SOMMAIRE

Une session de travail

Démarrage d'une session de travail

Le login permet à l'utilisateur de démarrer une session de travail.

L'entrée dans la session nécessite la saisie d'un nom d'utilisateur (username) et d'un mot de passe (password) pour l'authentifier.

Exemple:
login: toto
passwd: passToto

NB: le mot de passe est saisi en mode aveugle pour en assurer la confidentialité, car c'est lui qui protège l'accès au compte.

```
Debien Linux release 7.0 for x64

Kernel 3.6.17 on an i886

fcs login: root

Password: ******

Last login: Mon Aug 24 17:16:53 on tty2

[root@fcs]:/root #
```

La session de travail

Un prompt (en général le caractère #, ou $) indique que la connexion de l'utilisateur est établie. L'utilisateur bénéficie alors des ressources auxquels ses droits lui donnent accès, qu'il partage avec les autres utilisateurs de la machine. L'utilisateur n'a pas un accès direct aux commandes du système, mais passe par un interpréteur de commandes (shell) qui interprète et exécute séquentiellement les commandes entrées par l'utilisateur. Les principaux interpréteurs de commande sous UNIX sont le C-shell, le Bourn shell et le Korn shell.

A chaque début de session, des fichiers d'initialisation créent l'environnement de l'utilisateur nouvellement connecté. Ces fichiers sont des fichiers de type texte que l'on peut éditer. Ils auront une portée sur toute la session de travail courante (initialisation du terminal, protection des fichiers ...). Ces fichiers sont interprétés et exécutés ligne après ligne par l'interpréteur utilisé. Les fichiers d'environnement exécutés se trouvent dans le répertoire principal (home directory) de l'utilisateur courant.

Déconnexion

La procédure de déconnexion dépend du shell employé, mais en général la commande exit permet de se déconnecter de n'importe quel shell.

Un prompt de connexion tel que ¨login:¨ apparaîtra à l'écran, indiquant ainsi que vous êtes déconnecté et qu'un autre utilisateur peut à nouveau se connecter.

Une fois déconnecté, c'est-à-dire lorsque le prompt apparaît, vous pouvez éteindre l'ECRAN (pas la machine).

Attention :

Se déconnecter n'est pas équivalent à éteindre le système Linux. Quand je me déconnecte, j'indique que moi et moi seul ai fini. Linux continue toujours de fonctionner, peut-être pour d'autres utilisateurs connectés sur la même machine.

Eteindre le système stoppe également Linux et interrompt toutes les connexions courantes. Pour cela, il faut suivre une séquence spéciale d'arrêt, à laquelle seuls les utilisateurs privilégiés ont accès, avant d'éteindre le système Linux. Sinon on peut risquer de perdre la totalité du système (ce sont les utilisateurs qui vont être ravis !).

Découvrir les commandes de base

1. Loguez-vous sous votre nom d'utilisateur

2. Tapez votre mot de passe.

-> Vous êtes sous Linux

3. A l'aide de l'interface graphique, tapez les commandes suivantes, que signifient-elles ?

ls : elle donne l'accès au dossier Fichiers

ls -l : elle liste les éléments du dossier ``Fichier`` en détails

pwd : elle indique ou on se trouve

cd / : emmène a la racine ‘’/’’

cd /home : on se déplace dans le dossier /home

cd .. : on revient au dossier précèdent

cd : emmène au dossier personnel (quel que soit l`arborescence)

echo bonjour : affiche bonjour

clear : efface les commandes à l' écran

date : affiche la date et l' heure

whoami : affiche l' utilisateur

who : affiche les users connecté à la machine (users en ligne)

man ls : Affiche le manuel de la commande

mkdir : Créer des dossiers

rmdir : pour supprimer des dossiers

Présentation du *bash*

Le bash est le nom d'un des programmes exécutés sous Linux, il est spécifique car il sert d'interface utilisateur. L'utilisateur communique avec le système à travers lui. Il formule ses demandes en les tapant à la suite du prompt. La saisie est une chaine de caractères, terminée par la touche 'Entrée'

Le programme bash prépare le prompt et attend une saisie. Que se passe-t-il lorsque la saisie est validée par la touche 'Entrée' ? Comment fonctionne le programme bash ?

