

Chapitre 1 – Installation d'un système Unix

Introduction

Linux est une version libre de Unix, multi-tache, multi-user, multi-session, on peut y accéder à distance. Il existe plusieurs distribution de linux, il y a entre autre, Red Hat, Debian, Suse, chacun à sa spécificité cela montre que rien n'est figé dans le monde Unix.

Choisir la bonne distribution linux

Plusieurs raison poussent à choisir un linux plutôt qu'un autre, on choisit un système linux qui est déjà utilisé par des connaissances car ceci permet la discussion et la solution de certain problème rencontré.

Dans le choix d'un système linux le cycle d'évolution du produit garanti est important, par exemple Fedora core relache des nouvelle version tout les 4 à 6 mois, comparé à sa version payante Red Hat Entreprise dispose d'une nouvelle version tout les ans et demi, la raison est que les programmes en beta-test sont d'abord testé par les users de Fedora Core, et si les programmes fonctionne bien alors ceux-ci seront ajouté à RH. De plus il faut faire attention à la gestion de la maintenance, par exemple Fedora Core dispose d'une mise à jour de sécurité tout les 2 3 mois, cela veut dire que après un an, la version d'aujourd'hui ne sera plus supportée ce qui est dommageable car de grosse faille de sécurité peuvent restés. Dans le cas de RH, le constructeur garanti un support de la maintenance durant 7 ans cela veut dire que l'on peut gardé un même système RH durant 7 ans maximum sans le changer. Pour un serveur le choix de RH est sensé, au contraire de Fedora Core qui est plus axé poste de travail.

L'installation d'un linux

A l'installation d'une distribution Linux, il faut toujours vérifier les checksums de l'ISO utilisé car celui-ci peut être corrompus. Dans le cas d'une installation d'un serveur il vaut mieux laissé la langue par défaut en anglais, car il est plus simple lors de l'apparition d'un message d'erreur de rechercher sur un moteur de recherche (style google) le message d'erreur en anglais qu'en français, de plus dans la même vision l'anglais n'utilise pas d'accent (é, è, à) et on ne sait pas comment les moteurs de recherche interprète cela. Aussi une autre raison cela évite des problèmes linguistique.

Si on a déjà une partition linux sur une machine mais qu'on ne sait plus y accéder pour diverses raison (perte du pwd root, problème de boot, etc...), à l'écran d'installation, à l'invite « Boot : » il faut renseigner « linux rescue » et suivre la marche à suivre. Cette démarche fonctionne avec n'importe qu'elle distribution et pour n'importe la quelle.

On peut aussi installé une distribution autrement que par un ISO, exemple, par un ftp, ... Pour ça, à l'invite de « boot : » il faut renseigner la ligne « linux askmethod ».

Le choix de la partition des disques

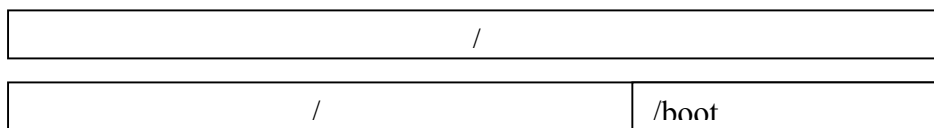
Le choix des partitions est très importants pour un serveur, il accroît la sécurité du système. Le principe est « qu'il ne faut pas mettre tout ses œufs dans le même panier ».

Pour une machine qui servira comme simple poste de travail, il n'est pas nécessaire d'avoir toutes ces partitions.

Lors de l'installation, nous pouvons choisir une configuration automatique des partitions ou manuelle. Si c'est en automatique, le LVM (Logical Volume Manager) s'occupera de partitionner le disque automatiquement. Le LVM permet de travailler dynamiquement avec les partitions, tel que la redéfinition de la taille d'une partition, l'ajout d'un disque, ... Cette façon de faire n'est préférée que pour un système local.

Par défaut, LVM partitionne le disque en 2, une partition pour racine (/) et une partition boot (/boot) de 100mo. Ceci n'est pas conseillé pour un serveur car, à un moment il va falloir réinstaller le serveur, et certains des fichiers appartiennent à des utilisateurs, d'autres fichiers qu'on a ajoutés soit même, qu'on a compilés et donc si on réinstalle le système on va virer toute cette partie là, il est donc nécessaire de séparer la partie users et la partie système. De plus il peut y avoir des fichiers qui peuvent être locaux à une machine et d'autres qui peuvent être partagés sur un réseau, il faut donc séparer ce qui peut s'exporter et ce qui ne peut pas. Troisièmement si le système est corrompu tout le système est corrompu, il est donc impératif d'utiliser les partitions car la corruption d'un système de fichiers ne se propage qu'à la partition. Quatrième raison, toutes les actions effectuées sont enregistrées dans un fichier log, il est donc important que ce fichier log soit séparé du reste, si quelqu'un se met en tête de bourrer ces logs il reste de compromettre tout le système car le système a besoin d'une certaine quantité d'espace disque pour bien fonctionner.

Partition pour un simple poste



Partition pour un serveur



/
 swap (sa taille = 2x la mémoire)
 /boot (128mo)
 /usr (pgm locaux)
 /var (pour les logs, taille très grande)

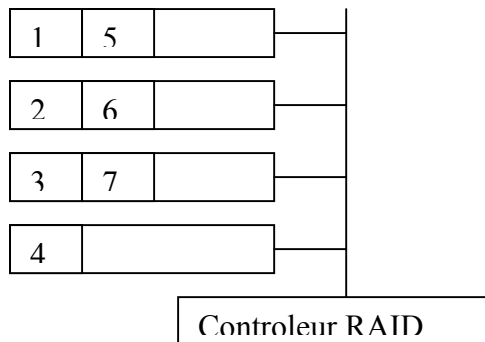
/boot est de taille petite car avant les boot loaders des vieilles machines ne savaient pas référencer plus de 1024 blocs (mais ceci est historique).

/home n'est pas à faire comme partition car /home est plus axé « local » à la machine.

Les RAIDs (Redondant Array Inexpensive Disk)

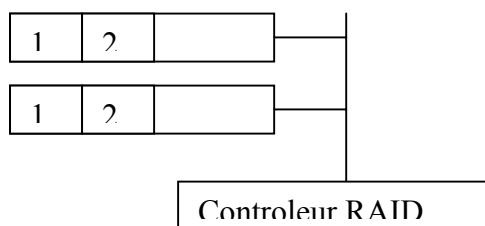
Le RAID est une solution hardware aux crashes disque. Il existe différentes versions du RAID, cela va du RAID 0 au RAID 7, tous proposant des solutions différentes. Dans ce chapitre nous allons voir que le RAID 0, 1 et 5¹.

Le RAID 0



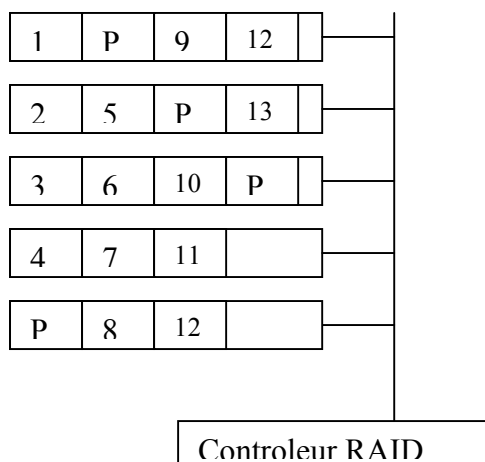
Le RAID 0 (RAID linéaire) ne propose aucune solution de récupération de données, son avantage est qu'il disperse les blocs écrits sur l'ensemble des disques. Par conséquent la vitesse d'écriture et de lecture est augmentée, du fait que un disque peut envoyer un bloc dans le buffer et qu'un autre simultanément peut déplacer son bras et se préparer à lire le bloc voulu. Par contre un disque crash une partie des data sont perduent.

Le RAID 1



Le RAID 1 appelé aussi mirroring, propose la duplication des données d'un disque sur un autre disque. Cette solution n'offre pas beaucoup d'avantage, si ce n'est que en cas de crash disque il y a toujours le second qui peut dépanner, pour la série des point noir, le RAID 1 coûte cher, est lent car pour une écriture d'un enregistrement il faut 2 écritures sur les disques. Ici les deux disques doivent avoir exactement la même taille. Un avantage qu'on peut retrouver est lors de la lecture, comme les mêmes data sont sur les deux disque on peut lire sur les deux disque et donc la vitesse peut être augmentée.

Le RAID 5



Le RAID 5 est une solution qui reprend l'avantage du RAID 0 (La rapidité) et en plus met en place un système de correction d'erreur (simple parité). La parité est vérifiée par « colonne », un bloc de parité est utilisé de façon à ce que ces blocs ne se retrouvent pas tous sur le même disque mais qu'il soit dispersé pour une plus grande sécurité. Nous pouvons donc lorsqu'un disque flanche reconstruire les blocs perdus grâce aux autres blocs de parité. La capacité totale est des n-1 disques.

Un disque supplémentaire peut être ajouté à ceux-ci, c'est un disque « SPARE », il sert de réserve en cas de crash d'un des disques.

¹ Pour plus d'information : <http://www.commentcamarche.net/protect/raid.php3>

Pour faire du RAID il faut avoir un contrôleur de disque RAID, ce qui est assez coûteux, Linux propose une solution Software au RAID.

