Aide Linux et Commandes de base

commandes externes (appel à un programme)
Commandes internes (commandes du shell)
Nommage : max 255 caracteres
Sensible à la casse
tilde : ALTGR 2

Obtenir de l'aide dans le prompt : 3 aides possibles

Pour installer les docs en FR :sur debian :
su
Mot de passe:
apt-get install manpages-fr manpages-fr-extra

1. Les pages de man

Le terme pages de man (prononcez « manne ») signifie « pages de manuel ». Ces pages décrivent essentiellement les commandes disponibles sous Linux, avec toutes leurs options. L'usage de ces pages de manuel se trouve donc principalement utile en mode console `man <nom-commande>`; c'est pourquoi nous présenterons d'abord l'utilisation de ces pages dans cet environnement.

L'ensemble des pages de man est organisé en sections numérotées de 1 à 9 pour les plus courantes :

1. commandes utilisateur pouvant être exécutées par tous ;
2. appels système, c'est-à-dire les fonctions fournies par le noyau ;
3. fonctions des bibliothèques ;
4. périphériques, c'est-à-dire les fichiers spéciaux que l'on trouve dans le répertoire /dev ;
5. descriptions des formats de fichiers de configuration (par exemple /etc/ passwd) ;
6. jeux ;
7. divers (macros, conventions particulières...) ;
8. outils d'administration exécutables uniquement par le super utilisateur (root) ;
9. autre section (spécifique à Linux) destinée à la documentation des services offerts par le noyau. Lorsque vous interrogez la documentation à propos d'un terme présent dans plusieurs sections (par exemple passwd , à la fois commande et fichier de configuration), si vous ne précisez pas le numéro de section, c'est toujours la section de numérotation la moins élevée qui sera affichée.

Organisation des pages de man

Chaque page de man est structurée en paragraphes contenant des informations organisées toujours de la même façon. Une fois que vous avez compris une page, vous avez compris toutes les autres. Voici donc les différents paragraphes que vous rencontrerez :

- **L'intitulé de la commande ou du fichier** et la section du manuel : ce détail est intéressant à connaître pour vérifier qu'on a récupéré la bonne documentation.
- **Nom** : il s'agit du nom de la commande ou du fichier avec une description synthétique.
- **Synopsis** : dans ce paragraphe, vous trouverez la syntaxe d'une commande, c'est-à-dire l'ensemble des options (introduites par un « - ») et des arguments (sans « - ») disponibles. Les éléments indiqués sans crochets sont obligatoires et, s'ils sont omis, provoqueront une erreur. Les éléments entre crochets sont facultatifs pour le fonctionnement de la

commande. Lorsque les options sont indiquées dans les mêmes crochets, elles peuvent être combinées. Dans le cas contraire, elles sont incompati-bles et devront être utilisées séparément. Enfin, les options peuvent être abrégées (ex : « -f ») ou complètes (ex : « --force »), mais la signification est la même et elle est développée dans le paragraphe suivant.

- **Description** : comme son nom l'indique, c'est une description (très com- plète) de la commande. Ce paragraphe peut être suivi de plusieurs autres selon les commandes et les informations à donner.
- **Voir aussi** : comme son nom l'indique, il s'agit d'une liste de commandes, fichiers, appels système... auxquels on vous renvoie pour compléter votre information.
- **Environnement** : ici sont spécifiées les variables d'environnement que vous pouvez configurer pour le fonctionnement de la commande ou du fichier.

less

En mode console, les pages de man sont affichées avec le paginateur `less` .

Pour pouvoir vous déplacer et faire des recherches dans les pages de man, il vous faudra connaître un minimum de raccourcis clavier. Si vous voulez en savoir plus, une solution évidente est : `man less` .

Commandes de déplacement dans la page de man :

- Entrée : faire défiler la documentation ligne par ligne en avant ;
- Espace ou page vers le bas : faire défiler la documentation page par page en avant ;
- b ou page vers le haut : faire défiler la documentation page par page en avant ;
- q : quitter le man et revenir à l'invite du shell.

Commandes de recherche de chaîne de caractères :

- /texte : recherche la chaîne « texte » dans la page de man ;
- n : aller à l'occurrence suivante ;
- N : aller à l'occurrence précédente.

2. Les pages d'info

Les pages d'info sont aussi de bonnes sources d'information sur les commandes. Elles sont structurées différemment et utilisent plus abondamment les liens hypertextes. En revanche, il est plus délicat de les trouver en français. Elles peuvent être un bon complément aux pages de man. Les pages d'info ne sont pas affichées avec `less` , donc les commandes sont légèrement différentes.

