Administration Linux

Sommaire

Présentation

Cette documentation présente des bases de l'utilisation de Linux mais orienté hébergement et administration système. Elle traite donc principalement d'utilisation et configuration en ligne de commande.

Généralités

Linux est d'abord le nom du Kernel (cœur du système) créé par Linus Torvald en 1991.

Les distributions GNU/Linux sont similaires sur beaucoup de points communs avec les Unix propriétaires.

Les particularités de l'environnement Linux sont les suivantes :

- Tout est fichier. L'accès aux périphériques et composants internes au système se font pas le biais de fichiers.
- Les extensions de fichier n'ont pas de signification réelle autre qu'aider l'utilisateur (et le cas particulier des environnements graphiques KDE, GNOME ...). Il faut comprendre par là qu'un fichier texte peut se nommer document.txt ou document.ext sans importance réelle.
- Les points de montage. Linux n'utilise pas de lecteur c: d : ... comme Windows. Toute partition ou lecteur est monté dans un sous-dossier de la racine / (/ étant aussi une partition montée à ce niveau).
- L'utilisateur root (super utilisateur). root est l'utilisateur qui a tous les pouvoir sur le système. Il faut faire attention aux manipulations réalisées avec ce compte.

Les principaux dossiers

- / (la racine du système)
- /etc (fichiers de configuration)
- /usr (la plupart des programmes sont dans ce dossier mais non vitaux au démarrage)
- /bin (commandes principales : cp, mv, mount etc …)
- /dev (fichier de périphériques)
- /var (fichiers de données : log, bases de données, etc)
- /sbin (commandes de gestion sytème : adduser, fsck …)
- /lib (bibliothèques)
- /home (dossiers utilisateurs)
- /proc (informations sur les processus et sur le système)
- /var/spool (divers files d'attente pour les mails par exemple)

Dossiers particuliers

Les dossiers . et ..

- . est le dossier courant
- .. est le dossier parant au dossier courant.

Ils sont présents pour tous les dossiers.

Utilisation du système

Chemin absolu et relatif, le PATH, lancer une commande

Considérons que nous sommes situés dans le dossier **/home/jerome**. Nous devons éditer le fichier documentation.txt situé dans ce dossier.

- ***/home/jerome/documentation.txt*** est le chemin absolu vers ce fichier.
- ***documentation.txt*** est le chemin relatif au dossier /home/jerome.

Le **PATH** indique au système où trouver les programmes à exécuter sans avoir à indiquer le chemin absolu vers ceux-ci. Il s'agit d'une variable système. Regardons sa valeur pour un utilisateur standard :

```
echo $PATH
/usr/local/bin:/usr/bin:/bin:/usr/bin/X11:/usr/games
```

Nous avons pu exécuter la commande **echo** directement car son chemin absolue est ***/bin/echo***

Il aurait donc été possible de lancer :

```
/bin/echo $PATH
```

Par défaut un exécutable dans le dossier courant (hors ceux indiqués dans le PATH) doit être lancé ainsi :

```
./executable
```

Pour modifier le PATH, il est nécessaire de modifier cette variable ainsi en ajoutant par exemple le dossier courant . :

```
PATH=$PATH:.
```

Et lancer :

```
executable
```

Utiliser et se déplacer dans le système de fichier

- cd, pwd

```
cd /home #permet de se positionner dans le dossier /home
pwd #indique le dossier courant
```

- mkdir, rmdir

```
mkdir /var/www/www.test.com #crée le dossier /var/www/www.test.com
rmdir /var/www/www.test.com #efface le dossier /var/www/www.test.com, le dossier
doit être vide
```

- rm, cp, mv

```
rm /var/www/www.test.com/index.html #supprime le fichier
/var/www/www.test.com/index.html
rm -fR /var/www/www.test.com/ #supprime le dossier /var/www/www.test.com ainsi
que toutes les fichiers et dossiers présents dans celui-ci.
```

- ls

```
ls /var/www/www.test.com/ # liste les fichiers du dossier en argument
```

```
ls -l /var/www/www.test.com # liste les fichiers du dossier avec des détails
ls -ld /var/www/www.test.com # liste les détails du dossier
```

- Détails de ls -l

-rw-r-----	1	root	www-data	407176081	2009-03-23 16:23	www.dj-j.net-access.log
type de fichier puis les 9 propriétés d'accès	nombre de lien physique	propriétaire	groupe	taille	date de modification	la référence

Le type de fichier vaut d, - ou l, respectivement pour dossier, fichier ou lien.

La référence est le nom du fichier, dossier ou lien.

- les fichiers cachés

Les fichiers cachés se distinguent par la présence d'un . devant le nom du fichier.

```
ls -al /var/www/www.test.com # liste les fichiers et fichiers cachés du dossier avec des détails
```

- Les liens symboliques

Un lien symbolique permet d'accéder à un dossier ou un fichier depuis un emplacement différent de celui où il se trouve à l'origine.

Par exemple, un fichier **/home/user/index.html** doit être accessible depuis **/var/www/site/index.html** :

```
ln -s /home/user/index.html /var/www/site/index.html
```

Le lien symbolique est **/var/www/site/index.html**.

Autre exemple, un dossier **/home/user** doit être accessible depuis **/var/www/user2** :

```
ln -s /home/user /var/www/user2
```

Le lien symbolique est **/var/www/user2**. Vous remarquerez que le lien peut avoir un nom différent de la source.

Trouver l’emplacement d’une commande

- **which**

which indique le chemin absolu d'une commande incluse dans un dossier du PATH

- **locate**

locate indique le chemin d'une commande ou d'un fichier sur tout le disque.

locate utilise une base de référence qui est mise à jour avec la commande **updatedb** (généralement mise dans une tâche planifiée).

Les droits d’accès aux fichiers

- rwx

Les droits possibles sur le système de fichiers sont :

```
r : lecture
w : écriture
x : exécution
```

Leurs équivalences en valeur numérique sont :

```
r : 4 (2 puissance 2)
w : 2 (2 puissance 1)
x : 1 (2 puissance 0)
```

Les valeurs numériques s'additionnent. Pour r+x, cela donne 5.

Ces droits vont s'appliquer soit au propriétaire, soit au groupe, soit aux autres.

Attention, le droit x sur un dossier ne veut pas dire le droit de lister celui-ci. Il permet de "passer" le dossier. Pour lister un dossier, r+x sont complémentaires. Juste le droit x permet par exemple de ne pas lister un dossier mais d'accéder à un sous-dossier et lister celui-ci.

Exemple : En tant que root :

```
sangoku:/tmp# ls -ld dossier/
drwx--x--- 3 root www-data 1024 2009-03-23 16:07 dossier/

sangoku:/tmp# ls -l dossier
total 1
drwxr-x--- 2 root www-data 1024 2009-03-23 16:07 sousdossier

sangoku:/tmp# ls -ld dossier/sousdossier/
drwxr-x--- 2 root www-data 1024 2009-03-23 16:07 dossier/sousdossier/

sangoku:/tmp# ls -l dossier/sousdossier/
total 0
```

En tant que www-data :

```
www-data@sangoku:/tmp$ ls -ld dossier/
drwx--x--- 3 root www-data 1024 2009-03-23 16:07 dossier/

www-data@sangoku:/tmp$ ls -l dossier/
ls: dossier/: Permission non accordée

www-data@sangoku:/tmp$ ls -ld dossier/sousdossier
drwxr-x--- 2 root www-data 1024 2009-03-23 16:07 dossier/sousdossier

www-data@sangoku:/tmp$ ls -l dossier/sousdossier
total 0
```

- ugoa : abréviations utilisés par les commandes de modification des droits

```
u : user = propriétaire
g : group = groupe
o : other = autres
a : all = tout le monde
```

- Exemples de droits avec ls -l :

sangoku:/var/log/apache2# ls -l www.dj-j.net-access.log -rw-r----- 1 root www-data 369676014 2009-03-23 15:37 www.dj-j.net-access.log Les droits accordés au fichier www.dj-j.net-access.log sont :

```
lecture + écriture au propriétaire root
lecture au groupe www-data
aucun aux autres
```

- Les commandes chmod, chown, chgrp

chmod permet de changer les droits de fichiers ou de dossiers. Il est possible d'utiliser les abréviations ou les valeurs binaires.

Pour rwxr-x---, il faut par exemple faire la commande :

```
chmod 750 fichier
```

A l'origine, un fichier a les droits rwxr-x---, vous souhaitez ajouter les droits de lecture aux autres :

```
chmod o+r fichier
```

chown permet de changer le propriétaire et le groupe de fichiers ou de dossiers :

```
chown proprietaire:groupe fichier
chown proprietaire fichier
```

chgrp permet de changer le groupe de fichiers ou de dossiers :

```
chgrp groupe fichier
```

Toutes ces commandes ont l'option -***R*** pour réaliser la modification récursivement à tous les sous-dossiers et fichiers du dossier en argument.

Gestion de l’espace disque

- Espace disque avec df :

df indique l'espace disque occupé et disponible.

L'option --si permet d'obtenir un affichage avec des valeurs en Mo, Go ... Exemple :

```
sangoku:~#  df --si
Sys. de fich.          Tail.  Occ.  Disp. %Occ. Monté sur
/dev/md0                78G    50G    25G  67% /
tmpfs                  1,1G      0   1,1G   0% /lib/init/rw
udev                    11M    70k    11M   1% /dev
tmpfs                  1,1G   4,1k   1,1G   1% /dev/shm
/dev/md1               502M    32M   445M   7% /tmp
```

L'espace sur les disques physiques sont indiqués par partition du type /dev/hda1, /dev/sda1 ou /dev/md0 mais aussi en fonction du point de montage.

