

Introduction à Linux

Marius Ambayrac, 1-2 Février 2024





Marius Ambayrac

Head of Engineering @Gojob

- Introduction à Linux (2 jours)
- Introduction à Docker et à la conteneurisation (2 jours)
- Gestion de projets Informatiques (5 jours)

Plan du Cours

1. Origine et philosophie Unix / Linux
2. L'arborescence de fichier et les permissions
3. Bien se repérer dans les commandes
4. Le terminal et sa customisation
5. Exécuter des programmes
6. Passage à la pratique !

Origine et philosophie Unix / Linux

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

"Un système d'exploitation est un logiciel qui pilote les dispositifs matériels et reçoit des instructions de l'utilisateur."

Les grandes catégories de systèmes d'exploitations



Origine et philosophie Unix / Linux

Concept de distribution

Au dessus de la base Unix, plusieurs distributions ont été développés

Elles comportent des variations graphiques, de paquets installés par défaut, etc.

Les principales à citer sont :

- Debian
- Ubuntu
- Arch
- Fedora
- etc.

Origine et philosophie Unix / Linux

L'utilisation intensive de Linux pour les serveurs de déploiements

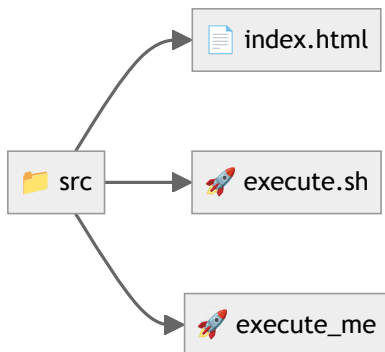
De part sa gratuité, le noyau Linux est très largement utilisé. Notamment dans sur les serveurs de déploiements

Linux est petit (avec beaucoup de versions light / small / alpine), modulaire et est donc très adéquat pour opérer des Virtuals Machines (VMs), des containers ou autres systèmes dans le Cloud / déployé.

L'arborescence de fichier et les permissions

Dossiers, fichiers, liens symboliques et exécutables

Sur les systèmes Unix, l'extension n'est pas obligatoire.



Sur les systèmes Unix, on utilise la notion de lien symbolique (= raccourci)



L'arborescence de fichier et les permissions

Les droits et permissions

d rwx r-x r-x

Type Permission_propriétaire Permission_groupe Permission_publicue

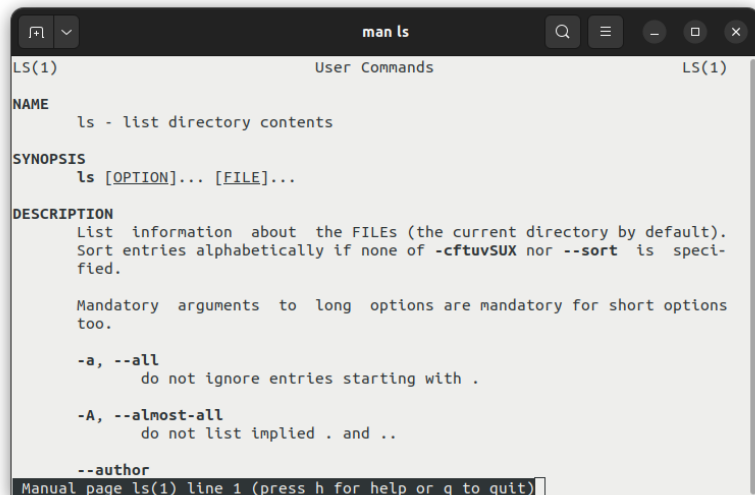
Exemple : - rwx r-- r--

⚠ ⚠ Attention, modifier les permissions de certains fichiers peut casser certains programmes (ex: clé ssh)

Documentation officielle

Bien se repérer dans les commandes

La commande "man" permet d'ouvrir le manuel d'une commande



```
man ls
LS(1)                                User Commands                                LS(1)

NAME
    ls - list directory contents

SYNOPSIS
    ls [OPTION]... [FILE]...

DESCRIPTION
    List information about the FILES (the current directory by default).
    Sort entries alphabetically if none of -cftuvSUX nor --sort is specified.

    Mandatory arguments to long options are mandatory for short options too.

    -a, --all
        do not ignore entries starting with .

    -A, --almost-all
        do not list implied . and ..

    --author
    Manual page ls(1) line 1 (press h for help or q to quit)
```

Bien se repérer dans les commandes

Le programme "tldr" permet d'ouvrir des exemples d'utilisation des commandes

```
→ ~ tldr tail
```

tail

Display the last part of a file.