Pour les afficher en mode console, tapez simplement : `info <nom-commande>`

Pour afficher la page d'info sur la commande <nom> .

Les liens hypertextes sont précédés d'une astérisque. Vous pouvez vous déplacer de lien en lien avec la touche de tabulation. Pour activer un lien, appuyez sur la barre d'espace. Voici d'autres commandes de navigation :

- les touches `p` et `n` permettent respectivement de passer à la page précédente ou suivante ;
- la touche `u` permet de remonter d'un niveau dans la documentation ;
- la touche `q` permet de quitter.

3. L'aide des commandes

Si vous voulez juste vous remémorer les différentes options ou paramètres d'une commande que vous connaissez déjà, l'usage des pages de man ou d'info est superflu. Il vous suffit simplement de taper dans une console le nom de la commande suivi de l'option `<nom-commande> --help` ou `-h`

Les Commandes de base

`sudo su` (donne les privilèges administrateur)
`su user` (log avec user dédié)
`su root` (log en admin)

Créer un répertoire

`mkdir <nom-repertoire>`

Créer un fichier

`touch <nom-fichier>` ..

Afficher le contenu d'un répertoire

Pour afficher le contenu du répertoire où l'on se trouve, il faut saisir la commande `ls` .
N.B : Cette commande n'affiche par défaut pas les fichiers et **répertoires cachés**.

Pour afficher les fichiers cachés, saisir : `ls -a`
L'option -a précise à la commande ls d'afficher tous (all) les fichiers contenus dans le répertoire, y compris ceux qui commencent par un « . ». Ceux-ci sont en général des fichiers ou des répertoires de configuration : ainsi .bashrc représente le fichier de préférences personnelles portant sur le comportement du shell Bash.
La deuxième option importante est `-l` , qui permet d'obtenir plus d'informations sur les fichiers en les affichant au format « long », avec la taille et la date de dernière modification du fichier. Elle n'affiche par défaut pas les fichiers et répertoires cachés. Pour les afficher, il suffit de combiner les options - a et - l : `ls -al`
Enfin, notez qu'il est possible de passer le nom d'un répertoire en argument.
Par exemple, pour connaître le contenu du répertoire racine, on saisira : `ls /` où « / » désigne le répertoire racine du système.

Naviguer entre les répertoires

La commande cd permet de changer de répertoire. Fort simple d'utilisation, elle attend en argument un nom de répertoire. Pour revenir au répertoire précédent, plusieurs solutions s'offrent à nous : repasser par la racine ou utiliser « .. », qui représente le répertoire parent.

Complétion de saisie

Il suffit de saisir les premières lettres de l'argument et, s'il existe, d'effectuer une pression sur la touche <tab> pour compléter la commande. Ainsi `cd /ho<tab>` affichera cd /home
`cd ~` : nous place sur espace personnel

Revenir dans le répertoire d'où vous venez, il suffit de saisir : `cd -`

Ctrl + R, ou la recherche inversée

Vous pouvez également utiliser la recherche inversée. Si vous saisissez Ctrl + R et commencez à taper, le shell cherchera dans l'historique la ou les commandes qui commencent par votre saisie. Au fur et à mesure de votre saisie, la recherche se restreindra, jusqu'à correspondre à la commande voulue.

Rappel des dernières commandes

Le shell permet de rappeler des commandes précédemment saisies. Il suffit d'appuyer successivement sur la touche flèche vers le haut - (on peut aussi utiliser Ctrl + P) pour faire dérouler au fur et à mesure toutes les commandes saisies en ordre chronologique inverse (de la plus récente à la plus ancienne). L'inconvénient de ce système est qu'il demande d'appuyer sur la touche - autant de fois que l'on souhaite remonter dans les commandes. Il est alors souhaitable de faire appel à la commande history qui affiche les
500 dernières commandes.

Pour ré-exécuter une des commandes de la liste, il faut alors taper : !<n° de la commande>
Enfin, il est également possible de taper ! suivi du début de la commande précédemment saisie – la plus récente étant par défaut exécutée.

Mais où suis-je ?

Pour terminer, si vous êtes perdu au fin fond d'un sous-répertoire et que vous vouliez obtenir votre emplacement exact dans l'arborescence, il suffit d'exécuter la commande **`pwd`** .

