LéaBook, Chapitre : **Réseau**

par tous les amis de Léa

Table des matières

Table des matières

Table des matières

Table des matières

Configurer les composants d'un réseau

29/5/2000 : un IP−Masquerade HOWTO [lien mort ?] est sorti ! Il met à jour l'ancien mini−HOWTO... en attendant la traduction anglaise...

Nous avons besoin de correcteurs ! Si vous voulez nous aider, prenez le fichier source (voir lien en bas de chaque page), corrigez−le et envoyez−le moi par mail !

Voici les documents qui concernent le réseau (local et internet) :

Configuration carte réseaux pour un réseau local

par Serge (quelques modifications par Fred et Jicé)

ou comment faire communiquer tous vos ordinateurs entre eux

Le protocole réseau de prédilection pour Unix et l'internet est le TCP/IP, mais pour de nombreux débutants et non informaticiens, configurer un réseau est plus qu'un casse tête. Alors on va voir comment configurer un tel réseau sous Linux (quelques PC reliés entre eux) , pour les connexions pour le net, voir les autres rubriques.

Le HARDWARE (carte réseau)

Pré−requis

Bon avant tout il faut configurer la partie hard du réseau. Dans le cas d'un réseau local, on a plusieurs machines, pour les connecter entre elles, il faut des cartes réseau et ce qu'on appelle un "média" ou "support". On a le choix entre plusieurs topologies que je ne vous explique que très rapidement :

- Câblage en bus coaxial ou "BNC" : le moins cher à mettre en place, de bonnes performances de communication jusqu'à quelques machines (moins de 10) et pour des applications non critiques, c'est a dire des partages de fichiers, un petit intranet local. Il suffit pour cela d'une carte réseau par machine avec connecteur BNC, un câble coaxial entre deux machines de 50 Ohms, avec un T BNC pour chaque machine et deux bouchons réseaux de 50 Ohms à chaque extrémité du support (c'est à dire pour la première et la dernière machine de la chaîne). Ça ne revient pas cher (cartes et câbles compris on arrive en gros à 200 fr par machine maximum). Il faut quand même respecter les normes, c'est à dire pas plus de 500m de câble, pas moins de 50 cm entre deux stations, bien mettre les bouchons à chaque extrémité et c'est bon.
- Câblage en étoile RJ45 : C'est un peu plus cher, mais ce type de câblage a de bien meilleures performances : on peut espérer aller jusqu'à 100 Mb/s avec du matériel fait pour (contre 10 Mb/s pour le BNC), on peut enlever une machine sans "rompre" le support (c'est à dire que les stations qui ne sont pas en relation avec la station qui vient d'être déconnectée continueront de fonctionner comme si de rien n'était). Bref c'est beaucoup mieux et c'est le standard pour des réseaux avec des applications gourmandes, un grand nombre de machines etc... Le câblage est différent, on utilise alors des paires torsadées en cuivre, chaque câble doit faire 100m maximum, on passe par un élément actif (Hub ou Switch) pour connecter les machines, les prises sont du type RJ45 (un peu comme les prises téléphoniques américaines). En fait chaque carte réseau est reliée par un câble à un élément actif, d'ou le nom de câblage en étoile (partant de l'élément actif). On distingue deux types d'élément actif :
- les HUBS qui en fait "émulent" une connexion en bus en envoyant sur chaque câble toutes les informations du réseau qu'ils reçoivent. Ceux−ci laissent à la machine destinataire des requêtes réseau le soin de les prendre en compte ou de les ignorer si elle n'est pas destinataire,
- les SWITCH qui sont beaucoup plus intelligents que les HUBS : ils envoient les informations du réseau UNIQUEMENT vers la machine destinataire, donc on gagne en performance (et en sécurité, car il devient plus difficile de "sniffer" le réseau), c'est pour cela que le câblage en RJ45 est plus performant que le BNC.

  Entre le 10 Mb ou le 100 Mb il suffit d'avoir le matériel qu'il faut, c'est a dire carte réseau, HUB ou SWITCH à 100 Mb. Seul les SWITCH permettent effectivement un "vrai" réseau en étoile. Certains HUBs ou SWITCH permettent de mélanger les deux types de vitesse, alors qu'en général, une carte 10 Mb/s oblige toutes les machines à communiquer en 10 Mb/s.*Remarque :* On trouve maintenant sur le marché des KITS carte réseau + HUB + câblage pour pas cher du tout (3 cartes, 1 HUB les câbles pour 400 fr), si vous devez faire un achat, prenez plutôt ça, attention quand même à ce que les cartes soient supportées par Linux (cf. : Hardware HOWTO).

Pour les cartes réseau, je vous conseille les cartes réseau "100% COMPATIBLE NE2000 PCI" ou "ISA NE2000" si vous ne pouvez pas prendre de PCI. Dans le cas de carte ISA je vous conseille les PnP (facile à configurer) alors que les non PnP passent souvent par un programme de configuration sous DOS, si votre machine est 100% linux sans partition DOS vous allez être embêté !
Remarque : si vous utilisez xxxBSD en plus de Linux, sachez que le driver de BSD pour les cartes NE2000 est pourri : si vous avez besoin de mettre plusieurs interfaces dans la même machine, n'utilisez pas de carte compatible NE2000, sinon chaque paquet va mettre dans les 1 seconde a passer !!! (si vous ne me croyez pas, demandez à BooK ce qu'il en pense...)

Pour les cartes plus exotiques, regardez le Hardware−HOWTO et/ou demandez dans des listes de diffusion ou dans les news si quelqu'un a déjà réussi à faire marcher cette carte sous Linux.

Paramétrage de la carte.

Bon, on va étudier les différents cas:

- Carte PCI : normalement vous n'avez rien à faire, juste pour test un "`modprobe module`" ou module est le nom du module pour votre carte (`modprobe ne2k−pci` pour les NE2000 PCI) et ça devrait passer sans le moindre problème ! (si ça ne passe pas, vérifiez que vous avez le support PCI dans votre kernel, voir rubrique compilation kernel, et que le module de votre carte est bien compilé, voir rubrique module et kernel).

- Carte ISA PnP : après avoir mis la prise en charge du PnP dans votre machine (rubrique PnP) il suffit de mettre dans `/etc/isapnp.conf` les ressources de votre carte réseau (attention à ne pas écraser les ressources d'une autre carte) en faisant un `pnpdump >/etc/isapnp.conf`. Décommentez alors dans ce fichier les ressources (voir la rubrique PnP !). Le reste de la configuration est identique à celle des cartes PCI : faites un test en chargeant le module ad−hoc (c'est à dire : `modprobe ne`) .
- Carte ISA non PnP : aïe ! (Serge exagère... ;) le plus dur! Soit vous avez de la chance et il suffit de configurer votre carte via des cavaliers (jumpers) et puis de tester en chargeant le module en indiquant les ressources en option du module (rubrique module). Soit vous avez pas de chance, il faut passer sous DOS (par exemple avec une disquette de boot FreeDOS), utiliser le programme de configuration fourni avec la carte, affecter les ressources de la carte (une seule fois heureusement − quand vous avez fait attention à ne pas entrer en conflit avec d'autres cartes...) et lors du chargement du module passez, là aussi, les options (genre : `append="macarte=iobase,irq,etc..."`.

Ça ne marche pas ! Bon pas de panique ! Vérifiez que les ressources de votre carte n'écrase pas celle d'une autre (typique si lors du chargement du module vous obtenez "device or ressource busy" , on vérifie ça dans `/proc/ioports`, `/proc/interrupts`. Vérifiez que l'adresse mémoire (ioport) et les interruptions (irq) ne sont pas déjà occupées par une autre carte. Vérifiez aussi à l'aide de la notice de la carte que vous n'essayez pas d'attribuer une ressource que cette carte ne peut pas prendre (normalement la plage d'irq et de mémoire valide est indiquée dans la notice). Vérifiez, suivant votre cas, que le PnP est bien validé dans le kernel, la prise en charge PCI, et que quelqu'un a déjà réussi à faire marcher cette &*#! de carte réseau.

Prise en compte de la carte par le kernel.

Bon le module se charge, ok, il reste juste à rajouter dans `/etc/conf.modules` les options pour le chargement automatique du module quand le kernel en a besoin, ainsi qu'un alias (lien) pour le kernel qui indique à celui−ci que ce module gère une carte réseau par :

```
alias ethX     nom_module
```

Avec: **X**=0 pour la première carte réseau, 1 pour la seconde etc. (si vous n'avez qu'une seule carte réseau − sur cette machine − alors X=0, soit `eth0`, est votre seule possibilité)
`nom_module` : le nom du module (driver) de la carte (`ne2k−pci` par exemple)

Pour savoir si des options sont nécessaires (cas des cartes NE 2000 ISA) voir la rubrique modules.

Configuration réseau de la carte

Bon la carte est installée, le module se charge sans erreur maintenant il va falloir lui affecter une adresse IP, etc... Pour ceux qui savent ce qu'est une adresse IP, un masque de sous réseau, une passerelle par défaut, etc. hop passez au paragraphe suivant, pour les autres : lisez la suite (autrement votre réseau ne marchera pas c'est sûr ! (La lecture d'un bon livre sur TCP/IP est certainement utile si vous comptez installer un réseau de plusieurs milliers de machines ;)

Le protocole TCP/IP

Bon je ne vais pas détailler à fond le protocole TCP/IP mais juste les bases nécessaires pour comprendre comment on configure un réseau TCP/IP.

