



Linux

Résumé du cours de Linux en Latex

Classe X81

Ambroise - Anouar - Quentin

LATEX

Table des matières

1	MODULE : Les outils essentiels	1
1.1	Les 7 premières commandes	1
1.2	Obtenir de l'aide	1
1.2.1	Avec l'attribut -help	1
1.2.2	Avec la commande man	1
1.3	Comprendre les outils du « SHELL »	2
2	MODULE : Les outils essentiels pour la gestion des fichiers	3
2.1	L'arborescence du système de fichier	3
2.2	Lister les fichiers avec ls	4
2.3	Utiliser les SHELL wildcards	4
2.4	Copier un fichier avec la commande cp	4
2.5	Travailler avec les dossiers	4
2.6	Utiliser les chemins absolus et relatifs	5
2.7	Déplacer un fichier avec la commande mv	5
2.8	Supprimer un fichier ou un dossier avec la commande rm	5
2.9	Comprendre le Hard-Link et le Symbolic-Link	5
2.10	Créer un link avec la commande ln	6
2.11	Trouver un fichier avec la commande find	6
3	MODULE : Travailler avec des fichiers textes	7
3.1	Comprendre VIM	7
3.2	Travailler avec le pager less	8
3.3	Lire un fichier texte grâce à cat et tac	8
3.4	Lire le début ou la fin d'un fichier avec head tail	8
3.5	Travailler avec la commande grep en dehors de 	8
3.6	Comprendre les « REGEXP »	9
3.7	Utiliser les commandes awk, sort, tr	9
A	Commandes utiles	12
A.1	Section d'annexe	12

<i>TABLE DES MATIÈRES</i>	2
Épilogue	13

Chapitre 1

MODULE : Les outils essentiels

1.1 Les 7 premières commandes

- **whoaim** : Renvoie votre login-name actuel.
- **hostname** : Renvoie le nom de la machine sur laquelle vous travaillez.
- **date** : Renvoie la date actuelle.
- **uname** : Renvoie des informations sur le système actuel.
- **passwd** : Permet au user de changer son mot-de-passe et permet à l'administrateur ou le root de changer le mot-de-passe d'un user.
- **touch** : Permet la création d'un fichier vide ou la mise à jour de la date de modification d'un fichier existant.
- **last** : Renvoie la liste des utilisateurs qui se sont récemment connectés au système

1.2 Obtenir de l'aide

1.2.1 Avec l'attribut -help

Pour obtenir une aide rapide sur une commande, on utilise l'attribut -help à la suite de la commande.

1.2.2 Avec la commande man

Man est une commande qui permet d'obtenir de l'aide quant à l'utilisation, la syntaxe et les attributs des autres commandes Linux.

La commande man s'utilise avec la syntaxe suivante : man [Nø section] [nom de la commande recherchée].

Il peut arriver que man ne soit pas à jours et ne vous renvoie rien ou des informations lacunaires : dans ce cas vous pouvez mettre à jours la base de donnée de man grâce à la commande mandb.

1.3 Comprendre les outils du « SHELL »

A) Le « **TAB Completion** » : le SHELL possède la capacité de compléter vos commandes si vous tapez sur « **TAB** » et que celle-ci ne souffre d'aucune ambiguïté. Si votre commande souffre d'ambiguïté, tapez 2 x sur « **TAB** » pour obtenir une liste réduite de commande.

B) **History** : Le SHELL référence l'ensemble des commandes que vous utilisez dans la console et est capable de vous les restituer.
History référence un fichier qui conserve une trace des commandes tapées et ce de manière persistante même après reboot.

C) **Les redirections** ⇒ il existe trois canaux principaux :

- 1) **STDIN** : c'est l'entrée standard (généralement le clavier)
- 2) **STDOUT** : c'est la sortie standard (généralement l'écran)
- 3) **STDERR** : c'est le canal d'erreur (généralement vers un fichier)

Les redirections permettent de rediriger chaque canal selon nos besoins.

- Exemple : il est possible de diriger la sortie standard vers un fichier plutôt que vers l'écran.

D) **Les pipes « | »** : Le pipe permet de rediriger la sortie d'une commande dans l'entrée d'une seconde afin que la deuxième commande effectue un traitement sur le résultat de la première.

Chapitre 2

MODULE : Les outils essentiels pour la gestion des fichiers

2.1 L'arborescence du système de fichier

Cette structure peut sensiblement varier en fonction des distributions.