Afin que la saisie soit correctement validée, certaines règles doivent être respectées :

- Le premier mot de la ligne de commande est la commande

ex : cal

Si vous tapez **cal date**, le bash interprète la ligne de commande comme l'appel de la commande cal, le mot date n'est pas compris en tant que commande, mais en tant qu'argument donné à la commande cal, qui, ne sachant pas comment le traiter, le bash émet un message d'erreur. Pour reconnaître le premier mot de la ligne de commande, la programme bash cherche un caractère de séparation, c'est à dire un espace ou une tabulation. Les caractères précédant le premier espace sont la commande, les caractères suivant désignent les arguments, et éventuellement les options

Utilisation du système de fichier

L'un des travaux les plus importants effectués par l'administrateur système est la gestion des données. La structure des données reste la même que sous Windows, sous forme d'arborescence : Les données sont enregistrées dans des fichiers qui eux mêmes sont rangés dans des répertoires ou dossiers. Nous voulons d'abord savoir dans quel répertoire nous nous trouvons, pour cela il existe la commande **pwd**, qui est l'abréviation de print working directory, indiquer le nom du répertoire courant. Si vous avez ouvert une session sous l'utilisateur *user*, **pwd** affiche probablement :

/home/user

Après l'ouverture d'une session, le répertoire home est le répertoire par défaut, cela est logique puisque l'utilisateur peut y travailler sans restrictions.

Pour passer dans un autre répertoire, il existe la commande **cd** (change directory), dont la syntaxe est :

cd [nouveau répertoire]

ex : cd /dev

puis : pwd

Nous sommes bien passés dans le répertoire /dev

puis : cd

Nous sommes revenu dans notre répertoire home. Cette dernière utilisation est très pratique dans le cas ou vous vous perdriez dans l'arborescence, un simple cd vous fait revenir dans votre répertoire personnel.

Pour vérifier le contenu d'un répertoire, il existe la commande ls, équivalente au dir du MSDOS, abréviation de list files.

Passons dans un autre répertoire :

cd /

puis : ls

Voici la liste des répertoires contenus sous la racine, nous en connaissons quelques-uns

puis : cd

Nous sommes dans notre répertoire home. Comment retrouver la liste des repertoires de la racine ?

man ls

Qui est le manuel de la commande ls. Nous pouvons ainsi remarquer que la syntaxe de la commande ls est : ls [-option] [liste de chemins]

Nous taperons donc : ls /

Chemins absolus

Linux distingue chemins absolus et chemins relatifs. Un chemin absolu commence par la racine et contient les noms des répertoires qui sont situés entre la racine et le répertoire

Le chemin absolu du répertoire correspondant à votre bureau est par exemple :

/home/user/Desktop

Chemins relatifs

Les noms de chemins relatifs s'appuient sur les noms absolus. Ils sont relatifs parcequ'au lieu de prendre comme point de référence la racine, ils prennent le répertoire courant.

Supposons que nous soyons dans notre répertoire home, le chemin relatif du répertoire de notre bureau est

Desktop

Une règle à retenir : les noms absolus commencent toujours par /, et les noms relatifs jamais.

Autre exemple, je me place dans le répertoire /home

- cd /home

je dispose de 3 moyens pour aller dans mon répertoire personnel

- cd
- cd /home/user (chemin absolu)
- cd user (chemin relatif)

Liste non exhaustive des commandes de base pour Linux

Attention :

Sous Linux les minuscules et les majuscules ne sont pas équivalentes.

Gestion des fichiers :

• *pwd (print working directory) :* Afficher le répertoire courant .

• *cd .. :* Remonter d'un répertoire dans l'arborescence .

• *cd :* Se placer dans le répertoire de l'utilisateur .

• *cd chemin_répertoire :* Aller dans le répertoire désiré .