Les commandes/ les utilitaires

df : Est une commande qui fournit la quantité d'espace occupé des systèmes de fichiers. Il affiche aussi toutes les partitions du système

fdisk : Est un utilitaire qui permet de modifier les partitions du système. Création de nouvelle, suppression d'ancienne, modification de taille.

Lorsqu'il est exécuté les commandes suivantes font :

N : crée une nouvelle partition sur l'espace libre du disque

W : sauve les nouvelles partitions et demande un reboot du user

D : delete une partition

mount périphérique répertoire : est une commande qui permet de monter le système de fichier de périphérique sur répertoire.

Le problème du mount est que si on veut que le montage d'un périphérique sur un répertoire soit persistant il ne fonctionne pas, en effet une opération d'umount est exécuté lors de l'arrêt du système, et donc après démarrage tout les périphérique monté précédemment ne le sont plus. Afin d'enlever ce problème linux a mis en place un fichier appelé fstab qui contient tout les périphériques à monter au démarrage.

Le fichier /etc/fstab

Exemple de fstab

/dev/hda1	/mnt/DOS_hda1	vfat	user,exec,conv=binary	0	0
/dev/hda5	/mnt/DOS_hda5	vfat	user,exec,conv=binary	0	0
/dev/hda3	swap	swap	defaults	0	0
/dev/sda4	/	ext2	defaults	1	1
/mnt/floppy	/mnt/floppy	supermount	fs=vfat,dev=/dev/fd0	0	0
none	/proc	proc	defaults	0	0
none	/dev/pts	devpts	mode=0620	0	0
/mnt/cdrom	/mnt/cdrom	supermount	fs=iso9660,dev=/dev/cdrom	0	0

La première colonne désigne le périphérique concerné (**hd pour IDE et sd pour SCSI**). Par exemple /dev/fd0 pour le lecteur de disquette.

La deuxième est le point de montage du système de fichier. (les partitions de mémoire secondaire utilisent le mot clef none)

La troisième indique le type de système de fichier (la valeur "ignore" fait ignorer la ligne dans le cas d'une partition qui n'est pas encore utilisée, la valeur "auto" signifie détection automatique)

La quatrième définit les options de montage du système de fichier séparées par une virgule

La cinquième donne la fréquence de sauvegarde du système de fichier. 0 est la valeur de défaut et signifie aucune sauvegarde.

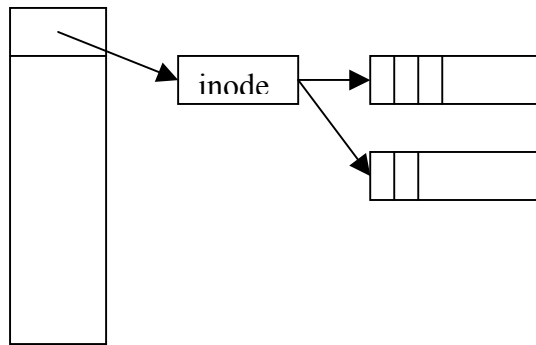
La sixième et dernière donne l'ordre de vérification du système de fichier par fsck lors de l'initialisation. Il est conseillé de laisser les valeurs par défaut.

Mkfs : est une commande qui permet de créer un système de fichier sur un périphérique, il connaît plus de 80 systèmes de fichier.

Utilisation : `mkfs -t typeSysFile sysFile`.

EXT2 & EXT3 file systems

Ext2 et ext3 sont des systèmes de fichier propres à linux. Le premier, ext2 est le système de fichier standard. Le second, ext3, est identique à ext2 avec un système journaliser. Un tel système de fichier a été conçu pour éviter la perte de donnée causée par une coupure de courant. Pour rappel, un file system de Unix est une table d'inode pointant sur chaque inode qui eux même pointe vers des blocs de données. Effacer un fichier, ça consiste à effacer l'inode et couper les pointeurs vers les blocs ou inode.



Imaginons un système de fichier ext2, on effacerait un fichier et d'un coup une coupure de courant survient. Le résultat est que le fichier l'inode n'existe plus, mais le bloc de donnée est toujours présent. Donc on a un fichier qui est orphelin (sans nom). Pour arranger ce problème on a mis en place un utilitaire qui s'appelait fsck qui était chargé de vérifier la consistance des disques (voir si les fichiers et répertoires sont à la bonne place) tout le disque à chaque redémarrage dont l'arrêt a posé problème. Le problème avec le fsck était que pour vérifier un disque c'était que cela pouvait prendre plusieurs heures.

C'est pourquoi on a repensé à un autre système, c'est à dire un système journaliser, cela consiste à placer un journal au file système. Et dans ce journal, lors de l'effacement du fichier il est écrit de manière atomique que l'on va l'effacer, et puis après seulement on efface le fichier. L'idée est que si une panne de courant se produit, si le système voit qu'on a voulu effacer le fichier, le système l'effacera.

Chapitre 2 : User & SSH

User

Après l'installation d'un système Unix, dans notre cas Red Hat, nous devons ajouter des utilisateurs (user). L'utilisation de user permet entre autre à définir les droits de chacun ouvrant une session sur le système. L'utilisateur par défaut est le root, il a tout les droits, de là, on peut voir l'intérêt des utilisateurs : si tout le monde utilise le root, tout le monde peut faire n'importe quoi sans qu'on sache qui à fait quoi.

Dans le monde Unix, chaque user est identifier par un UID. Lors de la création du premier user sur un système, traditionnellement, celui-ci reçoit l'UID 500. Les suivants, recevront le dernier UID attribué incrémenté de un.

Sans penser, à un système qui doit servir de serveur, nous pouvons ajouter un utilisateur assez facilement via l'interface graphique. Mais cette méthode à ces limite lorsqu'il s'agit de créer plusieurs dizaines de nouveaux users.

Lorsque l'on doit entrer le nom de famille du user il est préférable de le mettre en majuscule, cela évite des confusions style « Gaétan David ».

Attribution des UIDs

Lors de la création d'utilisateurs, il est intéressant de réfléchir à la meilleurs façon d'attribué les UIDs. A savoir, il existe 3 types de users, les users systèmes, les users physiques, et les users publics.

Les users systèmes, sont des users non physique, ayant certains privilèges sur des programmes. Par exemple, il existe un user « apache » ayant tout les droits pour gérer le serveur apache.

Les users physiques, ... no comment.

Les users publics, utilisable par plusieurs personnes, par exemple un login « guest ».

On pourrait donc imaginer de définir des ranges d'UID pour chaque types de users. De 0 à 499 pour les users systèmes, de 500 à 999 pour les users physique, et de 1000 à 65533 pour les users publics (Note : 65533 est le max pour UID car codé sur 2 bytes, et deux UID sont réserver).

L'avantage d'avoir une bonne organisation des users, que cela permet de cibler certains users et pas d'autre, exemple on désire envoyer un mail à tout les users physiques, il suffit de vérifier que l'UID se trouve en 500 et 999. Un autre intérêt, est que lors du monitoring du système, lorsqu'on voit des actions avec par un UID de type système il faudra être plus vigilant, car il dispose de plus de droit que les autres.

Organisation du /home

Lors de la création des users, comme ceux-ci peuvent être très nombreux sur un serveur, le /home peut devenir très vite énorme, il est possible que lors de l'appel de la commande ls, l'affichage de la liste du répertoire mette plus de temps selon le nombre d'éléments à afficher.

Plusieurs organisations existes aussi.

Décomposé /home en département, compagnie, ...

Ex : /home/exxon/ /home/total/ /home/shell/

Cette façon de faire réduira le nombre d'élément à afficher. Seulement, il existe deux problème à cette organisation. Premièrement, si un user se trouvant dans /home/exxon, et qu'il est muté vers total, celui-ci doit être déplacé. Deuxièmement, si exxon et total fusionne, il faudra fusionner ces deux directory.

Donner une organisation alphabétique à /home

Ex : /home/a /home/b /home/c ...

Déplacer /home sur une autre partition

Ex : /disk/home/

Création du répertoire /home

REEL : /disk/home/c/myUser/

LIEN : /home/c/myUser/

Création de users via la ligne de commande

La commande pw, permet d'afficher tout les users d'un système. Le problème est que en sortie les users ne sont pas trié par UID. Ceci peut se faire grâce à l'éditeur vi (très puissant lui).

Pw >> listeUser

Vi listeUser

:1,\$!sort -t: -k 3,3

La commande useradd permet d'ajouter un utilisateur. Elle demande en paramètre :

-g monGroupe1 : le groupe auquel appartient l'user, celui-ci doit exister.

-d /ma/home/directory/ : la directory doit exister avant la création du user.

-c "Un commentaire généralement Prénom NOM"

monUser1



La commande groupadd monGroupe1, permet d'ajouter un groupe.

Nous pouvons modifier les users, en accédant au fichier /etc/passwd. Soit en exécutant vi /etc/passwd, soit en exécutant vipw. Remarque la dernière commande à l'avantage d'afficher les messages d'erreurs qu'on aurait provoqué, et après modification de passwd, nous demande si on désire modifier le fichier /etc/shadow, pour modifier le mot de passe.