- Taille de fichiers ou dossiers avec du

du permet de connaître la taille occupée sur le disque par des fichiers ou des dossiers.

L'option -h permet d'obtenir un affichage avec des valeurs en Mo, Go ...

```
du -h fichier
```

Une utilisation pratique est de faire par exemple :

```
du -sch /var/log/*
```

Vous obtenez la liste des fichiers et dossiers de /var/log/ avec leur taille ainsi que la taille globale de /var/log.

Les processus

- Lister les processus avec ps :

ps avec – pour les options type unix

ps sans – pour les options type BSD

```
ps –ef # affiche tous les processus  (f : avec beaucoup d'informations)
ps –af # affiche uniquement les processus associés à un terminal
```

L'option –eF affiche plus d'information sur le processus

Il est possible de combiner les type BSD avec les type unix.

```
ps –ef f # affiche les processus sous forme de hiérarchie en art ascii.
```

Des commandes à peu près semblables :

```
ps –efH
ps auxf
```

Quelques états :

```
      D    Uninterruptible sleep (usually IO)
      R    Running or runnable (on run queue)
      S    Interruptible sleep (waiting for an event to complete)
      T    Stopped, either by a job control signal or because it is being
traced.
      Z    Defunct ("zombie") process, terminated but not reaped by its parent.
```

Si vous voyez beaucoup de processus ou un processus qui est très souvent avec l'état ***D***, c'est qu'il y a probablement un problème de performance d'accès aux disques.

- Arrêter un processus avec kill :

kill envoie un signal à un processus afin qu'il réalise une action, généralement son arrêt. Il prend en argument le numéro d'ID (obtenu avec ps)

kill –l : liste des signaux possibles avec leur numéro et nom associé Signaux les plus utilisés : TERM / SIGTERM / 15 (signal par défaut demandant l'arrêt du processus) KILL / SIGTERM / 9 (signal tuant le processus s'il ne répond pas au signal d'arrêt) HUP / SIGHUP / 1 (signal différent suivant les applications, cela peut être par exemple la mise à jour de la configuration)

Exemples d'appels :

```
kill 19140
kill –TERM 19140
kill -9 19140 19602
```

- Voir l'activité en cours avec top :

Il suffit de lancer la commande **top**

Lorsque top est lancé, vous pouvez modifier des options d'affichage, par exemple :

```
d # raccourcir le temps entre le rafraichissement
u # choisir d'afficher les processus d'un utilisateur
B # afficher certains paramètres principaux en gras
i # afficher uniquement les tâches actives
W # sauvegarder les options pour l'utilisateur
```

- Effectuer des tâches en arrière plan avec jobs, fg et bg :

Pour lancer une commande en arrière plan, faite la terminer par ***&*** :

```
./commande&
[1] 31344
```

Le retour [1] 31344 indique dans un premier temps entre [] le numéro du **job** puis l'ID du processus.

La commande jobs liste les processus mis en arrière plan par le '&' ou l'association CTRL-Z.

CTRL+Z met en arrière plan et stoppe le processus. (Etat T)

La commande fg permet de remettre en premier plan un processus.

```
fg 1
```

Si vous avez arrêté un processus et mis en arrière plan avec CTRL+Z. La commande bg permet de faire continuer le processus mais en arrière plan.

```
bg 1
```

Exemple avec la commande top :

```
sangoku:~# top&
[1] 31836

[1]+  Stopped                 top
sangoku:~# jobs
[1]+  Stopped                 top
sangoku:~# fg 1
```

Arrêt et redémarrage

- Arrêter le système :

```
shutdown –h now
```

Alias :

```
poweroff
```

- Rebooter le système :

```
shutdown -r now # reboot immédiat
shutdown –r 5 #reboot dans 5 minutes
shutdown –rF now #reboot immédiat avec un check disque au redémarrage
```

Alias pour un reboot immédiat :

```
reboot
```

Les services

Il y a 7 niveaux de démarrage :

- 0 : utilisé pour l'arrêt du serveur
- 1 : mode "single user", utilisé en cas de problème pour démarrer directement en root
- 2 à 5 : démarrage des services au boot du serveur, le niveau est différent selon les distributions (voir après)
- 6 : utilisé pour le reboot du serveur

Pour connaître le niveau de démarrage utilisé par le serveur, il faut regarder le fichier **/etc/inittab** à

la ligne :

```
id:2:initdefault:
```

Ici le niveau est 2. En général, pour une distribution Debian ou basée sur Debian, le niveau est 2. Pour une distribution Red-Hat ou basée sur Red-Hat, le niveau est soit 3 soit 5 respectivement pour un démarrage sans serveur X ou un démarrage avec serveur X.

Les scripts de démarrage des services (Apache, MySQL ...) sont mis dans le dossier : **/etc/init.d**.

Des liens symboliques sont placés dans les dossiers /etc/rc.X où X est le niveau de démarrage. Cela permet d'activer ou non un service en fonction du niveau de démarrage.

Exemple :

```
ls -l /etc/rc2.d/S91apache2
lrwxrwxrwx 1 root root 17 2009-02-03 12:19 /etc/rc2.d/S91apache2 ->
../init.d/apache2

ls -l /etc/rc0.d/K09apache2
lrwxrwxrwx 1 root root 17 2009-02-03 12:19 /etc/rc0.d/K09apache2 ->
../init.d/apache2
```

Le script de démarrage d'apache2 est /etc/init.d/apache2. Un lien est placé dans le niveau 2.

Les liens sont créé de façon particulières. Ils doivent commencer par **S** lorsque ***start*** doit être passé en argument ou par **K** quand ***stop*** doit être passé en argument. Ensuite, un nombre permet d'ordonner les services. Avec 91 dans notre exemple, apache2 sera démarré après un service dont le lien commence par S50.

Pour lancer, arrêter ou redémarrer manuellement un service, il faut utiliser les scripts dans **/etc/init.d**. Par exemple pour apache2 :

```
/etc/init.d/apache2 stop # Arrêter Apache2
/etc/init.d/apache2 start # Démarrer Apache2
/etc/init.d/apache2 restart # Relancer Apache2
```

Parfois les scripts servent aussi à d'autres fonctionnalités comme par exemple :

```
/etc/init.d/apache2 reload # Recharger la configuration d'Apache2
```

Gestion des utilisateurs et groupes

- Généralités :

Un utilisateur appartient à un groupe. Ce groupe peut être commun à d'autres ou être un groupe unique pour cet utilisateur (dans ce cas généralement le groupe se nomme comme le login de l'utilisateur).

Un utilisateur ou un groupe possède un ID unique permettant de les distinguer des autres. On parle de UID pour les utilisateur et GID pour les groupes. Il est possible de voir les ID du propriétaire et groupe d'un fichier avec :

```
ls -ln synchro.log
-rw-r--r-- 1 1000 1000 1282 sep  6 11:31 synchro.log
```

- Ajouter ou supprimer des utilisateurs avec adduser et deluser

Par défaut, un utilisateur aura son dossier dans /home/login.

Lors de l'ajout d'un utilisateur, des options intéressantes sont :

```
--home dossier #pour spécifier un autre dossier que /home/login
--ingroup groupe #pour spécifier un groupe déjà existant pour l'utilisateur
--no-create-home #pour ne pas créer le dossier s'il existe déjà
--uid ID #pour spécifier un ID à l'utilisateur
```

Attention, l'ordre des options est importante, voir l'aide complète suivante :

```
adduser [--home RÉP] [--shell INTERPRÉTEUR] [--no-create-home] [--uid ID]
   [--firstuid ID] [--lastuid ID] [--gecos GECOS]
   [--ingroup GROUPE | --gid ID] [--disabled-password]
   [--disabled-login] UTILISATEUR
 Ajout d'un utilisateur ordinaire
```

Il est possible d'ajouter un utilisateur "système" pour un service par exemple. Dans ce cas, utiliser :

```
adduser --system [--home RÉP] [--shell INTERPRÉTEUR] [--no-create-home]
   [--uid ID] [--gecos GECOS] [--group | --ingroup GROUPE | --gid ID]
   [--disabled-password] [--disabled-login] UTILISATEUR
 Ajout d'un utilisateur système
```

Il est convenu que les ID des utilisateurs systèmes sont inférieurs à 500.

- Modifier le mot de passe d'un utilisateur avec **passwd** :

```
passwd # sans argument vous changer le mot de passe de l'utilisateur courant
(attention lorsque vous êtes connecté en root)
passwd login #modifier le mot de passe d'une autre personne lorsque vous avez
les droits root
```

- Ajouter un groupe avec **addgroup** :

Il n'y a pas beaucoup d'option pour l'ajout d'un groupe. Il faut indiquer un nom de groupe et si besoin le fait qu'il s'agisse d'un groupe système (pour un service par exemple). Il est aussi possible de préciser un ID spécifique.

```
addgroup --system [--gid ID] GROUPE
```

- Les fichiers /etc/passwd, /etc/group et /etc/shadow :

Le fichier ***/etc/passwd*** contient les informations de configuration des utilisateurs (ID,dossier personnel, login, shell ...).

Le fichier ***/etc/group*** contient les informations de configuration des groupes (ID, nom du groupe...).

Le fichier ***/etc/shadow*** contient les mots de passes cryptés des utilisateurs et les informations d'expiration de comptes. Sauf utilisateur expérimenté, il ne faut surtout pas modifier ce fichier.

