

Linux

Amnézic

Contents

1	Command Line Skills	3
2	Getting help	4
3	Navigating the filesystem	5
4	Managing files and directories	6
5	Working with text	7
6	Where data is stored	8
7	System and user security	9
8	Creating users and groups	10
9	Ownership and permissions	11
10	Special directories and files	13
11	Notes supplémentaires	14

1 Command Line Skills

Le shell est un *command line interpreter* et la CLI est une *command line interface*.

Le prompt est, initialement, sous cette forme là : `username@systemName:$pwd`.

Les commandes de bases : (on aura la commande avec ses principales options)

- `ls` : liste les fichiers du dossier
- `history` : liste des commandes
 - pour avoir accès à la n-ième commande : `!n`
 - pour avoir accès à la dernière commande commençant par *command* : `!command`
- `cp`
- `rm`
- `type command` indique si *command* est une commande interne (built-in) ou externe
- `which command` indique où se trouve l'exécutable de *command*

On peut créer des nouvelles variables/commandes grâce aux alias : `variable=value`. Pour trouver la valeur d'une variable, on peut :

- `echo $variable`
- `env | grep variable` (ssi la variable est globale (faire `export variable` pour la mettre globale))
- `unset variable` pour supprimer la variable

La variable *PATH* contient l'ensemble des dossiers contenant les exécutables des commandes. Pour ajouter un dossier à *PATH*, il suffit de faire `PATH=aNewPath:$PATH`.

Display variables :

- rien ou double quote " " : interprète les variables si ces dernières sont précédées par un \$
- single quote ' ' : n'interprète jamais les variables mêmes si elles sont précédées par un \$
- si on ne veut pas que la variable avec le \$ soit interprétée, il faut rajouter un devant le \$
- back quote ` ` : à n'utiliser qu'autour du nom seul de la variable qu'on veut afficher (pas de backslash ni de \$ nécessaire)

On peut exécuter plusieurs commandes à la suite grâce à :

- `;` : permet d'exécuter inconditionnellement les instructions les unes à la suite des autres
- `&&` : permet d'exécuter les instructions les unes à la suite des autres, mais si une commande renvoie une erreur, aucune des autres instructions à sa droite ne sera exécutée
- `||` : permet d'exécuter les instructions les une à la suite des autres jusqu'à ce qu'une commande ne renvoie pas d'erreur

2 Getting help

Pour obtenir de l'aide sur une *command*, il suffit de faire : `man command`. Les différentes sections des manuels sont :

1. commandes générales, qui contient les informations suivantes :
 - nom
 - synopsis
 - description
 - options
 - files
 - author
 - report bugs
 - see also
 - copyright
2. system calls
3. library calls
4. special files
5. file formats and conventions
6. games
7. miscellaneous
8. system administration commands
9. Kernel routines

Pour :

- connaître quelles pages possède `man command` : `man -f command` (== `whatis command`)
- aller à une page *n* de `man command` : `man n command`
- chercher un manuel en rapport avec une action *action* : `man -k action` (== `apropos action`)

Il est aussi possible de retrouver cette documentation dans les dossier `/usr/shar/doc` ou `usr/doc`

3 Navigating the filesystem

Pour se déplacer, on utilise la commande `cd` avec un chemin (absolu ou relatif). Pour savoir où on se trouve exactement, il suffit d'utiliser la commande *`pwd`*.

Pour lister le contenu d'un dossier, on utilise la commande *`ls`* avec ces options là :

- `-l` : long listing
- `-a` : affichage des fichiers cachés
- `-r` : ordre désalphabétique
- `-h` : human readable (taille des fichiers exprimés en o/ko/mo/go)
- `-d` : listing de dossier(s) uniquement
- `-R` : listing récursif
- `-S` : trie par ordre alphabétique les fichiers
- `-t` : liste les fichiers du plus récemment modifié au moins récemment modifié

4 Managing files and directories

Le file globbing est une manière d'englober un ensemble de combinaison de caractère bien précises. Ça se base sur le même principe que les RegEx avec les mêmes signes mais ne représentent pas forcément la même chose :

- `*` : signifie qu'il y a 0 ou + caractères (!= RegEx car il indique n'importe quel caractère)
- `?` : signifie qu'il y a exactement un caractère (`???` comprend tous les mots qui possèdent exactement 3 lettres par exemple)
-

`:` représente un ensemble de caractère :

`az`

représente tous les mots qui ne contiennent qu'un 'a' ou qu'un 'z' alors que

`a - z`

représentent tous les mots qui ne contiennent qu'une lettre entre 'a' et 'z' inclus. Les rangées se basent sur le tableau ascii donc `[a-Z]` comprendra aussi les caractères entre 'z' et 'A' dans la table ascii (accessible via la commande `ascii -d`)

- `!` ou `^` : représente la négation

5 Working with text

Pour lire un document comme si c'était un document papier (avec des pages donc), on peut utiliser la commande *less/more*. Si on veut rechercher un mot particulier, il suffit de taper : */mot* et se déplacer dans le texte en appuyant sur *n*.

