Book Initiation — V1

Club*NIX

https://clubnix.fr

Table des matières

1 Linux **3**
- 1.1 GNU-Linux . . . 3
  - 1.1.1 UNIX . . . 3
  - 1.1.2 GNU : "GNU's Not UNIX" . . . 3
  - 1.1.3 GNU/Linux . . . 4
- 1.2 Différents OS . . . 4
  - 1.2.1 Différents types d'OS basés sur différents noyaux . . . 4
  - 1.2.2 Différentes familles d'OS Linux . . . 5

2 Ecrire un document texte **6**
- 2.1 LaTeX . . . 6
  - 2.1.1 Histoire et Principe . . . 6
  - 2.1.2 Différences avec Word . . . 6
  - 2.1.3 Utilisation . . . 6
- 2.2 Markdown . . . 8
  - 2.2.1 Histoire . . . 8
  - 2.2.2 Instructions . . . 8

3 Terminal **10**
- 3.1 Terminal . . . 10
  - 3.1.1 Quoi et pourquoi ? . . . 10
  - 3.1.2 Ouvrir une console et s'en servir . . . 10
  - 3.1.3 L'auto-complétion . . . 11
  - 3.1.4 Les droits SUDO . . . 11
  - 3.1.5 Les questions des commandes . . . 11
- 3.2 Bash . . . 12
  - 3.2.1 Bash et scripts . . . 12

4 SSH **13**
  - 4.0.1 Définition . . . 13
  - 4.0.2 Les Commandes De Bases . . . 13

5 Exemples de logiciels libres **15**
- 5.1 Git . . . 15
  - 5.1.1 Définition . . . 15
  - 5.1.2 Commandes de base . . . 15
- 5.2 F-Droid . . . 16

6 Infrastructure réseau **17**
- 6.1 LDAP . . . 17
  - 6.1.1 Le LDAP, ou *Lightweigth Directory Acess Protocol* . . . 17
  - 6.1.2 Introduction . . . 17
- 6.2 Le LDAP du club Nix ou de l'école . . . 17
  - 6.2.1 Les entrées . . . 17
  - 6.2.2 Chercher des informations d'un LDAP . . . 17
- 6.3 NFS . . . 18
  - 6.3.1 Introduction . . . 18

6.3.2 Le NFS du Club*Nix ou de l'ESIEE 18
6.3.3 La sécurité du NFS 18
6.4 Fichiers logs 18
6.4.1 Définition 18
6.5 Virtual Machine 19
6.5.1 Définitions 19

1 Linux

1.1 GNU-Linux

1.1.1 UNIX

UNIX est un **systeme d'exploitation** multi-tâches et multi- utilisateurs créé en **1969**. Il est composé de plusieurs composants :

Le noyau : aussi appelé “kernel” en anglais. C'est un logiciel central au système d'exploitation ; il fait le lien entre le matériel et les logiciels plus normaux.

Un environnement de développement : ou tout simplement des outils qui permettent de créer de nouveaux logiciels/outils.

Des commandes qui permettent d'exécuter des actions qu'on attend d'un système d'exploitation, comme manipuler les fichiers, aller sur internet, etc.

De la documentation qui explique le fonctionnement du système et des outils fournis.

C'est une **marque deposée de l'OpenGroup**. Son nom est un dérivé de "Unics" (*Uniplexed Information and Computing Service*). C'est un jeu de mot avec "Multics" (un autre système) qui vise à offrir simultanément plusieurs services à un ensemble d'utilisateurs.

UNIX a donné naissance à une **famille de systèmes** comme BSD, GNU/Linux, iOS ou encore MacOS, eux-mêmes divisés en variantes de systèmes d'exploitation aux **normes POSIX**. Cette norme technique standardise des outils ou des fonctionnements que l'on devrait retrouver dans un système d'exploitation.

Il faut savoir que la quasi-totalité des systèmes PC ou mobiles (à l'exception des systèmes Windows) sont sur des systèmes basés sur la norme POSIX (y compris ceux de la marque Apple). D'une certaine manière, on peut dire qu'ils sont descendants ou directement reliés à UNIX.