Exemple de ligne pour passwd :

```
jlemaire:x:521:521::/home/jlemaire:/bin/bash
```

Les champs sont séparés par des : avec dans l'ordre :

- le login
- anciennement le mot de passe, x maintenant car dans le fichier shadow
- UID, l'id de l'utilisateur
- GID, l'id du groupe
- des informations comme le nom complet, etc ... vide dans l'exemple
- dossier utilisateur
- le shell par défaut, généralement bash ou par exemple /bin/false pour un compte FTP.

Exemple de ligne pour group :

```
jlemaire:x:521:
```

Les champs sont séparés par des : avec dans l'ordre :

- le ,nom du group
- x car la plus part du temps non utilisé
- GID, l'id du groupe
- vide lorsqu'il n'y a qu'une personne dans le groupe.

Dans /etc/passwd est défini que le groupe principale de l'utilisateur. Lorsque plusieurs personnes appartiennent à un même groupe, elles sont ajoutées à la fin de la ligne du groupe ainsi :

```
adm:x:4:root,adm,daemon
```

La commande su

La commande su permet principalement deux choses :

- changer d'utilisateur

```
su login # changement vers "login" en gardant les variables d'environnement de l'ancien utilisateur
su - login # changement vers "login" en chargeant les variables d'environnement de "login"
```

Si vous changer de l'utilisateur root vers un autre login, aucun mot de passe vous sera demandé, sinon le mot de passe associé au nouvel utilisateur sera demandé.

- exécuter une commande en tant qu'un autre utilisateur

Option utile pour des tâches planifiées pour lancer une commande sous le compte d'un autre utilisateur depuis le compte root :

```
su - login -c "path_to_commande"
```

Qu’est-ce qu’un shell

Le shell est l'interpréteur des commandes que vous lancez en console locale ou depuis une connexion SSH. Le shell donne des fonctionnalités permettant d'aider vos saisies et traitements. Les shell comment bash, ksh ou sh par exemples, permettre même de réaliser des programmes pour automatiser des tâches.

Le shell par défaut sous la plupart des distributions Linux est ***bash***.

Quelques fonctionnalités de bash

- Touche TAB

Vous pouvez utiliser la touche TAB par exemple pour deux choses, compléter la commande que vous tapez ou compléter un nom de fichier ou dossier.

Par exemple, tapez ***aw*** puis la touche TAB, le ***k*** est ajouté automatiquement. Ou alors tapez ***ls -l /ho*** puis la touche TAB, ***me/*** est ajouté automatiquement.

- !chaine

Vous venez d'exécuter la commande :

```
ls -l /home
```

vous souhaitez la relancer, tapez simplement :

```
!ls
```

bash lancera la dernière commande commençant par ls

- Ctrl+r

ctrl + r vous permettra de rechercher dans les dernières commandes lancées. Par exemple, après avoir lancé ***ls -l /home***, faite ***ctrl+r*** puis ***ho***, vous aurez la commande complète affichée :

```
(reverse-i-search)`ho': ls -l /home
```

- Ctrl+a, ctrl+e

Ctrl+a, ctrl+e permettent respectivement de déplacer le curseur au début et à la fin de la ligne de commande.

- Commande1;commande2

Il est parfois utile de préparer des commandes et les faire exécuter à la suite afin de gagner du temps entre chacune des commandes. Il suffit de les séparer par un **;**. Par exemple :

```
/etc/init.d/apache2 stop;umount /montageNFS;mount
/autremontageNFS;/etc/init.d/apache2 start
```

- Alias de commande

La commande alias, permet de réaliser des "raccourci", par exemple :

```
alias ll=’ls –l’
```

Fichier .bash_profile

Ce fichier, à la racine du dossier utilisateur (exemple : /home/user), permet de définir des préférences qui seront rechargées à la prochaine connexion. Il suffit d'un inscrire des commandes interprétables par bash. Par exemple, vous pouvez ajouter vos alias comme vu précédemment ou encore modifier le PATH.

Exemple de .bash_profile :

```
PATH=~/bin:"${PATH}"
alias ll='ls -l'
```

Utilisation avancée du système

Les tâches planifiées avec cron

Les tâches planifiées peuvent être configurées à plusieurs endroits selon les préférences et selon le type de tâches.

Tâches utilisateur

Les tâches utilisateur sont éditées avec l'outil **crontab** :

```
crontab -l #pour lister le contenu
crontab -e #pour éditer avec l'éditeur de texte par défaut.
```

Avec crontab, vous éditez les tâches de l'utilisateur courant, elle seront lancée en tant que cet

utilisateur.

Les fichiers se situent dans **/var/spool/cron/crontabs/**, il y a un fichier par utilisateur dont le nom est le login.

Tâches système

Ces tâches sont lancées à partir du fichier principal **/etc/crontab** qui appelle les autres ou depuis les fichiers mis dans **/etc/cron.d**.

Les tâches ne doivent pas être mises dans **/etc/crontab**, il faut les mettre dans :

- /etc/cron.d #pour des lancements à période personnalisée
- /etc/cron.hourly # pour des lancements toutes les heures
- /etc/cron.daily # pour des lancements tous les jours
- /etc/cron.weekly # pour des lancements toutes les semaines
- /etc/cron.monthly # pour des lancements tous les mois

Attention aux particularités suivantes :

- Dans /etc/cron.hourly, /etc/cron.daily, /etc/cron.weekly et /etc/cron.monthly, il s'agit directement de script exécutables à mettre. Il ne doit pas y avoir de . dans le nom du fichier. Ils sont exécutés en tant que root.
- Dans /etc/cron.d, les fichiers utilisent la même syntaxe que /etc/crontab, il faut préciser l'utilisateur lançant la commande avant celle-ci.

Syntaxe générale

Il faut préciser quand lancer la commande, puis indiquer la commande à lancer, sous la forme :

```
m h dom mon dow   command
```

Dans le cas de /etc/cron.d et /etc/crontab, il faut préciser l'utilisateur lançant la tâche :

```
m h dom mon dow  user command
```

- m : minutes (0 à 59)
- h : heures (0 à 23)
- dom : jour du mois (1 à 31)
- mon : mois de l'année (1 à 12)
- dow : jour de la semaine (0 à 6 où 0 est dimanche)

Exemple pour une exécution à 14h30 tous les lundi :

```
30 14 * * 1 /chemin/monprogramme
```

Il est préférable d'envoyer les sorties des tâches vers /dev/null ainsi :

```
30 14 * * 1 /chemin/monprogramme > /dev/null
```

S'il renvoie des erreurs dont nous avons connaissance et voulons les masquer :

```
30 14 * * 1 /chemin/monprogramme &> /dev/null
```

Il est possible de préciser plusieurs valeurs pour les minutes, heures ... :

- * : tout
- 3-6 : de 3 à 6 (3, 4, 5, 6)
- */5 : toutes les multiples de 5 (0, 5, 10, 15, ...)

- 4,12 : 4 et 12

Droits spéciaux

le droit « s »

```
ls -l /usr/bin/passwd :
-rwsr-xr-x 1 root root 28480 2007-02-27 08:53 /usr/bin/passwd
```

Le binaire passwd a le droit “s” , appelé aussi droit « setuid » qui permet de faire exécuter la commande passwd en tant que root (son propriétaire) même s’il est exécuté par un autre utilisateur.

Pour ajouter ce droit à un exécutable :

```
chmod u+s /pathto/command
```

Il existe l’équivalant pour le groupe avec le setgid. Le setgid sur un répertoire indique que les fichiers créés dans ce dossier appartiennent forcément au groupe du dossier.

Pour ajouter ce droit à un dossier :

```
chmod g+s /pathto/dossier
```

le droit « t »

Un exécutable peut être "sticky" (on dit aussi "avoir le sticky bit positionné"): cela signifie qu'il reste en mémoire même après la fin de son exécution, pour pouvoir être relancé plus rapidement.

Alors qu'un exécutable peut être déclaré setuid et setgid par son propriétaire, seul l'administrateur système peut positionner le sticky bit.

Le sticky bit sur un dossier avec les droits d’écriture pour tout le monde empêche qu’un utilisateur supprime les fichiers des autres. Exemple du dossier /tmp :

```
drwxrwxrwt   6 root root   4096 2008-04-25 15:35 tmp
```

Exemple pour affecter le sticky bit :

```
chmod +t /tmp
```

Valeurs Octales

- setuid=4
- setgid=2
- sticky=1

Les entrées/sorties, redirections et canaux de communications

Pour communiquer, les programmes utilisent des entrées et des sorties. Il existe une entrée et deux sorties :

- Stdin (0) : il s'agit de l'entrée, référencée 0, par défaut il s'agit du clavier.
- Stdout (1) : il s'agit de la sortie normale, référencée 1, par défaut, il s'agit de l'affichage écran
- Stderr (2) : il s'agit de la sortie pour les erreurs, référencée 2, par défaut, il s'agit de l'affichage écran

Elles se schématisent ainsi :

```
             -------------
stdin (0) -> | programme  | -> stdout (1)
             |            | -> stderr (2)
             -------------
```

Il est possible de réaliser des redirections sur ces entrées/sorties. Ainsi il est possible d'envoyer la sortie stdout vers un fichier ou d'envoyer la sortie d'un programme vers l'entrée d'un autre comme ceci :

```
             -------------                                    ---------------
stdin (0) -> | programme  | -> stdout (1) -> stdin (0) -> | programme2  | ->
stdout (1)
             |            | -> stderr (2)                  |             | ->
stderr (2)
             -------------                                    ---------------
```

- ***Redirections avec : >, >> et | (pipe)***

> permet de faire une redirection de la sortie vers un fichier. >> réalise la même chose mais en ajoutant à la fin du fichier au lieu de créer un nouveau fichier. Exemples :

```
echo "Texte"
```

Lancée ainsi, ***Texte*** est affiché à l'écran.

```
echo "Texte" > /root/texte
```

Lancée ainsi, ***Texte*** est mis dans le fichier de chemin /root/texte. Si /root/texte existe, il est écrasé.

```
echo "Texte" >> /root/texte
```

Lancée ainsi, ***Texte*** est ajouté au fichier de chemin /root/texte. Il y a alors deux fois Texte.