• *mkdir nom_répertoire :* Créer un répertoire dans le répertoire courant .

• *mkdir -p chemin_répertoire :* Créer un répertoire ainsi que ses répertoires parents à partir du répertoire courant .

• *rmdir nom_répertoire :* Supprimer un répertoire vide .

• *rm nom_fichier :* Supprimer un ficher .

• *rm -Rvf nom_répertoire :* Supprimer tous les fichiers et répertoires du répertoire sans demande de confirmation et éventuellement le répertoire s'il est vide

• *touch nom_fichier* : Créer un fichier texte vide.

• *ls :* Lister les fichiers du répertoire courant .

• *ls -a :* Lister tous les fichiers du répertoire courant .

• *ls -l :* Lister les fichiers du répertoire courant en affichant les attributs des fichiers .

• *mv source destination :* Renommer ou déplacer un fichier .

• *cp fichier_source fichier_destination :* Copier un fichier .

• *cp -Rvf source destination :* Copier de manière récursive un répertoire et ses sous répertoires .

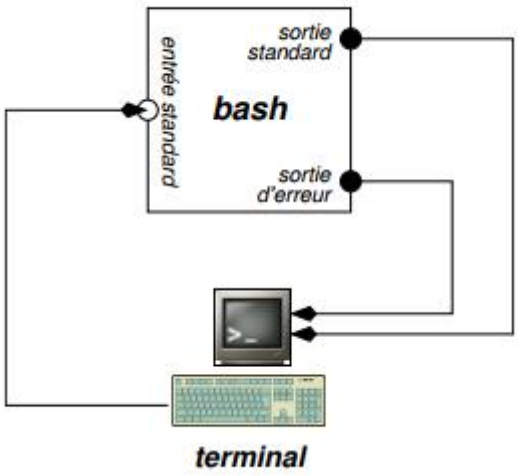
• *ln -s fichier_source fichier_destination :* Créer un lien symbolique entre deux fichiers

• *cat fichier :* Afficher le contenu d'un fichier texte à l'écran

• *cat -n fichier :* Afficher le contenu d'un fichier texte à l'écran en numérotant les lignes

• *more nom_fichier :* Afficher le contenu d'un fichier texte à l'écran

• *less nom_ficher :* Afficher écran par écran le contenu d'un fichier texte



• *commande_complète* > *nom_fichier* : Enregistrer la sortie standard dans un fichier

Exemple : la commande ls > save.txt (Enregistre la liste des fichiers et dossiers du répertoire courant dans un fichier texte créé dans le répertoire courant.)

• *type* *nom_fichier* *:* Afficher le chemin complet du fichier dans l'arborescence du système

Gestion des droits

-chgrp nom_groupe nom_fichier : Changer le groupe d'appartenance du fichier

-chown nom_utilisateur nom_fichier : Changer le propriétaire du fichier

-chown user.groupe fichier : Changer le propriétaire et le groupe d'appartenance du fichier

-chown -Rvf user.groupe répertoire : Changer le propriétaire et le groupe d'appartenance du répertoire, de ses fichiers, de ses sous-répertoires,...

-chmod options nom_fichier : Changer les permissions du fichier

La syntaxe des options de chmod est la suivante : appartenance +/-permissions, cela permet d'attribuer (le +) ou de retirer (le -) les permissions choisies au niveau d'appartenance du fichier.

Les niveaux d'appartenance :

u (user) : attribuer les permissions choisies seulement au propriétaire du fichier.

g (group) : attribuer les permissions choisies seulement au groupe du fichier.

o (other) : attribuer les permissions choisies seulement aux autres.

Les permissions :

r (read) : autoriser ou non à la lecture

w (write) : autoriser ou non à l'écriture

x : autoriser ou non à l'exécution

On peut bien sûr combiner toutes les options : chmod ug +rw nom_fichier

autorise à la lecture et à l'écriture du fichier choisi pour le propriétaire et le groupe d'appartenance du fichier.