1.1.2 GNU : "GNU's Not UNIX"

GNU est un **projet des années 1990** lancé par Richard Stallman. C'est un **système d'exploitation libre**, bien qu'incomplet, compatible avec la norme POSIX. GNU est incomplet car il lui manque un noyau viable ; il est donc très souvent utilisé avec le noyau Linux, développé par Linus Torvalds, ce qui a donné le projet GNU/Linux (c.f. section 1.1.3).

Sa **philosophie** est de maintenir intacts les **tradictions hackers de partage** dans un monde de plus en plus marqué par l'empreinte du droit d'auteur. Il se bat donc pour une **libre diffusion des connaissances**. "GNU vise à ne laisser l'homme devenir ni l'esclave de la machine ni de ceux qui auraient l'exclusivité de sa programmation".

Il est **utilisable et partageable librement par tous**, ainsi chacun complète petit a petit l'architecture initiale de GNU pour le rendre meilleur. C'est dans ce contexte que le projet invite la communauté de hackers à le rejoindre et à participer à son développement. Il faut savoir qu'il est composé :

— d'un éditeur de texte (emacs),
— d'un compilateur très performant (gcc),
— d'un débogueur (gdb),
— d'un langage de script (bash),
— de bibliothèques de systèmes (glibc),

1.1.3 GNU/Linux

GNU/Linux a été créé en **1991**. Il est **initié par le projet Debian** et la naissance du noyau Linux. Il crédite donc à la fois Linux et GNU *mais l'usage de Linux est plus connu au grand public.*

Il est alors toujours basé sur le **mouvement du logiciel libre et du mode opératoire du hacker**. Cette association a eu lieu pour combler le vide causé par le développement inachevé de GNU (c.f. section 1.1.2).

Il est utilisé sur la plupart des téléphones portables (par exemple Android) comme sur les super-ordinateurs. Ce projet eut un grand impact dans le monde des serveurs informatiques. GNU/Linux veut casser le fait qu'à l'origine il fallait des connaissances solides en informatique pour utiliser un système d'exploitation (*pas d'interface graphique et besoin d'installer toutes les applications soi-même*).

Il a donc été un important vecteur de **popularisation du mouvement de l'open source**. Il a eu des **centaines de milliers de redistributions** avec des versions différentes pour plaire à tous les goûts (en fonction des besoins, configurations, sécurité, ...*voir différents OS*). GNU/Linux a remplacé d'autres systèmes de type UNIX et/ou évite l'achat d'une licence Windows (qui est très chère à l'achat).

Aujourd'hui on peut retrouver tous les équivalents des logiciels/applications qu'il y a sous Windows mais en Open Source.

1.2 Différents OS

En informatique, un système d'exploitation est un ensemble de programmes qui dirige l'utilisation des capacités d'un ordinateur par des logiciels applicatifs.

1.2.1 Différents types d'OS basés sur différents noyaux

MacOS : série d'interfaces graphiques basée sur l'opération des systèmes développé par Apple pour leur Macintosh, originellement basé sur un système UNIX

iOS : système d'exploitation pour mobile développé et distribué par Apple pour iPhone et iPod

GNU/Linux : système d'exploitation pour ordinateur assemblé sous le modèle du "Free and Open Source Software" ou *FOSS*

Android : c'est un dérivé de GNU/Linux ; système d'opération désigné principalement pour les mobiles tactiles comme les smartphones et les tablettes, initialement développé par Android

Microsoft Windows : série d'interfaces graphiques développée et commercialisée par Microsoft, basé originellement sur MS-DOS qui était développé par IBM

BSD : réputé pour réhabiliter le rôle des serveurs, organisé par les programmeurs d'UNIX

1.2.2 Différentes familles d'OS Linux

Debian/Ubuntu : *1993*

- développé par SPI (*Software in the Public Interest*)
- caractère non commercial et mode de gouvernance coopératif
- déjà installé avec son noyau, ses pilotes, son programme d'installation de distribution, ses logiciels "utiles" (pour le WiFi, un navigateur web, etc.)
- réunit une dizaine de sous-familles Debian avec principalement les mêmes composants mais avec des différences plus ou moins mineures. Les plus connues sont : KaliLinux, Kubuntu, Raspbian, Ubuntu, Xubuntu, etc.