Il est possible de créer un script qui automatise la création d'un login. Nous pouvons en plus écrire un script qui automatise le processus de création un nombre fini de fois, en ayant juste un fichier avec tout les noms des nouveaux users.

//Exemple du script

Le fichier Passwd

Son path /etc/passwd

Il contient pour tout les users, le mot de passe crypté. Si un user n'a pas de mot de passe alors il sera écrit « !! » dans le fichier. Lorsqu'on crée, les users automatiquement, aucun mot de passe n'est attribué. Il est utile de mettre dans le fichier, autre chose que les « !! ». Par exemple on écrira « INACTIVE » à la place des « !! » pour chaque nouveaux user. Ceci permet de bloquer, des comptes pour une raison ou une autre.

La commande userdel, sert à supprimer un login et ses fichiers, il n'est pas à conseiller d'utiliser cette commande. Mais on utilisera plutôt, l'écriture d'un état à la place du mot de passe dans le fichier passwd. Par exemple, si on désire qu'un user ne se connecte plus mais que l'on puisse encore accéder à ses fichiers. On écrira un état du style « DELETED » dans le fichier passwd.

Attribution des permissions

Chaque utilisateur a besoin de faire tel ou tel sorte d'action sur le système. Il faut donc définir les droits que les users ont ou non. Au départ aucun users n'a de droit que sur sont /home. Il y a deux solutions, soit on donne le mot de passe root (chose à ne jamais faire), soit on lui attribue des droits.

Le fichier sudoers contient les droits des différents user en possédant.

On peut y accéder grâce à vi /etc/sudoers ou visudo

Le fichier est organisé en LOGIN/GROUP HOST=(sous le nom de) cmd.

LOGIN/GROUP : correspond au login du user, ou un groupe

HOST : est la machine par lequel il tente de faire une action

(sous le nom de) : certain user peuvent prendre d'autre identité

cmd : la commande que le user peut exécuter, doit toujours être avec le path entier ex : /bin/lis

par défaut : root ALL = (ALL) ALL

Configuration du réseau

Le fichier /etc/network/interfaces ou etc/ifcfg-ethX (X = le numéro de l'interface de la carte réseau)

Sert à configurer l'adresse ip, le gateway, le masque de sous-réseau, etc

/etc/network/interfaces

```
# /etc/network/interfaces
# Fichier de configuration d'exemple des interfaces réseau
# pour faire un serveur NAT
# Formation Debian GNU/Linux par Alexis de Lattre
# http://www.via.ecp.fr/~alexis/formation-linux/

# Plus d'informations dans "man interfaces"
```



```
# L'interface "loopback"
auto lo
iface lo inet loopback
    # Activation de la fonction de forwarding IP au niveau du noyau
    up echo "1" > /proc/sys/net/ipv4/ip_forward

# L'interface "eth0" connectée à Internet (configuration par DHCP)
auto eth0
iface eth0 inet dhcp

# L'interface "eth1" connectée au réseau local (IP privée fixe)
auto eth1
iface eth1 inet static
    address 192.168.0.1
    netmask 255.255.255.0
    broadcast 192.168.0.255
```

Le fichier /etc/hosts

Ce fichier sert à faire le mapping entre le hostname et une adresse ip

Le fichier /etc/hostname

Ce fichier contient le nom d'hôte de la machine

SSH ¹

La commande ssh permet de se connecter, ou d'exécuter des commandes sur un système distant. Dans une configuration standard, le système distant demande un login et un mot, ou une phrase, de passe. Cette configuration n'est cependant pas utilisable lorsque les commandes sont exécutées automatiquement, par exemple sous le contrôle du mécanisme crontab, ou l'utilisateur n'est pas présent pour répondre à ces questions.

Dans l'exemple ci-dessous, l'utilisateur lisa du système local localhost (lisa@localhost) souhaite exécuter, sans entrer de mot de passe, une commande sous le nom de martino sur le système distant sdsrv1 (martino@sdsrv1).

L'idée consiste d'abord à générer, sur le système local, une paire de clés, une privée et une publique, avec une phrase de passe vide.

```
[lisa@localhost lisa]$ ssh-keygen -t dsa
Generating public/private dsa key pair.
```

¹ Repris de www.unixacademy.org

```
Enter file in which to save the key (/home/lisa/.ssh/id_dsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/lisa/.ssh/id_dsa.
Your public key has been saved in /home/lisa/.ssh/id_dsa.pub.
The key fingerprint is:
3e:0d:95:f5:d3:fe:44:1d:6b:41:aa:8f:d0:56:24:7b lisa@localhost.localdomain
```

A chaque question posée par la commande, il faut répondre en acceptant la valeur par défaut (nom du fichier de clé, phrase de passe vide) en tapant simplement des return's. La commande a pour résultat la création de deux fichiers `id_dsa` et `id_dsa.pub`, resp. clé privée et clé publique, dans le répertoire `.ssh`.

```
[lisa@localhost lisa]$ ls -l .ssh
total 12
-rw----- 1 lisa lisa 672 Feb 1 14:32 id_dsa
-rw-r--r-- 1 lisa lisa 618 Feb 1 14:32 id_dsa.pub
-rw-r--r-- 1 lisa lisa 218 Feb 1 14:30 known_hosts
```

Il est intéressant de remarquer que le fichier `id_dsa`, clé privée, à un droit d'accès de 0600: seul l'utilisateur `lisa` a le droit de lire et de modifier le contenu de ce fichier.

Maintenant, la clé publique de `lisa@localhost` doit être ajoutée dans le fichier `~/.ssh/authorized_keys2` de `martino@sdsrv1`. Nous allons auparavant nous assurer que le répertoire `.ssh` existe bien chez `martino@sdsrv1` avec des droits d'accès corrects. A ce stade, il est encore nécessaire d'introduire le mot de passe de `martino@sdsrv1`.

```
[lisa@localhost lisa]$ ssh Martino@sdsrv1 "mkdir -p .ssh ; chmod 700 .ssh"
Martino@sdsrv1's password:
```

La clé peut maintenant être copiée dans le fichier `~/.ssh/authorized_keys2` avec des droits d'accès 0600:

```
[lisa@localhost lisa]$ cat .ssh/id_dsa.pub | ssh Martino@sdsrv1 "cat - >>
.ssh/authorized_keys2"
Martino@sdsrv1's password:
[lisa@localhost lisa]$ ssh Martino@sdsrv1 chmod 600 .ssh/authorized_keys2
Martino@sdsrv1's password:
```

A ce stade, il n'est plus nécessaire d'introduire un mot de passe pour passer de `lisa@localhost` à `martino@sdsrv1` !

```
[lisa@localhost lisa]$ ssh Martino@sdsrv1
[Martino@sdsrv1 Martino]$
[Martino@sdsrv1 Martino]$ exit
Connection to sdsrv1 closed.
[lisa@localhost lisa]$
```

Supplément SSH

Introduction

Il existe plusieurs protocoles Internet permettant de se connecter à un ordinateur distant : telnet, les r-commandes (rlogin, rsh ou encore rcp), ssh (*Secure Shell*). Cependant telnet et les r-commandes font circuler les informations par le réseau en clair, c'est pourquoi il est vivement recommandé d'abandonner ces 2 protocoles. Ssh, lui, est plus axé sécurité, et ce pour deux raisons :

- Le client et le serveur s'authentifient mutuellement, ce qui évite qu'un pirate se fasse passer pour l'un ou l'autre.
- Les données échangées sur le réseau sont cryptées

A savoir, SSH n'est pas une protocole utilisé que dans le monde UNIX, ce protocole fonctionne aussi bien sous Windows, un Mainframe, Mac OS...

Se connecter avec ssh

Il existe plusieurs façon de se connecter via ssh.

La première façon est de simplement indiquer au protocole la machine distante. Dans ce cas, il faut s'assurer que votre nom d'utilisateur est connu sur la machine distante, sans quoi l'accès vous sera refuser

```
Tantor$jungleCLT : ssh machinedist
```

Dans le cas ou la machine distante vous à créer un compte user, différent du votre sur votre machine local, il suffit d'indiquer dans la commande ssh le login à utilisé pour se connecter. Pour cela, il existe deux manière, soit on utilise l'option -l (L en minuscule).

```
Tantor$jungleCLT : ssh machinedist -l Tarzan
```

Soit on utilise la connexion du type user@machinedistante

```
Tantor$jungleCLT : ssh Tarzan@machinedist
```

Petite parenthèse, à chaque fois qu'on a taper machinedist, celle-ci est représenté, soit par son nom logique (ex : lePCdeMAchambre, totoServeur,...), soit il peut s'agir d'une adresse IP.

Si c'est votre première connexion, sur la machine distante, vous aurez un message tel que celui ci :

```
Tantor$jungleCLT : ssh toto@machinedist
The authenticity of host 'machinedist' (111.222.333.4)' can't be established.
RSA1 key fingerprint is 1z:2y:3x:4w:56:78:98:78:ab:cd:ef:01:23:45:67:89.
Are you sure you want to continue connecting (yes/no)?
```

Ce message indique simplement que le serveur vous a créé un accès temporaire, il suffit de répondre “yes”, et la connexion sera établie.

