# Formation Linux avancée

Sabrina DUBROCA <zappy@via.ecp.fr>
Thomas RICATTE <elenaher@via.ecp.fr>
Valentin ROUSSELLET<louen@via.ecp.fr>

9 juin 2008

# Table des matières



# 1 Introduction

Ce document résume ce qui a été dit lors des formations VIA-CTI organisées les lundi 26 mai et 2 juin. Elle commence là où la formation précédente(découverte de Linux) s'était arrêtée. Vous pouvez consulter les diapositives sur http ://people.via.ecp.fr/ louen/formations/Formation_linux.pdf

Le but de cette formation est de donner un aperçu de ce que peut être l'administration d'un serveur Linux, et d'apprendre quelques tâches simples (installer des programmes, modifier un fichier de configuration ... Il est bien entendu toujours possible de s'addresser à VIA ou au CTI pour toute question ou pour en savoir plus.

# 2 La distribution Debian

**Qu'est-ce que Debian ?** Debian est une distribution Linux reconnue pour sa stabilité, elle est donc choisie pour de nombreux serveurs (qui se doivent de rester en fonctionnement en permanence ). Elle est assez proche d'Ubuntu, la distribution vue dans la formation Linux de base (Ubuntu est en réalité un dérivé de Debian.

**L'outil APT** Un autre avantage de Debian est la facilité avec laquelle on installe les logiciels, grâce au système APT (Advanced Package Tool). APT s'occupe de tout : il installe le logiciel préparé par les développeurs de Debian sous la forme de "paquet", et il vérifie tout seul s'il faut installer des paquets supplémentaires dont le programme a besoin (les dépendances). Nous allons tout de suite l'utiliser pour installer notre premier logiciel : apache.

**Un petit rappel** Au cours de cette formation, nous n'utiliserons que la ligne de commande. Avec Linux, on peut absolument tout faire avec la ligne de commande, et notre serveur n'a pas besoin de s'encombrer d'une interface graphique qui le ralentirait, et cela permet de plus de l'administrer à distance par SSH. On pourra se reporter à la formation Linux de base pour les commandes les plus utiles. On a vu, au cours de cette formation, qu'il existe un utilisateur particulier, appelé root, qui possède tous les droits sur le système. Comme nous apprenons à administrer le serveur, la plupart des actions seront effectuées en tant que root. Cependant, il est recommandé de n'utiliser le compte root que lorsque cela est nécéssaire pour éviter de faire des erreurs (supprimer un fichier vital par exemple) et pour des raisons de sécurité. A partir de maintenant, je considère que vous êtes sous le compte root (accessible à l'aide de la commande su)

# 3 Installation d'apache 2

**Le serveur Web apache 2** Apache est le serveur web le plus utilisé dans le monde : c'est un programme qui tourne en permanence sur le serveur. Il écoute la connexion entrante sur le serveur, attend qu'un client demande à voir une page web, et envoie la page en réponse. Nous allons tout de suite l'installer avec APT en utilisant la commande :
`apt-get install apache2` Le serveur nous dit alors qu'il y a des dépendances à installer en plus de apache2. On lui répond oui sans hésiter.

**Configuration d'apache 2** Apache 2 permet d'activer certains modes. Par exemple, le mode "userdir" permet à chaque utilisateur de disposer d'un dossier accessible comme une page web. Il lui suffit pour cela de créer dans son répertoire personnel (/home/toto pour l'utilisateur toto) un dossier appelé public_html. Il sera alors accessible à l'addresse http ://monserveur.com/ toto.
Nous voulons donc activer ce mode, ce qui se fait à l'aide de la commande
`a2enmod userdir`
"a2enmod" signifie Apache2 ENable MODe. Un message nous demande de redémarrer apache 2 pour que la modification soit effective.

# 4 Arrêter et démarrer des services

**Les commandes init.d** Pour redémarrer apache 2 nous allons appeler un programme spécial dans le dossier /etc/init.d. Ce dossier contient un fichier par daemon (un daemon est un programme qui tourne en permanence sur le serveur) qui permet de le commander.
Par exemple la commande
`/etc/init.d/apache2 stop`
arrête apache 2, qui ne répondra alors plus aux demandes de pages web.
A l'inverse
`/etc/init.d/apache2 start`
démarre apache si celui-ci est arrêté.
et on peut faire les deux à la suite (arrêt - démarrage) avec
`/etc/init.d/apache2 restart`

Ce que le message nous a demandé de faire, donc on le fait.
A présent, le mode userdir est activé sur notre serveur !