Par défaut si cela n'est pas précisé, seule la sortie 1 est redirigée. Pour rediriger les erreurs, il faut le préciser ainsi :

```
commande 2> /chemin/fichier
```

Si vous souhaitez rediriger les deux sorties :

```
commande &> /chemin/fichier
```

ou équivalent en indiquant de rediriger 2 vers 1 :

```
commande > /chemin/fichier 2>&1
```

| (pipe) permet de rediriger la sortie d'un programme vers l'entrée d'un autre ainsi :

```
commande1 | commande2
```

Par exemple :

```
grep "Texte" /var/log/mail.info|more
```

La sortie de la commande grep est envoyée à la commande more.

- Commande tee

Si vous effectuez une redirection vers un fichier, vous n'avez plus l'affichage, tee permet de réaliser cela. Voici comment l'utiliser :

```
echo "Texte" | tee /root/texte
```

Ainsi Texte est affiché et /root/texte est créé avec Texte dedans.

- /dev/null

/dev/null est comme une poubelle, il est pratique de l'utiliser dans des redirections pour ne plus avoir d'affichage :

```
/chemin/programme > /dev/null
```

Pour ne plus avoir l'affichage ni les erreurs :

```
/chemin/programme &> /dev/null
```

Les commandes find grep et wc

find

find permet de réaliser des recherches sur les fichiers et dossiers. Sur ces recherches il est possible d'afficher les résultats ou d'exécuter une commande sur les résultats trouvés.

find permet de spécifier des critères de recherche, comme les types, les noms ou encore les permissions.

- Exemple d'options :
  - -type f : filtrer sur le type fichier
  - -type d : filtrer sur le type dosssier
  - -name *test* : filtrer sur un nom de fichier ou dossier incluant test
  - -perm -u+r : filtrer sur les permissions du fichier ou dossier
  - -mtime +30 : filtrer sur la date de dernière modification

Voir **man find** pour tous les détails.

Exemples de recherches avec affichage du résultat :

- Lister les fichiers en écriture pour tous :

```
find /home -type f -perm -o+w -ls
```

- Lister les dossiers en écriture pour tous :

```
find /home -type d -perm -o+w –ls
```

Exemples de recherches avec exécution d'une commande :

Lorsque vous souhaitez exécuter une commande, il faut indiquer **-exec** puis la commande. Celle-ci est terminée par '\' ou **\;**. Indiquez **{}** pour le faire substituer par le fichier ou dossier trouvé par la recherche.

- Effacer les fichiers dont la dernière modification date de plus de 90 jours (**Attention avec cette commande !**) :

```
find /chemindossier -type f -mtime +90 -exec rm -f {} \;
```

Ajouter le droit d'exécution pour le group à tous les dossiers :

```
find /dossier  -type d ! -perm -g+x -exec chmod g+x {} \;
```

Quel est l'intérêt de cette commande, si vous utilisez **chmod -R g+x /dossier**, à la fois les dossiers et les fichiers seront modifiés alors qu'avec la commande find et le filtrage sur le type dossier, les fichiers ne seront pas modifiés.

Remarquez l'utilisation de ! pour indiquer les dossiers qui n'ont pas la permission x pour le groupe. En effet, il n'est pas nécessaire de refaire la modification si la permission est déjà présente.

grep

grep permet d'effectuer un filtrage sur des chaînes de caractères. La recherche s'effectue sur chaque ligne.

- Recherche directe dans un fichier :

```
grep "expression" fichier
```

Par exemple, rechercher MaxClient dans un fichier de log apache :

```
grep "MaxClient" /var/log/apache2/www.exemple.com-access.log
```

Rechercher récursivement dans tous les fichiers d'un dossier, par exemple pour trouver le fichier de configuration Apache d'un site précis :

```
grep -r "www.exemple.com" /etc/apache2
```

- Recherche de chaînes en association avec une autre commande.

Pour cela, nous utilisons | (pipe), par exemple, chercher si un processus précis est lancé :

```
ps -ef|grep "mysql"
```

- Option de négation pour recherche toute occurrence ne présentant pas l'expression indiquée :

```
grep –v "200" /var/log/apache2/www.exemple.com-access.log
```

Donnera toutes les lignes n'incluant pas 200.

wc

wc permet de compter des lignes, caractères ou mots. Généralement, il sera utile pour compter un nombre de ligne.

- Exemple, compter le nombre de fichiers/dossiers dans un répertoire :

```
ls –l | wc –l
```

- Compter le nombre d'occurrences trouvées par grep :

```
grep "213.210.130.43" /var/log/apache2/www.exemple.com-access.log|wc -l
```

Archivages et compression de fichiers/dossiers

D'abord, il faut comprendre la différence entre tar et gzip/bzip2/zip/etc... En effet, tar permet de rassembler un ensemble de fichiers/dossiers dans une archive. Il n'effectue pas de compression contrairement à gzip ... Par contre il sait utiliser gzip ou bzip2 pour compresser l'archive.

Pour les exemples nous avons 2 fichiers non compressés :

```
-rw-r----- 1 root root 130M 2009-04-28 18:43 syslog.noncompresse
-rw-r----- 1 root root 257M 2009-04-28 18:45 syslog.noncompresse2
```

Différence entre gzip et bzip2, bzip2 compresse mieux mais est beaucoup plus consommateur de ressources CPU que gzip.

gzip/gunzip

- Compression d'un fichier avec gzip. Il suffit de lancer la commande suivante :

```
gzip syslog.noncompresse
```

syslog.noncompresse est compressé :

```
-rw-r----- 1 root root  16M 2009-04-28 18:43 syslog.noncompresse.gz
```

Attention, après compression syslog.noncompresse est supprimé.

```
gzip syslog.noncompresse syslog.noncompresse2
```

Dans ce cas, les deux fichiers sont compressé dans des fichiers séparés, ils ne sont pas archivés dans un seul fichier :

```
-rw-r----- 1 root root 32M 2009-04-28 18:45 syslog.noncompresse2.gz
-rw-r----- 1 root root 16M 2009-04-28 18:43 syslog.noncompresse.gz
```

- Décompresser un fichier compressé avec gzip (généralement extension gz) :

```
gunzip syslog.noncompresse.gz
```

bzip2/bunzip2

- Compression d'un fichier avec bzip2. Il suffit de lancer la commande suivante :

```
bzip2 syslog.noncompresse
```

syslog.noncompresse est compressé :

```
-rw-r----- 1 root root  12M 2009-04-28 18:43 syslog.noncompresse.bz2
```

Attention, après compression syslog.noncompresse est supprimé.

On remarque que le fichier compressé ne fait que 12Mo contre 16Mo avec bzip.

```
bzip2 syslog.noncompresse syslog.noncompresse2
```

Dans ce cas, les deux fichiers sont compressé dans des fichiers séparés, ils ne sont pas archivés dans un seul fichier :

```
-rw-r----- 1 root root 23M 2009-04-28 18:45 syslog.noncompresse2.bz2
-rw-r----- 1 root root 12M 2009-04-28 18:43 syslog.noncompresse.bz2
```

- Décompresser un fichier compressé avec bzip2 (généralement extension bz2) :

```
bunzip2 syslog.noncompresse.bz2
```

tar

- Archiver sans compression des fichiers et/ou des dossiers :

```
tar -cf archive.tar fichier1 fichier2 dossier1
```

L'option **c** indique que tar doit créer une archive.

L'option **f** indique que l'archive est à créer dans un fichier correspondant au premier argument suivant ***f***.

Par exemple, pour archiver tous les fichiers/dossiers dans le dossier courant :

```
tar -cf archive.tar *
```

- Extraire les fichiers/dossiers d'une archive :

```
tar -xf archive.tar
```

L'option **x** indique que tar doit extraire d'une archive.

L'option **f** indique que les fichiers/dossiers sont à extraire du fichier correspondant au premier argument suivant ***f***.

- Gestion de la compression avec tar :

tar permet d'ajouter une compression à l'archive. Il suffit d'ajouter une option à la commande.

Les options sont **z** pour la compression gzip et **j** pour la compression bzip2 :

```
tar -zcf archive.tar.gz *
tar -jcf archive.tar.gz *

tar -zxf archive.tar.gz
tar -jxf archive.tar.gz
```

Les devices

Les devices sont les fichiers spéciaux qui permettent d'accéder aux composants matériels comme les disques durs.

Ils sont dans le dossier /dev.

Par exemple pour les disques, ils sont créés ainsi :

- /dev/sd[a-z] pour les disques scsi/sata
- /dev/hd[a-z] pour les disques ide

Il y a des exceptions selon les versions de noyau. Il peut y avoir des disques sata en hd et des disques ide en sd.