Adresse, classe d'adresse et masque réseau

Bon vous devez sûrement vous demander comment deux machines entre elles arrivent à communiquer. Et bien, tout simplement avec une adresse. C'est comme pour le courrier−escargot (merci la francophonie), quand quelqu'un veut vous envoyer un courrier il envoie une lettre et le facteur vous trouve grâce à l'adresse que vous avez pris soin d'écrire sur celle−ci. Bon et bien pour comprendre ce qui suit vous gardez ça en tête, en remplaçant *trame réseau* par *courrier*, et *carte réseau* par *boîte à lettres*. Si notre analogie n'est pas trop farfelue (elle ne l'est pas ;) chaque carte doit avoir une adresse réseau (i.e. adresse IP) , mais comme pour les grandes villes où il y a beaucoup de maisons, on découpe souvent un réseau en plusieurs sous−réseaux (le meilleur exemple est Internet lui−même qui est constitué de divers − et nombreux − réseaux plus locaux), pour permettre de les reconnaître facilement, on leur donne un "bout d'adresse" en commun. Puis, comme on classe les maisons par rues et numéros, on classe les réseaux par adresse de réseau (le "bout d'adresse" en commun) et adresse de carte (adresse complète comprenant l'adresse de réseau ainsi qu'une partie spécifique à la carte). Comment cela s'exprime−t−il ? Par une adresse qui comporte une partie "réseau" et une partie "hôte" (la partie spécifique à la carte) tout cela sur 4 octets (si vous savez pas ce qu'est un octet, ce n'est pas grave, dites vous que c'est un nombre compris entre `0` et `255`). On représente alors l'adresse complète comme ceci :

`W.X.Y.Z` (où chaque lettre peut prendre une valeur entre `0` et `255` donc)

Bon d'accord, j'ai une adresse mais comment je reconnais le réseau et l'hôte? Et bien, on a coupé en classe de réseau toutes les adresses possibles suivant les valeurs de W. Je vous donne ce découpage, puis je l'explique:

Classe A: adresse comprise entre `1.0.0.0` à `126.255.255.255`
Classe B: adresse comprise entre `128.0.0.0` à `191.255.255.255`
Classe C: adresse comprise entre `192.0.0.0` à `223.255.255.255`

Il existe la classe D aussi mais qui n'est pas utilisable, elle sert en fait au protocole lui−même dans son fonctionnement pour "atteindre" plusieurs machines à la fois (on appelle ça le *multicast*), on ne va pas détailler ça ici `:−)`. Si vous avez bien suivi, vous avez pu constater que les adresses 0.x.x.x (utilisées pour l'adresse de route par défaut, c'est à dire le chemin à prendre pour aller sur un autre réseau), et 127.x.x.x (adresse interne de chaque machine pour les applications) ne sont pas utilisables non plus ...bref même si vous ne comprenez pas ce que je dis, rappelez vous simplement qu'elles sont tout simplement INUTILISABLES. Utilisez les adresses dans les plages ci−dessus (i.e.: correspondant aux classes A, B et C).

Chaque classe d'adresse a aussi son masque de sous réseau :
Classe A: `255.0.0.0`
Classe B: `255.255.0.0`
Classe C: `255.255.255.0`

Bon maintenant j'explique. Quand vous voulez faire un réseau local IP, vous choisissez d'abord une classe. Bon laquelle choisir ? Ça va dépendre du nombres de machines que vous comptez connecter à votre réseau.

Pour comprendre, on ne va s'occuper que des réseaux de classe C (i.e.: ceux ne pouvant contenir que peu de machines), mais tout ce qui va suivre s'applique très simplement aux autres classes. Le masque réseau de la classe C est : `255.255.255.0`. Cela veut dire que la partie `W.X.Y` de l'adresse `W.X.Y.Z` sert à adresser (contacter) le réseau (i.e. toutes les machines du sous−réseau auront le même `W.X.Y`) et le `Z` sert à adresser les machines. La partie du masque contenant les `0` nous donne la partie de l'adresse qui correspond à la machine. Donc cela nous donne :

Si le masque est `255.225.255.0` et que l'adresse d'une carte réseau est `W.X.Y.Z` alors `W.X.Y.0` est l'adresse du réseau. Ce réseau peut être constitué de toutes les cartes `W.X.Y.Z` ou `Z` varie entre `0` et `254` (pas `255` car c'est là encore une adresse spéciale).

Quand on veut désigner l'adresse du réseau , on remplace par des 0 les identifiants machines.

Pour la classe C, chaque réseau a donc 254 (de 1 à 254 pour Z) machines possibles. De même pour la classe B on a 65 000 (et quelques) machines possibles. Pour la classe A on dispose de plus de 16 millions de machines possibles.

Pour configurer un réseau local, il vous faut de choisir l'adresse réseau que vous voulez. Bon on prend un exemple, je choisis `192.1.3.0` comme adresse de réseau donc classe C. Je peux alors numéroter mes machines de `192.1.3.1` à `192.1.3.254`.

Bon vous allez me dire : On peux choisir n'importe quelle adresse ? Je ne vais jamais avoir besoin de classe A : je n'ai pas 16 millions de machines ? Etc... En fait ces classes ont été inventées pour l'internet, on a attribué alors les classes A aux très grandes organisations comme l'armée américaine (ça été fait pendant la guerre froide hein) pour leur permettre de relier toutes leurs machines. La B était pour les grand organismes (université, industrie, etc...) et les classes C pour les petits groupes. Comme vous vous en doutez, ces adresses sont attribuées par un organisme international qui régit tout ça pour que deux adresses ne soit pas dupliquées sur internet.
Donc en résumé si vous restez en local, sans aucune sortie vers l'extérieur (même pas un modem vers le net), vous utilisez l'adresse **que vous voulez** en respectant son masque et la numérotation réseau/machine.

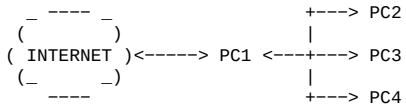
Comme on a vu que les adresses sont attribuées sur le net, que ce passe−t−il si vous utilisez une adresse déjà utilisée sur le net ? Et bien, dans un premier temps, toutes les machines de votre réseau qui vont vouloir aller sur les sites internet qui possèdent la même adresse de réseau IP que vous ne pourront pas y accéder, car pour elles c'est l'adresse de chez vous donc elles ne vont pas chercher l'adresse sur le Net. Dans un second temps, vous allez être en relation avec d'autres machines qui vont forcément confondre votre réseau avec celui qui est enregistré sur internet. Mais alors, quelle adresse je choisis afin de ne pas entrer en conflit avec d'autres machines sur Internet ? Et bien l'organisme international a réservé des adresses qui ne seront JAMAIS utilisées (tous les routeurs du monde les ignorent !) sur le net pour que vous puissiez les utiliser en local ! Elle est pas belle la vie ! Ces adresses sont :

classe A: `10.0.0.0`
classe B: `172.16.0.0` à `172.31.0.0`
classe C: `192.168.0.0` à `192.168.255.0`

Comme aucun site du net n'a cette adresse, si vous l'utilisez chez vous, vous ne risquez pas de ne pas pouvoir atteindre un site parce qu'il possède cette adresse vu qu'il n'en existe pas ! Donc pas de conflit entre votre réseau et le réseau des réseaux.

Donc, si vous avez un modem ou modem câble ou ADSL (voir rubrique ADSL) et si vous comptez surfer sur la toile un jour : UTILISEZ une des adresses réseau ci−dessus !!

Vous voulez que l'une de vos machines soit sur le net avec un beau site etc... : il vous faut alors acheter une adresse IP à votre fournisseur d'accès, soit pour un réseau complet et là vous n'avez pas le choix : on vous impose l'adresse, soit juste pour une machine, là votre machine connectée au NET a l'adresse que votre fournisseur vous impose et votre réseau local une des adresses réservées vue plus haut. En plus il va falloir faire du "masquerading" pour pouvoir utiliser l'accès au NET depuis toute les machines, voir pour cela la rubrique "masquerade" ou le HOWTO : vu d'internet, toutes les machines de votre réseau local ont l'adresse de la seule machine connectée au net (en fait, vu d'internet on ne voit que cette machine) :



En fait, vous pouvez aussi utilisez le système précédant pour créer des sous−réseaux à l'intérieur d'un sous−réseau de classe A, B ou C. Bon je ne vais pas expliquer ça ici non plus, reportez vous au site www.linuxenrezo.org pour plus de détail, de toute façon si vous devez faire de telles choses c'est que vous savez ce que vous faites.

Bon maintenant on va apprendre d'autres termes utile à une configuration réseau TCP/IP :

- *Passerelle par défaut* ou GATEWAY: c'est l'adresse de la machine qui est reliée aux autres réseaux (par exemple la machine qui possède le modem pour aller sur le net) (dans le dessin ci−dessus, `PC1` est la passerelle de votre réseau local)
- *Route par défaut* (default): c'est l'adresse `0.0.0.0`
- *Localhost*: adresse IP `127.0.0.1` (cela sert aux applications qui veulent accéder à la couche TCP/IP de la machine où elle tourne sans passer par le réseau, si vous ne comprenez pas c'est pas grave)

Avec ça, ça devrait aller :)

De la théorie à la pratique: Configuration statique (sans serveur DHCP).

Dans cette partie je traite de la configuration d'une carte réseau en statique, c'est à dire à la main. Il existe un autre moyen: par client DHCP, c'est à dire qu'un serveur va s'occuper d'attribuer adresse, masque et gateway (passerelle) automatiquement à votre station, c'est ce que fait votre provider à

chaque connexion pour la configuration IP de votre modem ou de votre carte pour des connexions modem câble ou ADSL. Pour l'instant voyons la manière "à la main". Dans cette partie on ne s'occupe que du cas où l'on n'a qu'une seule carte dans la station.

La carte est configurée au niveau HARD (module qui se charge bien), on a choisi/obtenu une adresse IP de réseau, on a décidé comment on adresse chaque machine : on peut réellement attribuer les adresses aux machines. Des utilitaires comme "`netcfg`", "`netconfig`" ou "`linuxconf`" permettent de configurer très facilement une carte réseau (voir www.linuxenrezo.org) mais bon je trouve plus ludique/instructif de vous l'apprendre à la main, comme ça on sait ce que l'on fait, de plus des distribution comme la Slackware ne possèdent pas ces outils par défaut.