⇒ Mais un tronc commun est communément admis c'est le « FHS : file hierarchy standard »

⇒ Chaque arborescence de fichier en Linux prend toujours naissance avec le « root directory » ou « / »

⇒ Depuis le « / » l'arborescence se dessine autour de dossiers fondamentaux pour le fonctionnement du système.

Ce système de fichier peut être héberger sur un seul device de stockage

⇒ HDD

⇒ SSD

⇒ Etc

- Cependant, il est courant et conseillé d'isoler certains dossiers sur des devices différents.

⇒ Exemple de dossiers couramment isolé sur un autre device de stockage :
/home : parce que c'est un dossier souvent très volumineux

/var : parce que c'est un dossier pouvant saturé le système puisqu'il héberge les fichiers de type « dynamique »

⇒ Pour pouvoir réaliser cette isolation Linux se repose sur le système de « MOUNT ».

⇒ Mount permet de connecter une partie du système de fichier à un stockage physique particulier de la machine.

Le principe du mount est donc de connecter des parties du système de fichier à la représentation du système de stockage.

2.2 Lister les fichiers avec ls

- Lister les fichiers en Linux est essentiel puisque nous travaillons principalement en ligne de commandes.

ls -a : renvoie la liste de tous les fichiers et des dossiers présent dans le répertoire courant.

ls -lrt : renvoie la liste des fichiers et des dossiers classés en fonction du temps de dernière modification

2.3 Utiliser les SHELL wildcards

- Le SHELL Linux possède la capacité de globbing :

⇒ C'est à dire que le SHELL est capable d'interpréter des symboles de remplacements dans les commandes.

⇒ * : remplace plusieurs caractère inconnus.

⇒ ? : remplace un caractère inconnu.

⇒ [a-9] : remplace un caractère par un des caractères du « range » défini.

2.4 Copier un fichier avec la commande cp

- Pour copier un fichier ou un dossier d'un emplacement à l'autre dans l'arborescence de fichiers, vous devez utiliser la commande :

⇒ Pour un fichier : cp [SOURCE] [DESTINATION]

⇒ Pour un dossier : cp -R [SOURCE] [DESTINATION]

2.5 Travailler avec les dossiers

- La commande cd (change directory)

⇒ Elle permet de se déplacer dans le système. Le chemin peut être absolu ou relatif.

⇒ cd . permet de rester dans le répertoire courant.

⇒ cd .. permet de remonter dans le répertoire parent.

- La commande mkdir(make directory)⇒ permet de créer un dossier dans le système.

- La commande rmdir (remove directory) ⇒ permet de supprimer un dossier dans le système.

2.6 Utiliser les chemins absolus et relatifs

- **Un chemin absolu** est un chemin qui commence à la racine du système de fichier. Dans notre cas cette racine est « / » aussi appelé « root ».
- **Un chemin relatif** est un chemin qui commence à la position actuelle dans le système de fichier.

2.7 Déplacer un fichier avec la commande mv

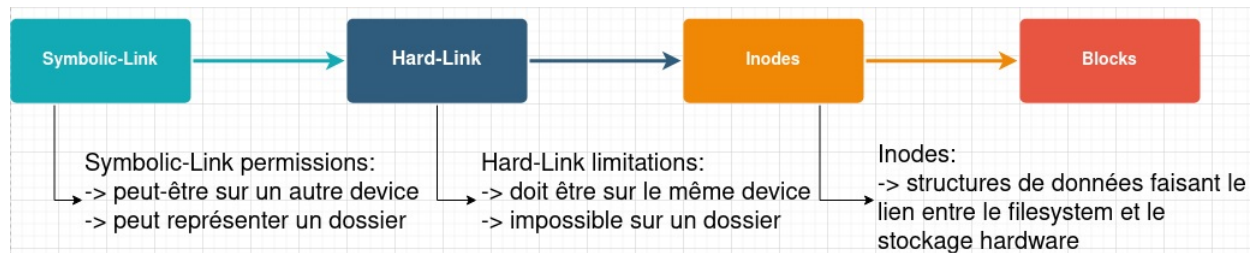
- Techniquement il est possible de déplacer un fichier avec la commande cp mais celle-ci a le désavantage de conserver une version du fichier à l'emplacement originel.
- Pour cela la commande mv a la capacité de recopier le fichier dans une autre partie de l'arborescence de fichier tout en effaçant le fichier de son emplacement originel.
- Il est à noter que d'un point de vue système, renommer un fichier revient à déplacer(mv) ou copier se fichier(cp) avec un autre nom dans une [DESTINATION == SOURCE]