See also: `head`.

More information: <https://www.gnu.org/software/coreutils/tail>.

- Show last 'count' lines in file:
`tail --lines count path/to/file`
- Print a file from a specific line number:
`tail --lines +count path/to/file`
- Print a specific count of bytes from the end of a given file:
`tail --bytes count path/to/file`
- Print the last lines of a given file and keep reading file until Ctrl + C:
`tail --follow path/to/file`
- Keep reading file until Ctrl + C, even if the file is inaccessible:
`tail --retry --follow path/to/file`
- Show last 'num' lines in 'file' and refresh every 'n' seconds:
`tail --lines count --sleep-interval seconds --follow path/to/file`

See also: `head`

Documentation tldr en ligne

Bien se repérer dans les commandes

La commande sudo

Sur chaque système Unix, pour éviter les fausses manipulations, le système propose un utilisateur Root. Cet utilisateur possède des droits très étendu (administrateur du système)

Pour accéder à cet utilisateur, si on a les droits "sudo", on peut utiliser la commande sudo et élever le niveau de privilèges

On peut ainsi lancer des commandes qui ont plus d'impact sur le système (supprimer des fichiers importants, changer des permissions, etc.)

⚠ ⚠ Attention à ne pas en abuser !

Le terminal et sa customisation




Les différents terminaux

- Bash : Standard Linux
- Zsh : Plus souple, par défaut sur Mac
- Oh-my-zsh : plugin pour Zsh

Beaucoup de paramètres sont customisables sur votre terminal. On peut ajuster l'affichage, configurer des plugins comme git, etc.

Le terminal et sa customisation

Les différents éditeurs de textes

-  Nano
-  Vi (-> Vim | NeoVim)
-  Emacs

Le terminal et sa customisation

Créer des alias

```
alias down="sudo shutdown -h now"  
alias <flag> <alias_name>="command"
```

La convention est de créer un fichier dédié contenant vos alias. Puis de l'importer dans votre .zshrc (fichier de configuration zsh) ou .bashrc

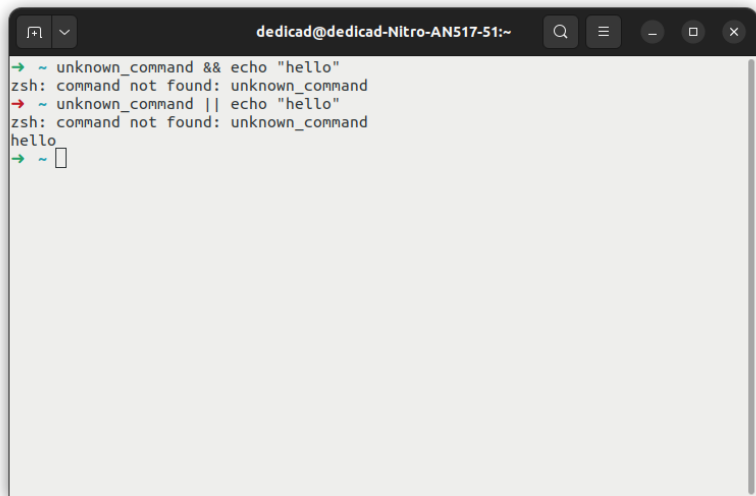
```
source ~/.zsh_aliases
```

Le terminal et sa customisation

Chainer des commandes

Pour exécuter des commandes en chaine, on peut utiliser le `&&`. Il permet de lancer la commande suivante seulement si la première commande a réussi.