Red Hat/Cent OS/Fedora : *1994*

- développé et distribué par l'entreprise Red Hat (entreprise, rachetée par IBM en 2018, dédiée aux logiciels Open source + distributeur de système d'exploitation GNU/Linux)
- plusieurs distributions importantes en sont issues : Fedora, Enigma, etc.
- principalement destinées aux serveurs des entreprises
- Fedora voulait faire passer progressivement les utilisateurs Windows sous GNU/Linux

Arch : *2002*

- accent sur la simplicité et légèreté et les utilisateurs avancés
- contribution ouverte (Open Source) tant que cela respecte sa philosophie

Suse/OpenSuse : *1994*

- distribution communautaire et commerciale
- destinée à l'utilisation en entreprise mais toujours Open Source
- cycle de développement long mais cycle de vie long
- disponible à la vente (licence et mise à jour)

2 Ecrire un document texte

2.1 LaTeX

2.1.1 Histoire et Principe

LaTeX a été créé en 1983, c'est un **langage et un système de composition de documents**. Il sert principalement à de la mise en page de documents et a pour but de séparer le fond et la forme.

LaTeX est devenu le langage privilégié pour les documents scientifiques et de beaucoup de livres.

Pour rédiger du LaTeX, il faut uniquement se concentrer sur la structure logique du document et son contenu car le logiciel s'occupe de la mise en page automatiquement. Vous pouvez écrire en LaTeX grâce à différents logiciels déditeurs de texte dédiés comme TexMaker, Texlive, TeXworks, TexMacs... Dans ces éditeurs, il est possible de voir directement la mise en page en format PDF lorsqu'il est compilé. Mais il est également possible de manipuler LaTeX simplement à partir d'un terminal ou d'un éditeur de texte.

L'évolution de LaTeX est assurée par la communauté d'utilisateurs qui regroupe des étudiants et des professeurs de mathématiques ou de physique, comme des musiciens et ingénieurs en informatique notamment.

2.1.2 Différences avec Word

Il faut d'abord savoir que les deux logiciels ne se comportent et ne s'utilisent pas de la même façon.

- La mise en page (images, figures, légendes, formules mathématiques, dessins, tableaux, ...) sur Word est une rude manipulation qui fait perdre du temps. LaTeX le fait tout seul, mais son interface austère fait "peur" aux débutants. Pourtant il suffit de lui dire quel type de documents on souhaite obtenir pour obtenir quelque chose de lisible et adapté avec les normes éditoriales
- LaTeX est gratuit (et même libre), contrairement à Word
- Les formules mathématiques sont simples d'écriture sur LaTeX
- Tout est modifiable et paramétrable avec LaTeX à n'importe quel moment (par exemple si au bout de la 100^{e} page vous vous rendez compte que vous voulez changer de police, uniquement sur tous les titres)
- La gestion de documents longs est intuitive sur LaTeX, contrairement à la complexité sur Word lorsqu'il faut gérer la mise en page identique
- LaTeX peut générer automatiquement des bibliographies, tables de matières ou glossaires beaucoup plus facilement que sur Word
- Accéder à la création des PDF rapidement sur LaTeX

2.1.3 Utilisation

Il est possible de **créer ou modifier des macro-commandes** afin d'ajouter des raccourcis, par exemple pour regrouper plusieurs instructions en une seule. Comme ce langage a été créé avant le Unicode, tous les caractères peuvent s'écrire en ASCII (mais ce n'est plus nécessaire).

Il faut tout d'abord inclure les "packages" à utiliser pour déterminer la langue décriture, les marges, polices décriture, couleur et taille de lécriture, style du document, etc. Tous ces packages **peuvent être trouvés documentés sur Internet**, vous pouvez donc simplement les importer.

Voici les principales instructions qui vous serviront :

Partie : `\part{nom de la partie}`

Section : `\section{nom de la section}`

Sous-section : `\subsection{nom de la sous-section}`

Paragraphe : `\paragraph{Nom optionel} contenu du paragraphe`

Paragraphe : `\newline` ou `\\`

Liste :

```latex
% Liste à point
\begin{itemize}
    \item Contenu
    \item de la
    \item liste
\end{itemize}

% Liste numérotée
\begin{enumerate}
    \item Contenu
    \item de la liste
    \item
        \begin{enumerate}
            \item Sous
            \item liste
        \end{enumerate}
\end{enumerate}

% Liste de descriptions
\begin{description}
    \item[Mot:] description
    \item[Autre mot:] autre description
\end{description}
```

Mise en avant : `\emph{mon texte}`

Gras : `\textbf{mon texte}`

Italique : `\textit{mon texte}`

Machine à écrire : `\texttt{mon texte}`

Exposant : `\textsuperscript{mon texte}`

Pour une excellente ressource sur le LaTeX et un format plus "tutoriel'' : https://en.wikibooks.org/wiki/LaTeX.