Copier des données

La copie de fichiers et de répertoires est très simple. Elle se fait via la commande cp , qui s'utilise tout naturellement selon la syntaxe :
`cp <fichier source> <fichier destination>`
N.B : Si le fichier source correspond à un répertoire, il faut utiliser l'option -R (copie récursive).

Déplacer des données

La commande `mv` sert à déplacer des fichiers et des répertoires. Il n'est pas besoin ici de spécifier d'option de récursivité, elle s'utilise simplement comme suit : **`mv <source> <destination>`** .
Chemin **relatif** ou **absolu**. `root@debian:# mv fichier1 /home/stephane/fichier1` déplacera le fichier

Effacer des fichiers et des répertoires

C'est la commande **rm** qui permet d'effacer des fichiers.
Sachez que vous ne pouvez effacer que les fichiers sur lesquels vous avez le droit d'écriture.
Normalement, tous les fichiers que vous avez créés dans votre répertoire personnel vous appartiennent ; vous pourrez donc les effacer sans problème. Pour effacer un seul fichier, il suffit de saisir **`rm <nom du fichier>`**.

Pour effacer un répertoire vide, on utilisera : **`rmdir .<nom-repertoire>`**
Pour effacer directement **un répertoire et son contenu,** il faut utiliser : **`rm -r <nom-repertoire>`**
Pour supprimer avec demande de confirmation : **`rm -ri <nom-repertoire>`**

Les liens symboliques et physiques

Les liens sont **des raccourcis** qui permettent **d'accéder** à un fichier sous plusieurs noms.

Un lien peut être :

– Physique ou matériel : sous Unix, un lien physique (parfois aussi nommé lien dur) **permet de donner à un même fichier plusieurs noms situés dans plusieurs endroits**. Les modifications effectuées dans le fichier sous l'un des noms apparaîtront aussi sous les autres noms. Aucun des noms ne représente plus le fichier que les autres. La suppression d'un nom ne supprime pas le fichier, qui continue d'exister sous ses autres noms. Un fichier n'est définitivement supprimé que quand son dernier nom est effacé. Il n'est pas possible de créer un lien physique vers un répertoire, ni vers un fichier situé sur une autre partition.
`ln Dossier1/F3 <LienF3>`

– Symbolique : à la différence d'un lien physique, celui-ci **ne contient que le chemin d'accès vers le fichier ou répertoire source**. Si la source est déplacée ou supprimée, le lien pointe dans le vide. Ces liens permettent de distinguer le fichier original, ce que ne permettent pas les liens physiques.
`ln -s / <Racine>`

Un lien physique est une référence supplémentaire dans le répertoire des inodes pointant vers la même inode. Il est évident que pour Linux aucune référence ne représente plus le fichier que l'autre, l'accès au fichier étant strictement identique quelle que soit la référence utilisée : référence/inode/bloc disque.

Un lien symbolique est un pseudo-fichier qui contient le nom et le chemin d'accès de la source (sa référence). L'accès au fichier se fera ainsi :référence symbolique/référence réelle/inode/bloc disque.

Les variables :

Création d'une variable : monmessage="Bonjour"
Afficher la variable : echo $monmessage
Bonjour

Portée d'une variable :
Par défaut elle est **locale** (que pour un bash)

Variable d'environnement (accessible par tous les programmes au sein d'un environnement utilisateur)
> Pour voir les variable d'environnement : commande `env`

Pour créer des variables d'environnement
MONMESSAGE="Ma variable d'environnement"
`export` MONMESSAGE

`unset` MONMESSAGE
Ce n'est plus une variable d'environnement mais locale
`echo $?` Pour vérifier la valeur de la dernière commande
retour 0 = ok
retour 127 = pas bon

Exercice :

Mavariable=”Bonjour”
MONNOM=”Stéphane”
Mavariable=”$Mavariable $MONNOM”

Redirection de Flux

echo “toto”
echo “toto” > MonFichier
cat MonFichier

Exercice A :
Aller au repertoire personnel : `cd ~`
Noter dans un fichier “Oujesuis” votre emplacemen:t: `pwd > Oujesuis`
Aller à la racine : `cd /`
A la suite du fichier “Oujesuis” marquer votre emplacement : `pwd >> ~/Oujesuis` ou `pwd >> /home/stephane/Oujesuis`

Exercice B :
Créer un dossier comprenant des fichiers :
```
mkdir Dossier1
touch fichier1Qsdfgh31
```
Tenter les commandes rm, rm -ir, rm -rf, rmdir

