Le système Linux

Claude Duvallet

Université du Havre
Courriel : Claude.Duvallet@gmail.fr

Plan de la présentation

Le système Linux (1/2)

- Introduction
  - Un clone d'unix : multi-tâche, multi-utilisateur, de nombreux outils (X-Windows, architecture TCP/IP, etc.).
  - Un engouement récent et une diffusion rapide ces dernières années.
  - Un noyau de système d'exploitation : il fournit des services de base tels que la gestion des processus, de la mémoire virtuelle et des entrées/sorties.
  - De nombreux outils (logiciels libres) viennent compléter ce système : environnement de travail et de développement tels que compilateurs, éditeurs, interfaces graphiques, outils bureautiques, jeux, etc.
  - Exploités par des entreprises, le monde de l'éducation et de la recherche, il fera sans doute bientôt son apparition chez les particuliers (non informaticiens).

Le système Linux (2/2)

- Introduction (suite)
  - Il sait piloter de gros serveurs, des machines, multi-processeurs ou des ordinateurs en réseaux.
  - Il est utilisé pour la mise en place de serveurs WEB, de serveurs de mail, de SGBD, etc.
  - Il est multi-utilisateurs :
    - gestion de groupes et d'utilisateurs,
    - fichiers /etc/passwd et /etc/group
    - changement de mot de passe au moyen de passwd ou yppasswd.

Le système de fichiers (1/3)

- L'arborescence
  - Une structure arborescente où les nœuds sont des répertoires et les feuilles des fichiers ordinaires.
  - Les fichiers contiennent les données. Vu du système (noyau), il ne s'agit que d'une suite d'octets, c'est aux applications de gérer les contenus (leur forme, leur organisation, etc).
  - 3 types de fichiers :
    - **Les fichiers ordinaires** servent à mémoriser les programmes et les données des utilisateurs et du système.
    - **Les fichiers répertoires ou répertoires** contiennent des listes et des références de fichiers placés sous leur contrôle ainsi que la référence du répertoire dont il dépend (répertoire père).
    - **Les fichiers spéciaux** gèrent les périphériques, les tubes ou autres support de communication interprocessus.

Le système de fichiers (2/3)

- La désignation des fichiers
  - Chemin d'accès absolu
    - permet d'accéder à n'importe quel fichier ou répertoire de l'arborescence quelque soit le point de départ.
    - composé d'une suite de répertoires débutant et séparé par des /.
    - longueur limité à 1024 octets.
  - Chemin d'accès relatif
    - permet de désigner un répertoire ou fichier en tenant compte du répertoire courant (répertoire de travail).
    - en début de session, l'utilisateur se trouve dans son répertoire d'accueil (home).
  - Quelques exemples :

    chemin absolu : /home/duvallet/Cours/Licences
    répertoire courant : /home/duvallet
    chemin relatif : Cours/Licences

Le système de fichiers (3/3)

- Les répertoires courants dans linux :

```
/
++ /etc
++ /usr
++++/usr/bin
++++/usr/local
++++++/usr/local/bin
++++++/usr/local/etc
++ /home
++ /var
++++/var/log
++++/var/spool
++++++/var/spool/mail
++++++/var/spool/lpd
```

Les interpréteurs de commandes

- Les différents shell possibles sur les systèmes Unix
  - le Bourne-shell (*sh*) : ancêtre commun à tous les shell, encore utilisé en programmation mais pas en interactif,
  - le C-shell (*csh*) et le tc-shell (*tcsh*) sont d'origines BSD, utilisables en interactif mais non compatible avec le Bourne-shell en programmation,
  - le Korn-shell (*ksh*) est l'un des plus répandu dans le monde Unix, utilisable en interactif et compatible avec le Bourne-shell en programmation,
  - le Bash est le plus répandu sous Linux, reprends les caractéristiques du Korn-shel et du C-shell, compatible avec le Bourne-shell.
- On peut regrouper des commandes shell au sein de fichiers appelés scripts.

L'interpréteur de commandes BASH (1/3)

- Les variables shell
  - Le prompt et la variable PS1
    - \h le nom de l'ordinateur
    - \u le nom du login
    - \t l'heure courante
    - \w le chemin du répertoire de travail
    - \W le nom du répertoire de travail
    - \ ! le numéro de commande dans l'historique
    - \$ # pour l'administrateur sinon $
  - PWD : nom du répertoire courant.
  - HOME : indique le répertoire d'accueil pour l'utilisateur courant.
  - HOSTNAME : indique le nom de l'ordinateur.
  - PATH : liste les chemins d'accès concernant les commandes. Chaque chemin d'accès est séparé par ' :'.