Ensuite, les partitions sont accessible en ajoutant le numéro de celle-ci à la fin. Par exemple, première partition du deuxième disque scsi :

```
/dev/sdb1
```

Attention pour les disques en raid matériel, un seul disque virtuel est vu par le système. Il n'y aura pas /dev/sda et /dev/sdb, uniquement, /dev/sda.

Par exemple, un périphérique usb peut être affecté à :

```
/dev/usb/hiddev0
```

La souris est affectée à

```
/dev/input/mice.
```

Gestion des systèmes de fichiers

Le fichier /etc/fstab

Dans ce fichier est listé l'ensemble des partitions et points de montages associés qui doivent être montées au démarrage. Ils indiquent aussi les montages NFS et autres montages réseaux.

Exemple :

```
/dev/hda3       /tmp            ext3    defaults        0       2
```

Extraits du man indiquant à quoi correspond chaque champs :

```
Le premier champ décrit le périphérique bloc ou le système de fichiers distant à
monter.
Le deuxième champ indique le point de montage du système de fichier.
Le  troisième champ décrit le type de système de fichiers.
Le quatrième champ définit les options de montage associées au système de
fichiers.
Le cinquième champ est utilisé par la commande dump(8) pour déterminer quels
sont les systèmes de fichiers à décharger. (Mettre 0 pour ne pas l'utiliser)
Le sixième champ est utilisé par le programme fsck(8) pour déterminer l'ordre de
vérification des systèmes de fichiers au démarrage. Le système de fichiers
racine doit avoir un champ de valeur 1, et les autres un champ de valeur 2. Les
systèmes partageant le même contrôleur  seront  vérifiés  à  la  suite,  mais
ceux utilisant différents contrôleurs seront vérifiés simultanément pour
profiter du parallélisme offert par le matériel. Si le sixième champ est absent
ou vaut zéro, fsck ne vérifiera pas ce système de fichiers.
```

Les 4 premiers sont les plus importants.

Les types les plus courant sont les suivants :

- ext2/ext3 : système de fichiers par défaut sous Linux, ext3 pour les systèmes récents (ext4 est en cours d'intégration dans les dernières distributions)
- nfs : système de fichiers par réseau, utilisé par exemple pour l'accès à une baie de stockage (EMC, Netapp...)
- swap : mémoire sur disque, utilisée lorsque la RAM n'est plus suffisante
- udf,iso9660 : pour les cdroms et dvd

Les options utiles peuvent être :

- defaults : options par défaut voir **man fstab** pour les détails.
- noauto : si vous souhaitez que la partition ne soit pas monté au démarrage.
- user : pour permettre à un utilisateur de réaliser les montages.
- uid=1000 ou gid=1000 : pour spécifier un uid et/ou gid à utiliser au montage.

Commandes fsck, fdisk, cfdisk, mkfs, mount, umount

- fsck permet de vérifier l'intégrité d'un système de fichiers. Il arrive que le système de fichiers soit dégradé et nécessite une réparation. Attention, il peut y avoir des pertes de données.

Il ne faut pas réaliser de fsck sur une partition montée..

Préférez l'utilisation de **fsck.ext3** pour un système formaté en ext3 ou **fsck.ext2** pour ext2.

Exemple de lancement :

```
fsck.ext3 -c /dev/sda1
```

Pour forcer le fsck :

```
fsck.ext3 -fc /dev/sda1
```

Si vous souhaitez toujours répondre oui aux questions sur le traitement des erreurs (**attention d'être sûr de ce que vous faites**) :

```
fsck.ext3 -cy /dev/sda1
```

- fdisk et cfdisk permette de créer, modifier ou supprimer des partitions. Ces commandes ne sont pas traités dans cette documentation.
- mkfs permet de formater une partition. Comme pour fsck, il faut utiliser le mkfs correspondant au système de fichier à utiliser. Par exemple, formater en ext3 :

```
mkfs.ext3 /dev/sda1
```

Il est possible de préciser un label pour la partition :

```
mkfs.ext3 -L "ROOT" /dev/sda1
```

- mount et umount

mount permet de monter une partition. Si elle est présente dans /etc/fstab, il est juste nécessaire de préciser le device ou le point de montage.

Exemple avec "/dev/hda3 /tmp ext3 defaults 0 2" dans /etc/fstab :

```
mount /dev/hda3
```

est équivalent à :

```
mount /tmp
```

S'il n'est pas présent dans fstab, il faut faire **mount DEVICE POINT_DE_MONTAGE** :

```
mount /dev/hda3 /tmp
```

Normalement le type de système de fichiers est détecté, pour le préciser, il faut ajouter l'option :

```
mount -t ext3 /dev/hda3 /tmp
```

umount permet de démonter une partition. Comme pour mount, si elle est présente dans /etc/fstab, il est juste nécessaire de préciser le device ou le point de montage.

Exemple avec "/dev/hda3 /tmp ext3 defaults 0 2" dans /etc/fstab :

```
umount /dev/hda3
```

est équivalent à :

```
umount /tmp
```

Il peut arriver que la partition ne puisse être démontée car elle est utilisée. Vérifier avec **lsof** quels processus l'utilisent :

Exemple pour /tmp :

```
lsof|grep "/tmp"
```

Screen

screen est un outil très pratique dans deux cas : besoin de reprendre une action en cours après déconnexion ou besoin d'avoir plusieurs fenêtres virtuelles sur une seule connexion.

Voici un résumé des commandes sous votre shell habituel :

- screen # pour le lancement au début
- screen -r # pour se reconnecter à l'ancien screen

Voici un résumé des commandes sous screen :

- CTRL+d # quit screen
- CTRL+A+d # détacher le screen courant et revenir au shell

Notion sur Netfilter/Iptables

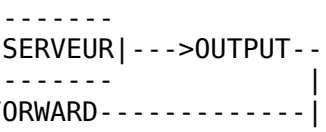
Netfilter/Iptables est le firewall pour Linux.

Voir aussi mon mini projet : [[Linux:sowall]

Le but est uniquement d'indiquer des notions sur iptables.

Netfilter est la partie au niveau du noyau Linux qui réalise le filtrage réseau. Iptables est la partie utilisateur permettant l'édition des règles firewall.

Les règles sont mises dans des chaînes représentant un flux. Il y a 3 chaînes INPUT, OUTPUT, FORWARD. Il est possible de créer des nouvelles chaînes utilisateurs. Voici le schéma :



La chaîne INPUT est donc utilisée pour les flux entrant vers le serveur, OUPUT pour les flux sortant du server et FORWARD lors de l'utilisation en tant que router.

Les cibles des règles indiquent l'action à prendre lorsque la règle correspond au packet en cours d'analyse. Il y a les cibles par défaut: ACCEPT, DROP et REJECT.

Le but des chaînes utilisateurs est de s'en servir en tant que cible et ainsi réaliser plusieurs traitement comme par exemple mettre le flux dans les logs puis l'accepter ou le refuser.

Exemples d'utilisation d'iptables :

- Listing des règles actuellement utilisées :

```
iptables –L
```

Avec les détails :

```
iptables –v –L
```

Afficher les IP au lieu des noms :

```
iptables -vnL
```

- Autorisation du port 25 pour le protocole tcp en entrée depuis n'importe quelle source :

```
iptables -A INPUT --proto tcp --dport 25 -j ACCEPT
```

- Bloquer en sortie le port 80 pour une IP de destination précise :

```
iptables -A OUTPUT -d 1.2.3.4 --proto tcp --dport 80 -j DROP
```

- Supprimer la règle :

```
iptables -D OUTPUT -d 1.2.3.4 --proto tcp --dport 80 -j DROP
```

Exemple d'affichage pour iptable -vnL :

```
Chain INPUT (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
  755 49876 firewall   all  --  *      *       0.0.0.0/0            0.0.0.0/0
state INVALID
    3  1617 firewall   tcp  --  *      *       0.0.0.0/0            0.0.0.0/0
state NEW,RELATED tcp  flags:!0x16/0x02
7093K 1741M ACCEPT     all  --  lo     *       0.0.0.0/0            0.0.0.0/0
  70M 8787M allowed_input  all  --  *      *       0.0.0.0/0
0.0.0.0/0           state RELATED,ESTABLISHED
 9768  776K allowed_ping  icmp --  eth0   *       0.0.0.0/0            0.0.0.0/0
    0     0 allowed_ping  icmp --  eth1   *       0.0.0.0/0            0.0.0.0/0
    5   240 allowed_input  tcp  --  eth0   *       81.56.179.14     0.0.0.0/0
state NEW tcp dpt:22

...

Chain allowed_input (33 references)
 pkts bytes target     prot opt in     out     source               destination
  70M 8812M ACCEPT     all  --  *      *       0.0.0.0/0            0.0.0.0/0
```

On voit qu'il y a une cible utilisateur **allowed_input**. Celle-ci est utilisé par exemple pour la règle :

```
5   240 allowed_input  tcp  --  eth0   *       81.56.179.14     0.0.0.0/0
state NEW tcp dpt:22
```

Donc si le packet vient de la source 81.56.179.14 et est à destination du port 22 en tcp, il est envoyé à la chaîne allowed_input. On voit aussi que la définition de allowed_input envoie vers la cible ACCEPT donc le packet est autorisé.

Notion de gestion de packages logiciels

Pour Debian

Voir la section dédiée : Linux:Administration_Debian#Gestion_des_packages_sous_Debian

Notion sur Grub et Lilo

Grub et Lilo sont des "boot loader". Ils permettent au système Linux d'être lancé.