2.2 Markdown

2.2.1 Histoire

Le Markdown est un **langage de balisage** crée en 2004. Il est facile à manipuler, donc simple à écrire et à lire sans connaître les balises. Le Markdown peut être écrit sur n'importe quel éditeur de texte, il suffit, lorsque le document est prêt a être enregistré de nommer le document avec l'extension : "**.md**".

2.2.2 Instructions

Les instructions sont **très simples et peuvent être combinées**. On va voir quelques instructions de base :

````
# Titre

## Sous-titre

### Sous-sous-titre (etc.)

Ceci est un paragraphe.
ceci fait partie du même paragraphe.

Ceci est un autre paragraphe.

- liste
- à
- puces

1. liste
2. numérotée
3. etc.

Il est possible de mettre le texte *en italique*, **en gras**,
_souligné_ ou encore ~~barré~~.

```
ceci est un extrait de code
```

```python
# Ceci est un extrait de code en Python
answer = 42
print("Hello, World: " + str(answer))
```

> Ceci est une citation
> sur plusieurs
> lignes
````

```
36
37 Les liens sont détectés automatiquement: https://ddg.gg/
38 [https://www.wikipedia.org/](Lien avec titre)
39
40 ![https://placekitten.com/400/400](Image avec description)
```

3 Terminal

3.1 Terminal

3.1.1 Quoi et pourquoi ?

Le terminal est un programme qui permet d'exécuter des commandes sous forme de texte.

Les commandes permettent en une ligne de texte d'effectuer des opérations qui peuvent s'avérer très longues avec l'interface graphique. Par exemple, modifier les droits d'accès ou d'écriture à un fichier s'effectue en une dizaine de clics avec l'interface graphique, alors que la commande `chmod` avec les droits voulus et le nom du fichier le fait instantanément.

3.1.2 Ouvrir une console et s'en servir

Pour ouvrir une console de terminal, on peut :
- chercher terminal dans la barre de recherche ;
- avec le raccourci clavier disponible sur la plupart des environnements de bureau avec Ctrl + Alt + T.

Une première ligne apparaît, et est comme ceci :
- `utilisateur@nom_du_pc:~$`

Tapez alors votre ligne de commande puis *Enter* pour l'exécuter.

Il existe de nombreux outils dans le terminal, que nous allons voir ici :

Arrêter une commande Il est possible de lancer une commande puis de l'arrêter manuellement sans attendre qu'elle se termine.

Par exemple, vous avez lancé une commande ping pour tester votre réseau. La commande ping ne s'arrête que si on lui demande. On peut alors l'interrompre avec le raccourci clavier suivant : *Ctrl + C*. Attention cependant. Même si sur la commande ping l'arrêt de la commande n'a pas d'impact, ce n'est pas le cas pour toutes les commandes. **Ctrl + C est une façon relativement brutale d'arrêter un programme.**

Copier-coller Copier-coller une ligne de commande depuis un terminal est possible, mais pas avec les raccourcis claviers classiques. En effet, Ctrl + C est déjà un raccourci clavier du terminal. Il faut donc faire *Ctrl + Shift + C* pour copier une ligne et *Ctrl + Shift + V* pour coller. Vous pouvez aussi, pour les ordinateurs en disposant, sélectionner la ligne et cliquer sur le clic du milieu (ou la molette de la souris) afin de coller la ligne.

Il faut faire attention toutefois avec le copier coller depuis les forums. En effet, si vous copiez collez une suite de commandes avec des retours à la ligne comme par exemple :
- *ls*
- *cd dossier*
- *cat fichier*

Le terminal exécutera les deux premières commandes car elles sont séparées par un retour à la ligne. Cela peut être très pratique mais aussi dangereux.