# 5 Configuration d'Apache 2

Nous allons maintenant modifier un peu la configuration d'Apache 2 pour que notre serveur puisse héberger un ou plusieurs sites (pouvant être différents). Supposons, pour l'exemple que nous voulons avoir sur notre machine le site du projet toto, à l'addresse toto.com ; et que nous voulons en plus avoir notre blog personnel monblog.com sur le même serveur. Nous allons utiliser une fonction d'apache appelée virtual host. En gros, nous allons dire à apache où trouver les fichiers correspondant à chaque site installé.

**Le premier site** Supposons que les fichiers du site toto.com (les fichiers HTML ) soient dans le dossier /home/toto/site. Nous allons donc créer le fichier correspondant au premier virtualhost avec la commande touch qui crée un fichier vide.
`touch /etc/apache2/sites-available/toto.com`
A présent on ouvre le fichier avec vim, un éditeur de texte en console
`vim /etc/apache2/sites-available/toto.com`
Pour modifier le fichier dans vim, on appuie sur la touche i (comme Insertion)
On écrit alors

```
<VirtualHost * >
ServerName toto.com
DocumentRoot /home/toto/site
</VirtualHost>
```

Ce qui signifie, pour apache : le serveur s'appelle toto.com et le dossier correspondant est dans /home/toto/site. On enregistre et on quitte vim en appyant sur echap pour quitter le mode insertion, puis en tapant
`:wq`
Pour write & quit.
Nous allons maintenant activer le site, ce qui va le rendre accessible.
`a2ensite toto.com`
Il faut maintenant recharger la configuration d'Apache à l'aide de la commande
`/etc/init.d/apache2 reload`
Reload est une commande qui fait relire à apache ses fichiers de configuration (sans toutefois le redémarrer), ce qui n'interrompt pas le service.

**Le deuxième site** On répète les mêmes opérations que pour le premier, les fichiers de monblog.com se trouvant dans le dossier /home/dupont/blog :
`touch /etc/apache2/sites-available/monblog.com`
`vim /etc/apache2/sites-available/monblog.com`
On insère dans ce fichier

```
<VirtualHost * >
ServerName monblog.com
DocumentRoot /home/dupont/blog
</VirtualHost>
```

et on quitte avec echap+ :wq, puis

```
a2ensite toto.com
/etc/init.d/apache2 reload
```

# 6 Quelques bons réflexes d'administrateur système

**Vérifier les logs** Pour être sûr du bon fonctionnement d'un programme, ou pour savoir ce qui se passe en cas d'erreur, le premier réflexe, c'est d'aller voir les logs. A chaque fois qu'un programme fait quelque chose d'important, il l'écrit dans un fichier. Ainsi, on peut consulter la liste des derniers messages pour vérifier que tout va bien ou pour essayer de savoir ce qui a causé une erreur.
Les logs se trouvent tous dans le dossier /var/log. Par exemple, nous allons voir dans le dossier /var/log/apache2 pour trouver les fichiers logs d'apache 2. Il y en a normalement deux, access.log qui liste toutes les requêtes auxquelles apache 2 répond (on peut ainsi savoir précisément qui a visité votre site à quelle heure) et error.log sur lequel sont écrits les messages d'erreur.
Il existe un fichier particulier dans /var/log : le syslog. C'est le fichier de log du noyau, le coeur du système. On le regarde généralement en cas de gros plantage pour savoir ce qui s'est passé. De manière générale, en cas d'erreur, regarder les derniers messages de log peut donner une indication sur l'origine du plantage. Même si on y connaît rien, on peut toujours rechercher le texte du message d'erreur dans Google : d'autres utilisateurs ont sûrement déjà eu une erreur similaire.

**Faire des mises à jour** Le fait que les logiciels que nous utilisons soient open source procure d'énormes avantages. Mais le fait que le code source soit à disposition de n'importe qui permet à n'importe qui d'y trouver des erreurs. La plupart du temps, les gens capables de déceler ces erreurs en font part aux développeurs du programme qui corrigent la faute dans une nouvelle version du programme. La mise à jour est annoncée sur des mailing-lists prévues à cet effet. Le problème, c'est qu'ainsi tout le monde est prévenu de l'erreur en question, et des pirates pourraient l'exploiter pour endommager un système qui aurait l'ancienne version du programme.
Heureusement avec Debian, les mises à jour sont très faciles. Il suffit de taper successivement deux commandes :

```
apt-get update
apt-get upgrade
```

Le système mettra automatiquement à jour les programmes qui en ont besoin sans que l'on aie tout à réinstaller. En restant à jour, on se prémunit de bon nombre de problèmes !

