Administration système GNU/Linux
Synthèse

Thibault Vanwersch
thibault@vanwersch.fr

Version du 13 janvier 2016

"Contrary to popular belief, Unix is user friendly. It just happens to be very selective about who it decides to make friends with." *Anonymous*

"La convivialité d'un système Unix réside dans la souplesse et la puissance des outils dont on dispose pour dialoguer avec le système. Il s'agit certes d'une convivialité à laquelle peu d'utilisateurs sont habitués ; essentiellement parce qu'elle demande un investissement sur le plan de la documentation et de l'apprentissage." *Vincent Lozano*[1]

Table des matières

1. dans "Pour aller plus loin avec la ligne de commande." Framabook

Introduction

Ce document a pour but de fournir un support théorique non-officiel au cours d'exercices d'administration Linux. Puisse-t-il vous être utile et vous éviter de finir complètement dégouté par Linux. Vous allez voir, c'est fun <3 !

Les commandes reprises ici (et dans plein de documentation sur internet) sont divisées en deux catégories : les commandes exécutées par un utilisateur normal et celles exécutées par le root. Les commandes normales sont précédées du symbole $, les commandes root d'un #.
Exemple :

```
# ceciEstUneCommandeRoot
```

```
$ ceciEstUneCommandeUtilisateurNormal
```

Ce document contient des références au manuel. Celles-ci sont incomplètes, je sélectionne l'information que je vous montre, n'hésitez pas à consulter le vrai manuel [2].

Il est possible que "votre man" ne fournisse pas le même contenu que mes copier-coller, il peut même ne pas être dans la même langue mais vous êtes une grande fille / un grand garçon, vous allez vous en sortir malgré tout, je n'en doute pas.

Pour maîtriser un système Linux (ou tout du moins pour espérer réussir ce cours), vous devez *absolument* être à même de trouver rapidement une information dans le manuel. Même lors de l'examen, vous ne devez en aucun cas hésiter à le consulter. Pour rappel, la touche "/" vous permet de taper une recherche, vous pourrez ensuite naviguer entre les résultats via "n" et "N".

Si vous souhaitez installer une machine virtuelle qui ressemble à celle de l'école, vous pouvez télécharger et installer une *vieille* debian. Aux dernières nouvelles, les machines de l'école ont un noyau Linux 2.16, au doigt mouillé ça correspond à du debian 6 : `http://cdimage.debian.org/mirror/cdimage/archive/6.0.10-live/amd64/iso-hybrid/`

N'oubliez pas de créer un deuxième disque dur virtuel dans la configuration de votre système de virtualisation et de rendre votre machine virtuel accessible par le réseau. Personnellement, j'aime bien l'accès par pont, ça rend mes VM accessibles depuis tout mon réseau local.

2. Il existe aussi des versions en ligne, vous n'avez pas d'excuses.

1 Premier cours

1.1 passwd et su

Vous venez de recevoir une machine virtuelle comme vos camarades. Toutes ces machines disposent encore toutes des mêmes comptes (lambin et root) et mots de passe (respectivement azerty et azerty). La première chose à faire une fois connecté sur *la votre* est donc de changer ces mots de passe avant qu'un de vos petits camarades indélicats ne prenne la peine de le faire à votre place...sans forcément vous préciser le nouveau mot de passe. La commande pour modifier les mots de passe est *passwd*. Et sur le coup, la page de manuel est simple et explicite ; vous n'avez pas d'excuse !

man : `passwd`

```
passwd [options] [LOGIN]
    La commande passwd modifie les mots de passe des comptes
    d'utilisateurs. Un utilisateur normal ne peut changer que son propre
    mot de passe, alors que le superutilisateur peut changer le mot de
    passe associé à n'importe quel compte.
```

Dans un premier temps [3], vous allez passer de nombreuses fois d'un utilisateur à l'autre. Pour ce faire, il n'est pas nécessaire de vous déconnecter/reconnecter à chaque fois, la commande su vous permettra de...

man : `su`

```
(...) devenir un autre utilisateur pour la durée d'une session. Invoqué
sans nom d'utilisateur, le comportement par défaut de su est de devenir
superutilisateur. Le paramètre optionnel - permet d'obtenir un
environnement similaire à celui que l'utilisateur aurait obtenu lors d'une
connexion directe.
```

1.2 vi(m)

Avant même de commencer, je vous conseille d'installer vim en lieu et place de vi [4]. En effet, jusqu'alors, vous n'avez été confronté qu'à vim en cours de C/Linux/Unix et pas encore vi. Si vous trouvez vim compliqué, vi est pire. Je m'explique :

— Certaines touches (ex : backspace) sur vi ne fonctionnent pas (ou pas comme vous en avez l'habitude). C'est probablement le plus perturbant.
— Oubliez la coloration syntaxique.

Pour moi, installer vim sur un nouveau système est donc un choix pertinent (particulièrement pour les débutants).

1.2.1 Les fichiers en "rc"

Les fichiers en "rc" sont des fichiers de configuration contenant des instructions qui sont exécuté au démarrage d'un programme pour le configurer. Chacune de ces instructions peuvent également être entrée à la main par l'utilisateur. Vous avez probablement déjà été confronté de près ou de loin à deux de ces fichiers : .bashrc et .vimrc respectivement pour bash et vim [5]. Le fichier .exrc dont il est fait mention dans l'énoncé sert à configurer vi, l'ancêtre de vim.

Ces fichiers trouvent généralement leur place à la racine du répertoire personnel de l'utilisateur : `"~/"`. **Vous pouvez facilement déployer automatiquement des fichiers de configuration comme ceux-ci dans le répertoire des nouveaux utilisateurs via la répertoire**

3. avant de découvrir sudo
4. Pour la procédure d'installation : voir le point suivant
5. Merci cap'tain obvious !

/etc/skel. Lors de la création de nouveaux utilisateurs sur votre système[6], ceux-ci recevront une copie des fichiers présent dans ce répertoire, placez-y donc la configuration par défaut pour les nouveaux.

1.2.2 .exrc

Voici quelques exemples d'instructions pouvant être placé dans le fichier .exrc. Le caractère " commente le reste de la ligne (comme le # en bash et python ou le // en java).

set nu " permet la numérotation des lignes[7]

set autoindent " indente automatiquement les nouvelles lignes

1.2.3 .vimrc

Voici quelques exemples d'instructions pour le .vimrc. Notons que les instructions du .exrc fonctionnent également ici. Le fichier .exrc est même pris en compte directement par certaines implémentations de vim.

syntax on " active la coloration syntaxique

colorscheme koehler " change le thème de couleur pour le thème "koehler[8]". Utile avec putty qui rend certaines couleurs illisibles.

set cursorline " surligne la ligne courante[9]

1.3 Création d'utilisateur

À moins de vous appeler Lambin, vous pourriez trouver intéressant de créer un compte utilisateur à votre nom, c'est ce que nous allons aborder à ce point. Pour ce faire, il existe deux commandes : *useradd* et *adduser*. *adduser* étant interactif, nous allons utiliser *useradd*[10]. La commande ressemblera à cela :

```
# useradd [options] LOGIN
```

Différentes options peuvent s'avérer fortement utiles aux futurs utilisateurs. Je vous propose les suivantes :

-m pour créer le répertoire personnel, c'est un peu vide une machine Unix sans répertoire personnel.

-s /bin/bash parce que /bin/sh (par défaut) sans l'auto-complétion, c'est sympa 5 minutes,...[11]

-c "Prénom Nom" c'est optionnel, mais c'est sympa quand le système connait votre nom complet.