3.1.3 L'auto-complétion

Certaines lignes peuvent être longues à taper. Le terminal met à disposition une touche permettant de compléter la fin de la commande et même parfois les arguments de la commande. C'est la touche Tab. Après avoir tapé 3 lettres, vous pouvez demander l'auto-complétion. C'est le cas par exemple pour un nom de fichier très long ou de paquets. Il suffit alors de taper "cd début + Tab " et le terminal finira à votre place. Lorsque plusieurs fichiers ont le même début de nom, le terminal vous les proposera alors en dessous de votre ligne de commande.

Le manuel La plupart des commandes disposent d'un manuel, qui renseigne sur les paramètres de la commande, son utilité, ou encore comment l'utiliser. Pour ouvrir le manuel d'une commande, on tape dans le terminal `man 'nom de la commande'`.

Retrouver une commande déjà tapée précédemment Pour retrouver une commande déjà tapée, on peut cliquer sur la flèche du haut. Un clic remonte d'une commande.

De ce fait, si vous souhaitez taper une commande très longue et que vous avez déjà tapée il y a quelques temps, cliquez sur la flèche du haut autant de fois que nécessaire pour la retrouver.

3.1.4 Les droits SUDO

Pour exécuter certaines commandes, notamment installer des paquets ou redémarrer la machine, le terminal a besoin des droits super-utilisateur. Cela se fait avec la commande SUDO, qui signifie Super Utilisateur DO et qui permet "d'élever" une commande avec les accès super-utilisateur. Quiconque détient ces droits peut passer des commandes capables de modifier gravement la machine sans restrictions, donc à utiliser avec précaution.

Pour lancer une commande administrateur système, il faut taper :
`sudo nom_de_la_commande`.
Par exemple pour installer un programme sous Debian/Ubuntu :
`sudo apt install nom_du_programme`.
Le terminal va alors vous demander votre mot de passe administrateur avant de lancer la commande.

Si le mot de passe ne s'affiche pas, ni même des astérisques ou autre, c'est normal : c'est pour renforcer la sécurité car personne ne saura la taille de ce mot de passe en regardant votre écran. Attention, vous êtes le seul responsable de votre machine et lancer des commandes sudo sans être sûr de leur impact pourraient complètement détruire votre machine.

3.1.5 Les questions des commandes

Certaines commandes vous posent des questions, comme par exemple lors de l'installation d'un paquet ces questions sont de la forme :

— "`question [Y/N]`". Il faut alors taper Y (pour yes) ou N (pour no) puis entrer afin de répondre à la question.

— “`question [Y/n]`” ou “`question [y/N]`” C'est une variante dans laquelle vous pouvez toujours taper y ou n mais aussi directement entrée. La réponse prise en compte sera celle en majuscule.

Il existe un poly regroupant les principales commandes terminal disponibles sur le GitHub du club.

3.2 Bash

3.2.1 Bash et scripts

Le bash est un interpréteur de ligne de commande natif aux systèmes d'exploitation UNIX et GNU/Linux.

A quoi ça sert ?

Bash est le programme par défaut sous GNU/Linux qui exécute des commandes. Il peut être utilisé en mode interactif, comme vu à la partie sur la ligne de commande ; mais il peut aussi être utilisé pour exécuter des scripts.

Un script est tout simplement un regroupement de lignes de commandes, qui vont être exécutées lignes par lignes.

L'interprétation d'une ligne de commande Chaque interprétation d'une ligne de commande, que ce soit en mode intéractif ou dans un script, repose sur ce format :

— Le premier mot de la ligne est interprété comme le nom de la commande
— Chaque mot est séparé par un ou plusieurs caractères de séparation (espace, tabulation)
— Un retour à la ligne, ou un “`;`” si l'on veut mettre plusieurs commandes sur une seule ligne

Bash propose différents traitements des commandes :

Commandes successives : `com1 ; com2 ; ... ; comN.` Les commandes com1 à comN sont exécutées les unes après les autres

Commandes simultanées : `com1 & com2 & ... & comN`. Les commandes com1 à comN sont exécutées simultanéments

Voici un exemple de script Bash :

```
1  #!/bin/bash
2
3  echo "Enter username"
4  read username
5  echo "Enter password"
6  read password
7
8  if [[ $username == "admin" && $password == "secret" ]]; then
9      echo "valid user"
10 else
```

```
11      echo "invalid user"
12  fi
```

Pour plus d'informations sur les lignes de commandes bash, il existe un poly sur les commandes du terminal disponible sur le GitHub du club.