Authentification par mot de passe

Si vous pouviez vous connecter comme cela à une machine distante, tout serait trop simple, pour vous ou pour le pirate. Après la connexion à cette machine distante, celle-ci vous demandera le mot de passe du user que vous prétendez être.

```
Tantor$jungleCLT : ssh Tarzan@machinedist
Tarzan@machinedist password:
```

Authentification par clé

Bien sur il existe une authentification permanente pour lequel nous ne devons plus entrer de mot de passe. Ssh utilise un système de clé d’authentification.

Il y a deux type de clé, la clé public et la clé privée. La machine local, elle, va générer ces deux clés, et elle va en gardé une copie, dans des fichiers spécifique. La machine distante, elle, va recevoir uniquement la clé public.

Pour commencer, il faut générer les clés, pour ca utilisons la commande ssh-keygen, qui va créer les clés et par défaut mettre la clé privée dans le fichier /home/.ssh/id_dsa (vous pouvez le changer).

```
Tantor$jungleCLT : ssh-keygen
Generating public/private dsa key pair.
Enter file in which to save the key (/usr/home/toto/.ssh/id_dsa):
```

Après avoir créé le fichier pour la clé privé, le système vous demandera si vous désirez utilisé une phrasepass, celle-ci sert de mot de passe lors de la connexion, si vous ne mettez pas de phrasepass alors la connexion se fera automatiquement sans rien demandé. Notez que l’utilisation de la phrasepass augmente la sécurité.

Dans tout les cas, vous aurez un message comme celui-ci à la fin qui indique que la clé public a été sauvé dans le fichier home/.ssh/id_dsa.pub.

```
Your identification has been saved in /usr/home/tarzan/.ssh/id_dsa.
Your public key has been saved in /usr/home/tarzan/.ssh/id_dsa.pub.
The key fingerprint is:
1a:2a:3e:4a:1a:65:1c:89:10:92:9c:5c:1f:75:cc:de
Tantor$jungleCLT :
```

Voilà, maintenant vos deux clés sont créés. Mais qu’en faire ? C’est pas trop compliqué ! Tout d’abord votre clé privé, si on dit qu’elle est privé c’est pas pour rien, il n’y a que vous et

vous seule qui puissiez l'avoir, la lire, ... Donc vous en faite rien et la laissé là où elle se trouve.

Ensuite la clé public, comme le mot l'indique, elle est public donc tout le monde peut y accéder. Tout le monde ? non pas vraiment uniquement les serveurs ssh chez qui vous voulez vous connecter sans passé par l'authentification par mot de passe cité là-haut. La clé public, doit être copié dans un fichier de la machine distance qui est /home/.ssh/authorized_keys.

Pour donner la clé public à la machine distante, il n'est pas nécessaire d'ouvrir une connexion ssh, et de retaper la clé tel-quel, dans le fichier /home/.ssh/authorized_keys de la machine distante, il y a plus rapide et plus efficace. Il suffit simplement de prendre votre fichier, et de le transférer à la machine distante.

Transfert de fichiers

Le transfert de fichier utilisant le protocole ssh existe. Expliquons-le à l'aide de notre transfert de clé public.

Tout d'abord la commande est scp. Le S pour ssh, et CP pour copy, et oui parfois on fait pas dans l'original dans unix !

Utilisation sans authentification par clé, nous désirons envoyé trois fichiers à un serveur, il suffit d'appelé la commande scp en mettant sur la ligne de commande les noms et directory des trois fichiers ainsi que l'adresse IP ou le nom logique de la machine distante sans oublier les « : » à la fin. La machine distante vous demandera le mot de passe, une fois entré, le transfert commencera.

NB : comme la connexion à ssh, scp exige que le user utilisé existe sur la machine distante, ref : voir « se connecter avec SSH », plus haut.

```
Tantor$jungleCLT : scp fichier1 fichier2 fichier3 monServeur:
Tantor$jungleCLT : scp fichier1 fichier2 fichier3 monServeur:
Tantor@monServeur's password:
fichier1    100% |*****| 2263    00:00
fichier2    100% |*****| 2263    00:00
fichier3    100% |*****| 2263    00:00
```

On peut aussi transférer des répertoires grâce à l'option -r

```
Tantor$jungleCLT : scp -r repertoire monServeur:
```

Maintenant si on veut changer le nom du fichier sur la machine distante par rapport au nom d'origine du fichier, il suffit d'indiqué le nouveau nom de ce fichier après les « : »

```
Tantor$jungleCLT : scp ./ssh/id_dsa.pub monServeur:coincoin.txt
```

Voilà maintenant on a tout pour upé la clé public sur la machine distante.

```
Tantor$ jungleCLT : scp fichier1 monServeur: ./ssh/authorized_keys
```

Vous pouvez dès lors vous connecter sur la machine distante sans utiliser de mot de passe, juste la phrasepass si vous l'avez ajoutée.

03_Daemon&SSH

Les DAMEONS

Les daemons sont des processus étant exécuté au démarrage du système. Ils servent en général à répondre à des requêtes du réseau, à l'activité du matériel ou à d'autres programmes ou services en exécutant certaines tâches

Il est possible de voir les services tel que ssh, qui sont exécuter pour le moment sur le système via la commande **ps -aux**.

Voir qu'un service utilisant un daemon est exécuter ne veut pas dire que le daemon est exécuter, pour cela on peut vérifier si le daemon est exécuter via la commande **lsuf**

La commande lsuf

Cette commande nous permet de voir tout les fichiers ouverts d'un processus actif. En utilisant les options **-iTCP**, ceci nous permet de voir tout les fichiers ouvert pour les connexions réseau, avec le numéro de port auquel il est lié. **Lsof -iTCP**, peut servir entre autre pour savoir si on est en train de se faire pirater.

Le boot

Linux utilise un système de démarrage nommé SysVinit. Il est basé sur le concept de *niveaux d'exécution*. SysVinit (qu'on appellera *init*) se base pour fonctionner sur un système de niveaux d'exécution. Ils sont au nombre de 7 (de 0 à 6) (en réalité, il y a plus de niveaux d'exécution que cela, mais ils sont réservés à des cas spéciaux et ne sont généralement pas utilisés. Le niveau d'exécution par défaut est le niveau 3.

Voici la description des différents niveaux d'exécution tels qu'ils sont fréquemment implémentés :

- 0: arrête l'ordinateur ;
- 1: mode mono-utilisateur ;
- 2: mode multi-utilisateur sans réseau ;
- 3: mode multi-utilisateur avec le réseau ;
- 4: réservé à la personnalisation, sinon identique au niveau 3 ;
- 5: identique au 4, utilisé généralement pour une connexion graphique (comme l'xdm de X ou kdm de KDE) ;
- 6: redémarre l'ordinateur.

Il existe un certain nombre de répertoires sous `/etc/rc?.d` où ? est le niveau d'exécution et `rcsysinit.d` qui contient un certain nombre de liens symboliques. Certains commencent par un K, les autres par un S et tous ont deux chiffres après la lettre initiale. Le K signifie d'arrêter (kill) un service, et le S (start) d'en démarrer un. Les chiffres déterminent l'ordre d'exécution des scripts, de 00 à 99 ; plus un nombre est petit, plus tôt il sera exécuté. Lorsque init passe à un autre niveau d'exécution, les services appropriés sont arrêtés et d'autres sont démarrés.

Le niveau d'exécution est déterminé (dans l'ordre) soit :

- lors du boot : si vous précisez un niveau sur la ligne de commande du noyau (par exemple, au prompt LILO, taper "linux 1"),
- dans le fichier `/etc/inittab`, où le runlevel par défaut est défini,

- par la commande `init <runlevel>` qui permet de changer de runlevel en cours de fonctionnement

Chaque service système ou réseau est contrôlé par un script présent dans le répertoire `/etc/init.d/`. Dans ce répertoire, se trouvent les scripts des différents services proposés par le système. Dans chacun de ces scripts, il existe les sous-routines `start` – `stop` – `reload`.

Le fichier `/etc/rc5.d` (à titre d'exemple)

A ce niveau il exécutera tout les fichiers commençant par S en suivant l'ordre croissant.

Le pare-feu (`iptables`) est lancé avant le réseau (`network`) pour ne pas créer de fenêtre de vulnérabilité

Lors du shutdown ce sont les fichiers commençant par K qui sont exécutés dans l'ordre croissant. De ce fait le réseau devrait s'éteindre avant le pare-feu.

La commande **CHKCONFIG**

La commande `chkconfig` peut également être utilisée pour activer et désactiver les services

Ajout d'un service à la base :

```
# chkconfig --add service
```

Suppression d'un service de la base :

```
# chkconfig --del service
```

Configuration des niveaux auxquels le service doit démarrer/s'arrêter :

```
# chkconfig --level <niveaux> service on|off
```

Pour ajouter un service dans tous les niveaux de démarrage:

```
#chkconfig --add le_service
```



Pour lister les services qui se lancent à n'importe quel niveau

```
#chkconfig--list
atd      0:off
1:off 2:off 3:on
4:on 5:on 6:off

xfs      0:on
1:on 2:on 3:on
4:on 5:on 6:on
keytable
0:off 1:off 2:on
3:on 4:on 5:on
6:off
gpm      0:off
1:off 2:on 3:on
4:on 5:on 6:off
```


Le daemon ssh : sshd (pour ssh daemon)

Sshd est un daemon qui écoute sur une porte (=un socket) dont le numéro par défaut est le 22. Cela fait de ssh une grosse faille de sécurité, car les pirates commenceront souvent par attaquer cette porte. Il est donc préférable de modifier le numéro par défaut.