Supposons, que l'on souhaite attribuer l'adresse X.X.X.X à la carte eth0 sur le réseau dont le masque de sous réseau du LAN est Y.Y.Y.Y (c'est pareil pour les autres cartes...). Rien de plus simple, on tape (en root) :

```
# ifconfig eth0 X.X.X.X netmask Y.Y.Y.Y
```

et votre réseau est configuré ! (Il faut bien le reconnaître : c'est pas la mort). Bon pour ne pas avoir à configurer ça à chaque reboot, ajoutez cette commande dans un script de démarrage comme par exemple `/etc/rc.d/rc.local`.
Cependant, la plupart des distributions ont des utilitaires pour configurer ça et lancer un script automatiquement, lancez par exemple `netconfig` ou `netcfg` ou `linuxconf`, vous avez bien au moins l'un de ces utilitaires (sinon : télécharger `linuxconf` sur rufus par ex.).
Pour la Slackware, placez ces commandes dans le `/etc/rc.d/inet1`. Éditez−le, cherchez les lignes suivantes et renseignez−les avec vos paramètres :

```
IPADDR="192.168.1.1"      (remplacez par l'adresse de votre machine)
NETMASK="255.255.255.0"   (mettez votre masque de réseau, ici classe C)
NETWORK="192.168.1.0"     (mettez votre adresse de réseau)
BRODCAST="192.168.1.255" (votre adresse de broadcast, c'est a dire votre adresse réseau avec les identifiants machines à
255)
```

Vérifiez que plus bas dans ce fichier il y a bien '`DHCP="no"`'. Le reste du fichier est OK. Toujours pour la Slackware, pour charger votre module automatiquement, éditez le fichier `/etc/rc.d/rc.modules`, decommentez la ligne (c'est à dire enlever le # en début de ligne) qui correspond au module de votre carte réseau, par exemple pour une pci ne2000:

```
/sbin/modprobe ne2k−pci
```

Configuration par client DHCP

Bon maintenant on va voir comment configurer sa carte automatiquement par client DHCP, c'est à dire que c'est un serveur DHCP (soit sur votre réseau local d'entreprise, soit votre provider) qui va vous attribuer un adresse IP, un masque, une gateway (et même plus si affinités ;). Bon pour cela il faut quand même avoir configuré le module de la carte, le "driver" comme vu précédemment.

Maintenant il reste à dire au démarrage de Linux que cette carte va être configurée par un serveur DHCP, pour cela il faut un client DHCP, je traite ici de `dhcpd` (en fait il fait serveur et client à la fois). Il existe aussi `pump` et `dhclient` mais bon pour des goûts personnels que beaucoup partagent, je traite ici de `dhcpd`. Récupérez le donc (allez sur freshmeat par exemple ou rpmfind). Pour les kernels 2.2.X prenez une version au moins égale à la 1.3.x.

Il suffit alors de taper une ligne de commande du type :

```
dhcpd −d ethX
```

Puis vérifier que votre carte à bien une adresse ip par:

```
ifconfig
```

Vous devrez voir une ligne avec `lo` et une autre avec `ethX`. Vérifiez que `ethX` n'a pas pour adresse `0.0.0.0`, cela voudrait dire que cela n'a pas marché.

Bon après on automatise ça en incluant cette commande dans un script de démarrage de la machine (`/etc/rc.d/localou rc.local` etc...). Pour les RedHat et Mandrake, `linuxconf` permet ça, pour la Slackware, éditer `/etc/rc.d/rc.inet1d` et mettez l'option yes pour DHCP

```
DHCP="yes"
```

Si vous avez des machines non linux sur votre réseau

Il suffit de configurer les autres machines avec les mêmes règles de réseaux TCP/IP. Dans Win9x, ça ce trouve dans le panneau de configuration, réseau, ajouter le protocole microsoft TCP/IP et réglez les valeurs comme il se doit.

- Pour partager des ressources avec ce type de machines diaboliques ;) lisez la doc SAMBA sur ce site,
- Pour accéder au net depuis un de ces machines, en partageant l'accès (modem RTC, modem câble, etc.) avec la machine Linux, lisez le IP−Masquerade−HOWTO (en anglais) ou le mini−IP−Masquerade−HOWTO (en français).

Se monter son propre gateway (passerelle) sous Linux

par Hervé J. Lombaert

10 Janvier 2001

permettre à votre LinBox de partager une connexion internet.

Introduction

Le but de ce document est de vous aider à monter un ordinateur afin qu'il puisse partager une connexion internet avec plusieurs de vos ordinateurs. Le contenu repose sur mon expérience personnelle, c'est pourquoi seules les configurations avec le matériel qui m'était disponible seront traitées. Un langage aussi clair que possible sera utilisé sans pour autant se limiter aux commandes que vous aurez à taper. Il ne s'agit pas de recopier des commandes à l'écran, mais de comprendre ce que vous faites. C'est donc ainsi que sera abordé dans un premier temps l'installation de linux, et en un second temps sa configuration. La configuration passe par celle du matériel puis des serveurs dhcp, DNS et samba. Finalement, l'utilisation d'`ipchains` et de `ipmasqadm` seront expliquées pour rediriger l'internet.

Installation de Linux

Choix d'une distribution

Plusieurs distributions sont actuellement disponibles à ce jour. Il faut donc choisir la distribution la plus adéquate à l'utilisation d'un gateway (ou **passerelle** en français). Le gateway idéal aura comme rôle principal la redirection de l'internet vers les ordinateurs locaux. Il ne s'agit donc pas d'un ordinateur de bureau.

La distribution à choisir devra être alors légère, autant en espace mémoire qu'en ressources matérielles. Une installation graphique ainsi que d'autres gadgets superflus ne feraient qu'alourdir inutilement l'exploitation du gateway. Le choix s'est donc dirigé vers la distribution Debian Potato. Elle a été choisie pour sa légèreté, sa robustesse et surtout pour la rigueur demandée par sa configuration.

Préparation des disquettes d'amorçage

Avant de commencer l'installation telle quelle, il faudra d'abord s'assurer qu'il nous est possible d'amorcer l'installation. C'est−à−dire, soit un BIOS permettant de booter sur un cdrom, soit dans le cas contraire, des disquettes d'amorçage. Étant donnée qu'un 486, et encore moins un 386, ne possèdent de bios supportant des cdroms bootables, il faudra dans ce cas créer les disquettes d'amorçages. La distribution Debian 2.2 demande la création de deux disquettes, l'une nommée *rescue*, et l'autre *root*.

Sous Windows, ou même sous DOS, allez dans le répertoire *D:\boot* de votre premier cdrom Debian et lancez le programme *rawrite2.exe*. Créez dans un premier temps une disquette à partir de l'image *rescue.bin*, et dans un second temps une disquette à l'aide de l'image *root.bin* :

```
D:\install> rawrite2.exe
enter image file : rescue.bin
enter target drive : a:
```

Vos deux disquettes en main, il est maintenant possible de commencer l'installation.

Insérez la disquette *rescue* et lancez l'ordinateur. Après avoir démarré le processus d'amorçage, il vous sera demandé d'insérer la disquette *root* contenant les programmes d'installation.

Configuration du système

Toutes les sous−sections de l'installation vont être présentées ci−dessous avec un bref descriptif de ce qui a été effectué :

Configure the keyboard

Le clavier par défaut est choisi, qwerty/us.
Note de Jicé : selon la disposition de votre clavier, en France choisissez un clavier *azerty* par exemple.

Partition a Hard Disk

Cette sous−partie fait appel au programme `cfdisk`, assez simple d'utilisation. Si vous êtes dérouté au départ, il y a le menu **help** à votre disposition. N'oubliez pas de créer une partition de swap.

Initialize and Activate a Swap Partition

Appuyez la touche entrée jusqu'à la prochaine sous−partie.

Initialize a Linux Partition

Personnellement, parmi les questions posées, je ne retiens pas la compatibilité avec les noyaux 2.0. Si vous utilisez plus d'une partition, le logiciel d'installation vous demandera d'abord d'initialiser votre partition monté sur "`/`"

Install Operating System Kernel and Modules

L'installation se fait à partir d'un CDROM, les choix par défaut proposés sont corrects.

Configure Device Driver Modules

Dans un premier temps, il n'est pas nécessaire de modifier ce qui est proposé par défaut. En effet, après l'installation, tous les modules seront présent dans le répertoire `/lib/modules/2.2.17`. Vous pouvez quitter la configuration des modules pour le moment.

Configure the hostname

Entrez ici le nom que portera votre ordinateur sur votre réseau. Dans mon cas, j'ai choisi `headquarters`.

Install the Base System

Les options par défaut sont bonnes, appuyez la touche entrée jusqu'à la prochaine sous−section.

Configure the Base System

Il vous est ici demandé de choisir votre fuseau horaire, utilisez les flèches de direction pour choisir. Attention à ne pas spécifier que votre horloge utilise l'heure GMT.

Make Linux Bootable Directly from Hard Disk

L'installation de LILO est fortement conseillée sur le Master Boot Record, ou MBR.

Make a Boot Floppy

C'est à votre choix, la disquette qui sera créée, et bootera votre système en cas de pépin, mais entre vous et moi, vous avez déjà créé tout à l'heure la disquette *rescue* qui est elle même bootable...

Reboot the System

Retirez la disquette insérée, ou bien le cdrom si vous avez un ordinateur bootant sur le cdrom.

À ce stade l'ordinateur est bootable, l'installation des programmes pourra débuter.

md5 password

Comme expliqué à l'écran, une réponse positive risque de provoquer des problèmes avec un serveur NIS. Dans l'éventualité d'un cas où vous installerez un serveur NIS sur votre ordinateur, il est déconseillé d'utiliser des mots de passe md5. Dans ce cas, seuls des mots de passe de moins de 8 caractères pourront être utilisés.

Shadow password

Pour des raisons de sécurité, il vaut mieux crypter les mots de passe. Par contre, une réponse négative permettra de voir les mots de passe des utilisateurs, ça présente l'avantage de retrouver un mot de passe perdu par exemple.

Root Password

Entrez ici votre mot de passe pour l'administrateur système, appelé root.

Normal Account

Il est conseillé de créer à ce stade un compte utilisateur, celui avec lequel vous vous loguerez pour une utilisation normale de l'ordinateur.

