut ensuite recharger notre shell :

```
# source ~/.bashrc
```

Il est également possible de voir tous les alias créés avec la commande alias :

```
# alias
```

10.8 Supprimer un alias

Pour supprimer un *alias*, utilisez la commande *unalias* suivi de l'alias que vous désirez supprimer.

```
# unalias rn
```

10.9 PATH

PATH est une variable d'environnement. Pour afficher le contenu d'une variable d'environnement, on utilise la commande *echo* :

```
$ echo $PATH
```

La variable *PATH* contient la liste de tous les répertoires dans lesquels le système va chercher les exécutables des commandes que vous tapez au prompt, séparés par des *:* . Par exemple, le répertoire */bin/* contient les commandes Unix de base, et vous pouvez vérifier qu'il est bien dans le *PATH*.

La commande *printenv* affichent les noms et les valeurs de toutes les variables d'environnement définies.

Pour modifier le *PATH*, éditez le fichier de configuration *~/.profile* et ajoutez à la fin du fichier

```
PATH="rep_chemin:$PATH"
--> Remplacer rep_chemin par le nom de répertoire à rajouter
```

11 Les processus

11.1 Gestionnaire de fenêtres

En système de fenêtrage un *gestionnaire de fenêtres* (« window manager » en anglais) est un logiciel chargé de l'affichage et du placement des fenêtres d'applications. Les plus connus sont ceux utilisé par le système de fenêtrage X (sur les systèmes Unix, Linux et BSD).

Le gestionnaire de fenêtres constitue l'intermédiaire entre le système de fenêtrage et l'environnement graphique.

Ici, on parle plus particulièrement des gestionnaires basés sur le système de fenêtrage X.

Étant lui-même un client sur serveur X, le gestionnaire de fenêtres offre des moyens pour déplacer, redimensionner et icônifier les fenêtres affichées par les autres clients. De plus, il ajoute une décoration aux fenêtres qui consiste souvent en un cadre et une barre de titre. La majorité des gestionnaires sait de plus gérer plusieurs bureaux virtuels et des raccourcis clavier.

11.2 X Window System

X Window System ou *X11* ou simplement X est un environnement graphique de type « fenêtré » qui gère l'interaction homme-machine par l'écran, la souris et le clavier de certains ordinateurs en réseau. Il est souvent appelé X Window. C'est le système standard ouvert d'interaction graphique avec l'utilisateur sur les UNIX (Linux, BSD, etc.). Le serveur X est optionnel.

11.3 OpenBox

Openbox est un gestionnaire de fenêtres pour le système X Window. Il est distribué sous licence GPL.

Openbox est conçu pour être petit, rapide et entièrement compatible avec le ICCCM et le Extended Window Manager Hints (en). Il supporte de nombreuses fonctionnalités comme les menus d'applications ou l'affichage dynamique de différentes informations.

Son principal avantage est son extrême légèreté, qui favorise stabilité et réactivité du système, et pas seulement sur des machines peu puissantes.

Son interface très simple rend son utilisation très facile. Par contre son paramétrage peut demander des compétences minimales en informatique, surtout si on passe directement par les fichiers de configuration et non par des programmes comme obmenu, obconf et lxappearance.

Openbox peut s'utiliser seul, sans environnement de bureau, ou en remplacement du gestionnaire de fenêtres intégré dans un environnement de bureau.

11.3.1 Installation de OpenBox

```
# apt-get install x-window-system-core
# apt-get install xdm numlockx xterm xserver-xorg
# apt-get install openbox
# apt-get install obconf obmenu
```

11.4 RunLevel

Le niveau d'exécution (*runlevel*) de Linux contrôle le choix des processus ou services qui sont démarrés automatiquement par le système (ou, plus exactement, par Init). Le niveau d'exécution est désigné par un chiffre de 0 à 6. Les niveaux d'exécution 0, 6 sont réservés respectivement à l'extinction du système, au redémarrage.