2.8 Supprimer un fichier ou un dossier avec la commande rm

- La suppression de fichiers ou de dossiers en Linux se fait via la commande rm.
- ⇒ Pour supprimer un fichier : rm [SOURCE]
⇒ Pour supprimer un dossier : rm -r [SOURCE]
⇒ Pour supprimer un dossier sans confirmation : rm -rf [SOURCE]

2.9 Comprendre le Hard-Link et le Symbolic-Link

- Les systèmes Linux possèdent une caractéristique très utiles que l'on appel « Link »
 - Il existe deux types de « Link »
- ⇒ Le Hard-Link : est un nom qui référence un « inode » qui lui même référence un bloc sur le périphérique de stockage.
- ⇒ Le symbolic-Link : est un nom qui référence un Hard-link

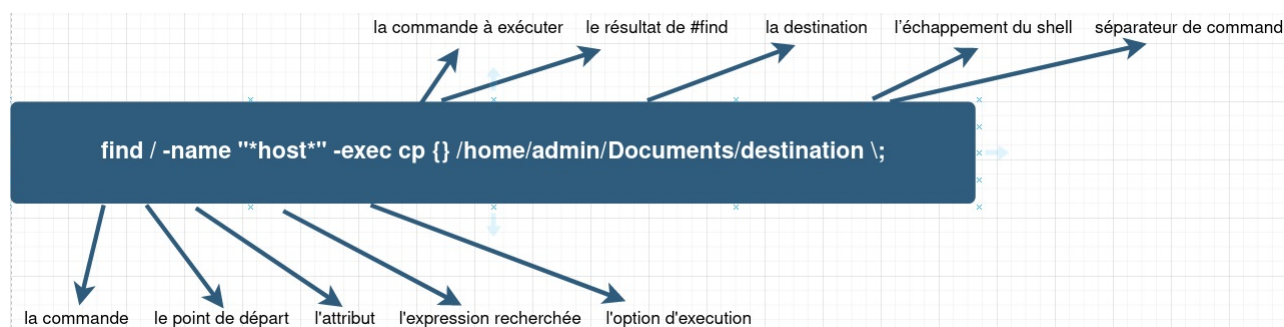


2.10 Créer un link avec la commande ln

- Pour créer un « Link » dans un système Linux, vous devez utiliser la commande `ln`
- Peut importe que vous vouliez créer un Hard-Link ou un Symbolic-Link
- `ln [SOURCE] [LINK-NAME]` : permet de créer un HARD-LINK
- `ln -s [SOURCE] [LINK-NAME]` : permet de créer un Symbolic-Link

2.11 Trouver un fichier avec la commande find

- Pour configurer certains services ou simplement parfois pour exécuter un script, il est nécessaire de trouver le fichier de configuration ou le fichier script dans le système de fichier.
 - Cela peut s'avérer difficile étant donné la quantité de fichier contenue dans un OS.
- ⇒ `find [START-POINT] -name « [FILENAME] »` : permet de chercher un fichier/dossier de manière récursive grâce à son nom à partir du point de départ.
- ⇒ `find [START-POINT] -user « [USERNAME] »` : permet de chercher tous les fichiers/dossiers appartenant à un utilisateur de manière récursive grâce au nom de l'utilisateur à partir du point de départ.



Chapitre 3

MODULE : Travailler avec des fichiers textes

3.1 Comprendre VIM

- Par tradition, il existe deux logiciels d'édition couramment utilisés pour réussir à éditer des fichiers en ligne de commande afin de ne pas devoir sortir du « SHELL »

⇒ VIM (celui que nous allons voir)

⇒ Emacs (le plus compliqué des deux à maîtriser)

- Pour éditer un fichier texte avec « VIM » vous devez utiliser la commande vim [SOURCE DU FICHIER]

- vim fonctionne sous 3 modes distinct :

- 1) Le mode « command » : permet de sauvegarder, quitter, rechercher ...
- 2) Le mode « insert » : permet d'éditer le texte ...
- 3) Le mode « visual » : permet d'effectuer des sélections dans le texte ...