L'interpréteur de commandes BASH (2/3)

- Affichage de la valeur d'une variable shell
  - Cela se fait en utilisant la commande `echo` suivi du nom de la variable dont on veut afficher la valeur, ce nom devant être précédé par un $.
  - Exemple : `echo $HOME` affiche la valeur de la variable HOME.
- Définition d'une variable shell
  - pour affecter une valeur à une variable on utilise simplement le signe = sans espace de part et d'autre (`nom_variable=valeur`).
  - pour déclarer une variable interne en une variable d'environnement, il faut utiliser la commande export (`export nom_variable`)
  - Exemple : `export PATH=$PATH:/home/duvallet/bin`

L'interpréteur de commandes BASH (3/3)

- Les fichiers d'initialisation permettent de configurer l'environnement et définissent les variables shell nécessaires à cet effet.
  - `/etc/profile` et `.bash_profile` sont exécutés lors de la connexion juste avant l'apparition du prompt. Le fichier /etc/profile est géré par l'administrateur alors que `.bash_profile` est géré par l'utilisateur.
  - `.bash_login` est exécuté si le fichier `.bash_profile` n'existe pas.
  - `.profile` est exécuté si `.bash_profile` et `.bash_login` n'existent pas.
  - `.bashrc` est exécuté lors que le Bash est en mode interactif et n'a pas été obtenu après une connexion.

Une aide sur les commandes

- la commande `man`

syntaxe : `man [options] nom_de_commande`

description : Elle permet d'afficher un manuel en ligne sur la commande spécifiée. Cet affichage s'effectue en mode console. La commande `q` permet de quitter le manuel.

options : `-h` : affiche l'aide concernant la commande man et donc permet notamment de savoir que `q` permet de quitter.
`-w` or `-path` : affiche le lieu où est stocké le fichier contenant le manuel de la commande.

Lister les fichiers

- la commande `ls`

syntaxe : `ls [options] [liste_de_fichiers]`

description : Affiche l'ensemble des fichiers passés en argument puis la liste des fichiers contenus dans les répertoires passés en argument. Par par défaut, elle affiche le répertoire courant ".".

options : `-l` : permet d'obtenir des informations détaillées sur chaque fichier listé (date, taille, droits, etc.).
`-a` : permet d'afficher tous les fichiers contenus dans un répertoire, y compris les fichiers commençant par un point.
`-R` : permet d'afficher récursivement le contenu des sous-répertoires.
`-color, -colour, -color=yes, -colour=yes` : permet d'afficher les fichiers en couleur selon leur type.
`-color=no, -colour=no` : annule l'affichage en couleur des fichiers.

Changer de répertoires

- la commande `cd`

syntaxe : `cd [chemin]`

description : Elle permet de se déplacer dans l'arborescence du système de fichier. Le chemin peut-être absolu ou relatif.

Exemples : `cd ..` : permet de remonter d'un cran dans l'arborescence.
`cd` ou `cd ~` : permet de se placer directement à la racine de votre répertoire de travail.
`cd /usr/local` : déplacement selon un chemin absolu.
`cd bin` : déplacement selon un chemin relatif.

Créer des répertoires

- la commande `mkdir`

syntaxe : `mkdir [options] repertoire`

description : Elle permet de créer un nouveau répertoire.

options : `-m, -mode mode` : permet de spécifier un mode lors de la création. Par défaut, le mode est 0777 moins les bits positionnés dans le umask (système ou utilisateur).
`-p, -parents` : s'assure que chaque répertoire spécifié existe et crée les répertoires parents manquants.

Manipulation des répertoires

- la commande `pwd`

syntaxe : `pwd`
description : Elle permet de connaître le répertoire courant.

- la commande `rmdir`

syntaxe : `rmdir [options] repertoires`
description : Elle permet de supprimer un plusieurs répertoires vides.
options : `-p, -parents` : efface les répertoires parents s'ils deviennent vide.