L'élément `||` permet lui de lancer la commande suivante seulement si la première commande a échoué.

A terminal window titled 'dedicad@dedicad-Nitro-AN517-51:~' with standard window controls. It shows three command attempts: 1) 'unknown_command && echo "hello"' which fails with 'zsh: command not found: unknown_command'; 2) 'unknown_command || echo "hello"' which also fails with 'zsh: command not found: unknown_command'; 3) 'hello' which successfully outputs 'hello'. The prompt returns to '~' after each command.

```
dedicad@dedicad-Nitro-AN517-51:~  
→ ~ unknown_command && echo "hello"  
zsh: command not found: unknown_command  
→ ~ unknown_command || echo "hello"  
zsh: command not found: unknown_command  
hello  
→ ~
```

Exécuter des programmes

Rendre un fichier executable

Comme nous l'avions vu sur la partie des droits. Tous les fichiers en sont pas exécutables

Sur Windows, on détermine d'après l'extension (.exe) si un fichier est exécutable. Sur Linux, tous les fichiers peuvent être exécuté, et il s'agit plutôt d'une question de permissions.

```
chmod +x test.sh
```


Exécuter des programmes

Utiliser son répertoire de binaires

Finalement dans votre terminal, chaque commande est un exécutable (vous pouvez utiliser `which` pour le voir)

Pour rendre un fichier exécutable partout, et facilement depuis votre terminal, il suffit de le stocker dans `/usr/bin`

Passage à la pratique !

A. Naviguer sur son ordinateur

Dans l'ensemble des sessions pratiques, les commandes "man" et "tldr" sont vos amies !

1. Ouvrez un terminal, et utiliser une commande pour connaître le chemin de là où vous vous trouvez
2. Comment remonter dans l'arborescence d'un étage ?
3. Comment accéder en une ligne à un dossier particulier ?
4. Utilisez la commande tree pour afficher l'arborescence avec 2 niveaux de profondeurs de votre Home.
5. Comment lister la liste des fichiers, incluant les fichiers cachés dans le répertoire /usr/bin ?

Passage à la pratique !

B. Manipuler des alias

1. Installez un alias "show" qui, lorsqu'on l'utilise "show filtre" renvoie la liste des fichiers du dossier courant dont le nom contient "filtre". Par exemple "show test" doit renvoyer tous les fichier qui contiennent le mot test.
2. Utilisez la commande "alias" pour obtenir la définition de votre alias
3. Quel fichier binaire est exécuté lorsque l'on utilise la commande "shutdown" ?

Passage à la pratique !

C. Créer des fichiers

Pour quitter vim sans sauvegarder, vous pouvez utiliser `:q!` comme combinaison ou `:wq` pour sauvegarder.

1. Utilisez vim pour créer un fichier "exercice.txt"
2. Copiez collez le texte suivant dans le fichier, en utilisant le mode Insertion (touche "i") de vim puis `ctrl + v` (ou `ctrl + maj + v`)

Just like Windows, iOS, and Mac OS, Linux is an operating system. In fact, one of the most popular platforms on the planet, Android, is powered by the Linux operating system. An operating system is software that manages all of the hardware resources associated with your desktop or laptop. To put it simply, the operating system manages the communication between your software and your hardware. Without the operating system (OS), the software wouldn't function.

3. Enregistrez le fichier et quittez vim (`:wq`)
4. Depuis votre terminal, Ajouter une ligne au bout du fichier à l'aide de `>>`

Vim et Neovim sont des outils très puissants pour gérer du texte (et donc du code). Un tutoriel est intégré et permet de rapidement construire des automatismes pour ceux qui seraient intéressés ! (En [voilà un

Passage à la pratique !