-k /un/répertoire/de/squelette si vous ne souhaitez pas utiliser le répertoire par défaut (/etc/skel) ou si vous en avez plusieurs différents. Nécessite qu'on crée un répertoire via "-m".

-G groupe1,groupe2,... Ajout l'utilisateur à un ou plusieurs groupes *déjà existant*. Par défaut, l'utilisateur appartient au moins à un groupe portant son nom[12].

Par exemple, vous pouvez créer l'utilisateur bob avec la commande suivante :

```
# useradd -m -s /bin/bash -c "Bob Brown" -G eleve bob
```

6. Voir plus loin
7. Par contre, chez moi, ça marche mal/pas avec vi.
8. Ce n'est qu'un exemple, avec tab pour pouvez faire défiler les différents thèmes installé sur le système.
9. Pour les bigleux avec un écran trop grand comme moi.
10. C'est tellement plus fun quand on peut scripter les choses.
11. Si vous n'effectuez pas cette manipulation, vous pouvez toujours changer à postériori le shell par défaut d'un utilisateur en modifiant le fichier /etc/passwd
12. Modifiable avec la commande -g. À éviter si vous ne savez pas ce que ça fait. C'est le groupe qui sera attribué par défaut aux fichiers créés par l'utilisateur. (cfr : théorie sur les permissions en 1ère : user/group/other)

Une fois vos utilisateurs créés via cette commande, pensez à leur fournir un mot de passe via la commande passwd :

```
# passwd nouvelutilisateur
```

1.4 Gestion de paquets (apt)

Une majorité de systèmes Unix/Linux disposent d'un gestionnaire de paquet permettant d'automatiser les processus d'installation, de désinstallation et de mise à jour des logiciels installés. Deux commandes nous seront utiles pour gérer notre système :

```
# apt-get {update | upgrade | install paquet ...}
```

man : apt-get

```
update
    La commande update permet de resynchroniser un fichier d'index
    répertoriant les paquets disponibles et sa source. Ces fichiers sont
    récupérés aux endroits spécifiés dans /etc/apt/sources.list. (...) On
    devrait toujours exécuter une commande update avant les commandes
    upgrade ou dist-upgrade.
upgrade
    La commande upgrade permet d'installer les versions les plus récentes
    de tous les paquets présents sur le système (...) On doit d'abord
    exécuter la commande update pour que apt-get connaisse l'existence de
    nouvelles versions des paquets.
install
    La commande install est suivie par un ou plusieurs paquets à
    installer. (...)
remove
    La commande remove est identique à la commande install, les paquets
    étant alors supprimés et non installés. Veuillez noter que la
    suppression d'un paquet en laisse les fichiers de configuration sur le
    système.
purge
    La commande purge est identique à remove mais les paquets indiqués
    sont supprimés et purgés (leurs fichiers de configuration sont
    également effacés).
```

```
$ apt-cache {search | show} paquet
```

man : apt-cache

```
show paquet
    La commande show (...) affiche des informations sur les paquets donnés
    en argument.
search expression_régulière
    La commande search recherche l'expression rationnelle POSIX donnée en
    paramètre sur tous les paquets disponibles, voir regex(7). Elle
    cherche une occurrence de la chaîne dans les noms de paquets et dans
    les descriptions.
```

Vous voila maintenant avec les outils nécessaires pour installer un serveur openssh sur votre machine. Commencez par resynchroniser l'index des paquets puis cherchez-y le paquet correspondant au serveur openssh pour ensuite l'installer [13].

13. Indice : la recherche ciblée "apt-cache search openssh serveur" ne devrait vous renvoyer qu'un seul résultat :

1.5 Serveur SSH

Sur une machine Linux, la majorité des fichiers de configuration se trouvent dans le répertoire /etc. En vous y rendant, vous constaterez la présence d'un répertoire /etc/ssh/ contenant, entre autre, les deux fichiers "ssh_config" et "sshd_config". Le "d" signifiant daemon [14], c'est donc le fichier "sshd_config" qui nous intéresse pour configurer le serveur ssh.

Pour sécuriser (un peu) notre serveur, nous allons faire trois choses :

— Changez le port par défaut (22)

— Empêcher le root de se connecter directement

— Édicter la liste des seuls utilisateurs autorisés à se connecter

Les deux premièress sont relativement simples à repérer dans le fichier de configuration. Les lignes "Port 22" et "PermitRootLogin yes" sont à changer. La troisième peut se montrer un peu plus surprenante. En effet, l'option n'est pas présente par défaut dans le fichier. "Mais comment suis-je sensé la connaître alors ?" allez-vous sans doute vous exclamer. Je vous répondrai sans aucune hésitation le mot magique : "man" ! En effet, certains fichiers de configuration disposent d'une page de manuel dédiée :

```
man : sshd_config

AllowUsers
    This keyword can be followed by a list of user name patterns,
    separated by spaces.
PermitRootLogin
    Specifies whether root can log in using ssh(1). The argument must be
    ‘‘yes’’, ‘‘without-password’’, ‘‘forced-commands-only’’, or ‘‘no’’. The
    default is ‘‘yes’’.
Port
    Specifies the port number that sshd(8) listens on. The default is 22.
    Multiple options of this type are permitted.
```

Une fois le fichier de configuration édité, si vous essayez de vous connecter sur le port 22 avec root, la connexion devrait encore être possible. Vous venez d'éditer un fichier texte. Le daemon correspondant ne le "surveille" pas en temps réel ! Vous devez lui signifier qu'une modification a été effectuée et qu'il doit la prendre en compte. Pour ce faire, vous devez utiliser les scripts présents dans /etc/init.d/ [15]. Ceux-ci offrent généralement plusieurs options : "start", "stop", "restart", "reload"...Dans ce cas-ci, nous nous contenterons de recharger ssh :

```
# /etc/init.d/ssh reload
```
[16]

1.6 Connexion sans mot de passe (Authentification par clé publique/privé)

Je vous invite à lire les points 3 et 4.2 sur le site indiqué sur la feuille d'exercice : `http://formation-debian.via.ecp.fr/ssh.html` C'est bien expliqué et j'ai assez de copier-coller à faire avec les pages de man.

Attention, la commande pour transférer votre clé publique sur la machine distante expliquée sur cette page ne fonctionnera pas tel quel car vous avez modifié le port par défaut du serveur ssh, il vous faudra donc rajouter une option pour le port :

```
$ ssh-copy-id -i  /.ssh/id_dsa.pub "login@nom_DNS_du_serveur -p
<port>"
```

le paquet à installer.

14. Dans le monde Unix, un daemon est un programme qui s'exécute à l'arrière plan et est généralement lancé au démarrage du système. Un exemple typique de daemon est un programme servant à répondre à des requêtes réseaux : serveur web, SSH, SQL,...

15. Notons que dans les versions plus récentes des distributions Linux, la tendance est au nouveau programme "systemd" pour la gestion des daemons. Sur d'autres machines que celles de l'école, vous devrez probablement utiliser la commande "systemctl restart sshd.service"

16. Identique à la commande : "service ssh reload"

2 Deuxième cours

2.1 sudo

Une politique de sécurité intéressante consiste à éviter le plus possible d'utiliser l'utilisateur root. Pour ce faire, il est courant d'utiliser la commande "*sudo*" dont le nom est plutôt explicite "su do". Elle nous permet de "faire faire" quelque chose en tant que le SuperUtilisateur.

L'utilisation de sudo vous permettra entre-autre de remplacer toutes les commandes :

```
# commande
```

par :

```
$ sudo commande
```

si vous utilisez un utilisateur ayant le droit d'utiliser sudo (voir plus bas : la configuration de sudo).