Pcmcia package

À moins d'utiliser un ordinateur portable, ce qui n'est pas pratique pour partager l'ordinateur d'ailleurs, répondez oui à la question demande de supprimer les package pcmcia.

PPP

Si vous utilisez un modem, installez le support PPP. Ce document ne traite pas de partager une connexion téléphonique à internet.
Note de Jicé : Pour partager une connexion par modem, la démarche est quasiment la même.

Scan another CD

C'est à cet endroit où vous ferez scanner vos CD pour que debian sache ce qu'il peut installer.

Add another apt source

Il est possible ici de rajouter une source de programme d'installation sur l'internet. C'est à votre choix, personnellement pour ne pas être attardé par des délais de download, je ne choisis pas cette option.

Simple Advanced

L'option *simple* vous présentera plusieurs profils à installer, ne vous laissant pas le choix de choisir les programmes à installer au début. Tandis que le mode *expert* vous permettra de choisir les programmes que vous voulez installer. N'oubliez pas que vous pouvez rajouter vos logiciels par la suite à l'aide du logiciel `dselect`. Le choix de l'option *simple* vous épargnera du temps pour le moment, vous pourrez toujours rajouter ou supprimer vos logiciels par la suite.

Installations des logiciels

Packages to install en simple mode

À ce stade, il faudra bien installer les logiciels, choisissez les catégories de logiciels voulus.

Packages to install en advanced mode

Ce mode vous permettra de choisir les logiciels à installer à l'aide du programme `dselect`.

Dans ces deux modes, il faudra installer un serveur DHCP, DNS et Samba. Il faudra installer les logiciels de forwarding : **ipchains** et **ipwadm**.

Par la suite, je vous laisse découvrir les différentes manipulations que vous devrez effectuer afin d'avoir tout ces serveurs installés. C'est à vous de voir quels programmes vont être installés ; il est impossible de choisir à votre place et de fournir dans ce document toutes les manipulations pas à pas que vous aurez à effectuer. Gardez en tête que les choix par défaut font l'affaire la plupart du temps, et que chaque choix est correctement expliqué.

Configurations principales

Préparatifs

Avant de commencer les configurations des différents serveurs, la configuration de la console et de son aspect devra être effectuée pour faciliter la suite du travail. Tout commence par de la couleur : lors des listages des dossiers, il est préférable de différencier les fichiers des répertoires, des liens symboliques ou des fichiers exécutables. Il faudra donc décommenter certaines lignes du fichier */root/.bashrc*. Voici à quoi devra ressembler le fichier */root/.bashrc* après modifications :

```
# /.bashrc: executed by bash(1) for non-login shells.

export PS1='\h:\w\$ '
umask 022

# You may uncomment the following lines
# if you want `ls' to be colorized:
export LS_OPTIONS='-color=auto'
eval `dircolors`
alias ls='ls $LS_OPTIONS'
alias ll='ls $LS_OPTIONS -l'
alias l='ls $LS_OPTIONS'

# Some more alias to avoid making mistakes:
alias rm='rm -i'
alias cp='cp -i'
alias mv='mv -i'
```

À vous ensuite de rajouter les alias que vous souhaitez. L'alias *clean='rm * -f'* pourra par exemple être utile pour supprimer les fichiers de sauvegarde *.

Le matériel

L'ordinateur en tant que tel n'a pas besoin d'être du dernier cri, personnellement, j'utilise un Pentium 166, mais un 486, ou même un 386 pourrait très bien faire l'affaire. Et pour ceux qui douteraient des capacités d'un 386, l'ordinateur ne fait que rediriger des paquets d'information vers vos ordinateurs locaux, ce n'est tout de même pas la résolution de systèmes d'équations différentielles à quinze inconnues qui lui est demandé !

Mis à part l'ordinateur, vous vous doutez bien qu'il faut deux cartes réseau (ou une carte réseau et un modem dans le cas d'un partage de connexion internet par modem). Les explications suivantes se baseront sur mon matériel, vous aurez donc à vérifier les drivers ou modules[1] respectifs à utiliser dans votre cas.

1ère carte réseau

La première carte, une 3Com 3C905B sera utilisée pour recevoir la connexion internet. Elle est branchée soit à un réseau d'entreprise, d'université, ou à un modem câble. Elle est activée par l'appel de son module `3c59x.o`, à l'aide de la commande :

```
bash# insmod 3c59x
```

Aucun paramètre supplémentaire n'est nécessaire : l'IRQ et le port IO seront trouvés automatiquement. Sous Debian, pour que ce module soit chargé automatiquement au démarrage, il faudra rajouter une ligne contenant "3c59x" au fichier `/etc/modules` :

```
bash# echo "3c59x" >> /etc/modules
```

```
bash# cat /etc/modules
```

ou bien utilisez un éditeur de texte quelconque pour rajouter votre ligne vous même. N'oubliez pas de mettre à jour votre fichier */etc/modules.conf* à l'aide de la commande :

```
bash# update-modules
```

Note de Jicé : pour un partage de connexion par modem, le modem remplace cette première carte. Il faut donc s'assurer de la présence du support PPP dans le noyau, ainsi que du paquetage `pppd`.

2ème carte réseau

Pour ce qui concerne la deuxième carte, cela a été un peu plus délicat pour ma part. En effet, étant donné qu'avec 5 dollars, on ne peut trouver facilement que des cartes ISA, il m'a fallu fournir le canal IRQ et le port IO manuellement à la ligne de commande. Après avoir cherché un moment sur l'internet les manuels de ma carte réseau ISA, une digital DE201, il a finalement été possible de comprendre comment utiliser les cavaliers ("jumpers" pour les semi-francophones).

En vérifiant le fichier */proc/interrupts* et */proc/ioports*, il a été possible de trouver un canal IRQ et un port IO disponibles. Dans mon cas, l'IRQ 5 et le port IO 0x300 ont été choisis. Le module se charge donc comme suit :

```
bash# insmod depca "irq=5" "io=0x300"
```

Comme vous devinez peut-être, le driver de la carte réseau DE201 est le module `depca.o`. Ce module supporte d'ailleurs plusieurs vieux modèles de cartes réseaux de marque Digital. J'ai remarqué également que si les paramètres n'étaient pas fournis, l'auto-détection ne fournissait pas le bon canal IRQ, ni le bon port IO, et qu'en plus de cela, l'ordinateur gelait ! Faites donc attention à fournir les bons paramètres.

Pour que ce module soit correctement chargé au démarrage avec les bons paramètres, il a fallu rajouter la ligne contenant "`depca`" au fichier */etc/modules* et créer un fichier */etc/modutils/depca* contenant les paramètres à prendre en compte :

```
bash# echo "depca" >> /etc/modules
bash# cat /etc/modules
bash# echo "options depca irq=5 io=0x300" > /etc/modutils/depca
```

et mettez à jour le fichier `/etc/modules.conf` à l'aide de :

```
bash# update-modules
```

Voilà en ce qui concerne pour la configuration des cartes réseau. Vous pourrez vérifier quel module utiliser pour vos cartes réseaux respectives dans un ethernet howto.

Branchement des cartes réseau

Après avoir configuré les cartes réseau, il faudra les brancher respectivement à l'internet, et au réseau local. Il ne s'agit pas de leurs branchements physiques qui consiste à bien brancher les câbles, mais à leurs attribuer les paramètres IP.

La première carte, la 3Com sera branchée à l'internet, elle recevra automatiquement tous les paramètres par un serveur DHCP situé chez le fournisseur internet. Il faudra donc rajouter la ligne "`iface eth0 inet dhcp`" au fichier `/etc/network/interfaces`.
Note de Jicé : dans le cas du partage d'une connexion par modem, vous devez configurer de manière classique cette connexion à votre fournisseur d'accès.

La seconde carte, la Digital, sera branchée au réseau local. Il faudra lui attribuer les paramètres IP manuellement. L'adresse IP réservée de classe A, 192.168.1.1 sera celle choisie pour notre ordinateur. Cela se configure également dans le fichier */etc/network/interfaces*.

Voilà donc de quoi aura l'air le fichier */etc/network/interfaces* :

```
# /etc/network/interfaces - configuration file for ifup(8), ifdown(8)

# The loopback interface
iface lo inet loopback

# eth0 - Reseau internet
iface eth0 inet dhcp

# eth1 - local network
iface eth1 inet static
address 192.168.1.1
netmask 255.255.255.0
broadcast 192.168.1.255
```

Pour brancher une carte réseau, il faut se servir de la commande `ifup`, pour la débrancher, la commande `ifdown` sera d'usure. La commande `ifconfig` permettra de vérifier les paramètres des cartes réseau branchées :

```
bash# ifup eth0
bash# ifup eth1
```

```
bash# ifconfig
```

Serveur DHCP

Le serveur DHCP permettra à tout ordinateur connecté au réseau local de pouvoir se procurer automatiquement les paramètres IP nécessaires, tel que son adresse IP, ou les adresses de passerelles, ou encore les masques de réseau. Tout ceci est configuré dans le fichier `/etc/dhcpd.conf`. Pour connaître toutes les possibilités offertes par le serveur DHCP, il y a les pages man dhcpd.conf. Voilà un exemple de fichier `dhcpd.conf` :

```
# dhcpd.conf
# Herve J. Lombaert, 01/05/2001

default-lease-time 600;
max-lease-time 7200;

option subnet-mask  255.255.255.0;
option broadcast-address 192.168.1.255;
option domain-name  "ch19231.rez";
option domain-name-servers 192.168.1.1;
option routers   192.168.1.1;

subnet 192.168.1.0 netmask 255.255.255.0 {
 range 192.168.1.11 192.168.1.20;
}

subnet 132.204.210.0 netmask 255.255.254.0 {
}
```

Notre réseau nommé `ch19231.rez`, il s'agit dans mon cas de la chambre 19231 d'une résidence universitaire, possède l'adresse `192.168.1.0`. Notre ordinateur principal nommé *headquarters* possède l'adresse IP `192.168.1.1`. Il est possible de monter 10 ordinateurs sur notre réseau, les adresses IP disponibles vont de 192.168.1.11 à 192.168.1.20. Le masque étant de 255.255.255.0 pour ne tenir compte que du dernier triplet de l'adresse IP.