Debian définit sept niveaux d'exécution (0-6).

```
0      (arrête le systéme)
1 - S  (simple utilisateur / mode minimal)
2 à 5  (modes multi-utilisateurs)
6      (redémarre le systéme).
```

L'installation par défaut de Debian ne fait pas de différence entre les niveaux d'exécution de 2 à 5. Vous pouvez les personnaliser à votre guise. Le niveau d'exécution 1 est utilisé pour la maintenance du système. Il lance le minimum de services pour éviter de possibles problèmes.

Votre système démarre au niveau d'exécution spécifié dans */etc/inittab*.

Par exemple

```
id:2:initdefault:
--> lance le systéme au niveau d'exécution 2
    par défaut dans Debian
```

Les niveaux d'exécution peuvent être modifiés manuellement en modifiant les scripts de contrôle dans */etc/init.d* et les liens symboliques dans */etc/rc0.d … /etc/rc6.d*. Veuillez lire attentivement les instructions contenues dans les références ci-dessous. Comme la modification manuelle est une tâche fastidieuse, on vous recommande de vous servir d'un éditeur de niveau d'exécution. Avec Debian, installez le paquet *sysv-rc-conf*. Vous pouvez ensuite modifier les niveaux d'exécution en ouvrant simplement un terminal en tant que super-utilisateur puis en exécutant le programme *sysv-rc-conf*.

Vous pouvez aussi changer le niveau d'exécution pendant l'exécution. Utilisez seulement les niveaux 1 à 5. Utilisez la commande *init runlevel* ou *telinit runlevel*. La seconde est préférable.

12 IP

12.1 Principes

Les réseaux informatiques utilisent un modèle composé de plusieurs couches de protocoles. Nous nous intéressons ici à la troisième couche, dite couche réseau, qui utilise le protocole IP (Internet protocol), dans sa version 4 ou 6 : c'est cette couche qui définit la topologie des réseaux, et dont la configuration est par conséquent très importante.

On se connecte à un réseau en utilisant une carte ou une clef réseau. Du point de vue du système d'exploitation, ce périphérique est une interface réseau. Sous Linux, ces interfaces sont nommées eth0, eth1… pour des interfaces filaires, et wlan0, wlan1 pour des interfaces sans fil (wifi, wimax…). Il existe également une interface spéciale, nommée lo (pour loopback) qui désigne toujours votre propre ordinateur.

12.2 Éléments de configuration

Une configuration réseau complète, permettant de profiter d'un réseau ou de l'Internet, est constituée des éléments suivants :

une adresse IP

cette adresse identifie votre hôte sur le réseau où il est connecté ;

un masque de sous-réseau

cette donnée indique la partie de votre adresse qui caractérise le réseau local sur lequel votre hôte est connecté, et lui permet de déterminer, pour n'importe quelle adresse IP, si celle-ci fait ou non partie du réseau local ;

une passerelle par défaut

c'est l'adresse IP à laquelle il faut transmettre les paquets IP destinés à des hôtes situés hors du réseau local, pour qu'ils soient routés vers le réseau local de leur destinataire ;

des serveurs DNS

ce sont les adresses de serveurs auxquels votre système ira demander les correspondances entre noms de domaine (www.debian.org) et adresses IP (194.109.137.218).

Chaque élément de configuration est nécessaire pour pouvoir utiliser normalement le réseau ou l'Internet :

- sans adresse IP, il est impossible de recevoir les réponses à ses requêtes ;
- sans masque de sous-réseau ou sans passerelle par défaut, il est impossible de communiquer avec les hôtes situés hors du réseau local ;
- sans serveur DNS, on ne peut pas désigner un hôte par son nom de domaine, et il faut donc connaître les adresses IP de tous les serveurs que l'on souhaite utiliser.

13 Archivage

tar (*tape archiver*) est un outil très puissant pour la manipulation d'archives sous les systèmes Unix et les dérivés dont Linux. Il ne compresse pas les fichiers, mais les concatène au sein d'une seule et même archive. La majorité des programmes Linux utilisent ce système d'archivage.

13.1 Installation

Le programme *tar* est disponible par défaut sous Debian. Il fait partie de l'installation minimale.