4 SSH

4.0.1 Définition

SSH signifie "Secure Shell", c'est un protocole sécurisé pour les communications à distance créé en 1995.

Pour rappel, un Shell va permettre de dialoguer avec une machine ou un serveur (grâce au terminal qui est une application graphique) via l'exécution de différentes commandes qui retourneront des informations. Il existe toutes sortes de Shell mais le plus utilisé reste Bash.

Un shell va donc nous permettre d'administrer nos serveurs Linux, en local, c'est-à-dire lorsque l'on se trouve physiquement en face de notre serveur, mais aussi à distance ! Notamment grâce au Secure Shell (SSH).

L'administration à distance est aujourd'hui vitale lorsque l'on gère un seul serveur, comme des milliers qui sont la plupart du temps difficiles d'accès car ils sont stockés dans des datacenter et isolés géographiquement.

Autrefois, d'autres protocoles étaient utilisés pour accéder à distance à un serveur Linux. Le protocole Telnet a pendant longtemps été utilisé, il permet également d'accéder à distance à une machine Linux, mais Telnet est aujourd'hui délaissé au profit de SSH et cela pour une raison très simple : son manque de sécurité. A l'inverse, SSH va créer un tunnel sécurisé entre la machine locale et la machine/serveur distant (il va crypter les données).

Comme dit précédemment nous utilisons SSH surtout pour faire le lien entre nos machines et nous.

4.0.2 Les Commandes De Bases

- Toutes les commandes du terminal sont applicables
- Connexion à la machine distante avec le login john :
  `ssh john@remotehost.example.com`
- Création d'une paire de clef publique/privée :
  `ssh-keygen -t rsa -b 4096 -C "your_email@example.com"`
  Cela va permettre de se connecter aux machines distantes possédant notre clef publique de manière plus sécurisée et sans mot de passe.
- Copie de la clé publique sur la machine distante :
  `ssh-copy-id -i ~/.ssh/id_dsa.pub john@remotehost.example.com`
- la commande **scp** va permettre de copier des fichiers entre la machine locale et le serveur en utilisant le protocole SSH :
  `scp my_file john@remotehost.example.com:/home/john/a_folder`
  pour un seul fichier et :

`scp -r my_folder john@remotehost.example.com:/home/john/another_folder`
pour un dossier (ajout de largument de récursivité). Il est indispensable de préciser le dossier de destination (après les ' :'). Avec cette commande il est possible de copier dans les deux sens : de notre machine au serveur, ou du serveur à notre machine, selon l'ordre des arguments de la commande.

5 Exemples de logiciels libres

Un logiciel libre est un logiciel qui est distribué sous une “licence libre” et qui fournit son code source en accord avec dite licence.

Une licence considérée comme “licence libre” doit définir ces 4 règles principales :

— Permission d'utiliser le logiciel sans restrictions
— Permission d'étudier le logiciel et son code source
— Permission de le modifier, ce qui permet de l'adapter aux besoins des utilisateurs
— Permission de redistribuer, peut-être sous certaines conditions précises

Cela a beaucoup d'avantages comme la possibilité de **corriger rapidement des bogues et des failles de sécurité**.

L'initiateur du mouvement du logiciel libre est le projet GNU. Ainsi une des licences préfaites les plus utilisées est la **licence publique générale GNU** (appelé **GPL**). Il existe cependant beaucoup d'autres licences, plus ou moins permissives sur d'autres points.

Attention : un logiciel libre n'est pas nécessairement gratuit et inversement un logiciel gratuit n'est pas forcement libre !

5.1 Git

5.1.1 Définition

Git est le logiciel de gestion de version le plus couramment utilisé dans le monde (il en existe d'autres comme SVN mais beaucoup moins utilisés), il fût développé en 2005 par Linus Torvalds (créateur du noyau Linux).

Git est un outil de versionnement, c'est à dire que lorsqu'une ou plusieurs personnes sont sur un même projet, Git va garder les différentes versions dans le temps.