2.1.1 Installation

L'installation ne devrait pas vous posez de problème depuis que vous savez rechercher dans les dépôts (apt-cache search) et afficher le détails d'un paquet (apt-cache show), je ne vais donc pas m'attarder sur ce point [17].

Vous connaissez la chanson, nouveau paquet = nouvelle page de manuel :

man : `sudo`

```
DESCRIPTION
    sudo allows a permitted user to execute a command as the superuser or
    another user, as specified by the security policy.
    (...)
    The default security policy is sudoers, which is configured via the
    file /etc/sudoers, (...)
```

Intéressons nous donc à ce fichier ...Un commentaire nous arrête rapidement : "This file MUST be edited with the 'visudo' command as root." suivi par "See the man page for details on how to write a sudoers file." dont nous reparlerons plus tard.

man : `visudo`

```
visudo edits the sudoers file in a safe fashion, analogous to vipw(8).
visudo locks the sudoers file against multiple simultaneous edits,
provides basic sanity checks, and checks for parse errors.
```

2.1.2 Editeur par défaut

Il est plutôt probable que l'exécution de "*visudo*" démarre l'éditeur de texte "nano" plutôt que votre éditeur préféré : vi(m). En effet, celui-ci est l'éditeur par défaut car considéré comme plus "n00b friendly", mais comme vous êtes des experts dans le maniement de vim [18], nous allons modifier ça ! (NB : tout ce qu'il vous faut savoir sur **nano**, c'est comment le **quitter** : faites **ctrl**+**x**)

Il existe plusieurs méthodes pour modifier l'éditeur utilisé. La première ne le fait que pour un seul utilisateur (vous-même) en définissant la variable d'environnement "EDITOR" :

```
$ export EDITOR=vim
```

17. Juste un spoiler pour vous dire que c'est bien le paquet "sudo" qu'il faut installer pour utiliser sudo.
18. Non non, je ne suis vraiment pas du genre à me moquer de vous ! ;D

Si vous souhaitez faire persister cette instruction, vous pouvez la mettre dans votre fichier .bashrc.

Vous pouvez également modifier les programmes par défault pour tout le système via la commande update-alternatives :

```
# update-alternatives --config editor
```

2.1.3 Configuration de sudo

Passons à la configuration à proprement parlé de sudo.

Premièrement, nous avons besoin d'un utilisateur qui aura les pleins pouvoirs dans son utilisation de sudo. Vous constaterez qu'il existe déjà une ligne dans le fichier sudoers permettant aux membres du groupe "sudo" de faire ce que nous recherchons, il nous suffit donc d'ajouter l'utilisateur souhaité au groupe sudo via la commande *adduser*.

man : `adduser`

```
adduser [options] utilisateur groupe
(...)
Ajouter un utilisateur existant à un groupe existant.
    Lorsqu'il est appelé avec deux paramètres n'étant pas des options,
    adduser ajoutera un utilisateur existant à un groupe existant.
```

Ça nous donnera donc tout simplement :

```
# adduser admin sudo
```

Vous pouvez facilement vérifier la liste des groupes d'un utilisateur via la commande :

```
$ groups <utilisteur>
```

Passons à la configuration d'un nouveau groupe dédié à la gestion des utilisateurs. Le groupe "mgmtuser" réclamé par l'énoncé d'exercice n'existant pas, nous allons le créer avec la commande *addgroup* puis ajouter les utilisateurs concernés [19] à ce groupe comme vu ci-dessus.

```
# addgroup mgmtuser
```

Il ne nous reste plus qu'à ajouter une règle spécifique à ce groupe. Pour ce faire, lisez le manuel . . .Nan, je troll, la page de sudoers est complètement imbuvable, j'ai du me faire violence pour vous faire ce résumé [20] et voila la ligne de configuration à ajouter au fichier sudoers via la commande visudo :

```
%mgmtuser ALL=(root) /usr/sbin/adduser, /usr/sbin/deluser,
/usr/bin/passwd
```

On peut traduire par : les membres du groupe "mgmtuser", ont l'autorisation sur tout les hôtes (ALL) [21] de se faire passer pour "root" pour utiliser les commandes : *adduser*, *deluser* et *passwd*. NB : pour connaitre la localisation des commandes (/usr/machin/. . .), vous pouvez utiliser la commande "*whereis*" (ex : "whereis adduser").

2.2 Contrôler l'accès au système

Je vais partir du principe que vous avez réussi les cours de Linux en première et le cours de Unix en deuxième [22]. Vous savez donc coder en bash et en perl et je ne m'attarderai pas sur le contenu du/des scripts à coder mais sur les informations à traiter et comment automatiser le traitement.

19. À vue de nez, c'est l'utilisateur avec votre prénom, l'énoncé ne précise pas.
20. En fait, j'ai triché. Je suis allé directement à la section exemple de la page de man :D mais vous pouvez quand même venter mes mérites !
21. Il n'y a qu'une seul machine ici, donc on s'en fiche un peu.
22. Et si, grâce au décret Marcourt, vous suivez ce cours sans ses pré-requis, débrouillez-vous !

2.2.1 Récupérer les informations

Dans la majorité des systèmes Linux, un nombre important d'informations sont journalisées (en franglais : "logées" [23]), vous pouvez lire ces journaux de log dans le répertoire /**var**/**log**. Vous remarquerez que le logger (loggeur) utilise un système de roulement, les fichiers sont automatiquement numérotés et compressés en .gz.

Pour surveiller l'activité de l'utilisateur privilégié, je vous invite à vous intéresser au fichier **auth.log**. Vous y trouverez, entre-autre, des informations qui sont pertinentes dans ce contexte :

— Qui utilise "su" pour devenir quelqu'un d'autre

— Qui utilise "sudo" pour faire quelque chose au nom de quelqu'un d'autre

Pour rappel, la commande "*tail*" permet avec l'option "-f" d'afficher en temps réel les données ajoutée en fin de fichier. Vous pouvez donc facilement voir en temps réel ce qui est ajouté à votre fichier auth.log via la commande :

```
# tail -f /var/log/auth.log
```

Un deuxième terminal vous permettra de "faire des trucs" en même temps pour créer des nouvelles entrées dans le fichier log. Voici un ligne qu'on peut retrouver dans mon fichier auth.log :

```
Dec 3 14:17:51 debian sudo: thibault : TTY=pts/0 ; PWD=/tmp/purgatoire ;
USER=bob ; COMMAND=/bin/echo J'aime pas windows
```

Dans le répertoire "/tmp/purgatoire", l'utilisateur thibault a fait dire à l'utilisateur bob : "J'aime pas windows" via la commande "echo". Ceci s'est déroulé sur le terminal "pts/0".

2.2.2 Automatisation

Sur la majorité des systèmes Linux [24], il existe un démon permettant de lancer des commandes différées. Ce démon s'appelle **cron** et comme pour *visudo*, vous pouvez modifier le fichier de configuration de vos crons via la commande crontab.

man : `crontab`

```
crontab [ -u utilisateur ] [ -i ] { -e | -l | -r }
DESCRIPTION
    crontab est le programme utilisé pour installer, désinstaller ou
    afficher le contenu des tables permettant de piloter le fonctionnement
    du démon cron(8) de Vixie Cron. Chaque utilisateur dispose de sa
    propre crontab,
    (...)
    L'option -e permet de modifier la crontab en cours, en utilisant
    l'éditeur indiqué par les variables d'environnement VISUAL ou EDITOR.
    Après avoir quitté l'éditeur, la table modifiée sera installée
    automatiquement. Si aucune des variables d'environnement n'est
    définie, alors l'éditeur par défaut /usr/bin/editor [a] est utilisé.
```

a. modifiable via update-alternatives comme nous l'avons vu précédemment

Chaque utilisateur dispose de sa propre crontab, ainsi la commande que vous configurez dans votre propre contab sera executé en tant que *vous* par cron. Si la commande nécessite les droits root, vous devez donc la mettre dans le crontab de root.