Pour afficher seulement une partie d'un document, on peut utiliser les deux commandes suivantes :

- *head* (-*n* le nombre de ligne qu'on veut afficher) : affiche *n* (10 par défaut) lignes à partir du début du fichier
- *tail* (-*n* le nombre de ligne qu'on veut afficher) : affiche *n* (20 par défaut) lignes à partir de la fin du fichier

Lorsqu'on travaille avec du texte, il est très pratique d'utiliser des pipes (*|*). Ça s'utilise comme ça :

command1 | command2

Cette instruction va exécuter *command1* et va renvoyer le résultat pour être utilisé par *command2*. Une notion importante est ce qu'on appelle les channels, il en existe 3 :

- STDIN (0) : standard input, récupère les informations généralement via ce que l'utilisateur tape sur son clavier (on peut récupérer STDIN grâce à '<')
- STDOUT (1) : standard output : ce qui est affiché dans la console et qui n'est pas une erreur (on peut renvoyer STDOUT grâce à '>')
- STDERR (2) : standard error : ce qui est affiché dans la console et qui est une erreur (on peut récupérer STDERR grâce à '2>')

Il peut être intéressant de trier les données que l'on a dans un fichier. Il existe deux commandes pour aider à sélectionner les bonnes données et les trier correctement :

- *sort* :
 - -*n* : permet de trier un ensemble de données numériques dans par ordre croissant
 - -*r* : permet d'inverser le tri
 - -*kn* : permet de sélectionner uniquement la colonne *n* (*n* peut être un ensemble) (commande à utiliser nécessairement avec -*t*)
 - -*t'délimiteur'* : indique à la machine que les différents champs sont séparés par *délimiteur*
- *cut* :
 - -*cn* : indique à la machine de ne garder que la colonne (un seul caractère)*n* (*n* peut être un ensemble)
 - -*d'délimiteur'* : indique à la machine que les colonnes seront séparées par des *délimiteur* (à utiliser avec -*f*)
 - -*fn* : indique à la machine de ne garder que la section *n* (*n* peut être un ensemble)
 - -*output-delimiter='char'* : indique à la machine de retourner le résultat avec des *char* pour séparer les résultats

6 Where data is stored

Les informations sur les programmes en cours sont stockés dans le pseudo-dossier `/proc`. Les informations sur les composants électroniques sont stockés dans le dossier `/dev` ou `/sys`. Lorsque l'ordinateur démarre, il commence l'init process et lui assigne un PID (program ID) de 1, les suivants ont une valeur dans leur ordre d'exécution.

On peut voir chaque programme en cours et ses enfants et récursivement avec la commande *pstree* ou *ps -forest*. Pour afficher l'ensemble des programmes en cours, on utilise la commande *ps aux* ou *ps -ef*. Pour voir les programmes en cours d'un autre utilisateur (si on fait partie du groupe root, on peut utiliser la commande suivante : *ps -u <login>*).

On peut aussi utiliser la commande *top*:

- permet de voir en temps réel certaines informations sur les programmes en cours
- il est affiché le load average sur une période (dans l'ordre) : d'une minute, de 5 minutes et de 15 minutes. Si un des nombres est égal à 1, cela veut dire qu'il y a un coeur qui est complètement utilisé, mais pas nécessairement les autres
- il est affiché un PID qui est le dernier process à avoir été exécuté
- K PID : ferme le programme ayant pour PID PID
- R : ajuste la priorité d'un programme ($-20 \leq R \leq 19$)

Pour voir l'utilisation de la mémoire, on utilise la commande *free* (-s n pour rafraichir toutes les n secondes (n=1 par défaut)).

7 System and user security

Pour changer d'utilisateur on utilise la commande `su` :

- `su -` (root par défaut)
- `su -l <login>`
- `su -login <login>`

Ça va permettre de pouvoir effectuer des tâches que seul root peut exécuter avec la commande *sudo* `<command>`.

Le fichier `/etc/passwd` permet de trouver plusieurs informations sur tous les utilisateurs :

1. le nom
2. anciennement, le mot de passe (trouvable maintenant dans le fichier `/etc/shadow`)
3. le user ID
4. le primary group ID
5. des commentaires
6. chemin du home directory
7. chemin vers le shell

Le fichier `/etc/shadow` contient les informations sur les mots de passe des différents utilisateurs :

1. le nom d'utilisateur
2. le mot de passe : (plusieurs champs séparés par des \$)
 - (a) un nombre (généralement 6)
 - (b) le sel
 - (c) le mot de passe hashé
3. last change
4. minimum de jours
5. maximum de jours
6. warning
7. inactive
8. temps avant expiration

Pour avoir accès aux informations d'un utilisateur en particulier, on peut faire :

- groupes : `id <username>`
- mot de passe : `getent <type de données (ex: passwd)> <login>`

Pour trouver les groupes d'un utilisateur, on peut aussi aller les chercher dans le fichier `/etc/group` : la première section contient le groupe primaire de l'utilisateur et la dernière l'ensemble des groupes dont fait partie l'utilisateur.