Après la configuration du serveur, il faudra l'activer. Il est possible de l'activer manuellement à l'aide de l'appel au serveur `dhcpd`. L'option -d permettra d'afficher à l'écran ce qu'il se passe au niveau de ce serveur :

```
bash# dhcpd -d eth1
```

Lancez un ordinateur sur votre réseau local et vous vous remarquerez que le serveur DHCP affecte une adresse IP à l'ordinateur du réseau.

Lors du démarrage du système, le serveur est lancé par */etc/init.d/dhcp*. Lorsqu'on démarre le serveur, un message explique qu'il faut modifier le fichier de lancement */etc/init.d/dhcp* à la ligne 11, afin de mettre l'option run_dhcpd à 1. Par la même occasion, il faudra spécifier sur quelle interface le serveur DHCP doit écouter. Dans les modes start et restart, l'interface `eth1` sera donc spécifié :

```
ligne 10: # Set run_dhcpd to 1 to start dhcpd at boot or 0 to disable it.
ligne 11: run_dhcpd=1 ligne 35 : -exec /usr/sbin/dhcpd eth1
ligne 44 : -exec /usr/sbin/dhcpd eth1
```

À présent, lorsque le script */etc/init.d/dhcp* est redémarré à l'aide du paramètre `start`, le serveur DHCP se met en route.

Serveur DNS

Voici peut-être la parti la plus ardue, il faut manipuler différents fichiers de configuration, passer au travers de la compréhension de différentes notions de bases sur les serveurs de nom de domaine (DNS).

Il est possible de passer au travers de cette étape en entrant tous les ordinateurs présents sur le réseau local dans le fichiers */etc/hosts*, mais il est souhaitable de mettre en route un serveur DNS pour que chacun des ordinateurs sur le réseau puisse communiquer entres eux à l'aide de noms concrets. Le serveur DNS à mettre en route doit donc en plus de servir des noms de domaines de l'internet, les noms des ordinateurs du réseau.

Note de Jicé : si vous disposez de seulement 2 ou 3 ordinateurs sur votre réseau local, il est plus facile de simplement entrer les coordonnées de ces ordinateurs dans le fichier `/etc/hosts` de chaque ordinateur (`c:\windows\hosts` sous windows).

Le fichier de configuration principal est le fichier */etc/bind/named.conf* ; voici de quoi il doit avoir l'air :

```
// This is the primary configuration file for the BIND DNS server named.
//
// Please read /usr/share/doc/bind/README.Debian for information on the
// structure of BIND configuration files in Debian for BIND versions 8.2.1
// and later, *BEFORE* you customize this configuration file.
//
options {
 directory "/var/cache/bind";

// If there is a firewall between you and nameservers you want
// to talk to, you might need to uncomment the query-source
```

```
// directive below. Previous versions of BIND always asked
// questions using port 53, but BIND 8.1 and later use an unprivileged
// port by default.

// query-source address * port 53;

// If your ISP provided one or more IP addresses for stable
// nameservers, you probably want to use them as forwarders.
// Uncomment the following block, and insert the addresses replacing
// the all-0's placeholder.

//forwarders {
// 0.0.0.0;
//};

};

// reduce log verbosity on issues outside our control ;
logging {
; category lame-servers { null; };
; category cname { null; };
;};
// prime the server with knowledge of the root servers zone "." {
 type hint;
 file "/etc/bind/db.root";
};

// be authoritative for the localhost forward and reverse zones, and for
// broadcast zones as per RFC 1912

zone "localhost" {
 type master;
 file "/etc/bind/db.local";
};

zone "127.in-addr.arpa" {
 type master;
 file "/etc/bind/db.127";
};

// Local Network
zone "ch19231.rez" {
 type master;
 file "/etc/bind/db.ch19231.rez";
};

zone "1.168.192.in-addr.arpa" {
 type master;
 file "/etc/bind/db.192.168.1";
};

zone "0.in-addr.arpa" {
 type master;
 file "/etc/bind/db.0";
};

zone "255.in-addr.arpa" {
 type master;
 file "/etc/bind/db.255";
};

// add entries for other zones below here
```

Il faut distinguer les fichiers de configuration qui serviront à traiter les résolutions de noms de domaine en adresse IP, et également la résolution d'adresses IP en nom de domaine. Par rapport à la configuration d'origine, il a fallu prendre en compte notre réseau, il y a eu donc rajout de la configuration du nom de domaine local `ch19231.rez`. Ces configurations étant référés au fichier `/etc/bind/db.ch19231.rez` et `/etc/bind/db.192.168.1`, il faudra alors créer ces fichiers.

Il faudra commencer par modifier le fichier */etc/bind/db.127* :

```
;
; BIND reverse data file for local loopback interface
;
@ IN SOA headquarters.ch19231.rez. root.ch19231.rez. (
```

```
 1  ; Serial
 604800  ; Refresh
 86400  ; Retry
 2419200  ; Expire
 604800 ) ; Negative Cache TTL
;
NS headquarters.ch19231.rez
1.0.0PTR localhost.
```

Et également modifier le fichier */etc/bind/db.local* :

```
;
; BIND data file for local loopback interface
; $TTL 604800
@ IN SOA localhost. root.localhost. (
 1 ; Serial
 604800 ; Refresh
 86400 ; Retry
 2419200 ; Expire
 604800 ) ; Negative Cache TTL
;
;@ IN NS localhost.
;@ IN A 127.0.0.1
;
NS localhost
localhost A 127.0.0.1
headquarters A 192.168.1.1
```

À partir de ce moment, il est possible de créer les fichiers */etc/bind/db.ch19231.rez* et */etc/bind/db.192.168.1*. Voici le premier fichier :

```
; db.ch19231.rez
; Herve J. Lombaert, 01/05/2001
; configuration file for the local network
; This file puts names on various IPs
;
@ IN SOA headquarters.ch19231.rez. root.ch19231.rez. (
 1
 8H
 2H
 1W
 1D)
NS headquarters
;
; headquarters computer (the gateway)
;
localhost A 127.0.0.1
headquarters A 192.168.1.1
headquarters HINFO "Pentium 166MHz" "Linux 2.2.17"
;
; hosts on the network using dhcp
;
comp-one A 192.168.1.11
comp-two A 192.168.1.12
comp-three A 192.168.1.13
comp-four A 192.168.1.14
comp-five A 192.168.1.15
comp-six A 192.168.1.16
comp-seven A 192.168.1.17
comp-nine A 192.168.1.18
comp-height A 192.168.1.19
comp-ten A 192.168.1.20
;
; end of file
```

Et voici le fichier */etc/bind/db.192.168.1* :

```
; db.192.168.1
; Herve J. Lombaert, 11/28/2000
; configuration file for the local network (reverse name)
;
@ IN SOA headquarters.ch19231.rez. root.ch19231.rez. (
 1
 8H
 2H
```

```
 1W
 1D)
NS headquarters.ch19231.rez
;
; headquarters computer
;
1  PTR headquarters.ch19231.rez.
;
; hosts on the local network (using dhcp)
;
11  PTR comp-one.ch19231.rez.
12  PTR comp-two.ch19231.rez.
13  PTR comp-three.ch19231.rez.
14  PTR comp-four.ch19231.rez.
15  PTR comp-five.ch19231.rez.
16  PTR comp-six.ch19231.rez.
17  PTR comp-seven.ch19231.rez.
18  PTR comp-height.ch19231.rez.
19  PTR comp-nine.ch19231.rez.
20  PTR comp-ten.ch19231.rez.
;
; end of file
```

Le fichier */etc/bind/db.root* contient les serveurs DNS principaux, il ne faudra pas toucher à ce fichier.

Une attention particulière est portée sur le point (.) après les noms de machines dans les lignes SOA, son oubli est grave, et il peut parfois être long avant de trouver d'où peut bien provenir l'erreur.

Il faudra ensuite indiquer quel serveur DNS utiliser. On crée pour cela le fichier */etc/resolv.conf* :

```
search ch19231.rez nameserver 127.0.0.1
```

Il est également possible de modifier 127.0.0.1 pour 192.168.1.1, il s'agit en effet de la même machine.

Il restera ensuite à lancer le serveur DNS :

```
bash# /etc/init.d/bind restart
```

Pour vérifier si le serveur fonctionne comme il faut, on peut utiliser le programme `nslookup`. En entrant différents noms de domaine, le programme nous retourne l'adresse IP du nom de domaine entré :

```
bash# nslookup comp-six
Server : localhost
Address : 127.0.0.1
Name : comp-six.ch19231.rez
Address : 192.168.1.16
```

N'oubliez pas non plus de tester la conversion d'adresse IP vers des noms de machine, à la place d'entrer comp-five, entrez par exemple 192.168.1.15, vous devriez avoir une réponse adéquate. Pour de plus amples informations sur `nslookup`, vous pouvez consulter ses pages man (`man nslookup`), vous y découvrirez par exemple qu'en entrant "`set query=ANY`", vous aurez tous les détails possibles fournis par le serveur de noms.

Serveur Samba

Le serveur Samba permettra à des ordinateurs sous windows dans votre réseau local d'utiliser de l'espace disque ou l'imprimante de votre ordinateur gateway. Il sera ainsi possible de mettre à disposition de tout le monde une imprimante.