Pour ce qui est du fonctionnement du fichier crontab, je vais juste vous copier-coller le commentaire qui est de base dans ma debian au début du fichier :

Each task to run has to be defined through a single line indicating with different fields when the task will be run and what command to run for the task

23. Oui, les français ont des traduction franchement bizarres !

24. Vous trouvez que je recycle mes introductions ? Si je prétend que ça marche sur toutes les Linux, il y aura un rigolo pour m'en trouver une où ce n'est pas le cas. Bienvenue dans le monde de la diversité.

To define the time you can provide concrete values for minute (m), hour (h), day of month (dom), month (mon), and day of week [25] (dow) or use '*' in these fields (for 'any'). (...)

For example, you can run a backup of all your user accounts at 5 a.m every week with :

```
0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
```

Vous l'aurez compris, il vous suffit donc de faire appel à votre script bash/perl dans une de ces lignes en configurant le délais à votre guise.

2.3 find

Comme pour les scripts bash/perl, c'est du déjà vu, mais cette fois-ci je fais un effort et je vous les donne. Notons que je redirige la sortie d'erreur (stderr) vers la poubelle avec "2> /dev/null" car find produit facilement des erreurs en essayant de lire des fichiers illisibles ou inexistant.

Lister les applications (= les executables) dont le suid (4000) est positionné à ON :

```
# find / -executable -perm -4000 2> /dev/null
```

Intérêt : Le suid permet d'exécuter un fichier en tant que son créateur et non en tant que soi-même. Ça peut s'avérer très vite très dangereux si on oublie des droits d'écritures sur le même fichier : le premier venu peut y mettre ce qu'il veut et se faire passer pour vous en l'exécutant.

Lister les répertoires accessibles en lecture-écriture par les autres utilisateurs du système (006) :

```
# find / -type d -perm -006 2> /dev/null
```

Intérêt : "autre", c'est tout le monde et n'importe qui. Et "écriture", bha c'est clair, non ?

Lister les fichiers système modifiés durant les 48 dernières heures :

```
# find /etc -type f -mtime 2
```

Intérêt : La surveillance très régulière des fichiers de configuration vous permet de vous assurer qu'aucune personne nuisible ne modifie *vos* fichiers de configuration dans votre dos.

3 Troisième cours

3.1 Théorie

3.1.1 Représentation des disques et partition

Sous Linux, les différents support de stockage branché à la machine sont représenté par des fichiers présent dans le répertoire /dev. Ces fichiers sont composé de trois lettres par exemple : /dev/hda ou /dev/sda. Le "h" désigne des disques branché par IDE et le "s" quant à lui désigne SCSI ou SATA. La dernière lettre est un lettre de l'alphabet servant à désigner le disque, le premier disque sata s'appellera /dev/sda, le deuxième /dev/sdb, etc... [26]

Les différentes partitions sont désignées par un chiffre après le nom du disque. Par exemple : la deuxième et la troisième partition du deuxième disque sata seront désignées par /dev/sdb2 et /dev/sdb3.

Outre le parcours du répertoire /dev, vous pouvez obtenir plus d'information sur les différents disques de votre machine avec la commande [27] :

```
# fdisk -l
```

Ce n'est qu'une des fonctionnalités de fdisk, nous y reviendrons plus tard.

25. Lundi = 0, mardi = 1, etc ...
26. Si on dépasse 26 disques : aa, ab, ac
27. Et si votre machine est plus récente que les antiquités de l'école, essayez donc "lsblk". L'affichage est vachement plus compréhensible <3

3.1.2 Le système de fichier

Sur une machine windows, on peut résumer la situation par : *"tout est dans c et vous avez accès automatique à toutes les autres partitions via la page "Poste de travail", "Ordinateur" ou "Ce PC" ".*

Sur une machine Linux, c'est un petit peu plus complexe[28], la partition principale sert de "/" (le top level directory), un peu comme c mais il n'y a pas automatiquement accès aux autres partitions. En effet, celle-ci doivent être "montée"[29] quelque part dans l'arborescence de "/"[30]. Une partition peut être montée n'importe où dans le système de fichier, vous pouvez même monter des partitions dans d'autres partitions.

Exemple concrète (Et un poil tordu aussi) :

Vous faites du montage de vidéo, votre machine est donc configurée pour que vous puissiez faire ça de manière optimal :

- La racine et vos programmes sont placés sur une partition de disque dur rapide (ex : SSD)
- Votre répertoire personnel se trouve sur un disque normal (ex : 7200 rpm) sauf :
  - le répertoire des vidéos est sur un deuxième SSD
  - le répertoire backup et le répertoire de vos photos de vacance sont sur deux partitions d'un gros disque lent (ex : 4To à 5400rpm).
  - le répertoire avec vos documents bancaires qui est sur une autre partition du disque normal, celle-ci est chiffrée.

Sur une windows, vous pouvez toujours vous en sortir à coups de raccourci d'un disque à l'autre et de changement de disque (c, d, e, f...). Avec Linux, une fois les partitions montées, vous ne voyez qu'un seul système de fichier dans lequel vous pouvez vous promener à coup de cd :

```
# cd /usr/bin # vous êtes sur le 1er SSD

$ cd /home/bob # vous êtes sur le disque "normal"

$ cd /home/bob/Vidéos # vous êtes sur le 2ème SSD

$ cd /home/bob/Backup # vous êtes sur le disque "lent"

$ cd /home/bob/private # vous êtes sur la partition chiffrée du disque
normal
```

3.2 Espace sur les disques : du et df

La commande df vous permet d'afficher les différentes partitions montée sur le système, leurs points de montage et l'espace utilisé sur celle-ci. Si la taille en nombre de bloc ne vous parle pas, vous pouvez toujours utiliser l'option "-h".

```
$ df -h
```

man : `df`

```
NAME
    df - report file system disk space usage
OPTIONS
    -h, -human-readable print sizes in powers of 1024 (e.g., 1023M)
```

Intéressons-nous maintenant aux fichiers eux-même plutôt qu'aux partitions, la commande "du" vous permet de vous renseigner sur la taille des répertoires.

28. mais surtout plus souple.
29. Nous verrons la procédure de montage manuel et automatique plus loin.
30. La plupart des interfaces graphique (unity, gnome, kde,...) montent automatiquement les disques (ex : clé usb) dans un sous-répertoire de /media et l'affiche dans l'explorateur de fichier graphique. C'est une sur-couche de l'interface graphique pour vous faciliter la vie et non un comportement "classique" d'une Linux *nue*.

man : du

```
NAME
    du - estimate file space usage
DESCRIPTION
    -h, -human-readable print sizes in human readable format (e.g., 1K
    234M 2G)
    -s, -summarize display only a total for each argument
```

Ainsi donc, si vous souhaitez connaître l'espace utilisé par chaque utilisateur dans le répertoire /home, vous pouvez utiliser :

```
# du -hs /home/*
```

3.3 Création de partition

3.3.1 fdisk

Maintenant que nous avons une idée plus clair de ce que sont les partitions, il est temps d'en créer nous-même. En effet, les machines de l'école disposent d'une deuxième disque dur vide sur lequel nous allons pouvoir faire nos expérimentations. Pour rappel, fdisk nous permet d'afficher les disques présent sur la machine. Vous pourrez donc voir ce second disque : /dev/sdb.