Le noyau linux est placés dans le dossier /boot. C'est Lilo ou grub qui lance son chargement.

Le boot loader est installé dans la partie nommée MBR du disque. C'est l'endroit où le bios va lancer le démarrage du boot loader (c'est le même principe pour Windows par exemple). Il y a des cas spécifiques où un second boot loader est installé à la racine d'une partition précise par exemple quand plusieurs systèmes Linux sont installés.

Grub

Les fichiers de configuration sont situé dans /boot/grub. Il y a le fichier menu.lst qui définit les

options et la liste des noyaux disponibles pour le démarrage et device.map qui définit les disques du système.

Pour installer grub dans le MBR, il faut lancer la commande **grub-install DEVICE**. Par exemple si le disque est /dev/sda :

```
grub-install /dev/sda
```

L'avantage de grub est qu'une fois installé, il n'est pas nécessaire de le refaire lorsqu'un nouveau noyau est installé. Il suffit de modifier /boot/grub/menu.lst, celui-ci est chargé dynamiquement.

Lilo

Le fichier de configuration de lilo est généralement /etc/lilo.conf.

Pour installer lilo dans le MBR, il faut lancer la commande **lilo**, ajouter -v est préférable pour voir si tout se passe bien :

```
lilo -v
```

Pas besoin d'indiquer le disque avec lilo, il utilise le paramètre **boot=/dev/sda** dans le fichier de configuration.

L'inconvénient de lilo est de devoir être réinstallé dans le MBR à chaque installation d'un nouveau noyau (même une mise à jour sans changement de version).

UDEV

Liens intéressants :

- Généralités Debian : http://www.fr.debian.org/doc/manuals/debian-reference/ch03.fr.html#_the_udev_system
- Comment créer des règles : http://www.reactivated.net/writing_udev_rules.html
- Généralités présentées par Gentoo : http://www.gentoo.org/doc/fr/udev-guide.xml

Utilisation de Linux pour l'hébergement

Le réseau : ping, ifconfig, route ...

ping

ping permet d'envoyer une demande d'echo icmp de type 8 à un serveur, celui-ci doit répondre par echo avec une réponse icmp de type 0. Cela permet de vérifier si le serveur répond bien sur le réseau ou s'il est totalement inaccessible.

Il suffit pour cela de réaliser la commande suivante :

```
ping nomserveur.domaine.com
```

ou avec l'IP :

```
ping IP
```

Contrairement à la version windows, ping sous linux ne nécessite pas d'option pour envoyer en continue des requêtes, ce comportement est par défaut.

Il faut taper "ctrl+c" pour terminer le ping.

Exemple :

```
djj@tortuegeniale:~$ ping krilin.dj-j.net
PING krilin.dj-j.net (192.168.100.2) 56(84) bytes of data.
64 bytes from krilin.dj-j.net (192.168.100.2): icmp_seq=1 ttl=64 time=0.157 ms
64 bytes from krilin.dj-j.net (192.168.100.2): icmp_seq=2 ttl=64 time=0.156 ms
64 bytes from krilin.dj-j.net (192.168.100.2): icmp_seq=3 ttl=64 time=0.154 ms
^C
--- krilin.dj-j.net ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1998ms
rtt min/avg/max/mdev = 0.154/0.155/0.157/0.014 ms
```

On voit qu'il n'y a pas de problème dans ce cas car les temps de réponse sont faibles moins d'une milliseconde (time=0.157 ms).

Ensuite, autre point à vérifier, au niveau du résumé à la fin, il y a **3 packets transmitted, 3 received, 0% packet loss**. Cela indique qu'il y a autant de paquets envoyés que de paquets reçus donc 0% de perte. S'il y a des pertes, c'est qu'il y a potentiellement un problème réseau ou au niveau des interfaces réseaux du serveur émetteur ou du serveur récepteur.

Les interfaces réseaux et ifconfig

- Les interfaces réseaux sont visibles grâce à la commande **ifconfig**. Une interface **lo** est toujours présente et représente l'interface local nécessaire au système même si le serveur n'est pas connecté. Elle a pour adresse IP **127.0.0.1**.

ifconfig sans argument liste toutes les interfaces configurées, il est possible d'en préciser une en particulier :

```
tortuegeniale:/home/djj# ifconfig lo
lo        Link encap:Boucle locale
          inet adr:127.0.0.1  Masque:255.0.0.0
          adr inet6: ::1/128 Scope:Hôte
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:128 errors:0 dropped:0 overruns:0 frame:0
          TX packets:128 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:0
          RX bytes:10370 (10.1 KiB)  TX bytes:10370 (10.1 KiB)
```

Les interfaces réseaux sont généralement nommées **ethN** où N est un nombre distinct par interface. Il existe aussi les interfaces nommées **bondN** lorsque le **bonding** est utilisé. Le **bonding** permet de lier deux interfaces de type ethN pour améliorer la haute disponibilité d'un serveur en cas de panne sur une des interfaces.

```
tortuegeniale:/home/djj# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:1e:4f:ad:5b:67
          inet adr:192.168.100.101  Bcast:192.168.100.255  Masque:255.255.255.0
          adr inet6: fe80::21e:4fff:fead:5b67/64 Scope:Lien
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:173400 errors:4 dropped:0 overruns:0 frame:35
          TX packets:169477 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:1000
          RX bytes:232195063 (221.4 MiB)  TX bytes:17964174 (17.1 MiB)
          Interruption:18
```

ifconfig est utile pour vérifier si une adresse et un masque réseau sont bien configurés sur une interface. Il permet aussi de voir s'il y a des erreurs sur cette interface **errors:** sur les lignes TX et RX.

route

route permet d'afficher comment le serveur communique les autres serveurs (en utilisant quel chemin). Lancez la commande **route -n** pour ne pas avoir la résolution DNS :

```
tortuegeniale:/home/djj# route -n
Table de routage IP du noyau
Destination     Passerelle      Genmask         Indic Metric Ref    Use Iface
192.168.100.0   0.0.0.0         255.255.255.0   U     0      0        0 eth0
192.168.101.0   0.0.0.0         255.255.255.0   U     0      0        0 eth0
0.0.0.0         192.168.100.1   0.0.0.0         UG    0      0        0 eth0
```

Généralement, il y a une route avec pour destination **0.0.0.0** correspondant à la route par défaut utilisée pour contacter tout serveur hors des réseaux du serveur. Dans ce cas, le serveur passe par une passerelle pour contacter les autres serveurs, ici **192.168.100.1** (colonne Gateway).

Il faut comprendre que s'il n'y a pas de route avec '0.0.0.0' comme destination, le serveur ne pourra pas aller sur Internet.

Il y a aussi une route par interface réseau configurée. Par exemple, pour eth0, on voit qu'il y a une route pour le réseau 192.168.101.0/255.255.255.0 dont fait parti tortuegeniale. Dans ce cas, le serveur n'a pas besoin de passer par une passerelle, il contacte directement le serveur.

Il peut y avoir des routes supplémentaires pour des cas particuliers.

Les fichiers de configuration

Debian

La configuration réseau est réalisée avec le fichier **/etc/network/interfaces**. Toutes les interfaces sont configurées dans le même fichier.

Après modification vous pouvez relancer avec ***/etc/init.d/networking restart***. Attention à toute manipulation, la réaliser que si vous êtes sûr de vous. Vous pouvez de plus perdre la connexion. Une autre méthode qui ne touche qu'à une interface est d'utiliser :

```
ifdown eth1; ifup eth1
```

Exemple de configuration :

```
auto eth0
iface eth0 inet static
        address 192.168.100.101
        netmask 255.255.255.0
        network 192.168.100.0
        broadcast 192.168.100.255
        gateway 192.168.100.1
        dns-nameservers 192.168.101.4
        dns-search dj-j.net
        dns-domain dj-j.net
        up /usr/local/scripts/fullduplex.sh eth0
```

- auto indique que l'interface est montée avec le script de démarrage /etc/init.d/networking. **allow-hotplug eth0** peut être mis à la place de auto eth0. Dans ce cas, le restart ne fonctionnera pas mais cela permet de reconfigurer après un rebranchement de câble.
- iface eth0 inet static indique que l'interface réseau eth0 est configuré de manière manuelle (le contraire de DHCP).
- dns-nameservers et dns-search modifie **/etc/resolv.conf** au démarrage
- up permet de lancer un script lorsque l'interface est montée. Il est possible d'utiliser down

pour lancer un script lorsque l'interface est démontée.

Il ne doit y avoir qu'une seule fois **gateway**.

Red-Hat

La configuration réseau est réalisée avec le fichier **/etc/sysconfig/network-scripts/ifcfg-ethN** où N est le numéro de l'interface. Il y a un fichier par interface.

La gateway est configurée dans le fichier **/etc/sysconfig/network**.

Après modification vous pouvez relancer avec ***/etc/init.d/network restart***. Attention à toute manipulation, la réaliser que si vous êtes sûr de vous. Vous pouvez de plus perdre la connexion. Une autre méthode qui ne touche qu'à une interface est d'utiliser :

```
ifdown eth1; ifup eth1
```

Exemple pour **/etc/sysconfig/network-scripts/ifcfg-eth1**

```
DEVICE=eth1
BOOTPROTO=none
HWADDR=00:04:23:D0:86:8D
ONBOOT=yes
TYPE=Ethernet
NETMASK=255.255.255.240
IPADDR=10.100.1.114
ETHTOOL_OPTS="speed 100 duplex full autoneg off"
```

- DEVICE correspond au nom de l'interface réseau configurée
- HWADDR indique l'adresse MAC de l'interface
- ONBOOT=yes indique que l'interface est montée au démarrage
- ETHTOOL_OPTS="speed 100 duplex full autoneg off" indique qu'il faut configurer l'interface avec ethtool pour qu'elle soit forcée en Full Duplex au montage de celle-ci.