Afficher le contenu d'un fichier

- la commande `cat`

syntaxe : `cat [options] [fichier]`

description : Elle affiche sur la sortie standard le contenu de chacun des fichiers indiqués (ou le contenu de l'entrée standard si aucun nom de fichier n'est fourni, ou si le nom '-' est indiqué).

options : `-n, -number` : numérote les lignes en sortie en commençant à un.

- la commande `more` et la commande `less`

description : D'un usage similaire à la commande `cat`, elles permettent une manipulation un peu plus étendue comme un déplacement progressif dans le contenu visualisé.

Compter les éléments d'un fichier

- la commande `wc`

syntaxe : `wc [options] [fichiers]`

description : wc compte le nombre d'octets, de mots séparés par des blancs, et de lignes dans chacun des fichiers indiqués.

options : `-c, -bytes, -chars` : affiche uniquement le nombre de caractères.
`-w, -words` : affiche uniquement le nombre de mots.
`-l, -lines` : affiche uniquement le nombre de lignes (saut de lignes).

Exemples :
- `wc /etc/passwd` a pour résultat `43 73 1822 /etc/passwd`.
- `wc -l /etc/passwd` permet de connaître le nombre d'utilisateurs qui ont un compte sur la machine.

Qui est connecté ?

- la commande `who`

syntaxe : `who [options] [fichier] [am i]`

description : Si aucun argument n'est fourni, hormis d'éventuelles options, who affiche les informations suivantes pour chaque utilisateur connecté : nom de connexion, terminal, heure de connexion, nom d'hote distant, ou numéro de terminal X. Par défaut, who va chercher ses informations /var/run/utmp mais il peut aussi aller chercher ses informations dans d'autres fichiers comme /var/log/wtmp qui permet de connaître les utilisateurs qui se sont connectés.

options : `-m` : identique à who am i, elle permet d'afficher les informations relative à l'utilisateur connecté.
`-q, -count` : affiche uniquement le nombre d'utilisateurs qui sont connectés.

Supprimer des fichiers

- la commande `rm`

syntaxe : `rm [options] nom`

description : Elle efface chaque fichier spécifié et par défaut n'efface par les répertoires.

options : `-f, -force` : efface les fichiers en ignorant ceux qui n'existent pas et en ne demande de confirmation à l'utilisateur.
`-i, -interactive` : demande à l'utilisateur de confirmer chaque suppression.
`-r, -R, -recursive` : supprime récursivement les contenus des répertoires et le répertoire lui-même.

Les liens sur les fichiers

- la commande `ln`

syntaxe :
```
ln [option] source [destination]
ln [option] source repertoire
```

description : Elle permet de créer des liens entre les fichiers. Par défaut, il s'agit de lien matériel ($\neq$ symbolique). La création d'un lien matériel est une façon de donner plusieurs nom à un même fichier. Un fichier est effacé réellement que lorsque tous ses noms ont été effacés. Un lien symbolique est un petit fichier spécial qui permet de pointer vers un autre endroit du système de fichier. Par conséquent, si on supprime la source, le lien devient mort.

options : `-s` : permet de créer un lien symbolique plutôt qu'un lien matériel.
`-f, -force` : force l'écrasement du fichier destination s'il existe.
`-i, -interactive` demande confirmation avant de supprimer les fichiers de destination.

Copie de fichiers

- la commande `cp`

syntaxe :
```
cp [options] nom1_fichier nom2_fichier
cp [options] nom1_fichier nom2_repertoire
cp [options] nom1_repertoire nom2_repertoire
```

description : Elle sert à copier des fichiers et éventuellement des répertoires depuis un endroit précis vers une destination précise ou un répertoire.

options :
`-i` : interroge l'utilisateur avant de supprimer les fichiers réguliers.
`-R` : copie récursivement les répertoires et gère correctement les fichiers spéciaux.
`-f` : force l'effacement des fichiers cibles existants.
`-p` : conserve le propriétaire, le groupe, les permissions d'accès, et les horodatages du fichier original.

Déplacement de fichiers

- la commande `mv`

syntaxe :
```
mv [options] nom1_fichier nom2_fichier
mv [options] nom1_fichier nom2_repertoire
mv [options] nom1_repertoire nom2_repertoire
```

description : Elle sert à déplacer ou renommer les fichiers. Si le dernier argument est un nom de répertoire alors tous les fichiers sources seront déplacés, en conservant leur nom, vers ce répertoire sinon il déplacera le premier pour remplacer le second.

options : `-i` : interroge l'utilisateur avant de supprimer les fichiers réguliers.
`-f, -force` : écrase les fichiers de destination existants sans demander de confirmation à l'utilisateur.
`-u, -update` : ne pas déplacer un fichier régulier qui écraserait un fichier destination existant ayant une date de modification plus récente.