man : fdisk

```
NAME
    fdisk - manipulate disk partition table
SYNOPSIS
    fdisk [options] device
    fdisk -l [device...]
DESCRIPTION
    fdisk is a dialog-driven program for creation and manipulation of
    partition tables.
```

"dialog-drives program", voila, tout est dit, c'est interactif j'ai moins de boulot de rédaction de résumé puisque le programme vous posera des questions. Lancez-vous donc et exécutez fdisk sur le disque dur a partitionner :

```
# fdisk /dev/sdb
```

fdisk devrait vous afficher ceci :

```
Command (m for help):
```

La première chose a faire sera donc de prendre connaissance de l'aide, voici une courte sélection de ce qui pourrait vous être utile :

```
d delete a partition
m print this menu
n add a new partition
p print the partition table
q quit without saving changes
w write table to disk and exit
```

Au moment de la création de votre première partition (commande "n"), fdisk vous interroge sur le type de partition que vous souhaitez créer : "primary" (une partition de base) ou "extended" (une partition qui contient d'autres partitions "logique"). Une limitation hardware [31] ne permet que la création de 4 partitions principales sur un disque dur [32]. Pour atteindre l'objectif de

31. Si ça vous intéresse, renseignez-vous sur le master boot record (MBR).
32. qui sont d'ailleurs numérotées de 1 à 4 dans fdisk

l'énoncé d'exercice, c'est-à-dire la création de 5 partitions de tailles égales, vous devrez donc créer 0 à 3 partitions primaires et une partition étendue qui contiendra le reste de vos partitions (2 à 5).

Les partitions sont alignées sur les cylindres du disque dur, fdisk vous demandera des valeurs en cylindre quand vous lui indiquerez vouloir créer des partitions. La commande "p" vous permet d'afficher le nombre total de cylindre disponible sur le disque que vous êtes en train de modifier. Je vous laisse à vos calculatrices pour calculer des cinquièmes de la taille total avant d'entamer la création de celles-ci.

Exemple : Si vous avez un total de 50 cylindres, vous pouvez créer 3 partitions primaires de 10 cylindres et une partition étendue de 20 cylindres ((3*10)+20 = 50) [33]. Vous créerez ensuite 2 partitions logiques de 10 cylindre dans la partition étendue.

À la fin de vos manipulation, n'oubliez pas d'utiliser "w" pour écrire toutes vos modifications sur le disque, aucun changement ne sera réellement effectué sur le disque avant ce moment là. Vous pouvez ensuite vérifier l'apparition des nouvelles partitions avec "fdisk -l".

3.3.2 Préparation d'une partition : mkfs

Maintenant que vos partitions sont crées, il est temps d'y mettre un système de fichier. Sur une machine Linux, vous pouvez facilement formater des partitions avec le programme mkfs. Celui-ci nécessite des paramètres mais comme nous sommes de parfait flemmards, utilisons une "variante toute faite" de mkfs : mkfs.ext4.

```
# mkfs.ext4 <partition>
```
[34]

Il existe d'autres variantes vous permettant de faire notamment du fat ou du ntfs. Vous pourrez les découvrir à coup de tabulation ou dans le "see also" du man de mkfs.

3.4 Déplacer l'espace disque alloué aux utilisateurs

La nouvelle partition est prête, il ne nous reste plus qu'à "l'allouer aux utilisateurs en respectant les standards habituels d'une système Linux". Pour parler plus clairement que l'énoncé, on va y déplacer le répertoire "/home". Comme je vous disais au début de ce chapitre, on peut "monter" une partition n'importe où, le but de ce point sera donc de monter la nouvelle partition sur le répertoire /home.

Mais un problème se pose à nous, il y a déjà des fichiers dans /home, on ne peut donc pas monter une partition "par dessus" ces fichiers. Il nous faut donc commencer par déplacer physiquement le contenu de home vers la nouvelle partition. Mais tant que la partition n'est pas montée, elle n'apparait pas dans le système de ficher et vous n'avez donc nul part où "mv" les fichiers [35].

3.4.1 Déplacement physique des fichers vers la nouvelle partition

Pour résoudre ces problèmes, nous allons monter temporairement la nouvelle partition ailleurs pour y avoir accès le temps de déplacer tout le contenu de home. Il existe déjà un dossier dédié au "montage temporaire" de partition : le dossier "/mnt" (abréviation de mount). Créez donc y un dossier qui accueillera notre nouvelle partition :

```
# mkdir /mnt/homeTMP
```

Passons enfin à la commande mount :

33. Toi aussi apprend à compter au cours d'admin' Linux ! :D
34. exemple: /dev/sdb3
35. NB : /dev/sdXX est un fichier spécial qui représente la partition, ce n'est pas son contenu.

man : `mount`

```
NAME
    mount - mount a filesystem
DESCRIPTION
    All files accessible in a Unix system are arranged in one big tree,
    the file hierarchy, rooted at /. These files can be spread out over
    several devices. The mount command serves to attach the filesystem
    found on some device to the big file tree. Conversely, the umount(8)
    command will detach it again.
    The standard form of the mount command, is
        mount -t type [a] device dir
```

a. "-t type" que je n'utilise pas personnellement. Il reconnait facilement les types les plus courant.

Pour monter la partition sur notre dossier temporaire la commande est donc la suivante :

```
# mount /dev/sdXX /mnt/homeTMP
```

Pensez à remplacer le XX par la partition correspondante sur votre machine.

Astuce : avant de commencer à déplacer le contenu de home, il peut être pertinent d'avertir les autres utilisateurs connecté sur le système [36]. La commande "w" vous permet de consulter la liste des utilisateurs connectés. Vous pouvez leur envoyer un message via la commande "wall" par exemple :

```
# echo "Pause café. Tout le monde déco' et revient dans 15 minutes" |
wall
```

C'est bon, tout le monde est parti et personne ne verra ses fichiers disparaitre pendant qu'il travaille ? Bien, passons au déplacement des fichiers :

```
# mv /home/* /mnt/homeTMP
```

Notons que si vous préférez utiliser la commande "cp" [37] pour faire cela, vous ne devez surtout pas oublier l'option "-p" (préserve) ! En effet, dans le cas contraire, les fichiers copier par cp appartiendrons au "copieur" et non à l'utilisateur original. Comme ce genre de déplacement massif ne peut qu'être fait par un administrateur, tout les utilisateurs normaux se feront bloquer leurs fichiers. Il sera ensuite particulièrement ennuyeux d'essayer de restaurer les droits sur les fichiers des utilisateurs. "mv" préserve les droits sur les fichiers dans tout les cas.

Une fois les fichiers déplacés, vous pouvez démonter le répertoire temporaire /mnt/homeTMP et remonter la partition sur /home. Si tout s'est bien passé, tout est en apparence "comme avant" : home est dans home mais home est sur la nouvelle partition.

```
# umount /mnt/homeTMP
```

NB : umount peut prendre du temps si tout les fichiers n'ont pas fini d'être déplacé depuis ou vers la partition concernée. mv vous rend la main mais l'OS fini de copier à l'arrière plan. Ça arrive souvent pour les gros transferts.

```
# mount /dev/sdXX /home

# rmdir /mnt/homeTMP # on en a plus besoin
```

Il nous reste un détail à régler, il faut que la partition soit montée automatiquement au démarrage de la machine. Il serait embêtant qu'à chaque redémarrage de la machine [38] les utilisateurs doivent attendre le passage d'un administrateur pour avoir accès à leurs fichiers. Et pour revenir sur l'exemple de notre monteur de vidéos, il serait complexe/long de remonter toutes les partitions à chaque fois. C'est là qu'intervient le fichier fstab.