Connexion SSH

Dans cette section, nous ne parlerons pas de Putty utilisé sous Windows :-p ni de la configuration du serveur SSH.

Les deux commandes principales pour réaliser des connexions et transferts de fichiers par SSH sont **ssh** et **scp**.

Nous ne parlerons pas de la connexion par clefs ssh pour l'instant car il s'agit d'un mode avancé.

Connexions distantes

Pour se connecter à un autre serveur, il suffit de lancer la commande suivante :

```
ssh serveur.dj-j.net
```

Dans ce cas, ssh se connecte avec l'utilisateur courant. Si vous êtes connecté en root, il se connectera en root sur le serveur distant et vous demandera le mot de passe root du serveur distant. Si vous êtes connecté en tant qu'utilisateur et que vous souhaitez utiliser un autre compte sur le serveur distant il faut préciser ainsi :

```
ssh login@serveur.dj-j.net
```

Exemple, je suis connecté sur ***tortuegeniale*** avec mon utilisateur **djj** et je souhaite me connecter sur krilin.dj-j.net en tant que root :

```
djj@tortuegeniale:~$ ssh root@krilin.dj-j.net
The authenticity of host 'krilin.dj-j.net (192.168.100.2)' can't be established.
RSA key fingerprint is 38:39:db:e6:c0:03:42:dc:f2:f3:89:0c:81:f7:5d:ee.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'krilin.dj-j.net' (RSA) to the list of known hosts.
root@krilin.dj-j.net's password:
Linux krilin 2.6.26-1-686 #1 SMP Sat Jan 10 18:29:31 UTC 2009 i686
...
Last login: Tue Sep  8 21:56:48 2009 from bulma.dj-j.net
```

La première fois que vous vous connectez à un serveur, il vous demande de valider la clef serveur, répondre **yes** :

```
The authenticity of host 'krilin.dj-j.net (192.168.100.2)' can't be established.
RSA key fingerprint is 38:39:db:e6:c0:03:42:dc:f2:f3:89:0c:81:f7:5d:ee.
Are you sure you want to continue connecting (yes/no)? yes
```

Lancer des commandes à distance

Il est possible par l'intermédiaire de ssh, de lancer des commandes sur un autre serveur. Il faut utiliser une commande du type :

```
ssh login@serveur.dj-j.net "commande distante"
```

Exemple :

```
ssh root@krilin.dj-j.net "cat /etc/hostname"
krilin.dj-j.net's password:
krilin
```

On voit bien que le hostname affiché est bien celui du serveur distant.

Transferts de fichiers

Pour réaliser des transferts par ssh, il faut utiliser la commande ***scp***. La syntaxe est proche de **cp**.

```
scp source destination
```

Il est possible de faire des transferts dans les deux sens, par exemple :

```
scp fichier_source root@serveur.dj-j.net:/dossier_distant/
```

ou

```
scp root@serveur.dj-j.net:/fichier_distant dossier_local/
```

On remarque que pour indiquer que la source ou la destination est sur le serveur distant, il suffit de mettre login@serveur.dj-j.net suivi de **:** puis le fichier ou dossier concerné.

Si vous ne précisez pas la racine / après le : alors le chemin indiqué est relatif au dossier personnel de l'utilisateur de la connexion.

Exemple : Transférons le fichier de log syslog de smoky10 dans le dossier courant de pcw :

```
djj@tortuegeniale:~$ scp root@krilin.dj-j.net:/var/log/syslog .
root@krilin.dj-j.net's password:
syslog                                        100%  164KB 164.0KB/s   00:00
```

Serveur Web Apache

Voir la page dédiée Linux:Administration_d'Apache2

Explorer les logs, rechercher des informations en s'aidant de grep, vi, tail, head et more

Comme vu plus haut, nous pouvons utiliser plusieurs commandes les unes imbriquées à la suite des autres afin de réaliser des recherches plus précises.

Nous allons voir quelques exemples dans cette section.

- Par exemple, recherchons si apache a atteint le nombre maximum de connexions autorisées :

```
grep MaxClients /var/log/apache2/*log
```

Dans ce cas, uniquement les logs courants sont pris en compte et pas ceux des jours précédents.

- Recherchons les erreurs WARNING sur un serveur de mail :

```
grep warning mail.log
```

Il y a beaucoup trop de résultat, dont une ligne du type :

```
Jul 20 09:05:32 sangoku postfix/smtpd[17920]: warning: 82.166.8.146: hostname
82-166-8-146.barak-online.net verification failed: Name or service not known
```

Supprimons là avec **grep -v** :

```
grep warning mail.log|grep -v "verification failed: Name or service not known"
```

Il y a encore les erreurs "address not listed for hostname localhost" qui ne nous intéressent pas, nous pouvons cumuler le grep -v :

```
grep warning mail.log|grep -v "verification failed: Name or service not known"|
grep -v "address not listed for hostname localhost"
```

Nous aimerions voir que les 10 premiers lignes, ajoutons **head** :

```
grep warning mail.log|grep -v "verification failed: Name or service not known"|
grep -v "address not listed for hostname localhost"|head
```

Au contraire, nous aimerions les 100 derniers résultats, utilisons alors **tail** :

```
grep warning mail.log|grep -v "verification failed: Name or service not known"|
grep -v "address not listed for hostname localhost"|tail -n 100
```

Dans notre grep, comme il y a beaucoup de warning, le traitement est long. Il est possible pour aller plus vite de d'abord limiter à 100 lignes mail.log puis filtrer, mais le résultat n'est pas le même car il n'y avait peut être que des warning "verification failed" dans les 100 dernières lignes de mail.log. Nous aurons alors aucun résultat (c'est d'ailleur le cas lors tu testes pour la rédaction du document).

```
tail -n 100 mail.log |grep warning |grep -v "verification failed: Name or
service not known"| grep -v "address not listed for hostname localhost"
```

Si vous avez beaucoup de résultat et que vous voulez tous les consulter, vous pouvez ajouter **|more** à la fin :

```
grep warning mail.log|grep -v "verification failed: Name or service not known"|
grep -v "address not listed for hostname localhost"|more
```

Une autre solution qui peut être pratique pour effectuer d'autres recherches ensuite est de renvoyer le résultat dans un fichier et ensuite le lire avec vi :

```
grep warning mail.log|grep -v "verification failed: Name or service not known"|
grep -v "address not listed for hostname localhost" >
/tmp/resultat_recherche.txt
```

Donc édition avec vi :

```
vi /tmp/resultat_recherche.txt
```

Recherchons maintenant la chaine MAIL :

```
/MAIL
```

vi nous envoie à la premier occurrence de MAIL. Si vous souhaitez voir la prochaine, il suffit de taper sur la touche **n** (next). Pour revenir au résultat précédent, utiliser **N**.

Nous souhaitons encore enlever des lignes trop récurrentes, toujours dans vi :

```
:g/.*Illegal address .*/d
```

Nous parcourons tout le fichier pour supprimer toutes les lignes incluant "Illegal address ". Notez bien que l'espace après address est pris en compte dans la recherche de la chaine.

Les caractères spécifiques ^ $ permettent de préciser la recherche.

```
^ : pour le début de ligne
$ : pour la fin de ligne

.* : indique n'importe quel caractère avec une occurrence de 0 à n (n est
indéterminé).
```

Par exemple, supprimons les lignes dont l'heure est 15:4x :

```
:g/^Jul 20 15:4.*/d
```

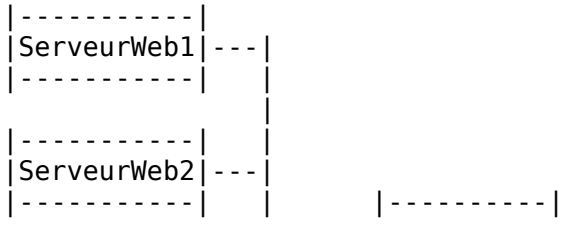
Ou alors les lignes terminant par nameservices.net :

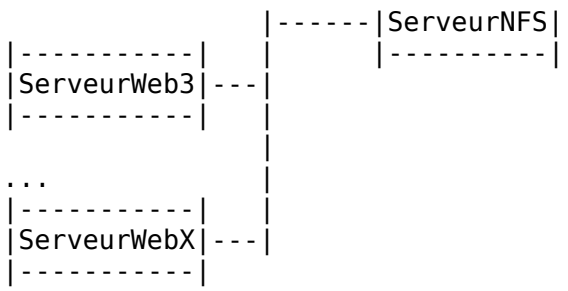
```
:g/.*nameservices.net$/d
```

NFS - système de fichiers par le réseau

NFS est le "Network File System", système de fichiers par réseau le plus utilisé sous Linux. Il permet d'accéder à des dossiers/fichiers distants comme s'il s'agissait de dossiers/fichiers du système local.

L'utilisation courante de NFS est de partager les données de sites web à plusieurs serveurs. Ainsi, il est possible de répartir la charge avec plusieurs serveurs, et lorsqu'une modification est faire sur un des serveurs sur le site alors tous les serveurs voient la modification. Il n'est pas nécessaire de mettre en place un système de synchronisation. Il est alors nécessaire d'avoir un serveur ayant le rôle de serveur NFS ou d'utiliser un espace sur une baie de stockage qui dispose du service de serveur NFS.