Compression de fichiers

- la commande `gzip`

syntaxe : `gzip [options] fichiers`

description : Elle permet de réduire la taille des fichiers (compresser) suivant le codage de Lempel-Ziv (LZ77). Lorsque c'est possible chaque fichier est remplacé par un fichier portant l'extension `.gz`.

options : `-d, -decompress, -uncompress` : permet de décompresser un fichier au format gzip. La commande `gunzip` effectue un travail similaire.

`-f, force` : force la compression ou décompression même si le fichier correspondant existe ou est cible de plusieurs liens matériels.

`-r, -recursive` : parcours la structure du répertoire récursivement. Si le nom précisé en ligne de commande est un répertoire, il compressera ou décompressera tous les fichiers présents dans les sous-répertoires.

Archivage de fichiers

- la commande `tar`

syntaxe : `tar [options] fichiers_ou_repertoires`

description : permet d'archiver un ensemble de fichiers ou répertoires, ou d'extraire le contenu d'une archive.

options : `-c, -create` : créer une nouvelle archive.

`-u, -update` : ajoute seulement les fichiers plus récents que ceux de l'archive.

`-x, -extract, -get` : extrait les fichiers contenus dans une archive.

`-f, -file F` : utilise le fichier d'archive spécifié. L'extension généralement employée pour créer un fichier d'archive est `.tar`.

`-z, -gzip` : compresse l'archive avec gzip. L'extension est alors en général `.tgz`.

`-v, -verbose` : affiche la liste des fichiers traités.

`-t, -list` : liste les fichiers contenus dans une archive.

Impression de fichiers

- la commande `lpr`

syntaxe : `lpr [options] fichier`

description : Cette commande permet d'envoyer vers l'impression son entrée standard et plus généralement le fichier qui est précisé en ligne de commande.

options : `-Pimprimante` : permet de spécifier l'imprimante à utiliser. Par défaut, `lpr` utilise l'imprimante spécifiée par le système ou la variable `PRINTER`.
`-#nombre` : désigne le nombre de copies voulues pour chaque fichier.

Travaux d'impression en cours

- la commande `lpq`

syntaxe : `lpq [options]`

description : Elle examine la file d'attente d'une imprimante et l'affiche. Cette commande permet notamment de connaître les numéros des jobs d'impression nécessaire lors de la suppression.

options : `-Pimprimante` : permet de spécifier l'imprimante à utiliser. Par défaut, `lpr` utilise l'imprimante spécifiée par le système ou la variable `PRINTER`.
`-l` : Toute l'information disponible sur un job est affichée. Normalement, l'information affichée est tronquée pour tenir sur une ligne.

Supprimer une impression

- la commande `lprm`

syntaxe : `lprm [options] [job #]`

description : Cette commande permet de supprimer un job d'impression dans la file d'attente. S'il n'y a pas de job précisé, le job actif est supprimé s'il appartient à l'utilisateur.

options : `-Pimprimante` : permet de spécifier l'imprimante à utiliser. Par défaut, `lpr` utilise l'imprimante spécifiée par le système ou la variable `PRINTER`.

`-` : Avec un - unique, `lprm` supprime tous les jobs appartenant à l'utilisateur.

`job #` : suppression du job indiqué par le numéro spécifié.

Trouver un programme ou un fichier

- la commande `which`

syntaxe : `which nom_de_program`

description : Elle prend une série de noms de programmes et affiche les chemins d'accès complets que le shell utiliserait pour les exécuter. Ceci est réalisé en simulant la recherche que le shell effectuerait dans la variable d'environnement $PATH.

- la commande `locate`

syntaxe : `locate [options] chaine`

description : Elle permet de retrouver rapidement le ou les fichier(s) correspondant à la chaîne de caractère passée en ligne de commande. Elle utilise une base de données qui est mise en jour par la commande `updatedb` en mode super-utilisateur.

options : `-i` : effectue une recherche sans se préoccuper de la case (minuscule/majuscule).