36. Rappel : Linux est un OS conçu pour être multi-utilisateurs
37. Vu dans un résumé d'élève pour ce cours
38. Notons qu'un serveur Linux, ça ne se redémarre PAS sauf pour mettre à jour le kernel ! Dans tout les autres cas de figure, c'est optionnel. Si vous aimez redémarrer, retournez sous windows ! Ho, mais attendez, maintenant même eux utilisent du Linux pour faire tourner leur cloud azure ! (Ouai ouai, c'est bon, je me calme :-P)

3.4.2 Automatisation du montage : /etc/fstab

man : `fstab`

```
NAME
    fstab - static information about the filesystems
DESCRIPTION
    The file fstab contains descriptive information about the various file
    systems. fstab is only read by programs, and not written; it is the
    duty of the system administrator to properly create and maintain this
    file. Each filesystem is described on a separate line; fields on each
    line are separated by tabs or spaces. Lines starting with '#' are
    comments.
```

En ouvrant le fichier une première fois, vous constaterez que chaque ligne à la forme suivante :

```
<file system> <mount point> <type> <options> <dump> <pass>
```

et qu'il existe déjà une ligne pour monter automatiquement la partition principale "/". Il vous faut taper une ligne supplémentaire correspondant à votre nouvelle partition.

Les trois premiers paramètres doivent vous sembler plutôt évidant. Pour le choix des options, vous pouvez consulter le détail dans la page de manuel de mount [39] celles-ci étant les mêmes que celle du "-o" de mount, le plus simple étant de mettre defaults [40].

man : `fstab`

```
The fifth field (<dump>), is used for these filesystems by the dump [a]
command to determine which filesystems need to be dumped.
(...)
The sixth field (<pass>) is used by the fsck [b] program to determine the
order in which filesystem checks are done at reboot time. The root
filesystem should be specified with a value of 1, and other filesystems
should have a value of 2.
```

a. man : "ext2/3/4 filesystem backup"
b. man : "check and repair a Linux file system", parfois lancé au boot en cas de problème.

Ça nous donne donc une ligne du style :

```
/dev/sdXX /home ext4 defaults 1 2
```

Pour aller plus loin :

Notons que si vous changer l'ordre de disque sous le capots de votre machine, par exemple en ajoutant un nouveau disque dur, il est probable que cela casse cette configuration en changeant la lettre des disques sda, sdb, etc. . .Utilisez /dev/sXX est ce que l'on pourrait qualifier de "rapide et sale". Pour "fortifier" votre système, vous pouvez utiliser l'identifiant de votre partition (le UUID) plutôt que ça représentation /dev/sXX. Par exemple :

```
UUID=3e6be9de-8139-11d1-9106-a43f08d823a6 /home ext4 defaults 1 2
```

Vous obtiendrez les UUID des partitions via la commande "blkid".

3.5 Gestion de quotas

Vous le savez probablement aussi bien que moi, le monde est remplis de maladroits et de personnes malveillante. C'est surement aussi valable pour les autres utilisateurs des machines

39. section "FILESYSTEM INDEPENDENT MOUNT OPTIONS"
40. Dans certains cas de figure, certaines options pourrait vous être utile comme : "noexec" si vous investiguez sur un support de stockage douteux ou encore "noatime" si vous voulez désactiver l'écriture des "access time" des fichiers pour des raisons qui vous regardent :-P.

que vous administrées. Nous allons donc mettre en place un système de quota pour les empêcher de faire déborder vos disques dur.

Il existe deux types de quota :

— Les quotas d'espace disque. L'utilisateur a un espace disque défini pour lui. Ex : 100 Mo.

— Les quotas en nombre de fichier (inode). L'utilisateur a un nombre de fichier maximum. [41]

Chacun de ces types peut être divisé en deux catégories :

— Les quotas soft. Si l'utilisateur dépasse le quota soft, il est prévenu et a un délais pour se mettre en ordre.

— Les quotas hard. L'utilisateur ne sait plus créer de fichier, il est bloqué.

Pour respecter l'énoncé d'exercice, nous allons mettre en place une limitation d'espace soft de 100 Mo et une limitation d'espace hard de 110 Mo. Nous ne toucherons pas aux quotas d'inode [42].

3.5.1 Installation et édition de /etc/fstab

La première étape est, comme souvent, l'installation d'un paquet : "quota". Une fois cela fait, il vous faut éditer le fichier /etc/fstab et rajouté deux options concernant la partition où vous souhaitez appliquer des quotas çàd la partition où se situe /home. Ces options sont : "usrquota" et "grpquota".

Rappelez-vous la page de man de fstab, les différentes options sont séparées par des virgules et les différents paramètres par des espaces et des tabulations. N'ajoutez donc pas d'espace entre les nouvelles options, juste des virgules :

```
/dev/sdXX /home ext4 defaults,usrquota,grpquota 1 2
```

Le fichier fstab étant utilisé par le système quand il démarre, vos nouvelles options ne sont pas prises en compte après une simple édition du fichier. Pour cela, vous pouvez utiliser mount avec l'option remount pour forcer le système à prendre en compte vos modifications.

```
# mount -o remount /dev/sdXX
```

3.5.2 Démarrage des quota : quotacheck et quotaon

Maintenant, quota va surveiller les écritures sur ce disque mais il ne prend pas en compte ce qui est déjà présent sur le disque. Pour ce faire, il y a une commande pour lui faire lire le disque et enregistrer ce qui a été fait avant son installation :

```
# quotacheck -vug /dev/sXX
```

man : `quotacheck`

```
NAME
    quotacheck - scan a filesystem for disk usage, create, check and
    repair quota files [a]
SYNOPSIS
    quotacheck [ -gubcfinvdMmR ] [ -F quota-format ] -a | filesystem
OPTIONS
    -v, -verbose
    -u, -user Only user quotas listed in /etc/mtab or on the filesystems
    specified are to be checked. This is the default action.
    -g, -group Only group quotas listed in /etc/mtab or on the filesystems
    specified are to be checked.
```

a. Les "quota files" (aquota.user et aquota.group) sont les fichiers interne de quota : la "base de donnée" de qui en est où dans ses fichiers.

Juste après ça, il est temps de démarrer les quotas. Pour ça, il vous faut utiliser la commande "quotaon" qui prend exactement les mêmes paramètres que quotacheck pour les même raisons.

41. Pour empêcher un petit rigolo de créer, par exemple : 200 millions de fichiers vides parce que yolo.
42. Il vous restera la possibilité d'insulter copieusement l'utilisateur qui créera 200 millions de fichiers vide.

3.5.3 Configuration des utilisateurs : edquota

La commande pour configurer les limitations est "edquota".

man : `edquota`

```
NAME
    edquota - edit user quotas
OPTIONS
    -u, -user Edit the user quota. This is the default.
    -p, -prototype=protoname Duplicate the quotas of the prototypical user
    specified for each user specified.
    -t, -edit-period Edit the soft time limits for each filesystem. (...)
    Time units of 'seconds', 'minutes', 'hours', and 'days' are
    understood. Time limits are printed in the greatest possible time unit
    such that the value is greater than or equal to one.
```

Comme nous l'avons déjà vu plusieurs fois, cette commande lance votre éditeur de texte sur un fichier de configuration qu'il vous faut modifier.

Commençons par modifier un utilisateur :

```
# edquota -u bob
```

Le fichier se présente sous forme d'un tableau, une ligne par système de fichier soumis au quota. Les six colonnes restante représentent (dans l'ordre) : le nombre de blocks [43] actuellement utilisés par l'utilisateur, les limites soft et hard de blocks, le nombre d'inodes de l'utilisateur, les limites soft et hard d'inode. Vous ne devriez pas avoir de mal à vous souvenir de l'ordre puisque qu'il y a des titres (probablement décalé) sur les colonnes.