Par défaut, si on ne signifie aucun niveau d'appartenance, les permissions seront accordées ou retirées à tous les niveaux d'appartenance.

Gestion du système

• *df (disk free) :* Afficher l'espace libre disponible sur toutes les partitions montées

• *df -h :* Afficher l'espace libre disponible sur toutes les partitions montées en KB, MB, GB

• *du (disk usage) :* Afficher la taille de tous les répertoires et sous-répertoires du répertoire courant (ne donne aucune information sur les fichiers)

• *du nom_répertoire :* Afficher le taille du répertoire et de tous ses sous-répertoires (ne donne aucune information sur les fichiers)

• *du -s nom_répertoire:* Afficher la taille du répertoire

• *nom_commande &* : Exécuter et mettre l'application en tâche de fond

-ps : Afficher la liste des processus de l'utilisateur en cours

-ps a : Afficher la liste complète des processus en cours

-ps x : Afficher la liste des processus en cours en prenant en compte ceux ne dépendant d'aucun terminal

-ps u : Afficher la liste des processus en cours en donnant leur apartenance utilisateur

-pstree : Afficher les processus sous forme d'arborescense

-top : Afficher l'état du système et des processus, affiche les plus gourmands en temps de calcul

-kill numéro_pid : Tuer le processus correspondant (pid : processus identity)

-killall nom_processus : Tuer tous les processus portant ce nom

• su - : S'approprier provisoirement l'identité de l'admin

• *su nom_utilisateur :* S'approprier provisoirement l'identité d'un utilisateur

• *useradd nom_utilisateur :* Création d'un compte utilisateur

• adduser : même chose mais simplifié

-passwd nom_utilisateur : Définir un mot de passe pour l'utilisateur

-userdel -r nom_utilisateur : Supprimer un compte ainsi que son répertoire personnel

-find / -name fichier -print : Rechercher dans l'arborescense un fichier et afficher son emplacement

-find / -type d -name répertoire -print : Rechercher un répertoire dans l'arborescence et afficher le résultat

• *grep mot nom_fichier :* Rechercher dans un fichier texte le mot voulu

• *grep -n mot nom_fichier :* Rechercher dans un fichier texte le mot voulu en donnant le numéro de ligne où il se trouve

• *mount -t type nom_device chemin_montage :* Monter le périphérique désiré dans le répertoire désiré au format adéquat (le chemin de montage doit exister)

-umount chemin_montage : Démonter le périphérique monté dans ce chemin

-shutdown -r now : Rebooter le système maintenant

poweroff : Arrêter le système maintenant

Exemple : shutdown -r +10 : Reboote le système dans 10 minutes

-who -u : Connaître les utilisateurs ayant ouvert une session

-whoami : Afficher les renseignements en rapport avec sa propre session

-cal : Afficher le calendrier du mois courant

-cal -y : Afficher le calendrier de l'année

-cal numéro_mois année : Afficher le calendrier du mois et de l'année spécifiés

Exemple : cal 12 1999 : affiche le calendrier du mois de décembre 1999

• *date :* Afficher la date du systèmels-

• *echo texte :* Afficher, envoyer vers le terminal l'argument texte

• *man nom_commande :* Afficher la page de manuel de la commande

• *clear :* Effacer l'écran

-set : Afficher les variables d'environnement

-history : Afficher l'historique des commandes utilisées

Gestion des packages et fichiers compressé

gzip :

Les fichiers compressés avec gzip se présentent sous la forme suivante : nom_fichier.gz

- Pour compresser un fichier avec gzip : *gzip nom_fichier*
- Pour décompresser un fichier avec gzip : *gunzip nom_fichier.gz*

Attention : Dans les deux cas, le fichier initial est supprimé. Dans le premier cas, un fichier fichier.gz est créé dans le répertoire courant, et dans le deuxième le fichier nom_fichier est créé. On ne peut compresser que des fichiers avec gzip. Pour les répertoires, il faut archiver avant.