La configuration du serveur samba est rien de plus simple, le fichier de configuration */etc/samba/smb.conf* par défaut contient tous les commentaires nécessaires à la compréhension du serveur. Voici un exemple de fichier */etc/samba/smb.conf* :

```
;
; /etc/smb.conf
;
; Sample configuration file for the Samba suite
; for Debian GNU/Linux
;
; Please see the manual page for smb.conf
; for detailed description of ; every parameter.
;
[global]
; Interface configuration
interfaces = eth1
bind interfaces only = yes
; interfaces = 192.168.1.0/24 127.0.0.1/32
```

```
; Authorized computers to gain access to samba server
hosts allow = 192.168.1. 127.

; time out in minutes
deadtime = 2

; printing = bsd
; printcap name = /etc/printcap
; load printers = yes
guest account = nobody
invalid users = root

; "security = user" is always a good idea.
; This will require a Unix account
; in this server for every user accessing the server.
; share –> everyone can access samba server
security = share

; Change this for the workgroup your Samba server will part of
; workgroup = WORKGROUP
workgroup = KLAN−SPECIAL

server string = %h server (Samba %v)

; If you want Samba to log though syslog only then set the following
; parameter to 'yes'. Please note that logging through syslog in
; Samba is still experimental.
syslog only = no

; We want Samba to log a minimum amount of information to syslog. Everything
; should go to /var/log/{smb,nmb} instead. If you want to log through
; syslog you should set the following parameter to something higher.
syslog = 0;

; This socket options really speed up Samba under Linux, according to my
; own tests.
socket options = IPTOS_LOWDELAY TCP_NODELAY SO_SNDBUF=4096 SO_RCVBUF=4096

; Passwords are encrypted by default. This way the latest Windows 95 and NT
; clients can connect to the Samba server with no problems.
encrypt passwords = yes

; It's always a good idea to use a WINS server. If you want this server
; to be the WINS server for your network change the following parameter
; to "yes". Otherwise leave it as "no" and specify your WINS server
; below (note: only one Samba server can be the WINS server).
; Read BROWSING.txt for more details.
wins support = no

; If this server is not the WINS server then specify who is it and uncomment
; next line.
; wins server = 172.16.0.10

; Please read BROWSING.txt and set the next four parameters according
; to your network setup. There is no valid default so they are commented
; out.
; os level = 0
; domain master = no
; local master = no
; preferred master = no

; What naming service and in what order should we use to resolve host names
; to IP addresses
name resolve order = lmhosts host wins bcast

; This will prevent nmbd to search for NetBIOS names through DNS.
dns proxy = no

; Name mangling options
preserve case = yes
short preserve case = yes

; This boolean parameter controlls whether Samba attempts to sync. the Unix
; password with the SMB password when the encrypted SMB password in the
```

```
; /etc/samba/smbpasswd file is changed.
unix password sync = false

; For Unix password sync. to work on a Debian GNU/Linux system, the following
; parameters must be set (thanks to Augustin Luton
; <aluton@hybrigenics.fr> for sending the correct chat script for
; the passwd program in Debian Potato).
passwd program = /usr/bin/passwd %u
passwd chat = *Enter\snew\sUNIX\spassword:* %n\n \ *Retype\snew\sUNIX\spassword:* %n\n .

; The following parameter is useful only if you have the linpopup package
; installed. The samba maintainer and the linpopup maintainer are
; working to ease installation and configuration of linpopup and samba.
; message command = /bin/sh -c '/usr/bin/linpopup "%f" "%m" %s; rm %s' &

; The default maximum log file size is 5 MBytes. That's too big so this
; next parameter sets it to 1 MByte. Currently, Samba rotates log
; files (/var/log/{smb,nmb} in Debian) when these files reach 1000 KBytes.
; A better solution would be to have Samba rotate the log file upon
; reception of a signal, but for now on, we have to live with this.
max log size = 1000

;[homes]
; comment = Home Directories
; path = /home
; browseable = yes

; By default, the home directories are exported read only. Change next
; parameter to "no" if you want to be able to write to them.
; read only = no

; File creation mask is set to 0700 for security reasons. If you want to
; create files with group=rw permissions, set next parameter to 0775.
; create mask = 0700

; Directory creation mask is set to 0700 for security reasons. If you want to
; create dirs. with group=rw permissions, set next parameter to 0775.
; directory mask = 0700

;[printers]
; comment = All Printers
; browseable = no
; path = /tmp
; printable = yes
; public = no
; writable = no
; create mode = 0700

;
-------------------------------
; SHARED PATH
;
-------------------------------

; A sample share for sharing your CD-ROM with others.
[cdrom]
comment = Samba server's CD-ROM
writable = no
locking = no
path = /cdrom
public = yes

; The next two parameters show how to auto-mount a CD-ROM when the
; cdrom share is accesed. For this to work /etc/fstab must contain
; an entry like this:
;
; /dev/scd0 /cdrom iso9660 defaults,noauto,ro,user 0 0
;
; The CD-ROM gets unmounted automatically after the connexion to the
;
; If you don't want to use auto-mounting/unmounting make sure the CD
; is mounted on /cdrom ;
preexec = /bin/mount /cdrom
postexec = /bin/umount /cdrom
```

```
; /tmp on debian for temporary transfers
[tmp]
comment = /tmp on debian (erased at each reboot)
writable = yes
path = /tmp
public = yes

; configuration files
[etc]
comment = configuration files
writable = no
path = /etc
public = yes

; whole system
[all]
comment = whole system
writable = no
path = /
public = yes
```

Utilisation d'ipchains et de ipmasqadm

ipchains

Le rôle principal d'`ipchains` dans notre cas est de distribuer l'internet sur le réseau local. Il peut également faire office d'un puissant firewall. La configuration d'ipchains peut se faire manuellement, mais en gardant l'esprit de monter une machine gateway, il faudra trouver un moyen de charger des règles de fonctionnement pour ipchains dès le lancement de l'ordinateur. Le programme `ipmasq` est fourni dans la distribution Debian 2.2, mais il est bien trop complexe pour débuter correctement, s'il est installé, désinstallez−le pour le moment. Un petit script de lancement tapé à la main fera très bien l'affaire.

Avant toute chose, il faudra s'assurer que la redirection de paquet d'information est possible, il faudra donc que le fichier `/proc/sys/net/ipv4/ip_forward` contiennent le chiffre 1 et non 0 qui le désactive. Pour que ce fichier soit toujours à 1, il faudra également modifier le fichier `/etc/network/options` pour que la modification soit prise en compte à chaque démarrage de l'ordinateur. Voici de quoi aura l'air le fichier `/etc/network/options` :

```
ip_forward=yes
spoofprotect=yes
syncookies=no
```

Pour rendre possible la redirection, nous avons expliqué qu'il fallait modifier le fichier */proc/sys/net/ipv4/ip_forward* :

```
bash# echo "1" > /proc/sys/net/ipv4/ip_forward
```

La seule et unique règle ipchains nécessaire à la redirection d'internet consiste à forwarder les paquets d'information provenant du réseau vers l'extérieur. Ceci ce fait à l'aide de la commande :

```
bash# ipchains −A forward −j MASQ −s 192.168.1.0/24 −d 0.0.0.0/0
```

La règle s'inscrit dans la chaîne forward, elle s'applique aux paquets d'information provenant de votre réseau local, 192.168.1.xxx sur 24 bits, allant n'importe où, 0.0.0.0 sur 0 bit. Son état est MASQ pour masquer le paquet, c'est−à−dire que l'entête du paquet IP traité sera modifié pour contenir à la place de l'adresse IP locale, l'adresse IP de votre gateway.

À ce stade, vous pourrez vérifier que vous êtes bien capable d'accéder à l'extérieur depuis un ordinateur sur votre réseau local. Pinger par exemple un ordinateur quelconque :

```
bash# ping info.polymtl.ca
PING info.polymtl.ca (132.207.12.85):
56 data bytes 64 bytes from 132.207.12.85:
icmp_seq=0 ttl=250 time=4.7 ms 64 bytes from 132.207.12.85:
icmp_seq=1 ttl=250 time=2.7 ms
```

Ceci montre bien qu'il est possible d'accéder à l'extérieur.

Les modules ipchains

Informations sur la redirection de données

Vous remarquerez que si vous vous contentez seulement de cette règle, beaucoup de logiciels fonctionneront : Netscape, ICQ pour les messages, plus au moins les sites FTP. Mais vous remarquerez également que tout ne marche pas comme il faut. Les transferts de fichier via ICQ ne fonctionnent pas, le listing des répertoires d'un site FTP ne fonctionne pas également.
Pour comprendre cela, il faudra comprendre ce que fait ipchains. Ipchains ne fait simplement que remplacer l'adresse IP locale située dans l'entête des paquets par l'adresse IP de votre gateway. Imaginez maintenant qu'un programme place l'adresse IP dans le corps du paquet, c'est−à−dire dans les informations qu'il veut envoyer. Ipchains n'étant pas supposé modifier le contenu utile des paquets d'information, il laissera l'adresse IP locale dans le

corps du paquet. Le programme qui recueillera le paquet à l'autre bout aura du mal à communiquer, car il ne connaîtra pas l'adresse de retour car une adresse locale réservé comme 192.168.15.x n'est pas valide sur l'internet.
Un autre exemple d'un cas typique où la redirection de l'internet ne fonctionnera pas est un programme qui dans la manière dont il fonctionne se comporte comme un serveur pour recueillir des informations. Un ordinateur de l'extérieur voulant communiquer avec un logiciel à l'intérieur de votre réseau local ne pourra jamais l'atteindre. En effet, l'ordinateur externe n'est pas supposé savoir que vous possédez un réseau local. À ses yeux, il communique uniquement avec votre gateway. Personnellement, je soupçonne le transfert de fichier ICQ de fonctionner de cette manière.

Pour remédier à ces problèmes, il existe des modules qui apporteront les modifications nécessaires au bon fonctionnement de certains logiciels tels que ICQ, ou des clients FTP. Ces modules se chargent comme nous l'avions expliqué pour le chargement des modules des cartes réseau :

```
bash# insmod ip_masq_ftp
bash# insmod ip_masq_irc
```

Pour le module ICQ, il n'est pas inclus dans la distribution Debian 2.2, de plus, ce module est experimental. Il faudra donc recompiler le noyau pour qu'il puisse accepter des modules expérimentaux. Veuillez pour cela vous referez à des kernel−howtos ou à l'article sur le noyau, ce n'est pas si sorcier que ça peut en avoir l'air.