C. Créer des fichiers

5. Créez un fichier "script" en bash/shell, rendez le exécutable à l'aide de la commande chmod
6. Ce script doit faire une boucle pour afficher (commande "echo") 10 lignes :
 1. Ligne 1
 2. Ligne 2
 3. etc.
7. Le script peut être utilisé de la façon suivante : "sh script"
8. Déplacez le script dans /usr/bin pour pouvoir peu importe le dossier ou vous vous trouvez
9. Modifier le script pour qu'il puisse prendre un argument et afficher le nombre de ligne en conséquence :
"script 5" ne renverra que 5 lignes.

Passage à la pratique !

D. Se repérer dans un fichier texte

1. Télécharger le fichier html suivant à l'aide de la commande curl
 1. <https://agilemanifesto.org/principles.html>
2. Utilisez la commande grep pour retourner toute les lignes contenant le mot "software"
3. Pouvez vous afficher toute les lignes les précédant ?
4. Utilisez la commande cat pour afficher l'ensemble du fichier
5. Utilisez la commande tail pour afficher les 20 dernières lignes du fichier

Passage à la pratique !

E. Manipuler des logs

Beaucoup de programmes tournent en parallèle sur des systèmes linux. Les logs sont par défaut localisés dans `/var/log`

1. Les logs systèmes sont écrites en temps réel dans le fichier `/var/log/syslog` : Affichez ces logs en temps réel dans votre terminal via la commande `tail`
2. Use command "grep" to visualize all errors in system logs
3. Dans beaucoup de systèmes les logs, ou d'autres fichiers sont au format JSON
 1. Faites un appel à l'api OpenWeather : <https://openweathermap.org/api/one-call-3> pour connaître les métriques météo d'Orléans.
 2. Stocker le résultat de cet appel dans un fichier `.json`
 3. Utilisez l'utilitaire `jq` pour extraire la vitesse du vent (`wind_speed`)
 4. Concevez un programme bash "meteo" de telle sorte qu'il affiche la vitesse du vent pour n'importe quelle latitude et longitude.

Passage à la pratique !

F. Bonus

Lors du cours, nous avons parlé du système de distribution Linux.

Sur le site suivant [distroSea](#) vous pouvez tester quelques distributions et appréhender les différences entre chacune. Vous pouvez tester Ubuntu et Arch par exemple.

- Vous pouvez également prendre de l'avance sur demain, en essayant d'utiliser une clé usb Live pour booter votre ordinateur sur Linux. N'hésitez pas à m'appeler !

Passage à la pratique !

F. Bonus n°2

1. Utiliser le "Get Started" de docker pour l'installer sur votre machine et lancer le container hello_world
2. Comment docker fonctionne-t-il ?
3. Lancer une image de docker avec la distribution Alpine ou Debian Slim
4. Accéder au shell dans le container docker et éteindre le container depuis l'intérieur
5. Rallumez le container. Transférez des fichiers entre votre machine et le container docker (qu'on appellera container-server)
6. Ajouter un serveur http qui écoute sur le port 80 et répond hello_world (langage de votre choix, vous pouvez en récupérer des déjà écrits)
7. Lancer ce serveur http dans le container
8. Utilisez docker-compose pour ajouter un second container alpine (appelé container-client)
 1. Permettez la communication sur le port 80 entre les deux containers via la configuration du docker-compose
 2. Depuis container-client, installer un script qui fait 4 appels à une adresse précise (celle de container-

Passage à la pratique !

F. Bonus n°3

1. Installer fnm et la version 20 de NodeJS
2. Installer tldr sur votre ordinateur via npm (node packet manager)
3. Créer un programme exécutable en ligne de commande (dans le langage de votre choix, pour node les outils suivants sont assez agréables mais c'est possible en C++ également) :
 1. Le programme doit faire le bootstrap d'une application web en architecture MVC (définition Wikipédia, exemple sur OpenClassrooms)
 2. Le programme doit demander à l'utilisateur le nom du projet
 3. Dans un premier temps le langage du projet est fixe (par exemple Php ou javascript)
 4. Dans un seconde temps, permettez à l'utilisateur de sélectionner dans une liste déroulante le langage qu'il souhaite (php, javascript, typescript, nodeJS, etc. à vous d'ajoutez ceux que vous pouvez !)