- *gzip -l nom_fichier.gz :* Afficher des informations sur le fichier compressé : taux de compression,...

bzip2 :

Les fichiers compressés avec bzip2 se présentent sous la forme suivante : nom_fichier.bz2

-Pour compresser un fichier avec bzip2 : *bzip2 nom_fichier*

-Pour décompresser un ficher avec bzip2 : *bunzip2 nom_fichier.bz2*

Les mêmes remarques que celles pour gzip sont valables.

tar :

Les archives tar se présentent sous la forme : nom_archive.tar . Si, de plus, l'archive est compressée elle sera sous la forme : nom_archive.tar.gz, nom_archive.tgz pour une compression avec gzip ou nom_archive.tar.bz2 pour une compression avec bzip2.

Création d'archive tar :

- *tar cfv archive.tar fichier :* Créer une archive classique du ficher/répertoire désiré dans le répertoire courant.
- *tar zcfv archive.tar.gz fichier :* Créer une archive compressée avec gzip du ficher/répertoire désiré dans le répertoire courant.

Attention : le propriétaire des fichiers dans l'archive sera l'utilisateur ayant lancé tar (sauf si on est root, les permissions sont alors conservées).

Extraction d'archive tar :

-*tar xvf archive.tar :* Extraire l'archive dans le répertoire courant.

-*tar zxvf archive.tar.gz :* Extraire l'archive compressée avec gzip dans le répertoire courant.

-*tar zxvf archive.tar.bz2 :* Extraire l'archive compressée avec bzip2 dans le rép courant.

<u>Syntaxe générale de tar</u> : tar fonction option fichier1 fichier2 ....

-Les valeurs du paramètre fonction :

c : créer une nouvelle archive

x : extraire des fichiers d'une archive

t : afficher le contenu d'une archive

r : rajouter des fichiers à une archive

u : mettre à jour les fichiers d'une archive

d : comparer les fichiers sur le disque dur et à ceux de l'archive

-Les valeurs du paramètre option :

v : afficher plus d'informations

k : conserver les fichiers existants avant extraction

f : spécifier un nom_fichier d'archive à lire ou à écrire

z : option de compression ou décompression avec gzip des archives

<u>Remarque</u> : l'ordre des fonctions ou options n'importe pas.

-Voici quelques exemples d'utilisation :

*-tar cvf nom_archive.tar * :* archiver les fichiers du répertoire courant.

-tar tvf nom_archive.tar : lister le contenu de l'archive.

-tar xvf archive.tar fic1 ficr2 ... : extraire de l'archive que les fichiers : fic1, fic2,... (répertoire également).

Par exemple, taper tar xvf home.tar /home/nom_utilisateur pour extraire son répertoire utilisateur de l'archive home.tar (en supposant bien sûr que home.tar soit l'archive du répertoire /home).

-tar zcvf archive.tar.gz fic1 ficr2 .: créer une archive "gzippée" contenant fic1 fic2 ..

Par exemple, tar zcvf maisons.tar.gz /root /home permet d'archiver les répertoires de tous les utilisateurs du système, root compris.

<u>Remarque</u> : la commande tar a été créée à l'origine pour faire des sauvegardes sur bandes, on peut donc mettre directement ses archives sur un support amovible comme un lecteur de bande, un lecteur de disquettes, un lecteur Zip,.... en tapant tar zcvf nom_device fichier1 fichier2 ...

Par exemple, tar zcvf /dev/fd0 /home archive le répertoire /home sur la disquette du premier lecteur. L'extraction se fait de la même manière : tar zxvf /dev/fd0 .

La commande tar fourmille d'autres possibilités, consulter sa page de man pour en faire le tour.

La gestion des paquets

La gestion des paquets sous Debian se fait par l'outil APT (Advanced Packaging Tool). Les dépots sont renseignés dans le fichier /etc/source.list

• apt update : permet simplement d'actualiser la liste des paquets disponibles pour votre système

• apt upgrade : Installer les dernières versions d'un paquet (doit toujours être précédé de la commande apt update)

• apt install <paquet> : Installer un nouveau paquet (Paquet provenant d'un dépôt du sources.list) <paquet> correspondant au nom exact du logiciel que l'on veut installer. On peut installer plusieurs paquets en même temps, en séparant simplement le nom des paquets par un espace.