Un projet est constitué de plusieurs fichiers amenés à évoluer dans le temps. Git va conserver les informations sur Qui a modifié Quoi, Quand et Pourquoi. Cet outil est donc primordial durant la conception d'un projet à plusieurs et même seul (lorsque l'on revient sur du code écrit il y a 3 mois on ne sait pas toujours à quoi il correspond).

Attention, il ne faut pas confondre Git avec GitHub et GitLab. GitHub et GitLab vont se servir de Git mais ils vont aussi rajouter une interface graphique et vont permettre le stockage sur un serveur distant. De plus il ne faut pas confondre GitHub et Gitlab, GitHub est géré par une entreprise privée qui entre temps a été rachetée par Microsoft (entreprise vouée à l'échec) et l'autre est une entreprise qui fourni aussi son logiciel, qui est partiellement Open Source.

5.1.2 Commandes de base

Voir le poly Git du club qui est sur GitHub.

5.2 F-Droid

F-Droid est un magasin d'application (au même titre que le PlayStore ou l'AppleStore) pour smartphones Android qui met à disposition plus de **12 000 applications libres et gratuites**.

Il a été créé en 2010 et est promu par la Free Software Foundation Europe. Son architecture de sécurité est basée sur le modèle de Debian. Comme toute construction Open Source, F-Droid est tenu et développé par un grand nombre de contributeurs faisant partie de la communauté. Ainsi, chaque personne peut créer sa propre application et la mettre à disposition gratuitement sur ce magasin. F-Droid assure un certain niveau de sécurité : les applications publiées sont vérifiées.

Lavantage d'utiliser F-Droid plutôt que le magasin d'application par défaut est principalement la préservation de vos données. Mais aussi la **sécurité** de vos applications et le fait qu'il n'est pas nécessaire de s'identifier pour pouvoir télécharger des nouvelles applications.

Pour un développeur d'application Android, un avantage est aussi le fait que mettre à disposition son application sur F-Droid est gratuit, contrairement au PlayStore et à l'AppleStore.

Ce magasin facilite la découverte et l'installation de multiples applications. De plus, contrairement à la plupart des applications, vous nêtes pas obligé de faire les mises à jour, et pouvez ainsi garder une ancienne version.

6 Infrastructure réseau

6.1 LDAP

6.1.1 Le LDAP, ou *Lightweigth Directory Acess Protocol*

6.1.2 Introduction

Le LDAP est un protocole créé en 1995, succédant au protocole DAP et est une base de données sous forme d'arbre. Les serveurs LDAP sont principalement utilisés dans les grandes entreprises ou écoles/universités pour enregistrer les informations sur les employés/élèves, et vérifier leurs identifiants sur différents services.

Par exemple, une entreprise peut connecter leurs ordinateurs GNU/Linux, Windows, etc. et leurs services web à un serveur LDAP pour que les employés n'aient qu'un seul identifiant et un seul mot de passe sur tous ces services.

6.2 Le LDAP du club Nix ou de l'école

Le Club*Nix et l'école possèdent chacun un serveur LDAP contenant les informations des comptes des membres/élèves, leur permettant entre autres de se connecter aux ordinateurs avec le même mot de passe.

6.2.1 Les entrées

Une base de donnée de type LDAP contient des entrées (utilisateurs la plupart du temps), qui sont des collections d'attributs. Un attribut peut être un nom d'utilisateur, un périphérique ou autres.

Comme le LDAP est sous forme arborescente (comme un système de fichier), chaque entrée peut être accédée via son "chemin", ici appelé DN pour *Distinguished Name*. Contrairement à un système de fichier, le chemin se lit de droite à gauche (au lieu de gauche à droite), et chaque "dossier" est sous la forme "`clé=valeur`".

Ainsi un exemple de DN d'une entrée serait : `uid=john,ou=Membres,dc=clubnix,dc=org`

Dans cet exemple, en faisant le parallèle avec un système de fichier, ici la racine est "`dc=clubnix,dc=org`" (un peu comme le `C:` dans un système de fichier Windows), ensuite dans le "dossier" "`ou=Membres`", enfin dans le "fichier" (ou ici l'entrée) "uid=john".