Changer les droits d'accès d'un fichier

- la commande `chmod`

syntaxe : `chmod [option] mode fichier`

description : Elle permet de changer les droits d'accès sur un fichier.

options : `-R, -recursive` : permet de modifier les droits récursivement.
`-v, -verbose` : décrire les modifications apportées

mode : `###` : # est un chiffre entre 0 et 7 représentant les droits cumulés (4=lecture, 2=écriture, 1=exécution). Le premier chiffre représente les droits pour le propriétaire, le second pour le groupe et le troisième pour tous les autres utilisateurs.
`[ugoa][+-=][rwx]` : u=propriétaire, g=groupe, o=autres utilisateurs, a=tous les utilisateurs, r=lecture, w=écriture, x=exécution.

Changer le groupe, le propriétaire d'un fichier

- la commande `chgrp`

syntaxe : `chgrp [options] groupe fichier...`
description : Elle permet de changer le groupe propriétaires des fichiers.
options : `-R, -recursive` : permet de modifier le groupe récursivement.
`-v, -verbose` : décrire les modifications apportées.

- la commande `chown`

syntaxe : `chown [options] proprietaire[:groupe] fichier...`
description : Elle permet de changer le propriétaire et le groupe de fichiers.
options : `-R, -recursive` : permet d'effectuer les modifications récursivement.
`-v, -verbose` : décrire les changements de propriétaire.

Retrouver des fichiers ou des éléments de fichiers

- la commande `grep`

syntaxe : `grep [options] motif`

description : Recherche dans les fichiers d'entrée les lignes correspondant à un certain motif.

options : `-n` : ajoute à chaque ligne de sortie un préfixe contenant son numéro dans le fichier d'entrée.
`-H` : ajoute à chaque ligne de sortie un préfixe contenant le nom de fichier d'entrée lorsqu'ils sont plusieurs.
`-r` : effectue un parcours récursif dans les répertoires.
`-e motif` : utilisé le motif indiqué.

Retrouver des fichiers

- la commande `find`

syntaxe : `find [chemins] [expression]`

description : Elle recherche des fichiers dans une hiérarchie de répertoires. Pour plus détails, consulter le manuel en ligne (`man find`)

- la commande `sed`

syntaxe : `sed [options] [fichiers]`

description : Elle permet d'effectuer des transformations sur des fichiers textes (par exemple substitution de chaînes). Pour plus détails, consulter le manuel en ligne (`man sed`)

Créer des raccourcis vers des commandes

- la commande `alias`

syntaxe : `alias raccourci='commandes'`

description : Cette commande permet de donner une équivalence entre une ligne de commande Linux et une chaîne de caractères. Il s'agit bien souvent de définir des raccourcis pour des commandes utilisées de façon répétée.

exemples :

- `alias` : permet de connaître la liste des alias déjà définis dans votre environnement.
- `alias lsl='ls -l'` : permet de raccourcir une commande souvent utilisée.
- `alias ls='ls -color=auto'` : permet de redéfinir une commande.
- `unalias ls` : permet de supprimer un alias.
- `alias lsl='ls -al | more'` : permet d'afficher la liste détaillée des fichiers contenus dans le répertoire courant page par page.

Accès aux périphériques

- la commande `mount`

syntaxe : `mount [options] périphérique répertoire`

description : Pour pouvoir accéder à un périphérique (de stockage), il est nécessaire de l'associer à un répertoire.

options : `-t type` : permet de spécifier le type de système de fichiers du périphérique (vfat, ntfs, ext2, reiserfs, ext3, iso9660, auto, etc.)

exemples :
- `mount -t vfat /dev/fd0 /mnt/floppy` : permet d'accéder par la suite au lecteur de disquette.
- `mount -t ext3 /dev/hda2 /home` permet d'associer la seconde partition de votre disque dur au répertoire home.

remarques : les montages les plus courants et ceux du système sont spécifiés dans le fichier /etc/fstab.