• apt install <paquet1> <paquet2> <paquet3> (la commande install installera automatiquement toutes les dépendances nécessaires.)

• apt remove <paquet> : Supprimer un paquet (la commande supprimera également les dépendances qui ne sont plus nécessaires.)

• apt purge <paquet> : Supprimer un paquet et les fichiers de configuration

Editer le fichier sources.list

Il peut arriver de devoir modifier son fichier sources.list pour ajouter des paquets non-libres ou appartenant à des dépôts tiers. Cette commande permet de bénéficier de la coloration syntaxique et fournit des vérifications de sécurité de base.

apt edit-sources

Lors du premier lancement, vous pourrez choisir l'éditeur à utiliser.

Select an editor. To change later, run 'select-editor'.
  1. /bin/nano   <---- easiest
  2. /usr/bin/vim.tiny

Choose 1-2 [1]:
Si vous ne savez pas lequel choisir, utilisez nano

• apt list –upgradable : Connaître les paquets qui seront mis à jour

• apt show <paquet> : Trouver des informations sur un paquet

• apt list --all-versions <paquet> : Trouver les versions disponibles d'un logiciel

• apt search <terme> : Rechercher un paquet

• apt full-upgrade : Mise à jour pour l'ensemble du système

La commande full-upgrade remplit la même fonction que upgrade mais peut aussi supprimer des paquets installés si cela est nécessaire pour résoudre un conflit entre des paquets.

Les pipes ou tubes

C'est en fait une fonctionnalité très intéressante du shell bash, celà permet de passer le résultat d'une commande à une autre commande. Ce que j'appelle "résultat" d'une commande, c'est en fait la sortie standard de celle-ci (par exemple, la sortie standard de la commande ls, c'est la liste des fichiers du répertoire courant).

La syntaxe générale d'un tube ou pipe est la suivante :

commande1 | commande2 | commande3 | ... | commandeN

Le résultat de la commande1 est passé à la commande2, ce résultat est ensuite passé à la commande3 et ainsi de suite jusqu'à la commande N. De cette manière, on a en quelque sorte connecté les N commandes. Cette fonctionnalité est très utile et permet de rendre plus facile un certain nombre de tâches d'administration sur Linux.

Pour bien comprendre l'utilité des pipes ou tubes, quelques exemples d'utilisation :

-*ls -la | grep mot :* celà permet de chercher dans la liste des fichiers du répertoire courant les fichiers contenant mot.

-*ps aux | grep mot :* ce pipe permet d'afficher la liste des processus contenant le mot.

Ces petits exemples vous montrent une petite partie de ce qu'on peut faire avec des pipes. Au fur et à mesure vous allez découvrir qu'il est difficile de s'en passer.

Les caractères génériques

L'astérisque

L'un des caractères génériques les plus connus est l'astérisque. Celle-ci désigne n'importe quelle chaîne de caractères. L'interpréteur de commandes va remplacer ce caractère par tous les éléments du répertoire qui correspondent. Ainsi, dans un répertoire contenant les fichiers toto, titi et tototi, si l'on tape *, il va lire en fait titi toto tototi, et va donc tenter d'exécuter la commande titi, en lui donnant comme paramètres toto et tototi.

Note : En fonction des cas, le système va trouver une commande appelée titi, et l'exécuter, où indiquer qu'il ne trouve pas de commande de ce nom.

Si l'on tape rm to* (attention si vous le tester, cela va réellement enlever les fichiers, il va remplacer to* par l'ensemble des éléments qui correspondent à « to suivi de n'importe quoi », soit toto et tototi dans notre exemple.

On peut placer un caractère générique n'importe où dans une chaîne : to*to signifie « to, suivi de n'importe quoi, et se finissant par to ». Dans notre exemple, seul le fichier toto correspond.