Lorsqu'un client se connecte à un serveur via ssh, il le fait toujours par la même porte (22), celle ci peut alors être changée par le serveur dans son fichier de configuration de ssh. Mais le client lui aussi devra en prendre compte, pour cela il devra se connecter en spécifiant dans la commande ssh le numéro de port auquel il se connecte ex : ssh -p 12345 monServer. Le client a une autre possibilité, c'est de créer et de configurer un fichier ~/.ssh/config

Le fichier /etc/ssh/sshd-config

Ce fichier permet de modifier le port d'écoute de ssh.

```
Port 12345
```

Le fichier ~/.ssh/config

Dans ce fichier, on retrouve toute les différentes configurations pour chaque serveur auquel un client se connectera

```
Host monServer
Hostname mydomain.com
Port 12345
HostKeyAlias = monServer
CheckHostIP = no

Host monServer2
Hostname mydomain.com
Port = 12347
HostKeyAlias = monServer2
CheckHostIP = no
```

Le Super Daemon

inetD

Dans le monde UNIX, c'est le processus INETd qui est traditionnellement chargé d'écouter tous les ports et de basculer sur l'application demandée en fonction du numéro de port de la demande.

Le fichier /etc/inetd.conf

Il s'agit de la configuration de inetd, il contient pour chaque service, une entrée

```
# <service_name> <sock_type> <proto> <flags> <user> <server_path> <args>
#
#ftp      stream  tcp      nowait  root    /usr/sbin/tcpd  in.ftpd  -l -a
#telnet   stream  tcp      nowait  root    /usr/sbin/tcpd  in.telnetd
#
```

Le fichier /etc/services

Ce fichier contient pour chaque service le numéro de port lui étant associé.

```
#
# service-name  port/protocol  [aliases ...]  [# comment]

chargen          19/udp          ttytst source
ftp-data         20/tcp
ftp-data         20/udp
# 21 is registered to ftp, but also used by fsp
ftp              21/tcp
ftp              21/udp          fsp fspd
ssh              22/tcp          # SSH Remote Login Protocol
ssh              22/udp          # SSH Remote Login Protocol
telnet           23/tcp
telnet           23/udp
# 24 - private mail system
smtp             25/tcp          mail
smtp             25/udp          mail
time             37/tcp          timserver
```

XinetD

Les distributions récentes de linux remplacent maintenant INETd par xinetd. Par rapport à INETd, xinetd permet notamment :

- d'individualiser la configuration de chaque application
- des contrôles d'accès plus fins
- de faire un contrôle des services basé sur le temps
- de limiter le nombre de serveurs démarrés

Le fichier /etc/xinetd.d

Permet de configurer xinetd

Le partage de fichier : NFS (Network File System)

Le principe est que le serveur donne l'accès à ses file system aux clients qui se connectent. Le client va demander de monter à distance le file system du serveur. Un daemon s'occupera des requêtes.

Le fichier /etc/exports

Ce fichier contient la liste des répertoires a partager sur le réseau.

04_Sharing

Une des manière de se connecter sur le réseau est **ssh**, via un daemon appelé secure-shell », il existe un autre protocole qui s'appelle Telnet. Pour l'utiliser il faut que le serveur exécute un daemon qui s'appelle telnetd.

Telnet

Lorsqu'un client se connecte sur un serveur, il y a un daemon, telnetd qui est exécuter. Telnet est non sécurisé, c'est à dire que tout ce qui passe à travers le réseau passe en clair. Comme dit précédemment, ce daemon telnetd, est exécuter par le super-daemon : **xinetd**. Nous pouvons voir a l'aide de la commande **lsuf -iTCP**, les daemons étant exécuter à ce moment, **ps -aux** permet de voir les processus en cours.

Lors de la connexion telnet d'un client, ce client voit directement sur quel configuration de machine il se connecte, ce qui est déjà un gros point faible, car une personne mal intentionné n'a pas besoin de chercher sur quel configuration il se trouve et ainsi peut directement attaquer les failles de la configuration dite.

Nmap

nmap [typeDeScan] [Option] Cible

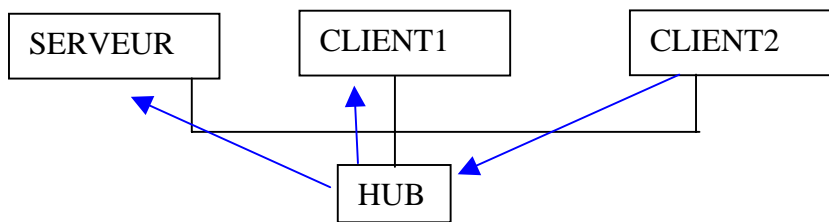
C'est un logiciel qui se connecte sur un serveur, et scanne tout les portes, et ce de manière discrète. La cible peut être de plusieurs types, un host, un ip, une adresse réseau ou un range d'adresse. Nmap peut détecter sur quel OS de la machine on se connecte : le protocole TCP, est un protocole adapter pour chaque OS, il va répondre de manière différente selon qu'on parle à un linux, mac OS, ect... de là nmap peut détecter avec une certaine marge d'erreur sur quel OS on se connecte.

Généralement les scans nmap sont détecter par les pare-feux.

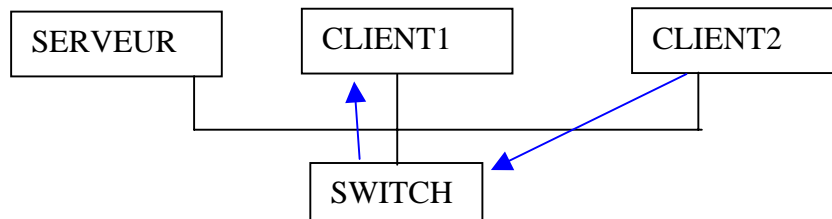
Les options :

- sL : liste les cibles a scanné
- sP : équivaux à ping, pour voir si la cible est online
- S adresseIP : permet de camoufler son ip par une adresse
- sS : mode discret, il n'enverra sur le réseau que les SYN et pas les ACK
- sT : scanne des ports TCP ouverts
- sU : scanne des ports UDB ouverts
- O permet de connaître sur quel OS est la cible
- p port[-port-port...] : scanne la liste des ports donnés
- F : scanne uniquement les ports se trouvant dans le fichier **/etc/services**
- I : offre plus d'information sur les ports ouverts
- o fichier : permet de logué le résultat dans un fichier
- v : pour le mode verbeux
- e int : permet de spécifier une interface particulière a scanné eth0, S0 etc...

Cas concret :



Pour le cas du Hub (ou répéteur), si une machine envoie un paquet sur le réseau, le hub se contentera de redistribuer ce paquet sur tout le réseau.



Dans le cas du switch (ou commutateur), celui-ci va d'abord inspecté l'adresse du destinataire du paquet, et le lui renvoyer et les autres machines sur le réseau n'y verront rien. Si notre serveur pour une raison ou une autre écoute le réseau, celui-ci, ne verra rien, contrairement au hub. Petit bémol, le switch fonctionne avec une table de routage, il suffit qu'on bombarde (flood-attck) le switch avec des messages incompréhensible ou bien formé pour foutre en l'air ces tables de routage, pour qu'ensuite le switch ne soit plus qu'un hub normal. Le deuxième problème, est que parfois il est nécessaire d'écouter le réseau, lors de debbuging, dans ce cas, il est permit dans certains switch, de configurer une des portes pour qu'elle soit le miroir d'une autre, ainsi tout le trafic qui passe d'un coté peut s'écouter par cette porte miroir.

TCPDUMP

Ce logiciel affiche en sortie ce qui se passe sur ses portes réseaux.

Les options :

- ithx : pour le choix de la carte réseau(x pour le numéro)
- s xxx : pour capturer xxx byte
- c xxx : capturer xxx paquet
- w xxx : ou x est le fichier de log

Ethereal

Une manière plus puissante de voir le trafic réseau est d'utilisé tcpdump, d'écrire les paquets (généralement 120 paquets) capturé dans un fichier au lieu de la sortie standard et décrypter le protocole/fichier à l'aide de Ethereal. On peut voir que avec l'utilisation de Telnet, tout passe en clair.

Les packages sous Red Hat/fedora Core

Les packages sont des fichiers qui contiennent sous un format compresser, un ensemble de fichiers (souvent) exécutables. Il se peut qu'il y ait aussi les sources de ces fichiers, et la documentation s'y référant. Les fichiers de package sont souvent de la forme xxx.rpm

La commande rpm

gère les packages

Les options :

-u : indique qu'on fait un update, donc si le paquet est déjà installé il n'y a aucune raison de le refaire.