Exercice C :
Créer un fichier ExerciceC composé de :

Mon historique est :
```
echo “Mon historique est : ” > ExerciceC
history >> ExerciceC
less historique (permet d’afficher tout le contenu)
```

Mon environnement est :
```
echo “Mon environnement est : ” >> ExerciceC
env >> ExerciceC
```

Le contenu de mon repertoire perso est :
```
echo “Le contenu de mon rep perso est : ” >> ExerciceC
ls ~ >> ExerciceC
```

Les expressions régulières

Certains caractères qui vont définir un groupe de caractères (sera utile pour des recherches)

`grep [option] regexp [fichier]`
`env | grep -i lang` : va rechercher le mot lang dans l’environnement en ignorant la casse (option i)

Exercice :

Avec le fichier dico,

Rechercher les mots contenant “bis” :

```
grep bis liste.de.mots.txt > result
grep -c bis liste.de.mots.txt >> result (-c affiche le count)
grep -i (l'option -i (ignorecase)
grep -w travail mon-fichier (grep ne recherche que les lignes où figure le mot tel quel, et non pas ses variantes)
```

Rechercher les mots dont le début est “te” ou “to” :

```
egrep '^(te|to)' liste.de.mots.txt > result2
egrep -c '^(te|to)' liste.de.mots.txt >> result2
```

Rechercher les mots dont le début est une voyelle suivie d’au moins 2 consomnes :

```
egrep -i '^[aeiouy][bcdfghjklmnopqrstuvwx]{2,}' liste.de.mots.txt > result3
egrep -ic '^[aeiouy][bcdfghjklmnopqrstuvwx]{2,}' liste.de.mots.txt >> result3
egrep -i '^[aeiouy][^aeiouy] [^negation]
```

Rechercher les mots dont la fin est x, y ou z

```
egrep -i '[xyz]$' liste.de.mots.txt > result3
```

Rechercher les mots contenant “libre” ou “ouvert”

```
egrep -i '(libre|ouvert)' liste.de.mots.txt > result3
```

Rechercher les mots contenant explicitement “libre” ou “ouvert” :

```
egrep -i '^(libre|ouvert)$' liste.de.mots.txt > result4
egrep -w 'libre|ouvert' liste.de.mots.txt > result4
```

Rechercher les verbes du premier groupe :

```
egrep -i '(er)$' liste.de.mots.txt > result5
```

Le nombre de verbes du premier groupe :

```
egrep -ic '(er)$' liste.de.mots.txt > result6
```

Rechercher les verbes du premier groupe avec 5 caracteres :

```
egrep -i '^.{3}(er)$' liste.de.mots.txt > result7ich
```

Lancer une commande qui écrira dans un fichier ‘MotsenT” uniquement les mots entrés par un utilisateur qui commencent par T

```
egrep -i '^t.*' > MotenT
t beau
toto
```

ctrl-D pour arreter la commande.

Ecrire dans un fichier “ExB’ les noms de tous les dossiers contenus (directement) par le dossier racine (rappel : ls /)

```
ls -l / | egrep -i '^d' > ~/ExB
ls -p / | egrep -i '/$' > ~/ExB
```

```
ls -d */ > ~/ExB
```
(methode globbing)

Dans le fichier dico :
Ou que ça finisse par “ent” ou “ant” :
```
egrep -i '(ent|ant)$'
```

Ou que ça finisse par br et finissant par une voyelle :
```
egrep -i 'br[aeiouy]$'
```

Commencant par a ou e :
```
egrep -i '^[ae]'
```

Ou que ça finisse par “ent” ou “ant”, ou “br” et finissant par une voyelle, ou “a” ou “e”, ou contenant “tion” :
```
egrep -i '(ent|ant)$|br[aeiouy]$|^[ae]|tion'
```

Simplifier la dernière commande en créant un fichier de commande d’expression réguliere (**Avantage** > on est pas obligé de concatener avec | et le fichier peut être réutilisé) :
```
egrep -f mesexpression listedemot > exC
```

Contenu du fichier “meexpression” :
```
(ent|ant)$
br[aeiouy]$
^[ae]
tion
```

Ecrire les expressions dans le fichier à partir du bash :
```
echo "(ent|ant)$" > mesexpressions
echo "br[aeiouy]$" >> mesexpressions
echo "^[ae]" >> mesexpressions
echo "tion" >> mesexpressions
```

Plus d’exemple ici :
http://www.tuteurs.ens.fr/unix/exercices/solutions/grep-sol.html#grep1