Pour que toutes ces règles soient prises en compte dès le démarrage, il faudra créer un script de démarrage dans le répertoire */etc/init.d*. Créons par exemple le fichier */etc/init.d/ip_masquerade* :

```
#! /bin/sh
# ip_masquerade : setup ip_masquerade
# Herve J. Lombaert, 01/05/2001

# Rules
start_rules() {
 ipchains −P forward DENY
 ipchains −A forward −j MASQ −s 192.168.1.0/24 −d 0.0.0.0/0
 return 0
}

# Install
modules install_modules() {
 insmod ip_masq_ftp
 insmod ip_masq_icq
 insmod ip_masq_irc
 insmod ip_masq_cuseeme
 insmod ip_masq_quake
 insmod ip_masq_raudio
 insmod ip_masq_user
 insmod ip_masq_vdolive
 return 0
}

# Flush
rules flush_rules() {
 ipchains −F input
 ipchains −F output
 ipchains −F forward
 return 0
}

# Remove
modules remove_modules() {
 rmmod ip_masq_ftp
 rmmod ip_masq_icq
 rmmod ip_masq_irc
 rmmod ip_masq_cuseeme
 rmmod ip_masq_quake
 rmmod ip_masq_raudio
 rmmod ip_masq_user
 rmmod ip_masq_vdolive
 return 0
}

case "$1" in
start)
 echo −ne "\033[1;32m
 Starting ip_masquerade\033[0m\n"
 start_rules
 install_modules
;;
```

```
stop)
 echo −ne "\033[1;31m
 Stoping ip_masquerade\033[0m\n"
 flush_rules
 remove_modules
;;

restart)
 echo −ne "\033[1;32m
 Restarting ip_masquerade\033[0m\n"
 flush_rules
 remove_modules
 start_rules
 install_modules
;;

*)
 echo "Usage : /etc/init.d/ip_masquerade {start|stop}"
 exit 1

esac

exit 0

# End of file
```

Il faudra également penser à affecter les bons droits à ce script :

```
bash# chmod uga+x /etc/init.d/ip_masquerade
```

Le script de lancement étant maintenant créé, il va falloir créer les bons liens symboliques dans les répertoires */etc/rcN.d/* correspondant à chaque runlevel où sera exécuté le script. La distribution Debian déconseille de rajouter ces liens symboliques manuellement, mais suggère plutôt d'utiliser la commande :

```
bash# update−rc.d ip_masquerade defaults
```

Cela mettra le script crée dans les initialisations des runlevels 0 1 2 3 4 5 et 6.

Note de Jicé : pour d'autres distributions, vous pouvez utiliser des programmes en mode graphique comme `ksysv` pour faire ce travail.

Voilà ! À ce stade, vous avez un gateway opérationnel, vous pouvez redémarrer votre ordinateur et le laisser allumé aussi longtemps que vous voulez.

ipmasqadm

`ipmasqadm` permettra de rediriger des informations provenant de l'extérieur vers une machine à l'intérieur du réseau. Ceci aura l'avantage de pouvoir utiliser un serveur, http par exemple, à l'intérieur même de votre réseau local.

Pour pouvoir l'utiliser, il vous faudra connaître les numéros de ports et le protocole utilisé par le serveur que vous voulez rendre accessible à l'extérieur de votre réseau local. L'exemple d'un serveur de page web sera pris. Il fonctionne sur le port 80 et utilise le protocole TCP. Voici donc la ligne de commande que vous aurez à taper :

```
bash# ipmasqadm portfw −a −P tcp −L 132.204.210.10 80 −R 192.168.1.11 80
```

Ceci aura pour but de retransmettre tout ce qui arrive sur le port 80 de votre gateway, sur la machine 192.168.1.11, également sur le port 80.

Vous pourriez rajouter vos règles dans un fichier placé dans `/etc/init.d/`, ou également le placer dans le fichier créé, */etc/init.d/ip_masquerade*

Utilisation d'iptables (noyaux 2.4)

Ci−dessous les étapes à suivre pour partager sa connexion au moyen de iptables. Pour plus d'infos sur iptables, conulter iptables par l'exemple.

Pour partager sa connexion avec iptables, rien de plus simple. Il suffit de suivre les 2 étapes ci−dessous :

- **Côté passerelle :**
  - ♦ activer le forwarding dans le noyau :
    `echo 1 > /proc/sys/net/ipv4/ip_forward`
  - ♦ cacher les machines forwardées par le firewall :
    iptables −A POSTROUTING −t nat −o ppp0 −j MASQUERADE
- **Côté client :**
  - ♦ ne pas oublier de renseigner `/etc/resolv.conf` avec les DNS fournis par votre provider
  - ♦ ajouter votre passerelle en route par défaut :
    `route add default gw ip_de_votre_passerelle`
  - ♦ si votre client est sur Windows :
    - ◊ clic droit sur Favoris réseau et choisir propriétés

◊ clic droit sur Connexion au réseau local et choisir propriétés de TCP/IP
◊ renseigner la passerelle et les DNS

.

Conclusion

Ce document a tenté de décrire aussi simplement que possible les différentes étapes à suivre afin de bâtir votre machine passerelle, sans pour autant vous écarter de différentes explications. Le but principal était de vous aider à configurer un petit ordinateur pour qu'il puisse s'occuper correctement d'un petit réseau local. Le système d'exploitation traité dans ce document a été Linux, distribution Debian 2.2, son installation a donc été expliquée avant d'entamer la configuration des serveurs DNS, DHCP, Samba, ainsi que l'utilisation d'`ipchains`.

Notes

... modules[1]

Pour qu'une carte réseau soit utilisable, il faut d'abord charger son driver. Sous Linux, les drivers se présentent sous la forme de modules, des fichiers `*.o` ou `*.o.gz`. Il s'agit de code compilé qui peut être chargé et déchargé dans le noyau à votre guise. Il est également possible d'incorporer ces modules directement dans le noyau lors de la compilation du noyau (voir les kernel howtos). Un module se charge à l'aide de la commande `insmod` ou `modprobe`, et se décharge à l'aide de la commande `rmmod`. Il est possible de lister les modules chargés ainsi que leurs états d'utilisation à l'aide de la commande `lsmod`. Pour voir ce qu'il se passe au niveau du noyau, utilisez la commande `dmesg` qui permet de voir les messages reportés par le noyau, les modules, etc.

Exploration de la configuration réseau

par Anne

Ou lorsque votre pingouin se met à communiquer avec le monde entier

Avant propos

Ce document se veut être un document synthétique qui vous permettra de répondre aux questions suivantes :

- Quelle est ma configuration réseau local et distant ?
- Quels sont les services réseaux configurés sur ma machine ?
- Quels sont les principaux outils de diagnostic réseau sur ma machine ?

Attention : la liste des commandes fournies **n'est pas exhaustive** et plutôt orientée sur des distributions Redhat / Mandrake. Quelques précisions sont toutefois apportées concernant Debian et Slackware pour les plus grosses différences (merci à Prae et ses connaissances Debian)

Ma configuration réseau

Le hostname

Le nom de machine, ou *hostname* en bon français, est extrêmement important et a des conséquences non seulement sur la configuration réseau mais aussi sur le fonctionnement (ou dysfonctionnement) du serveur X et donc de l'interface graphique.

- **afficher le hostname** : la commande `hostname`
- **modifier le hostname** : il suffit de modifier les fichiers suivants :
  `/etc/sysconfig/network` (sur Redhat et Mandrake), `/etc/HOSTNAME` (sur slackware), `/etc/hostname` (sur Debian), et `/etc/hosts`.
- **utilisation du hostname avec X Window** : le fonctionnement du serveur X se fonde sur la variable d'environnement **`DISPLAY`**. **exemple :**
  `anne@pingu~~# echo $DISPLAY`
  `:0`
  A gauche des "`:`", on trouve le `hostname`, ici sous~entendu : localhost ou 127.0.0.1, soit la machine locale. A droite, on trouve le numéro de serveur X et parfois le numéro d'écran.
  Attention : toute modification du `hostname` en cours de fonctionnement sous interface graphique risque fort de vous faire perdre la main.

L'adressage IP

Pour connaître l'adressage IP de la machine, quelle que soit la nature du réseau, une commande à connaitre : `ifconfig`

la commande vous retourne quelque chose comme ça :

```
eth0  Lien encap:Ethernet  HWaddr 00:10:5A:DA:D3:47
      UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
      RX packets:2006397 errors:0 dropped:0 overruns:0 frame:0
      TX packets:1863082 errors:0 dropped:0 overruns:0 carrier:0
      collisions:3679 lg file transmission:100
      RX bytes:1283974990 (1224.4 Mb)  TX bytes:590947572 (563.5 Mb)
      Interruption:10 Adresse de base:0xe800

eth1  Lien encap:Ethernet  HWaddr 52:54:05:F5:BB:9B
      inet adr:192.168.0.3  Bcast:192.168.0.255  Masque:255.255.255.0
      UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
      RX packets:1841782 errors:0 dropped:0 overruns:0 frame:48
      TX packets:1984028 errors:0 dropped:0 overruns:0 carrier:0
      collisions:6607 lg file transmission:100
      RX bytes:579039163 (552.2 Mb)  TX bytes:1261892485 (1203.4 Mb)
      Interruption:9 Adresse de base:0xe400

lo    Lien encap:Boucle locale
      inet adr:127.0.0.1  Masque:255.0.0.0
      UP LOOPBACK RUNNING  MTU:16436  Metric:1
      RX packets:442 errors:0 dropped:0 overruns:0 frame:0
      TX packets:442 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 lg file transmission:0
      RX bytes:278720 (272.1 Kb)  TX bytes:278720 (272.1 Kb)

ppp0  Lien encap:Protocole Point~à~Point
      inet adr:213.41.132.215  P~t~P:62.4.16.248  Masque:255.255.255.255
      UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1492  Metric:1
      RX packets:2001693 errors:2006395 dropped:0 overruns:0 frame:0
      TX packets:1858378 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 lg file transmission:3
      RX bytes:1238624343 (1181.2 Mb)  TX bytes:549766984 (524.2 Mb)
```

Les informations fournies par la commande :

	nom de l'interface réseau

eth0, ppp0, lo...	
HWaddr	adresse MAC ou matérielle
inet adr	adresse IP liée à l'interface réseau
Bcast	adresse de broadcast
Masque	masque de réseau
UP	état de l'interface réseau − un 1er élément de diagnostic d'une panne réseau (non fonctionnement : DOWN)
MTU	taille maximum des trames physiques
RX / TX packets	nombre de paquets arrivés à destination, perdus ou non reçus à cause de débordements (arrivent de manière trop rapide pour pouvoir être traités par le noyau) − autre élément de diagnostic notamment lorsque les transmissions deviennent très lentes voire inexistantes.
collisions	se produit lorsque 2 machines émettent en même temps sur le réseau. Il n'y a pas lieu de s'inquiéter tant que le rapport (nombre de paquets total / nombre de collisions) reste inférieur à 30%.