Se connecter à distance en mode sécurisé

- la commande `ssh`

syntaxe : `ssh [options] -l user serveur`
`ssh [options] user@hostname [commande]`

description : Cette commande permet de se connecter sur un serveur distant en utilisant un mode sécurisé.

options : `-p port` : permet de spécifier un port (par défaut le port 22 est utilisé.
`-x` : désactive X11.
`-X` : permet l'utilisation de X11 à travers la connexion distante.

exemples : `ssh -X duvallet@etoile`

Copier des fichiers en mode sécurisé

- la commande `scp`

syntaxe : `scp [options] [[user]@serveur1:]sources [[user]@serveur2:]destination`

description : Cette commande est une version étendue de la commande cp. Elle permet de copier des fichiers depuis ou vers un répertoire situé sur une machine distante.

options : `-r` : copie récursivement les répertoires.

exemples : `scp duvallet@etoile:oracle .`

Connexions sécurisées (1/3)

- Très souvent, on se connecte sur les mêmes machines. Il serait donc intéressant de passer plus rapidement la phase d'authentification (demande du mot de passe).
- Avoir une connexion sécurisée signifie que vos mots de passe ne seront plus transmis sur le réseau de façon lisible.
- Le système RSA implanté dans SSH repose sur une authentification avec deux clefs (clef publique/clef privée).
  - La clef publique est connue de tous et peut être transmise sur le réseau.
  - La clef privée est connue de vous seul et reste sur votre machine. Elle permettra d'encoder vos données et de s'assurer par conséquent que ces données proviennent bien de vous.

Connexions sécurisées (2/3)

1. Génération d'une paire de clefs
   - `ssh-keygen -t dsa` : taper une longue phrase.
   - deux fichiers dans `.ssh` :
     - la clef privée : id_dsa,
     - la clef publique : id_dsa.pub.
2. Copie de la clef publique sur le serveur distant.
   - `scp .ssh/id_dsa.pub duvallet@etoile:.ssh/authorized_keys`
   - S'assurer de l'existence du répertoire .ssh sur la machine distante.

Connexions sécurisées (3/3)

3. Gestion des connexions par ssh-agent :
   - `eval 'ssh-agent'`
     - Résultat :

       ```
       SSH_AUTH_SOCK=/tmp/ssh-XXGL8p9M/agent.2495;
       export SSH_AUTH_SOCK;
       SSH_AGENT_PID=2496; export SSH_AGENT_PID;
       echo Agent pid 2496;
       ```
   - `ssh-add`
     - Ajout de la clef dans le gestionnaire (ssh-agent).

La redirection des entrées-sorties (1/3)

- Chaque processus ouvre d'office :
  - une entrée standard (par défaut le clavier),
  - une sortie standard (par défaut l'écran),
  - une sortie erreur standard (par défaut l'écran).
- Chaque entrées-sorties standard peut être redirigée vers un fichier, un tube, un périphérique.
  - La redirection de la sortie standard consiste à renvoyer le texte qui devrait apparaître à l'écran vers un fichier par exemple,
  - La redirection de l'entrée standard revient à prendre les données depuis une autre source que le clavier comme par exemple un fichier (utilisation du caractère <). Le fichier doit exister.

La redirection des entrées-sorties (2/3)

- La redirection des sorties vers un fichier peut être réalisé deux façons :
  - en mode ajout (redirection de la sortie standard avec les caractères $>>$, de la sortie erreur standard avec 2$>>$), si le fichier n'existe pas, il est crée,
  - en mode création, si le fichier existe il est supprimé (redirection de la sortie standard avec les caractères >, de la sortie erreur standard avec 2>).

La redirection des entrées-sorties (3/3)

- Exemples :
  - `ls -l > temp.txt`
  - `ls -l >> temp.txt`
  - `ls /toto 2> erreur.txt`
  - `ls /toto >temp.txt 2> erreur.txt`
  - `cat temp.txt > temp2.txt`
  - `cat <temp.txt > temp2.txt`
  - `cat temp1.txt temp2.txt > temp.txt`
  - `cat temp2.txt >> temp1.txt`

Les tubes de communication et les filtres

- Un **tube** (**pipe** en anglais) est un flot de données qui permet de relier la sortie standard d'une commande à l'entrée standard d'une autre.
- Dans une ligne de commande le tube est formalisé par la barre verticale | que l'on place entre les commandes : P1 | P2 | P3
- Quelques exemples :
  - '`ls -l | more`' : affiche page par page le contenu du répertoire.
  - '`ls -l | grep "rwxr-xr-x"| more`' affiche page par page le contenu du répertoire dans les droits sont `rwxr-xr-x`
  - '`who | wc -l`' : compte le nombre de personnes connectés.
  - '`ls | wc -w`' : compte le nombre de fichiers contenus dans le répertoire.