-v : verbose

-h : affiche 50 marques de hash (?man)

-q : query

-a : all

installation : rpm -ivh nom_paquet.rpm OU rpm -uvh nom_paquet.rpm

désinstallation : rpm -e nom_paquet

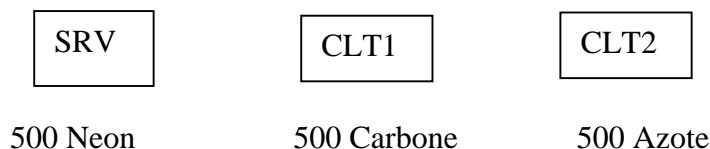
savoir si un paquet existe : rpm -qil nom_paquet_xxx.rpm

savoir a quel package appartient un fichier rpm -qf monFichier

savoir quel sont les packages installés : rpm -qa

Parfois certains logiciels ont besoin d'autre logiciel pour être installé et rpm ne gère pas toujours les dépendances, c'est pourquoi il existe sur certains OS a été mis en place une couche supérieur pour gérer les packages (Red Hat(payant) => up2date, Debian => apt, Mandrake =>urpmi). Ces couches ou gestion supérieure de package gèrent les packages mais aussi les dépendances et les conflits entre packages.

NFS – network file system



Le problème avec NFS est que lorsqu'on a un serveur, et des clients, il se peut que les uids du serveur et ceux des clients entres en contradiction. Les conséquences de cela sont que le client ayant un user d'un uid exactement le même qu'un user sur le serveur, se verrait recevoir les droits d'accès et de propriété de tout les fichiers et commande sur le serveur .

Pour résoudre ce problème il existe plusieurs solutions, la seule limite à ces solutions étant notre imagination. Nous allons en démontrer 2-3.

Le poussage : On s'arrange pour que les fichiers passwd, shadows, group, se recopie du serveur vers les clients de façon automatique. Ou la création de login sur le serveur se ferait de manière synchronisé sur les autres, de même que pour la suppression de login. Il faut penser aussi au cas ou le password changerait, alors il faut le changer sur toute les autres machine. Cette solution amène un problème, si il faut ajouter de nouvelles machines sur le réseau, il faut synchroniser ces nouvelles machines avec l'existant.


Le tirage : On créerait un daemon sur les clients, qui vérifierait de temps en temps si une mise à jour des fichiers passwd, ... sur le serveur et répercuter ces modifications sur ses fichiers locaux. Ceci implique que le daemon tournant sur les clients doit avoir les droits root sur le serveur pour pouvoir lire les fichiers passwd, group, shadows.

Une autre solution plus satisfaisante est d'utiliser des systèmes de directories.

Les système de directories

Il existe différent service de directories parmi eux on retrouve, Ldap (lite weight directory access protocol), NIS (Network information service).

NIS

Toutes les commandes commencent par YP, (YP pour Yellow Page). Le protocole utilisé pour YP est basé sur **RPC (Remote procedure call)**. Les RPC utilisent un ensemble de portes TCP-IP/UDP-IP. Ceci peut être vu comme un défaut de RPC, car il faut faire attention que le firewall ne bloque pas ce qui passe par les portes utilisées par RPC. Un autre défaut, RPC un est protocole non crypté, et donc non sécurisé. La sécurité de RPC est suffisante pour des petits réseaux. 

Comme dans tout les systèmes de directory, il faut qu'il y ait un maître (MASTER), dans notre cas ce sera le serveur. On va devoir configurer le serveur. Pour cela, on va définir un domaine NIS dans le fichier **/etc/sysconfig/network**

```
NISDOMAIN=monDomaine
```

Pour que ces base de données puissent voyagé d'un poste à l'autre via RPC, il faut qu'il y ait un daemon qui tourne. Sur le serveur, il y a 2 daemons qui tourne, le premier est **ypserver** et le second ybind. Sur les clients, on va faire tourner uniquement le **ypbind**.

Ypserver sert à indiquer qui est le serveur et d'où vient les base de données.

Ypbind sert aux postes à faire de la résolution de uid -> login – mot de passe.

Lorsque un client a un uid, et que celui-ci veut avoir le nom du login ou le mot de passe le ybind du client ira se renseigner sur le ypserver du maître.

```
/etc/init.d/ypserver start  
/etc/init.d/ypbind start
```



Le serveur, lui, lorsqu'il a besoin d'une information, il va aller voir dans ses bases de données construite à partir des fichiers /etc/passwd, group, shadow, hosts. Au préalable il faut, initialisé ces data base avec la commande /usr/lib/yp/ypinit -m, le « m » indique que c'est un serveur master.

Les fichiers de configuration se retrouve dans /var/yp/monDomaine. Ce sont ces fichiers qui seront utilisés par le serveur NIS. Pour mettre à jours les bases de données en cas d'ajout de

login où autre, la commande make est utilisée dans le répertoire /var/yp/, celle-ci va comparer les fichiers de passwd et shadow avec ce qui est dans la db, et mettre à jour si nécessaire.

Le client quand il sera démarré, enverra un broadcast sur le réseau pour demander si une machine peut lui servir de serveur NIS.

La commande ypwich sert à savoir qui est la machine qui fournit les informations login, pwd, ...

Un problème existe toujours, pour le moment il existe 2 fichiers de login et de mot de passe, le premier sur la machine local, le second est distribuer par le réseau. Il faut donc dire qui est l'information prédominante. Il faut savoir que information on va consulter d'abord, si c'est l'information du réseau ou du système local. Ce genre de problème est résolu grâce au fichier /etc/nsswitch.conf.

Passwd : NIS file Hosts : NIS file DNS Shadow : NIS file
--

La commande ypcat passwd, vas afficher le fichier passwd fournit par le réseau et non le fichier local.

La commande yppasswd vas afficher le fichier passwd fournit par le réseau.

05_ Les quotas

La gestion des quotas permet de limiter le volume en terme de nombre de bloc ou de nombre de fichier (inode), pour un utilisateur, ou pour un groupe d'utilisateur. Cette limitation se fait sur un file système.

Les limitations

Il existe deux sortes de limitations :

La limitation fixe

Permet d'empêcher l'utilisateur de dépasser le seuil mis en place. Passé cette limite toute écriture pour cet utilisateur sur ce file système est interdite. L'inconvénient de cette limitation est que l'utilisateur pourrait avoir besoin de rapatrier un fichier de grande taille pour un certain temps et qui risque de dépasser le seuil autorisé.

La limitation souple

Cette limitation permet à l'utilisateur de dépasser pour un laps de temps (appelé période de grâce) le seuil autorisé. Lorsque un user dépasse son seuil, le système peut en avertir l'administrateur.

Il est possible d'utiliser ces deux limitations en même temps, c'est à dire donner une certaine souplesse d'utilisation de l'espace disque à un utilisateur, mais en lui donnant tout de même un seuil à ne pas franchir.

Mise en place des quotas

Afin d'utiliser les quotas, il faut créer deux fichiers, dans chaque file système soumis aux quotas. Il faudra, juste après ça, leur donner les droits d'accès nécessaire.

```
touch /home/aquota.grp
touch /home/aquota.usr
chmod 666 /home/aquota.grp
chmod 666 /home/aquota.usr
```

Ensuite, il faut indiquer aux file systèmes qu'ils sont soumis à la gestion des quotas. Ceci ce fait dans le fichier /etc/fstab

On a l'habitude de voir ce fichier comme suit :

```
/dev/sda2 /home ext3 defaults 1 1
```

il suffit d'indiquer en plus, derrière le champs « defaults », *userquota* pour la gestion des quotas par utilisateur, et *groupquota* pour la gestion des quotas par groupe.

On obtient alors la ligne suivante :

```
/dev/sda2 /home ext3 defaults,userquota,groupquota 1 1
```

Maintenant nous pouvons démarrer le daemon de gestion de quota. Au préalable, après avoir modifier le fichier /etc/fstab, il faut toujours le recharger, on fait cela grâce à la commande `mount -a`.

Les commandes utiles :

aquotaon [file système/option]

-a : permet de démarrer la gestion de quotas pour tout les files systèmes

aquotaoff [file système/option]

permet d'arreter la gestion des quotas

quotacheck : vérifie les quota pour un file système

-a : pour tout les files systèmes

-u : pour les files systèmes étant soumis a la gestion des quotas pour utilisateur

-f : pour les files systèmes étant soumis a la gestion des quotas pour groupe

repquotas : affiche la configuration des quotas

-a : sur tout les systèmes de fichier

-u : sur les quotas utilisateurs

-g : sur les quotas des groupes

edquotas : attribution des quotas pour les différent user/groupe/files système

-u : pour un ou plusieurs utilisateurs

-g : pour un ou plusieurs groupes

-t : permet de définir le temps de grâce

exemple :

```
edquota -u babar
Disk quotas for user babar (uid 500):
Filesystem    blocks      soft      hard    inodes    soft      hard
/dev/hdc1      0         9000    10000      0     90000    10000
```

Le fichier se compose de 6 colonnes :

- Filesystem : système de fichiers concerné par les quotas
- blocks : nombre de blocs occupés par l'utilisateur dans le système de fichiers. Ici aucun fichier n'a encore été créé.
- soft : limite soft en nombre de blocs. Ici elle est fixée à 9 000 blocs soit environ 9 Mo
- hard : limite hard en nombre de blocs (environ 10 Mo)
- inodes : nombre d'inodes occupés par l'utilisateur dans le système de fichiers
- soft : limite soft en nombre d'inodes
- hard : limite hard en nombre d'inodes

Dépassement de quotas ??

Ceci est un cas concret de dépassement de quotas repris sur www.lea-linux.org.