**Les sauvegardes** Personne n'est à l'abris d'une panne. Coupure de courant, accident ou erreur, les données contenues sur notre serveur peuvent être endommagées, voire irrécupérables. Il est donc très important de faire des sauvegardes régulières des fichiers importants de votre système. Les fichiers de configuration du /etc sont très importants par exemple, ainsi que les données personnelles

des utilisateurs situées dans le /home. Il est conseillé de copier souvent ces dossiers sur des disques durs physiquement séparés de votre serveur. Vous pouvez par exemple utiliser un disque dur externe et copier ces dossiers à l'aide de la commande
`cp -r /chemin/du/dossier1 /chemin/du/dossier2`
qui copiera l'ensemble du dossier 1 vers le dossier 2
On peut aussi utiliser la commande scp pour copier les dossiers vers un autre serveur à distance :
`scp -r /chemin/du/dossier1 dupont@sauvegardes.net :/chemin/du/dossier2`
Cette commande copiera l'ensemble du dossier 1 vers le dossier 2 situé sur le serveur sauvegardes.net. Bien entendu, on vous demandera le mot de passe de dupont sur ce serveur, et on ne pourra écrire que dans un dossier dans lequel dupont a le droit d'écrire. Nous verrons un peu plus loin comment automatiser les sauvegardes.

# 7 Installation de MySQL et PHPMyAdmin

Nous allons installer MySQL, un serveur de bases de données.
`apt-get install mysql-server`
MySQL gère des bases de données relationelles. Il dispose d'un système d'utilisateurs et de droits comme un système Linux. Il y a un compte "root" qui possède tous les droits sur les bases de données, et on peut ajouter de nouveaux comptes avec des droits limités. De base, le compte root a un mot de passe vide, alors on va vite changer ça. MySQL s'administre en ligne de commande. On lance le programme :
`mysql`
L'invite de commande mysql s'affiche alors. On peut taper dans cette interface des commandes SQL pour créer ou modifier les bases de données ou faire des requêtes (select). Cependant, il faut convenir que ce n'est pas très pratique. Heureusement il exite un programme appelé PHPMyAdmin qui permet d'administrer ses bases de données au travers d'une interface web. Nous allons donc l'installer
`apt-get install phpmyadmin`
Notez que parmi les dépendances, apt installe le mode php5 pour apache2 et qu'il l'active tout seul.
A présent, les dossiers de phpmyadmin sont dans le dossier /var/www/phpmyadmin. Le dossier /var/www est celui où redirige apache2 lorsqu'il n'y a pas de virtualhost configuré. Pour y accéder, il faut se rendre sur l'addresse http ://monserver.net/phpmyadmin (s'il n'y a pas de virtualhost sur monserveur.net.).
De là, on se logge en root sans mot de passe, et on utilise l'interface qui est assez intuitive pour rapidement créer un mot de passe pour root, puis faire les bases de données dont on a besoin.

# 8 Le shell et les scripts shell

La ligne de commande est un outil extrêmement pratique car il s'agit en fait d'un véritable langage de programmation.

Lorsqu'on maîtrise les commandes, on peut faire des modifications bien plus rapidement qu'avec une interface graphique. Par exemple si je veux savoir dans un dossier de 200 fichiers combien sont de type JPEG, c'est possible en une seule ligne.
L'efficacité de la ligne de commande est due aux redirections d'entrée sortie qui permettent d'exploiter le résultat d'une commande dans une autre commande ou dans un fichier.
Par exemple, la commande mysqldump est utilisée pour faire des sauvegardes des bases de données. Elle écrit à l'écran les instructions SQL nécéssaires pour reconstruire la base et ses données.
`mysqldump -all databases`
Bien entendu, avoir ce tas d'instructions à l'écran ne nous est pas très utile. On voudrait le sauvegarder dans un fichier !
Pour celà, nous allons rediriger la sortie du programme qui s'affiche à l'écran (appelée *sortie standard* vers un fichier à l'aide du chevron.
`mysqldump -all-databases > sauvegarde.sql`
Cette commande n'affiche rien. Par contre, elle créée un fichier "sauvegarde.sql" et elle écrit le résultat de la commande mysqldump à l'intérieur.
Maintenant, supposons que je veuille que le résultat me soit envoyé par mail. On utilise pour cela la commande mail.
`mail -s "base de donnees" louen@via.ecp.fr <sauvegarde.sql`
Cette commande utilise le chevron inverse $<$ qui passe le contenu du fichier sauvegarde.sql en entrée de la commande mail. Le résultat sera que le contenu du fichier sera envoyé à l'addresse mail spécifiée avec le sujet "base de donnees".
De même, pour reconstruire la base de données avec notre fichier de sauvegardes (puisque c'est une suite de commandes SQL) : `mysql -u root -p <sauvegarde.sql`
Enfin, on peut combiner les deux à l'aide de | (le "pipe") qui redirige la sortie de la commande à sa gauche sur l'entrée de celle à sa droite.
`mysqldump -all-databases | mail -s "base de données" louen@via.ecp.fr`
Cette commande enverra directement à louen@via.ecp.fr le résultat de la commande mysqldump (sans passer par un fichier). Il est possible ainsi d'enchaîner les commandes avec des | pour obtenir le résultat souhaité.