Modifiez la ligne correspondant au système de fichier et ajouter les quotas correspondants. 100 Mo = 100000 blocks et 110 Mo = 110000 blocks. Laissez à 0 les quotas que vous ne souhaitez pas traiter.

Si vous avez d'autres utilisateurs à qui vous souhaitez donner exactement les mêmes droits, vous pouvez utiliser la commande de prototype pour "cloner" une configuration. Ici, la configuration de bob sera copiée pour alice et eve :

```
# edquota -p bob alice eve
```

3.5.4 Configuration du délais de grâce : edquota

La commande est toujours edquota mais cette fois-ci il vous faut utiliser l'option "-t". Si vous avez compris comment fonctionne le fichier de configuration des utilisateurs, celui-ci ne devrait pas poser problème. Il y a trois colonnes : le système de fichier concerné, la période de grâce pour les blocks et la période de grâce pour les inodes.

3.5.5 Tests (pour aller plus loin)

Procédons maintenant à quelques testes, juste pour vérifier que nos quotas réagissent comme attendu. Pour ce faire, découvrons une commande supplémentaire : dd [44]. dd permet de copier des fichiers, y compris les "fichiers spéciaux" comme des partitions. Vous pourriez par exemple créer l'image bit à bit d'une partition comme ceci :

```
# dd if=/dev/sdXX of=/tmp/sdXX.img
```

ou faire l'inverse pour restaurer une partition depuis une image.

Mais le dossier /dev ne contient pas que des fichiers représentant des partitions, laissez moi vous présenter /dev/zero. C'est un fichier "générateur", il crée des zéros [45] autant que vous

43. 1 block = 1024 bytes
44. Le cousin de ponséchossé et péage </référence kulturel complètement hors sujet>
45. une longue suite de bit à 0. Le résultat produit est illisible avec un éditeur de texte normal comme vim. Pour consulter le contenu de ces fichiers, vous pouvez essayer hexdump ou installer une variante de vi : bvi.

cherchez à en lire [46]. Vous pouvez donc créer facilement des fichiers de taille arbitraire en copiant /dev/zero avec dd.

Testons la limite de quota soft, connectez vous avec un utilisateur restreint par les quotas et essayez :

```
$ dd if=/dev/zero of=102mega.img bs=1M count=102
```
[47]

Un message devrait vous signaler que vous avez atteind la limite soft. Pour atteindre la limite hard, essayer de créer un deuxième fichier de 10 Mo :

```
$ dd if=/dev/zero of=10mega.img bs=1M count=10
```

Un second message devrait vous arrêter brusquement et si vous vérifiez la taille de votre fichier (ls -lh *.img), il ne devrait pas faire 10 Mo.

Supprimez rapidement ces deux fichiers pour respecter vos quotas.

4 Quatrième cours

NFS est un protocole réseau permettant de partager des répertoires et de monter des répertoires distant dans le système de fichier local via le réseau. En gros, les "mount" que nous faisions sur des partitions, nous pouvons maintenant les faire sur un dossier d'une autre machine.

4.1 nfs : partage d'un répertoire publique

4.1.1 Serveur

Commencez par installer les paquets nfs-common et nfs-kernel-server et créer le répertoire /data/pub que nous allons partager en lecture seule.

La configuration de nfs se fait par le billet du fichier /etc/exports. La syntaxe est la suivante :

```
/dossier/a/partager hôtes (option1,option2,...)
```

Vous pouvez spécifier le ou les hôtes sous la forme : ip/masque. Par exemple : 10.0.0.1/26.

Vous trouverez dans la page de man "exports" les options générales que vous pouvez mettre entre les parenthèses.

man : `exports`

```
General Options
    rw
        Allow both read and write requests on this NFS volume. The default
        is to disallow any request which changes the filesystem. This can
        also be made explicit by using the ro option.
```

Nous aurons donc une configuration qui ressemble à ça :

```
/data/pub <ip du réseau autorisé>/<masque du réseau>(ro)
```

Pour prendre en compte la nouvelle configuration, vous devez recharger le serveur nfs avec la commande :

```
# /etc/init.d/nfs-kernel-server reload
```
[48]

Vous pouvez interroger le serveur de l'extérieur (ou en localhost) via la commande showmount pour connaitre la liste des dossiers partagé disponible.

46. et /dev/urandom génère des valeurs aléatoire tandis que /dev/null est un "trou noir" : tout ce qu'on y écrit disparait.
47. Cela copie 102 blocks d'une BlockSize de 1M depuis /dev/zero vers le fichier 102mega.img.
48. identique à "# service nfs-kernel-server reload"

```
man : showmount

NAME
    showmount - show mount information for an NFS server
OPTIONS
    -e or -exports Show the NFS server's export list.
```

```
$ showmount -e host
```
[49]

4.1.2 Client

Puisque vous savez déjà vous servir de mount et que la différence est minime, ce paragraphe sera particulièrement cours. Créez le dossier qui accueillera le répertoire distant :

```
# mkdir /usr/local/pub
```

et montez le répertoire distant :

```
# mount <ip serveur nfs
```
[50]
```
>:/data/pub /usr/local/pub
```

Vous devriez vous retrouver maintenant avec deux répertoires contenant la même chose : /data/pub et /usr/local/pub. /data/pub est l'original et /usr/local/pub sa "copie distante" en local par nfs. Nous sommes donc ici avec un système client-serveur où les deux se trouvent sur la même machine : votre machine.

4.2 Déplacer les données des utilisateurs vers un serveur nfs

Bon, je vais aller droit au but : je pense qu'avec le reste du résumé, vous avez acquis la capacité à résoudre ce dernier problème tout seul. Vous avez déjà toutes les briques, il faut juste les ré-assembler dans le bon ordre.

Je vais résumé ce qu'il faut faire en quelques points, ça vous sera peut-être plus clair que l'énoncé :

- La nouvelle partition que nous avons crée ensemble plus haut ne sert plus à rien et est montée sur /media/sdb1.
- Le répertoire /data/users contiendra les répertoires personnelles des utilisateurs de votre binôme. Ce répertoire est partagé en nfs.
- Le dossier /media/nfs sera le "montage" du dossier /data/users de votre binôme.
- Vous aurez pris soin de déplacer les dossiers de vos utilisateurs de la nouvelle partition (monté sur /media/sdb1) vers le dossier /media/nfs qui est en réalité sur le serveur de votre binôme.
- Le dossier /home (qui n'est plus la partition) contiendra des liens symboliques ("ln -s") vers les répertoires personnelles de vos utilisateurs maintenant présent dans /media/nfs

5 Cinquième cours

5.1 Samba

"Samba est un logiciel d'interopérabilité qui permet à des ordinateurs Unix de mettre à disposition des imprimantes et des fichiers dans des réseaux Windows, en mettant en œuvre le protocole SMB/CIFS de Microsoft Windows. Il s'agit d'une re-implémentation des protocoles SMB/CIFS par ingénierie inverse." (Source : Wikipedia)

L'objectif de ce cours est d'installer et configurer un serveur et un client samba. Commencez par installer les paquets suivant : samba et smbclient.

49. L'adresse d'un autre serveur ou "localhost" pour soit-même.
50. Ici, ça peut être 127.0.0.1 ou localhost puisque le client et le serveur sont sur la même machine.