Pour voir les utilisateurs qui sont en train d'utiliser la machine, on peut utiliser la commande *who*.

8 Creating users and groups

Commandes pour gérer les utilisateurs et les groupes :

- `groupadd <name>` : permet de créer un nouveau groupe qui aura pour nom `<name>`
 - `-g n` : associe le groupe `<name>` ou GID `n`
 - `-r : ???` (les GIDs jusqu'à 500 pour RedHat et jusqu'à 1000 pour Debian sont réservés au système)
- `groupmod <name>` : permet de modifier des informations sur le groupe `<name>`
- `groupdel <name>` : permet de supprimer le groupe `<name>`
- `useradd <username>` : permet d'ajouter l'utilisateur `<username>`
- `usermod <username>` : permet de modifier des informations de `<username>`
- `userdel <username>` : permet de supprimer `<username>` (`-r` pour supprimer toutes ses données)
- `passwd <username>` : permet de changer le mot de passe de `<username>`
- `chage [options] <username>` : permet de modifier les options `[options]` de `<username>`

```
sysadmin@localhost:~$ ls -l /tmp/filetest1
-rw-rw-r--. 1 sysadmin sysadmin 0 Oct 21 10:18 /tmp
```

9 Ownership and permissions

Pour connaître toutes les informations d'un fichier, il faut utiliser l'option `-l` de `ls`. On aura alors quelque chose qui va ressembler à ça : On a dans l'ordre :

1. si c'est un fichier classique (-), un dossier (d), un lien vers un autre dossier (l)
2. les permissions, dans l'ordre, de l'owner, du groupe <group> et des autres
3. le nom de l'owner
4. le nom du groupe <group>
5. d'autres trucs pas importants
6. le nom du fichier

Pour changer le groupe qui a le meilleur accès au fichier <file>, il suffit de faire `chgrp <group> <file>` (-R si le fichier est un dossier).

Pour changer le propriétaire d'un fichier, il faut exécuter la commande suivante : `chown <newOwner> <file>`.

Pour changer les permissions de manière générale, on utilise la commande `chmod` suivi soit de l'écriture complète (cf image ci dessous) ou de l'écriture numérique.

`chmod` $\begin{pmatrix} u \\ g \\ o \\ a \end{pmatrix} \begin{matrix} + \\ - \\ = \end{matrix} \begin{pmatrix} r \\ w \\ x \end{pmatrix}$ filename...

abbreviation	interpretation
u	user
g	group
o	other
a	all
+	add
-	remove
=	set
r	read
w	write
x	execute

Avec l'écriture numérique, il suffit de faire : `chown value1value2value3 file` avec value1,2 et 3 la valeur des permissions pour, respectivement, l'owner, le group owner et les autres, sachant que :

- r = 4
- w = 2
- x = 1

- on fait la somme des permissions de chaque "entité" et ça donne la valeur de value1, value2 et value3

Attention, pour que quelqu'un puisse modifier un document, il doit avoir la permission de modifier le document mais aussi de pouvoir aller jusqu'au dossier contenant le document à modifier.

10 Special directories and files

Revoir la partie avec SUID et SGID

Sticky bit

Le sticky bit permet d'éviter de supprimer un fichier dans un dossier partagé s'ils ne sont pas le propriétaire de ce fichier ou le root user. Pour l'ajouter ou le supprimer il suffit de taper la commande *chmod o±t* ou *chmod (0 ou 1)nnn* avec nnn les permissions déjà existantes.

Les liens

Il peut être très pratique de créer des liens dans un système de fichier (notamment pour raccourcir le nom du chemin d'un fichier très loin dans l'arborescence et/ou qu'on utilise souvent). Pour créer un lien, il suffit de taper la commande :

ln (-s pour un soft link) <src> <dst>

Il existe 2 types de liens (hard et symbolic) et chacun a ses avantages et ses inconvénients :

- hard links
 - avantage : Si le fichier de base est supprimé/corrompu, le fichier "copié" reste intact
 - inconvénient : Ne peuvent "copier" que des fichiers fichiers
- soft links
 - avantage : Peut pointé sur n'importe quel type de fichier (fichier, dossier...)
 - inconvénient : Si le fichier initial est supprimé/corrompu, le lien ne fonctionnera plus non plus

11 Notes supplémentaires

- en terme de stream, channel 2 correspond à STDERR
- les log files ne contiennent pas uniquement du texte
- When you execute the dmesg command, the system displays messages that are generated by the kernel
- FHS : Filesystem Hierarchy Standard
- The sbin directories are primarily intended to be used by the root user.
- Traditional UNIX systems allowed users to belong to how many groups? 16 mais les systèmes plus récents permettent d'en avoir jusqu'à 65000
- Which command can be used to view the /var/log/wtmp file entries? last
- The sudo command allows regular users execute commands as another user.