**les scripts shells** On peut également écrire des véritables programmes en lignes de commandes. Ouvrons un nouveau fichier :
`vim script.sh`
On passe en mode d'insertion (avec i).
Dans ce fichier, nous allons écrire une suite de lignes de commandes pour qu'elles soient exécutées les unes après les autres.
Par exemple, nous voulons automatiser la sauvegarde de notre base de données.
Il nous faut :
– supprimer l'ancien fichier de sauvegarde
– créer le nouveau fichier
– l'envoyer par SCP sur un autre serveur
Le script ressemble donc à ça

```
#!/bin/sh
rm -f /home/admin/sauvegarde.sql
mysqldump -all-databases > sauvegarde.sql
```

```
scp sauvegarde.sql moi@monserveur.com :/home/moi/sauvegardes
```

La première ligne est une indication pour dire au système en quel language est écrit le script (un script en python commence par #!/usr/bin/python).
Ensuite, le script exécutera les instructions à la suite (sauf s'il rencontre une erreur, dans ce cas il s'arrête : par exemple si le fichier sauvegarde.sql n'existe pas, il ne pourra être supprimé). On quitte vim (echap + :wq). Maintenant, pour le tester, il suffit de le rendre exécutable avec
`chmod 744 script.sh`
Puis de lancer l'exécution du script par
`./script.sh`
Il nous faudra bien sûr avoir les mots de passes nécéssaires. Maintenant, nous pouvons passer à la dernière étape qui est d'automatiser l'exécution de notre script avec *cron* qui va nous permettre de faire exécuter le script à une heure fixe tous les jours, par exemple. (En réalité, pour réaliser cela, nous avons besoin de nous débarrasser des passages où le système nous demande un mot de passe avec des techniques qui sortent largement du cadre de cette formation).
Nous ouvrons la *crontab* avec
`crontab -e`
Nous voyons s'afficher à l'écran
`# m h dom mon dow command`
Signifiant respectivement minutes, heures, jour du mois (day of month), mois (month) et jour de la semaine (day of week).
Pour remplir la crontab, on doit mettre quelque chose dans chaque colonne : une valeur ou une *, et dans la dernière colonne, l'action à accomplir.
Décryptons ces deux lignes de crontab :
`# m h dom mon dow command 00 2 5 * * /root/script1.sh 05 * * 6 01 /root/script2.sh`
La première ligne est exécutée à 2 heures du matin tous les 5 du mois. La deuxième est exécutée toutes les heures (il y a une * dans la case des heures), mais seulement en juin (6 dans la case mois) et seulement les lundi (01 dans la case jour de la semaine).
Nous voulons que notre sauvegarde se fasse tous les jours à 6h du matin : on met
`00 6 * * * /root/script.sh` Puis, comme toujours, echap + :wq pour quitter.

# 9 Conclusion

Bien entendu, l'administration Linux est un sujet très vaste, et des livres entiers y sont consacrés. Nous allons terminer sur deux choses

**Les règles d'or du Linuxien** . S'il n'y a que deux commandes à retenir :
`man commande` vous donne le manuel d'une commande. Si vous ne vous souvenez plus de la syntaxe de telle ou telle commande, c'est là qu'il faut regarder. C'est parfois imbitable (essayez `man ps` pour voir), mais c'est extrêmement pratique.
`"man Google"` : chercher sur le Web si vous rencontrer un problème vous permet souvent de clarifier la situation : d'autres utilisateurs ont sûrement déjà rencontré ce problème !

Enfin, n'oubliez pas que les membres de VIA et du CTI seront sûrement heureux de vous aider à progresser avec Linux. N'hésitez pas à envoyer vos questions à **perms@via.ecp.fr** !