Une fois n'est pas coutume, on se place du côté utilisateur. Nous allons décrire les principaux cas de figure de dépassement de quotas et les messages envoyés à l'utilisateur.

Prenons l'exemple suivant : l'utilisateur Anne dispose de 9Mo en limite douce et 10 Mo en limite dure. Son délai de grâce est de 7 minutes. Ci-dessous le contenu du système de fichiers faisant l'objet de ces quotas :

```
anne@pingu$ ls -l /home/anne
total 1842
-rw-----  1 root    root      7168 fév 28 23:50 aquota.user
```

-rw-r--r--	1	anne	anne	1857516	mar	1	12:19	fic1
drwx-----	2	root	root	12288	nov	28	12:59	lost+found

Nous sommes largement en-dessous des quotas. Nous allons maintenant copier 4 fois le fichier fic1. Les 3 premières copies se passent bien et nous avons fic2, fic3 et fic4. Ci-dessous, la dernière copie

```
anne@pingu$ cp fic1 fic5
idel(22,10): warning, user block quota exceeded.
anne@pingu$ ls -l
total 9134
-rw----- 1 root root 7168 fév 28 23:50 aquota.user
-rw-r--r-- 1 anne anne 1857516 mar 1 12:19 fic1
-rw-r--r-- 1 anne anne 1857516 mar 1 13:18 fic2
-rw-r--r-- 1 anne anne 1857516 mar 1 13:18 fic3
-rw-r--r-- 1 anne anne 1857516 mar 1 13:18 fic4
-rw-r--r-- 1 anne anne 1857516 mar 1 13:18 fic5
drwx----- 2 root root 12288 nov 28 12:59 lost+found
```

La limite douce est dépassée. L'utilisateur reçoit un message mais l'écriture est réalisée car nous n'avons pas dépassé la limite dure.

Deux cas de figures peuvent alors se présenter si l'utilisateur ne contacte pas l'administrateur ou s'il ne libère pas de l'espace pour repasser en-dessous de la limite douce.

1er cas : l'utilisateur tente d'écrire dans le système de fichiers ce qui l'amène à dépasser la limite dure.

```
anne@pingu$ cp fic1 fic6
idel(22,10): write failed, user block limit reached.
cp: écriture de `fic6': Débordement du quota d'espace disque
L'opération échoue. Une partie du fichier seulement a été copiée.
l'utilisateur de pourra plus écrire dans le système de fichiers.
```

2ème cas : l'utilisateur laisse s'écouler le délai de grâce de 7 minutes fixé par l'administrateur. Il tente alors de copier le contenu du fichier /etc/passwd par exemple. Le total de l'espace occupé reste toutefois inférieur à la limite dure.

La sanction sera identique que dans le 1er cas. L'opération échoue.

```
anne@pingu$ cp /etc/passwd .
idel(22,10): write failed, user block quota exceeded too long.
cp: écriture de `./passwd': Débordement du quota d'espace disque
L'opération a échoué comme en témoigne le listage ci-dessous :
anne@pingu$ ls -l passwd
-rw-r--r-- 1 anne anne 0 mar 1 14:48 passwd
```

De même si vous essayez d'écrire dans le fichier passwd, vous obtiendrez le message suivant dans votre éditeur au moment de l'enregistrement :

```
"passwd" erreur d'écriture (système de fichiers plein ?) Appuyez sur ENTRÉE
ou tapez une commande pour continuer
```

Il vous est impossible d'écrire.

6 Installation des quotas

1) Installation du package quotas.rpm

Il faut d'abord effectuer la copie du .rpm se trouvant sur blacksad
/usr/local/linux3ème/quotas.rpm

Commande : `scp u3bin008@blacksad:/usr/local/linux3ème/quotas.rpm ./`

Une fois le package récupéré il faut l'installer .

Commande : `Rpm -Uvh quotas.rpm`

Mise en place des quotas, il faut mettre en place un mécanisme de quota sur /usr ainsi que sur /home .

On crée d'abord un fichier aquota.user et aquota.group.

Commande : `touch aquota.user`

`touch aquota.group`

Les fichiers doivent être créés avec les modes de permissions 0600.

Pour fixer les quotas sur un système de fichiers, il faut mettre à jour le fichier /etc/fstab. On va pour cela ajouter les options de montage pour le ou les systèmes de fichiers concernés

/dev/sda7	/home	reiserfs	defaults,usrquota,grpquota	1 2
/dev/sda6	/usr	reiserfs	defaults,usrquota,grpquota	1 2

Remonter le ou les systèmes de fichiers concernés pour prendre en compte l'utilisation de quotas pour ce système de fichiers.

Il faut initialiser la base des quotas en exécutant la commande suivante : `quotacheck -auvg`

-u : pour checker les quotas du user.

-v : pour une sortie des messages à l'écran

-g : pour checker les quotas du group.

-a : Check tout les filesystem non NFS dans /etc/mtab

Activer les quotas : `quotaon -a`

Définir des quotas pour un utilisateur :

`edquota -u prof`

Disk quotas for user prof (uid 500):

Filesystem	blocks	soft	hard	inodes	soft	hard
/dev/hdc1	0	9000	10000	0	90000	10000

Les limites soft et hard appliquées dépendent du type d'utilisateur présent sur le système, ceux donnés ici ne servent que de base d'exemple.

Informations Concernant les quotas

La commande `repquota` permet d'afficher un résumé de l'utilisation des quotas et délais de grâce.

Syntaxe : repquota [-vug] -a | filesystem

-v : mode verbeux, affiche des infos supplémentaires

-u : affiche des informations sur les quotas utilisateurs

-g : affiche des informations sur les quotas groupes

-a : affiche des informations sur tous les systèmes de fichiers disposant de quotas

filesystem : affiche des informations sur les quotas du système de fichiers spécifié

Il est aussi possible d'assigner une période de grâce ,délai fixé entre le moment où l'utilisateur atteint la limite soft et celui où on va lui interdire toute occupation supplémentaire dans le système de fichiers. On va donc fixer la durée de ce délai. elle sera la même quelque soit l'utilisateur et/ou le groupe.

Ceci est réalisable grace à la commande : edquota -t

06_WebHosting

Au sein d'un réseau local d'une entreprise, il peut être intéressant que celle-ci dispose d'un intranet pour fournir des services à ses employé. Il est inutile que cet intranet se trouve sur un serveur distant. Apache est un serveur web, il permet de faire tourner plusieurs serveur virtuel sur une machine ce qui lui permet de hoster plusieurs sites web sur une unique machine.

Si on snif un réseau à partir d'une ip, on ne sait pas voir quel site on a visité car il peut exister plusieurs sites sur un serveur. Dans ce cas il suffit d'aller voir le paquet de la couche ip, dans le champs appelé « la charge » les informations s'y trouvant. Dans la charge on retrouve le site qu'on a visite grâce aux mot GET:/index.htm et HOST :www.machin.be s'y trouvant.

Ether	0800	Ip-source	Ip-dest	Tcp	http	CHARGE
-------	------	-----------	---------	-----	------	--------

Une autre manière de faire mais plus lourde, est d'avoir une ip pour un site web héberger.

Configuration des postes clients et serveur

Dans un réseau local on a pas forcément un serveur DNS, pourtant il faut pouvoir traduire des adresse web en ip. Ceci ce fait à l'aide du fichier /etc/hosts en y ajoutant un alias sur la bonne ip, et en configurant le fichier /etc/nsswitch.conf

```
/etc/hosts
10.0.0.2  www www.ipl.be ipl.be
```

```
/etc/nsswitch.conf
passwd:      files nis
group:       files nis
hosts:       files nis dns
```

Configuration d'Apache

La configuration du serveur Apache se fait dans le fichier **/etc/httpd/httpd.conf**.

Il est important d'indiquer un serveur par défaut, car une personne mal intentionnée pourrait directement se retrouver dans les directories du serveur.

```
Port 80
ServerAdmin admin@www.ipl.be
ServerName www.ipl.be
DocumentRoot /var/www
<Directory /var/www>

</Directory>
```

On peut ajouter autant de serveur virtuel que l'on souhaite, il suffit de rajouter les lignes suivantes au fichier httpd.conf

```
<virtualhost ip :port>
ServerName
ServerAdmin
```

```
DocumentRott  
ErrorLog  
CustomLog  
</VirtualHost>
```

Rien n'empêche de mettre le site sur un autre port que le port 80. Cela peut être utile si on veut éviter que quelqu'un n'accède pas ou plus difficilement au site en question.

On peut aussi faire en sorte d'afficher le contenu d'un home directory d'un user directement sur le browser web. Il suffit de rajouter les balises suivantes.

```
<ifModule mod.userdir.c>  
useDir enable  
</ifModule>
```

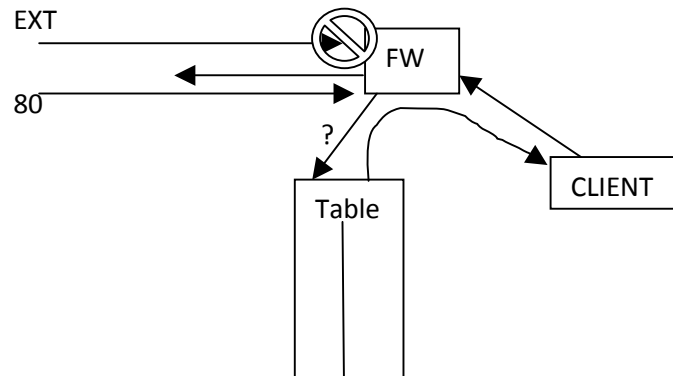
avec une tel configuration, si on tape www.ipl.be/~unUser, on verra afficher la home directory du user unUser.