5.1.1 Sécurisation du serveur

En cherchant un peu, vous ne devriez pas avoir de problème pour trouver le fichier de configuration du serveur samba qui nous permettra de faire cela. Comme expliqué plus tôt, la majorité des fichiers de configuration se trouvent dans le répertoire /etc, vous y trouverez donc un dossier samba contenant "smb.conf". Il commence par :

```
# This is the main Samba configuration file. You should read the
# smb.conf(5) manual page in order to understand the options listed
# here. Samba has a huge number of configurable options most of which
# are not shown in this example
```

Découvrons donc cette page de manuel [51] pour en apprendre plus sur le fonctionnement de ce fichier de configuration :

man : `smb.conf`

```
SECTION DESCRIPTIONS
    Each section in the configuration file (except for the [global]
    section) describes a shared resource (known as a "share"). The section
    name is the name of the shared resource and the parameters within the
    section define the shares attributes.
    There are three special sections, [global], [homes] and [printers],
    (...)
The [global] section
    Parameters in this section apply to the server as a whole, or are
    defaults for sections that do not specifically define certain items.
```

Vous l'aurez compris, pour créer facilement des règles communes à tout les *share*, il faut les placer dans la section [glogal]. C'est donc via cette section que nous pouvons sécuriser l'ensemble du serveur samba en limitant l'accès à celui-ci à la machine de votre binôme ainsi que uniquement à certains utilisateurs.

man : `smb.conf`

```
hosts allow (S)
    A synonym for this parameter is allow hosts. This parameter is a
    comma, space, or tab delimited set of hosts which are permitted to
    access a service.
    (...)
    Example 1: allow all IPs in 150.203.*.*; except one
        hosts allow = 150.203. EXCEPT 150.203.6.66
invalid users (S)
    This is a list of users that should not be allowed to login to this
    service. (...)
    A name starting with a '@' is interpreted as an NIS netgroup first (if
    your system supports NIS), and then as a UNIX group if the name was
    not found in the NIS netgroup database. (...)
valid users (S)
    This is a list of users that should be allowed to login to this
    service. Names starting with '@', '+' and '&' are interpreted using
    the same rules as described in the invalid users parameter. If this is
    empty (the default) then any user can login. (...)
    Example: valid users = greg, @pcusers
```

51. Rappelez-vous : dans man : /<pattern> pour chercher et 'n' et 'N' pour faire défiler. Sur un écran HD j'ai plus de 5000 lignes, personne ne veut lire tout ça !

Vous l'aurez compris, il vous faut ajouter deux lignes à votre fichier smb.conf :

```
hosts allow = 127.0.0.1,<ip de l'autre machine>
```

Et, à votre meilleur convenance :

```
valid users = bob,lambin,<encore un autre utilisateur>
```

ou

```
valid users = @ungrouppoursamba
```

En ayant pris soin de créer le groupe correspondant et d'y avoir ajouté tous les utilisateurs souhaités.

5.1.2 Configurer des partages

Passons maintenant aux sections "normales" :

man : `smb.conf`

```
The following notes apply to ordinary section descriptions.
    A share consists of a directory to which access is being given plus a
    description of the access rights which are granted to the user of the
    service.
    (...)
    The following sample section defines a file space share. The user has
    write access to the path /home/bar. The share is accessed via the
    share name foo:
        [foo]
        path = /home/bar
        read only = no
```

Nous devons créer des sections pour les trois partages que nous demande l'énoncé d'exercice mais il nous reste des options à découvrir. En voici :

man : `smb.conf`

```
path (S)
    This parameter specifies a directory to which the user of the service
    is to be given access.
read only (S)
    An inverted synonym is writeable [a].
    Default: read only = yes
comment (S)
    This is a text field that is seen next to a share when a client does a
    queries the server, either via the network neighborhood or via net
    view to list what shares are available.
```

a. Il est donc complètement superflu de mettre les deux dans un *share*.

NB : Les options comme "browseable = yes" ou "available = yes" présentent dans un autre résumé sont inutile, en effet, les valeurs par défaut de ces commandes (yes) sont les valeurs utilisées.

En parcourant attentivement le fichier smb.conf, vous constaterez qu'il existe par défaut une section ([homes]) permettant de partager en read only le dossier personnel de l'utilisateur connecté. Il ne vous faut donc que modifier cette règle dans la section pré-existante. Pour les deux partages restant, les sections ressemblerons à cela :

```
[pub]
comment =Publique
path = /usr/share/pub
```

```
[post]
path = /share/post
writable = yes
```

N'oubliez pas de créer ces dossiers et de leurs "chmoder" les permissions nécessaires. En effet, si bob n'a pas les droits d'écriture "classique" sur le dossier, même si le partage samba est configuré en "writable", bob ne saura rien écrire dans le dossier via samba.

5.1.3 Tester la configuration avec testparm

Comme vous avez sans doute déjà pris l'habitude de le faire, il ne vous reste plus qu'à recharger le service dont vous venez de modifier la configuration au point précédent. Mais avant, je vous invite à essayer une commande :

man : `testparm`

```
testparm is a very simple test program to check an smbd(8) configuration
file for internal correctness. If this program reports no problems, you
can use the configuration file with confidence that smbd will successfully
load the configuration file.
```

Et donc, si tout c'est bien passé, vous pouvez recharger le serveur avec une des commandes suivante. Dans le cas contraire, il vous faut corriger les erreurs dans le fichier de configuration.

```
# /etc/init.d/samba reload
```

```
# service samba reload
```

5.1.4 Ajout des utilisateurs samba

Samba utilise les comptes utilisateurs pré-existant sur la machine mais pas leurs mots de passe. Vous devez donc attribuer des mots de passe samba aux utilisateurs concerné. Pour ce faire, vous devez utiliser la commande "smbpasswd" qui fonctionne comme la commande "passwd" :

man : `smbpasswd`

```
By default (when run with no arguments) it will attempt to change the
current user's SMB password on the local machine. This is similar to the
way the passwd(1) program works.
(...)
When run by an ordinary user with no options, smbpasswd will prompt them
for their old SMB password and then ask them for their new password twice,
to ensure that the new password was typed correctly.
(...)
When run by root, smbpasswd allows new users to be added and deleted in
the smbpasswd file (...)
```

La première fois, la commande doit être utilisée par root pour créer un mot de passe samba pour l'utilisateur ce qui l'ajoute dans le fichier des mots de passe samba :

```
# smbpasswd <utilisateur>
```

Ensuite, l'utilisateur pourra ensuite modifier sont mot de passe en entrant le précédent :

```
$ smbpasswd
```

5.1.5 Client : smbclient

Le client smbclient sensiblement de la même manière que le client ftp. Lancez-le avec la commande suivante :

```
$ smbclient //<ip serveur>/<nom du partage> [-U <utilisateur>]
```

Une fois votre mot de passe entré, vous vous retrouvez dans un shell spécial permettant de parcourir le serveur samba. La commande "help" vous permet d'afficher l'ensemble des commandes utilisables dans ce shell. Voyons rapidement les bases. Certaines vous sont familières car elles fonctionnent comme leurs versions "habituelles" : **ls**, **cd**, **pwd**, **more**, **exit**. D'autres le sont moins, en voici deux autres expliquées :

put Prend en paramètre un fichier local et l'envoie sur le répertoire courant du serveur.

get Prend un paramètre un fichier distant et le place dans le répertoire courant du client.

Voila, vous devriez survivre à votre premier passage sur un shell ftp-like. N'hésitez pas à tester un put sur le read only.

5.1.6 Mount samba : smbmount

Pour utiliser smbmount, vous devez installer un nouveau paquet : smbfs. Mount vous étant déjà familier, je ne vais pas m'attarder sur la commande suivante :

```
# smbmount //<ip serveur>/<nom du partage> <point de montage> -o
user=<utilisateur>
```