- Il est très important de comprendre que chaque action ne peut se faire que dans le mode qui lui est dédié

- En fonction des modes « VIM » possède plusieurs options :

- En mode COMMANDE :

- :w (sauvegarde votre fichier)

- :q (quitte le fichier)

- :! (force l'action)

- :wq! (sauvegarde et quitte en forçant l'action)

- u (undo)

- En mode INSERT :
 - Vous pouvez simplement l'utiliser comme un éditeur de texte sans l'option copier – coller.
- En mode VISUAL :
 - D (delete la sélection)
 - Y (copie la sélection)
 - P (colle la sélection)

3.2 Travailler avec le pager less

- less [SOURCE] est une commande qui a la capacité d'organiser le texte en page pour le « SHELL ».

3.3 Lire un fichier texte grâce à cat et tac

- Certain fichiers texte sont suffisamment court pour ne pas requérir à less.
- La commande cat [SOURCE] est alors utile pour présenter le contenu du fichier dans le « SHELL »

3.4 Lire le début ou la fin d'un fichier avec head tail

- head -n[nombre de lignes] [SOURCE] : présente les 10 premières lignes du fichier dans le « SHELL »
- tail -n[nombre de lignes] [SOURCE] : présente les 10 dernières lignes du fichier dans le « SHELL »

3.5 Travailler avec la commande grep en dehors de |

- grep est une des commandes les plus utiles en Linux.
- Exemple : vous chercher tout les fichiers ou il est écrit « dhcp » dans votre système.
- grep -iR [Expression recherchée] 2>/dev/null
- -i : attribut qui rend grep case insensitive
- -R : attribut qui permet à grep de travailler de manière récursive
- 2>/dev/null : redirection des erreurs dans le /dev/null

3.6 Comprendre les « REGEXP »

- Les expressions régulières sont des modèles de texte utilisables par des outils présents dans Linux comme `grep`
 - Il ne faut pas confondre les « REGEXP » et le « globbing » Linux
- ⇒ Le « globbing » est interne au « SHELL »
- ⇒ Les « REGEXP » sont générales et utilisables par toutes les commandes qui traitent des chaînes de caractères.

Character	Definition	Example	Result
<code>^</code>	Start of a string	<code>^abc</code>	<code>abc, abcdef, abc123</code>
<code>\$</code>	End of a string	<code>abc\$</code>	<code>abc, blahabc, 456abc</code>
<code>.</code>	Any character except newline	<code>a.c</code>	<code>abc, aac, a2c</code>
<code> </code>	Alteration	<code>1 8</code>	<code>1,8</code>
<code>{...}</code>	Explicit quantity of preceding character	<code>ab{2}c</code>	<code>abbc</code>
<code>[...]</code>	Explicit set of characters to match	<code>a[bB]c</code>	<code>abc, aBc</code>
<code>(...)</code>	Group of characters	<code>(123){3}</code>	<code>123123123</code>
<code>*</code>	Null or more of the preceding character	<code>ab*c</code>	<code>ac, abc, abbbbc</code>
<code>+</code>	One or more of the preceding character	<code>ab+c</code>	<code>abc, abbbbc</code>
<code>?</code>	Null or one of the preceding character	<code>ab?c</code>	<code>ac, abc</code>

3.7 Utiliser les commandes `awk`, `sort`, `tr`

- `awk` est une commande qui permet de découper un texte en fonction de ses délimiteurs.

```
[admin@localhost ~]$ cat /home/admin/Documents/catfile
Colone1 : Colone 2
one      : 1
two      : 2
three    : 3
four     : 4
five     : 5
six      : 6
seven    : 7
eight    : 8
nine     : 9
ten      : 10
eleven   : 11
tweelke  : 12
thirdeeen: 13
fourteen : 14
fithteen : 15
sixteen  : 16
seventeen: 17
eighteen : 18
nineteen : 19
tweety   : 20
```

```
[admin@localhost ~]$ awk -F : '{print $1}' /home/admin/Documents/catfile
Colonel
one
two
three
four
five
six
seven
eight
nine
ten
eleven
twelve
thirteen
fourteen
fifteen
sixteen
seventeen
eighteen
nineteen
twenty
```

- `sort` est une commande qui permet de trier le texte dans l'ordre alphabétique ou numérique.

⇒ `sort [source]` (alphabétique)

⇒ `sort -n [source]` (numérique)

- `tr` est une commande qui permet réaliser une traduction de certains caractères du texte en d'autres

⇒ `tr [caractère(s) d'origine] [caractère(s) de remplacement]`

⇒ `tr [état d'origine] [état de remplacement]`

Annexe A

Commandes utiles

A.1 Section d'annexe

Épilogue