Côté serveur

Sur le serveur, il y a portmap et nfsd. Le fichier de configuration du serveur nfs est ***/etc/exports***.

```
/home/network/djj               192.168.100.111(ro,sync)
192.168.100.101(rw,sync) 192.168.100.102(rw,sync) 192.168.100.223(ro,sync)
```

Il y a d'abord le dossier qui est disponible par NFS puis la liste des serveurs qui ont l'autorisation d'accès avec des options mises entre parenthèses. Vous pouvez donner des droits différents par serveur.

Deux options courantes :

- rw/ro : permet d'indiquer si le serveur a les accès en écriture ou uniquement en lecture : ReadWrite ou ReadOnly
- no_root_squash : par défaut, pour une raison de sécurité, le root du serveur client n'a pas les accès d'écriture même avec l'option rw, en précisant no_root_squash, il a alors les droits.

Une fois une modification réalisée ou un dossier ajouté, il faut mettre à jour avec la commande :

```
exportfs -ra
```

Côté client

Sur le serveur client, il suffit de configurer le dossier distant dans **/etc/fstab** à la manière d'un disque local.

Avec la commande **mount', on voit qu'il y a :**

```
192.168.100.2:/home/network/djj on /home/djj/Documents type nfs
(rw,nfsvers=3,tcp,addr=192.168.100.2)
```

Dans le fichier /etc/fstab, il y a :

```
192.168.100.2:/home/network/djj        /home/djj/Documents nfs
defaults,nfsvers=3,tcp    0     0
```

En première colonne, il y a comme système de fichier distant le nom du serveur puis **:** puis le dossier configuré au niveau du serveur NFS.

En deuxième colonne, il y a le dossier local utilisé pour le point de montage. Ensuite il y a le type de système de fichier, il suffit d'indiquer **nfs**.

Il n'y a pas de 6ième colonne car il ne faut pas réaliser de fsck au démarrage sur un montage NFS. Il peut y avoir le 6ième champs s'il vaut 0.

Nous ne détaillerons pas les options.

Fonctionnement du système

A propos de la gestion mémoire

Contrairement à ce que l'on peut penser à première vue, Linux n'utilise pas toute la mémoire "pour rien". Si vous regardez la mémoire libre sur votre système avec top, vous verrez une information du type :

```
Mem:   2070900k total,  1878068k used,   192832k free,    36164k buffers
```

Il n'y a pas uniquement 192832k de libre, la mémoire utilisée l'est en grande partie par un cache du disque. Mais cette mémoire est rapidement disponible en cas de besoin par une application.

```
free -m
             total       used       free     shared    buffers     cached
Mem:          2022       1836        185          0         35       1060
-/+ buffers/cache:        741       1280
Swap:         4769         18       4751
```

Nous voyons plus précisément qu'il y a 1060M utilisé par ce cache. Aussi le total de la mémoire n'inclue pas la mémoire utilisée par le noyau et certains périphérique, c'est pourquoi il y a moins de 2048 dans cette exemple.

La ligne -/+ buffers/cache indique les valeurs sans les buffers et caches.

- **VIRT RES SHR**

Informations d'utilisation mémoire dans top :

```
 PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
3453 djj       20   0  379m 130m  25m S   10  6.4   6:16.13 firefox-bin
```

  - VIRT : la mémoire virtuelle d'un processus correspond à sa mémoire allouée, les fichiers sur disque allouée (bibliothèques partagées) et la mémoire partagée avec d'autres processus,
  - RES : la mémoire réellement utilisée par un processus
  - SHR : la mémoire partageable avec d'autres processus

Un programme utilisant uniquement certaines fonctions d'une librairie verra compter toute la librairie dans VIRT et SHR mais uniquement la fonction dans RES.

- Utilisation du swap

Le noyau a deux possibilité lorsqu'il n'y a plus de mémoire disponible, supprimer du cache les données anciennes ou déplacer sur swap les portions de mémoire des programmes les moins utilisées. Celui-ci essaie d'effectuer le meilleur choix en fonction de l'activité récente du système.

Depuis les noyau 2.6, il est possible de jouer sur la **Swappiness** pour réaliser plutôt une mise en swap ou un nettoyage du cache. Par défaut :

```
cat /proc/sys/vm/swappiness
60
```

Si vous souhaitez utiliser plus le swap, augmenter la valeur, si vous voulez limiter l'utilisation du swap, diminuez la.

```
echo 80 > /proc/sys/vm/swappiness
```

Dans /etc/sysctl.conf :

```
vm.swappiness = 80
```

La documentation

Sous linux, il y a plusieurs sources de documentations installées sur le serveur.

Il y a les **man**, les documentations dans **/usr/share/doc** et les options de commandes indiquées par celles-ci.

man

Il y a 9 sections dans le man :

```
1 : commandes utilisateurs
2 :  appels systèmes
3 : fonctions des bibliothèques
4 : périphériques (/dev/*)
5 : formats de fichiers de configuration
6 : jeux
7 : divers
8 : outils disponibles uniquement à root
9 : spécifique à Linux pour les services du noyau
```

man renvoie de base la première documentation disponible par ordre croissant. Pour spécifier une section, il suffit d'indiquer le numéro de section.

Par exemple, pour le fichier de configuration ***/etc/passwd*** il faut demander la section 5, sinon on obtient la documentation de la commande passwd :

```
man 5 passwd
```

si on ne connait pas exactement la commande :

```
man -k mot clef : man renvoie une liste avec des manuels correspondants.
```

/usr/share/doc

Contrairement aux documentations man, les documentations dans ce dossier non sont pas toujours installées. Il s'agit soit de fichier textes soit de pages HTML. Pour avoir une documentation sur un package, il faut aller tout simplement dans le sous-dossier du même nom.

Par exemple, nous cherchons de la documentation sur postfix. Le dossier postfix-doc contient des exemples au format texte compressé avec gzip. Vous pouvez l'ouvrir directement avec vim :

```
/usr/share/doc/postfix-doc/examples# ls -l
total 76
-rw-r--r-- 1 root root 5257 2006-07-12 02:17 access.gz
-rw-r--r-- 1 root root 3280 2006-02-03 02:16 aliases.gz
-rw-r--r-- 1 root root 3550 2008-08-19 07:51 bounce.cf.default
-rw-r--r-- 1 root root 3700 2006-07-12 02:17 canonical.gz
-rw-r--r-- 1 root root 3474 2006-07-12 02:17 generic.gz
-rw-r--r-- 1 root root 5068 2005-12-02 16:43 header_checks.gz
lrwxrwxrwx 1 root root   10 2009-04-24 09:08 LICENSE -> ../LICENSE
-rw-r--r-- 1 root root 5007 2008-08-19 07:51 main.cf.default.gz
-rw-r--r-- 1 root root 1017 2008-08-19 07:50 makedefs.out
-rw-r--r-- 1 root root 2212 2008-08-19 07:50 postfix-script.gz
-rw-r--r-- 1 root root 6148 2007-01-07 00:38 post-install.gz
drwxr-xr-x 2 root root 4096 2008-10-24 22:09 qmail-local
-rw-r--r-- 1 root root 2478 2006-07-12 02:17 relocated.gz
drwxr-xr-x 2 root root 4096 2008-10-24 22:09 smtpd-policy
lrwxrwxrwx 1 root root   14 2009-04-24 09:08 TLS_LICENSE -> ../TLS_LICENSE
-rw-r--r-- 1 root root 3867 2006-07-12 02:17 transport.gz
-rw-r--r-- 1 root root 3951 2006-07-12 02:17 virtual.gz
```

```
vim aliases.gz
```

On remarquera que sur le serveur où se trouve ces docs, un package spécifique est installé :

```
dpkg -l|grep postfix
ii  postfix                                  2.3.8-2+etch1
A high-performance mail transport agent
ii  postfix-doc                              2.3.8-2+etch1
Postfix documentation
ii  postfix-mysql                            2.3.8-2+etch1
MYSQL map support for  Postfix
ii  postfix-pcre                             2.3.8-2+etch1
PCRE map support for Postfix
```

Options de commandes

Pour obtenir les options d'une commande, elles sont généralement disponible ainsi :

```
commande -h
```

ou

```
commande --help
```

Exemple :

```
vim --help
VIM - Vi IMproved 7.0 (2006 May 7, compiled Feb 18 2009 14:51:48)

utilisation : vim [args] [fichier ...]  ouvrir le ou les fichiers spécifiés
   ou : vim [args] -               lire le texte à partir de stdin
   ou : vim [args] -t marqueur     ouvrir le fichier qui contient le marqueur
   ou : vim [args] -q [fichErr]    ouvrir à l'endroit de la première erreur

Arguments :
  --           Seuls des noms de fichier sont spécifiés après ceci
  -v           Mode Vi (comme "vi")
  -e           Mode Ex (comme "ex")
  -s           Mode silencieux (batch) (seulement pour "ex")
  -d           Mode diff (comme "vimdiff")
...
```

Souvent, une commande nécessitant des options ou des arguments affichera un résumé de ceux-ci si vous lancez la commande seule. Exemple :

```
postfix
postfix/postfix-script: fatal: usage: postfix start (or stop, reload, abort,
flush, check, set-permissions, upgrade-configuration)
```

L'éditeur de fichier Vim

Voir la documentation dédiée à VIM Linux:Utilisation_de_vim