6.2.2 Chercher des informations d'un LDAP

Le protocole LDAP fournit un ensemble de fonctions permettant d'interroger le serveur sur lequel est hébergé le serveur LDAP afin de modifier, ajouter ou supprimer des entrées. On peut citer notamment *add* pour ajouter une entrée, *delete* pour la supprimer, ou *rename* pour la renommer, afin de modifier l'arborescence du LDAP.

On utilise essentiellement les LDAP grâce à des interfaces graphiques ou à d'autres logiciels qui utilisent un LDAP, mais des outils en ligne de commande, tels que `ldapadd` ou `ldapsearch` peuvent aussi être utilisés.

6.3 NFS

6.3.1 Introduction

Un NFS, ou *Network File System*, à traduire par *Système de fichiers en réseau*, est un protocole qui permet à un client à accéder à des fichiers/dossiers stockés sur un ou plusieurs serveurs. Ainsi le(s) serveur(s) stocke les fichiers et les dossiers, tandis que les clients donne l'impression d'avoir les fichiers et les dossiers présents sur la machine.

6.3.2 Le NFS du Club*Nix ou de l'ESIEE

Le Club*Nix et l'ESIEE possèdent un NFS, permettant aux postes de travail d'avoir accès aux fichiers des élèves/membres.

Le NFS transmet aussi les informations sur les utilisateurs des fichiers. Ainsi, un serveur NFS est souvent utilisé avec un serveur LDAP pour gérer les utilisateurs (c.f. section 6.1)

6.3.3 La sécurité du NFS

Il existe actuellement 4 versions de NFS. Dans les 3 premières versions, le protocole NFS n'était pas sécurisé et permettait l'accès aux fichiers de n'importe qui ayant un accès réseau au serveur. Depuis la version 4.1, le protocole NFS défini un système d'authentification et de chiffrement des données transmises sur le réseau.

6.4 Fichiers logs

6.4.1 Définition

Dans le domaine informatique, le terme "log" désigne un type de fichier, ou une entité équivalente, dont la mission principale consiste à stocker un historique des événements. Diminutif de *logging*, le terme peut être traduit en français par "journal". Le log s'apparente ainsi à un journal de bord horodaté, qui ordonne les différents événements qui se sont produits sur un ordinateur, un serveur, etc.

Ils s'avèrent utiles pour comprendre la provenance d'une erreur en cas de bug. Ils permettent également d'établir des statistiques, comme le nombre de connexions à un site Web ou à un serveur, le nombre d'échec d'authentification, etc.

S'il s'agit d'un serveur Web, un fichier log va par exemple enregistrer la date et l'heure de la tentative d'accès, l'adresse IP du client, le fichier cible, le système d'exploitation utilisé, le navigateur, la réponse du serveur à cette requête, éventuellement le type d'erreur rencontré...

Les fichiers de log peuvent contenir des informations confidentielles

6.5 Virtual Machine

6.5.1 Définitions

La Virtualisation Au sens large, la virtualisation consiste à simuler l'existence de machines informatiques dans une machine informatique. Ceci permet en particulier de diminuer les coûts d'achat de matériel informatique et de rentabiliser leur utilisation. Exemples de logiciels de virtualisation : KVM, Xen, VMware, VirtualBox...

Un Hyperviseur

- assure le contrôle du processeur et des ressources de la machine hôte
- Alloue à chaque machine virtuelle (VM) les ressources dont elle a besoin
- S'assure que ces VM n'interfèrent pas l'une avec l'autre

Une machine virtuelle

Une machine virtuelle est généralement enregistré sous la forme d'un fichier, généralement appelé image, qui se comporte comme un ordinateur réel.

Cette image contient les informations des partitions de la machine et le contenu des partitions.

Lorsque la machine virtuelle est lancée, elle est placée dans un "bac à sable" qui lisole du reste du système, de sorte que les programmes de la machine virtuelle ne peuvent ni séchapper, ni modifier l'ordinateur hôte. Cela produit un environnement idéal pour tester d'autres systèmes dexploitation, dont des versions bêta, laccès à des données infectées par des virus.

Cependant, leur principale utilisation en entreprise est pour des raisons de sécurité : plusieurs services sont hébergés sur plusieurs machine virtuelles. Ainsi s'il s'avère qu'un système est vulnérable, un attaquant aura beaucoup de mal à sortir du système de virtualisation.