Le serveur Apache tourne sans privilège en particulier. Ainsi, si il existe une faille dans le serveur, on ne pourra que attaquer Apache et rien d'autre, ce qui limite les dégats.

Parfois il peut être utile de voir quel service tourne sur quel port pour ça le fichier **/etc/service** est très utile

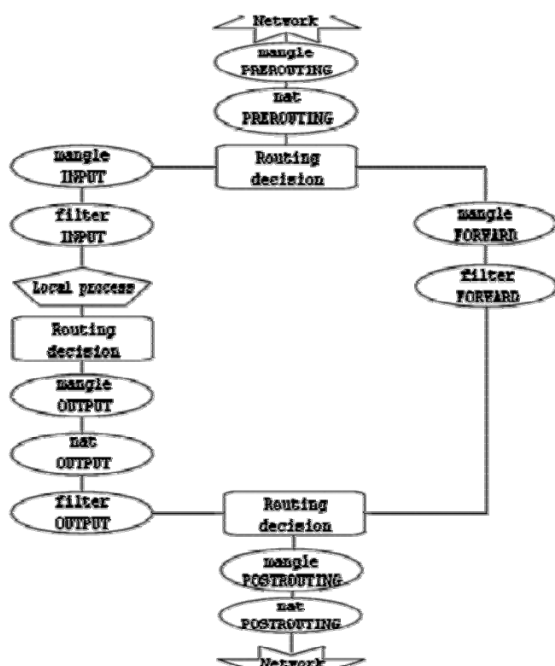
07_Firewall

Le firewall dans unix s'appelle iptables. C'est un pare-feu dit « state full » car il maintient une notion d'état. L'ancien pare-feu de unix s'appelait ipchain.



Lorsque un paquet réseau arrive sur le firewall, il devrait remonter jusqu'à la couche tcp/ip mais iptable fait en sorte que le paquet passe d'abord par lui et ses nombreuses tables. Iptable réécrit les paquets ip entrant. Les logs de iptables se trouvent dans **/proc/sys/net/ip-contrack**.

Dans la pile IP



Quelles que soient l'origine et la destination des paquets, ils vont entrer dans la pile de protocoles IP par le même point et en sortir par le même autre point.

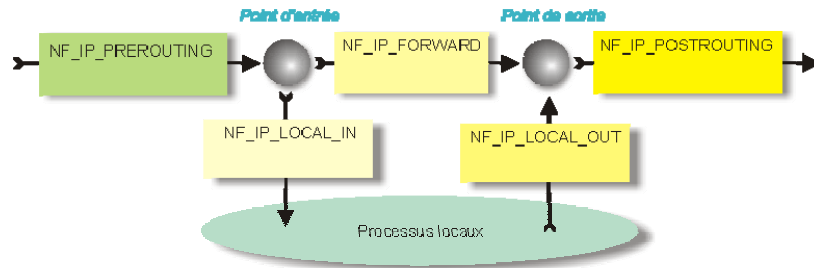
Iptable se présente comme une série de 5 "hooks" (points d'accrochage), sur lesquels des modules de traitement des paquets vont se greffer. Ces points sont:

- PREROUTING
- INPUT
- FORWARD
- OUTPUT
- POSTROUTING

La branche gauche représente le trajet des paquets qui entrent et qui sortent vers et depuis un processus local (SMB, FTP, HTTP etc.)

La branche de droite représente le trajet des paquets qui traversent notre passerelle dans sa fonction de routeur.

Une autre façon de voir les choses, un peu plus simpliste :



Les tables

NAT (Network Address Translation)

Cette table permet d'effectuer toutes les translations d'adresses nécessaires.

La chaîne PREROUTING.

Permet de faire de la translation d'adresse de destination. Cette méthode est intéressante si l'on veut faire croire au monde extérieur, par exemple, qu'il y a un serveur WEB sur le port 80 de la passerelle, alors que celui-ci est hébergé par un hôte du réseau privé, sur le port 8080.

La chaîne POSTROUTING.

Elle permet de faire de la translation d'adresse de la source, comme du masquage d'adresse, la méthode classique pour connecter un réseau privé comme client de l'Internet, avec une seule adresse IP "officielle".

La chaîne OUTPUT.

Celle-ci va permettre de modifier la destination de paquets générés localement (par la passerelle elle-même).

Filter

Cette table va contenir toutes les règles qui permettront de filtrer les paquets. Cette table contient trois chaînes:

La chaîne INPUT.

Cette chaîne décidera du sort des paquets entrant **localement** sur l'hôte.

La chaîne OUTPUT.

Ici, ce ne sont que les paquets émis par **l'hôte local** qui seront filtrés

La chaîne FORWARD.

Enfin, les paquets qui traversent l'hôte, suivant les routes implantées, seront filtrés ici.

Mangle

Cette table permet le marquage des paquets entrants (PREROUTING) et générés localement (OUTPUT).

Les chaînes

Ci-dessus, nous avons évoqué plusieurs fois le terme chaîne, expliquons un peu plus en détail. Les chaînes sont des ensembles de règles que nous allons écrire dans chaque table. Ces chaînes vont permettre d'identifier des paquets qui correspondent à certains critères.

Les cibles

Les cibles enfin sont des sortes d'aiguillage qui dirigeront les paquets satisfaisant aux critères . Les cibles préconstruites sont :

- **ACCEPT** : Les paquets qui satisfont aux critères sont acceptés, ils continuent leur chemin dans la pile,

- DROP : Les paquets qui satisfont aux critères sont rejetés, on les oublie, on n'envoie même pas de message ICMP . Un trou noir, quoi.
- LOG : C'est une cible particulière qui permet de tracer au moyen de syslog les paquets qui satisfont aux critères.

Configuration d'un iptable

Tout d'abord, il faut être sûr que les tables soient vides. Il suffit d'exécuter la commande :

```
iptables -F : cela va provoquer un flush de toutes les tables
iptables -P [INPUT,OUTPUT,FORWARD] : va flusher une des 3 chaines du filtre.
```

Ensuite, tout les paquets localhost venant de l'extérieur du réseau ainsi que les paquets d'un réseau privé venant de l'extérieur doivent être jetés.

```
iptables -A INPUT -i ppp0 -s 127.0.0.1 -j DROP
iptables -A FORWARD -i ppp0 -s 127.0.0.1 -j DROP
iptables -A INPUT -i ppp0 -d 127.0.0.1 -j DROP
iptables -A FORWARD -i ppp0 -d 127.0.0.1 -j DROP
```

```
iptables -A FORWARD -i ppp0 -s 192.168.0.0/16 -j DROP
iptables -A FORWARD -i ppp0 -s 172.16.0.0/12 -j DROP
iptables -A FORWARD -i ppp0 -s 10.0.0.0/8 -j DROP
iptables -A INPUT -i ppp0 -s 192.168.0.0/16 -j DROP
iptables -A INPUT -i ppp0 -s 172.16.0.0/12 -j DROP
iptables -A INPUT -i ppp0 -s 10.0.0.0/8 -j DROP
```

Tout ce qui vient du monde Windows doivent être jeter, entre autres les paquets arrivant sur les ports 135 et 445. On ne forwardera jamais des paquets dont les sources sont 137 et 139.

```
iptables -A FORWARD -i ppp0 -p tcp --dport 135 -j LOG
iptables -A FORWARD -i ppp0 -p tcp --dport 135 -j DROP
iptables -A FORWARD -i ppp0 -p tcp --dport 445 -j LOG
iptables -A FORWARD -i ppp0 -p tcp --dport 445 -j DROP
```

```
iptables -A FORWARD -o ppp0 -p tcp --sport 137:139 -j DROP
iptables -A FORWARD -o ppp0 -p udp --sport 137:139 -j DROP
iptables -A OUTPUT -o ppp0 -p tcp --sport 137:139 -j DROP
iptables -A OUTPUT -o ppp0 -p udp --sport 137:139 -j DROP
```

On accepte de sortir des paquets vers l'Internet que si on est sûr que cela vient de notre réseau.

```
iptables -A INPUT -i eth0 -j ACCEPT
iptables -A FORWARD -i eth0 -j ACCEPT
```

Le par feux acceptera des paquets sur la porte 22 (pour ssh). Et on autorisera celui-ci à établir des connexion.

```
iptables -A INPUT -i ppp0 -p tcp --dport 22 -j ACCEPT
```

On vérifie les règles entrée en les affichant

```
iptables -L
```

Pour finir on sauve la configuration avec la commande.

```
iptables -save
/etc/rc.d/init.d/iptables save
```

D'autre application utile

Squid : proxy tourne sur le port 3128

MAJ 2006-12-28

Dansguardian : content filter et url web tourne sur le port 8080

Un proxy à 2 rôle, le premier est de filtrer les urls, le second est de stocker les pages temporairement.

Source : <http://christian.caleca.free.fr/netfilter/>