Pour tous les formats à base de *tar*, vous verrez que les options de tar sont les mêmes :

- c : créer l'archive
- x : extraire l'archive
- f : utiliser le fichier donné en paramètre
- v : activer le mode « verbeux » (bavard, afficher ce qu'il fait).

Puis selon la compression souhaitée :

- z : ajouter la compression Gzip
- j : ajouter la compression Bzip2
- J : ajoute la compression Lzma

13.2 Utilisation en archivage simple

Utilisation tar seul : concaténation de fichiers

Création d'une archive, archivage de plusieurs fichiers :

```
    tar -cvf archive.tar fichierarchive1 fichierarchive2...
```

De même pour un dossier :

```
    tar -cvf archivedossier.tar dossier/
```

Pour l'extraction :

```
    tar -xvf archive.tar -C dossier
```

on remarque ici l'utilisation de trois options :

v pour un affichage des actions, mode verbeux
f pour indiquer que l'on va utiliser un fichier (on précise son nom ici : archivedossier.tar)
C pour indiquer le dossier de destination de l'extraction

13.3 Utilisation avec compression

Il est possible d'utiliser tar pour l'archivage et la compression de fichiers. Dans ce cas, il fait appel à d'autres utilitaires comme *Bzip2*, *Gzip*, *Gunzip*. L'archivage se fait alors en deux temps, d'abord la concaténation des fichiers en un seul puis la compression du tout.

13.4 Compression avec gzip (.tar.gz)

Création

```
    tar -zcvf votre_archive.tar.gz votre_dossier/
```

Extraction

```
    tar -zxvf votre_archive.tar.gz
```

13.5 Compression avec Bzip2 (.tar.bz2)

Remarques : Bzip2 crée des fichiers beaucoup plus petits que Gzip, mais utilise plus de ressources processeur surtout pour compresser.

Création

```
    tar -jcvf votre_archive.tar.bz2 votre_dossier/
```

Extraction

```
    tar -jxvf votre_archive.tar.bz2
```

13.6 Compression avec Lzma (.tar.xz)

Ces archives sont des archives Tar compressées avec Lzma, un utilitaire de compression libre parmi les plus puissants: c'est la même méthode de compression que celle utilisée par 7zip.

Pour utiliser le format « .xz », installez le paquet xz-utils.

Création

```
    tar -Jcvf votre_archive.tar.xz votre_dossier/
```

Extraction

```
tar -Jxvf votre_archive.tar.xz
```

13.7 Archivage incrémentiel

La taille des archives et leur stockage peuvent très vite poser problème. Voici un cas d'utilisation illustrant l'utilité de cet archivage. Vous désirez sauvegarder le /home (données des utilisateurs) toutes les semaines sur un second disque dur, tout en gardant les données antérieures. Vous avez en tout 50 Go de données et le second disque dur fait 500Go. 10% de ces données changent toutes les semaines (5Go). Dans le cas d'une sauvegarde complète, chaque archive fait 50Go, votre disque serait rempli en deux mois.

Nous remarquons que si 10% de ces données changent toutes les semaines, 90% sont identiques et ne nécessitent pas d'être sauvegardées à chaque fois. Il nous faut donc sauvegarder uniquement les nouvelles données en utilisant la sauvegarde incrémentielle. La première sauvegarde est complète, la suivante copie uniquement les nouveaux fichiers et ainsi de suite. Vous gardez ainsi chaque sauvegarde dans l'état à différentes dates.

13.8 Utilisation de l'archivage incrémentiel

Créer la première sauvegarde (sauvegarde complète) :

```
tar --create --file=archive.1.tar
--listed-incremental=/save/save.list /home
```

Seconde sauvegarde (incrémentée avec uniquement les fichiers ayant changé) :

```
tar --create --file=archive.2.tar
--listed-incremental=/save/save.list /home
```

Restauration :

```
tar --extract --listed-incremental=/dev/null
--file archive.1.tar
tar --extract --listed-incremental=/dev/null
--file archive.2.tar
```

Utiliser la date pour incrémenter le numéro :

```
tar --create --file=/save/archive. date --rfc-3339=date .tar
--listed-incremental=/save/archive.list /home
```