Le point d'interrogation

Le point d'interrogation est un caractère générique qui correspond à n'importe quel autre caractère, même à une absence de caractère. Il se comporte exactement comme l'astérisque, sauf qu'il ne correspond qu'à un seul caractère, et non pas à un groupe de caractères.

Les expressions rationnelles

Présentation

Les programmes Unix, dont les shells, sont capables d'interpréter des constructions beaucoup plus complexes que les deux caractères que l'on vient de voir. On appelle ces constructions des expressions rationnelles. Les caractères génériques constituent une partie des expressions rationnelles.

Avant tout, il faut savoir que chaque commande est libre d'interpréter comme elle veut ces expressions. Ainsi, le point, qui ne signifie rien de particulier pour le shell, est compris par le langage Perl comme l'équivalent de l'astérisque. Il faut donc s'assurer, avant d'utiliser des expressions rationnelles, que le programme que l'on utilise les comprend bien.

Expression correspondant à un ensemble de caractères

Une suite de caractères entre crochets droits [correspond à n'importe quel caractère de la suite. Pour reprendre l'exemple précédent, en tapant ls t[oi]t[oi], le shell va les remplacer par les fichiers existants dans l'ensemble toto, toti, tito, titi, et va donc faire l'équivalent d'un ls toto titi. Note : Il aurait été dans ce cas plus rapide de faire ls t?t? dans ce cas. On peut placer des suites de caractères entre crochets, reliés par un tiret. Ainsi, [A-Z] est remplacé par l'ensemble des lettres majuscules de A à Z.

Exclure des caractères d'une suite

Si le premier caractère d'une suite entre crochets est un accent circonflexe, les crochets sont remplacés par tous les caractères SAUF ceux qu'ils contiennent

Expressions complexes

Juxtaposition d'expressions

Il est possible de placer, on l'a vu, plusieurs expressions dans la même ligne. En fait, on peut dire que tout caractère dénué d'un sens spécifique est une expression rationnelle qui correspond au caractère en question.

Construire une alternative

Il est possible de créer des expressions correspondant à plusieurs groupes de caractères, plutôt qu'à un seul. On utilise pour cela des accolades. Il suffit d'y placer les différentes chaînes de caractères séparées par des virgules, pour que le shell les essaie à tour de rôle.

Ainsi, ls to{to,ta} va être équivalent à ls toto tota et ce même si tota n'existe pas (contrairement à un ls tot[oa]).

La banalisation de caractères

Nous avons vu maintenant plusieurs caractères réservés, qui ont une signification particulière pour le shell : les symboles « supérieur à », les caractères génériques utilisés dans une expression rationnelle...

Il peut arriver que l'on souhaite leur faire perdre leur sens particulier. Il faut alors les banaliser, d'une des trois manières suivantes :

Les précéder d'une barre oblige inverse (\) ;

Les entourer d'apostrophes ('') ;

Les entourer d'apostrophes doubles (" ").

Répétitions d'expressions

Toute expression rationnelle peut être répétée une ou plusieurs fois, en utilisant un des suffixes suivants :

{n,} : l'élément précédent doit apparaître n fois ou plus;

{,m} : l'élément précédent est facultatif et doit apparaître m fois au plus;

{n,m} : l'élément précédent doit être présent au moins n fois, mais au plus m fois.

Petite liste de trucs et astuces

A l'aide de la souris dans un terminal X ou la console, sélectionner un texte avec le bouton gauche puis lâcher et enfin cliquer sur le bouton droit : copie la sélection dans la ligne de commande.

Pour lancer des shell scripts ou des programmes n'étant pas renseignés dans le PATH taper: ./nom_script ou ./nom_programme

Dans /home/nom_utilisateur, le fichier .bash_history contient toutes les commandes utilisées et les met à disponibilité de l'éditeur de la ligne de commande (bien sûr si on utilise le shell bash).

passwd nom_utilisateur : Change le mot de passe de l'utilisateur spécifié sans demander l'initial, seulement en tant que root.