Pour de plus amples informations concernant le fonctionnement de l'adressage IP, lire l'article sur TCP/IP.

Dans quels fichiers trouver la configuration IP ?

- *configuration de la (ou les) carte(s) réseau* : les principaux fichiers de configuration se situent dans `/etc/sysconfig/network-scripts` pour Redhat et Mandrake, ou dans `/etc/rc.d` pour slackware ou `/etc/networks`. Les fichiers s'appellent respectivement `ifcfg-ethx` (ou x est le numéro d'instance de la carte réseau), `inet1` et `interfaces`.
- *configuration du réseau et du routage* : un fichier à connaitre, `/etc/sysconfig/network` pour Mandrake et Redhat, `rc.inet1` pour slackware et `interfaces` pour Debian.
- *une astuce pour récupérer uniquement l'IP* : ci-dessous un exemple de script pour recupérer votre adresse IP (merci à FRED :))

```
root@pingu# cat ifip
#!/bin/bash
if [ -z "$1" ]
then
       IF=`cat`
else
       IF=$1
fi
ifconfig $IF|grep inet|tr -s " "|cut -f3 -d" "|cut -f2 -d:
```

L'avantage de ce script : il fonctionne soit avec un argument, soit en récupérant la sortie standard.
Exemple :

```
root@pingu# ifip eth1
192.168.0.3
root@pingu# echo eth1 | ifip
192.168.0.3
```

Les services réseaux configurés sur ma machine

Service géré par le super démon `inetd` ou `xinetd`

(selon la version de la distribution, l'un ou l'autre est utilisé, `xinetd` étant le plus récent). `inetd` (ou `xinetd`) agit comme un standardiste. Dès qu'un client fait appel à un service autorisé, il passe la ligne au dit service.

- **`inetd`** : le fichier de configuration à connaître est `/etc/inetd.conf`. Quand vous ne pouvez pas accéder à votre machine en `telnet` ou que votre serveur FTP est inaccessible, vérifiez ce fichier. Il est constitué d'une ligne par service du type :

  `<service_name> <endpoint-type> <protocol> <wait-status> <uid> <server_program> <server-arguments>`
  avec :
  − `service_name` : nom du service
  − `endpoint-type` : type de socket (`stream` pour TCP, `dgram` pour UDP)
  − `protocol` : protocole de transport utilisé
  − `wait-status` : soit l'option est `wait` et le service est dit mono-thread (une seule connexion par service peut être gérée), soit l'option est `nowait` et le service est dit multi-thread (chaque requête vers le service génère un nouveau serveur)
  − `uid` : utilisateur sous lequel est lancé le service
  − `server_program` : nom du programme à lancer pour activer le service
  − `server-arguments` : arguments éventuels pour le lancement du service

  Exemple :

  ```
  telnet stream tcp nowait root /usr/sbin/tcpd in.telnetd
  ```

  En cas de dysfonctionnement d'un service, **vérifier la syntaxe de la ligne**. De plus, pour désactiver l'accès à un service il suffit simplement de commenter la ligne le concernant. Pour réactiver un service, il suffit donc de supprimer le # démarrant la ligne.

- **`xinetd`** : C'est une version améliorée de `inetd`. Il permet une configuration plus fine de l'accès aux services (interdiction d'utilisateurs, d'adresses,...). Il n'y a plus un fichier unique mais un fichier `/etc/xinetd.conf` qui renvoie à un répertoire `/etc/xinetd.d`. Celui-ci

contient un fichier par service configuré.

La cause la plus fréquente de non fonctionnement d'un service c'est la désactivation de celui−ci (désactivation effectuée de base à l'installation du service pour des raisons de sécurité). Il suffit alors d'ouvrir le fichier et de vérifier la valeur de la variable `disable` qui, par défaut, est `yes`.

Exemple :

```
root@pingu# vi /etc/xinetd.d/telnet
service telnet
{
        flags           = REUSE
        socket_type     = stream
        wait            = no
        user            = root
        server          = /usr/sbin/in.telnetd
        log_on_failure  += USERID
        disable         = yes
}
```

Dans ce cas de figure, **le service `telnet` est désactivé**. Pour l'activer, 2 étapes :

1. *Modifier le fichier* `/etc/xinetd.d/telnet` en remplaçant `yes` par `no` dans la variable `disable`.
2. *Relancer xinetd* pour la prise en compte de ce nouveau paramètre :

```
root@pingu# ps −ef | grep xinetd
root   973   1  0 Oct09 ?   00:00:00 xinetd −stayalive −reuse −pidfil
root@pingu# kill −HUP 973
root@pingu#
```

Service tournant en *standalone*

Standalone : votre service tourne sans être lancé et contrôlé par le super démon inetd (ou xinetd).

- **Vérifier l'état d'un service** : tous les scripts de lancement se situent dans `/etc/rc.d/init.d`. Selon le runlevel dans lequel on se situe, le service est arrêté ou démarré. Pour cela, consulter les répertoires `/etc/rc.d/rcN.d` où `N` est le numéro de runlevel. Ces répertoires sont constitués de liens symboliques. Si le lien commence par un `S`, le service est démarré. S'il commence par un `K`, il est arrêté. Le `S` ou `K` est ensuite suivi par un nombre à deux chiffres. Ce nombnre permet de déterminer l'ordre de lancement des scripts (ordre croissant).

  La plupart des scripts est utilisable avec 3 arguments : `start` pour le démarrer, `stop` pour l'arrêter et `status` pour connaitre son état. La plupart des scripts acceptent également l'argument `restart` pour redémarrer le service (`stop` puis `start` ;)

  exemple :
  − *mon serveur Apache ne fonctionne pas.*
  ```
  root@pingu# /etc/rc.d/init.d/httpd status
  httpd est arrêté
  ```

  − *on démarre le serveur Apache*
  ```
  root@pingu# /etc/rc.d/init.d/httpd start
  Démarrage de httpd : [ OK ]
  ```

  − *on vérifie l'état du serveur*
  ```
  root@pingu# /etc/rc.d/init.d/httpd status
  httpd (pid 14113 14112 14111 14110 14109 14106) en cours d'exécution
  ```

- **Configurer le démarrage d'un service** (cas général, applicable quelle que soit la distribution) : il suffit de créer le script de gestion du service dans `/etc/rc.d/init.d` puis les liens symboliques de démarrage et d'arrêt dans les répertoires d'état de marche.
  Exemple : je veux que mon service `bidule` démarre aux niveaux 3, 4 et 5. On va procéder en 2 étapes :
  − création d'un script de gestion des arguments start / stop : /etc/rc.d/init.d/bidule
  − création des liens symboliques suivants :

  ```
  root@pingu# ln −s /etc/rc.d/init.d/bidule /etc/rc.d/rc0.d/K05bidule
  root@pingu# ln −s /etc/rc.d/init.d/bidule /etc/rc.d/rc1.d/K05bidule
  root@pingu# ln −s /etc/rc.d/init.d/bidule /etc/rc.d/rc2.d/K05bidule
  root@pingu# ln −s /etc/rc.d/init.d/bidule /etc/rc.d/rc3.d/S95bidule
  root@pingu# ln −s /etc/rc.d/init.d/bidule /etc/rc.d/rc4.d/S95bidule
  root@pingu# ln −s /etc/rc.d/init.d/bidule /etc/rc.d/rc5.d/S95bidule
  root@pingu# ln −s /etc/rc.d/init.d/bidule /etc/rc.d/rc6.d/K05bidule
  ```

- **Configurer le démarrage d'un service** : la commande `chkconfig`. Cette commande fort pratique est spécifique aux distributions Redhat et Mandrake. L'ensemble des services est intégré à une base de données gérée par `chkconfig`. Grâce à cette commande, vous pouvez ajouter / supprimer / modifier les niveaux auxquels sont démarrés ou arrêtés un service.

  Ajout d'un service à la base :
  ```
  root@pingu# chkconfig −−add service
  ```

  Suppression d'un service de la base :
  ```
  root@pingu# chkconfig −−del service
  ```

Configuration des niveaux auxquels le service doit démarrer/s'arrêter :
```
root@pingu# chkconfig --level <niveaux> service on|off
```

Pour tout complément d'information consulter l'article sur la gestion des services.

Les principaux outils de diagnostic réseau

arp

Permet de visualiser et modifier la table arp stockée en cache. La table arp est une table de correspondance entre adresses IP et adresses matérielles (ou MAC). Elle est générée au fur et à mesure grâce au protocole ARP et stockée en cache.

La commande permet de détecter 2 types de problèmes : le **fonctionnement au niveau hardware**de la carte et la bonne correspondance entre une adresse IP et une carte réseau au moyen de son adresse MAC. (cf. double attribution d'une adresse IP sur un réseau, spoofing, ...).

Exemple : Vous voulez vérifier que la seconde carte réseau appelée `eth1` est opérationnelle. Dans un 1er temps, on va pinger la carte, ce qui va permettre d'alimenter la table arp.

```
root@pingu# ping 192.168.0.4
```

Une fois la table alimentée, on vérifie son contenu : la présence de l'adresse IP ET de l'adresse MAC

```
root@pingu# arp -a
? (192.168.0.4) at 52:54:05:F5:AD:0C [ether] on eth1
? (192.168.0.1) at 00:04:76:8E:B2:90 [ether] on eth1
```

En cas d'erreur dans la table, il est inutile de pousser plus avant le diagnostic... On est face à un problème matériel (certainement le plus pénible à détecter).

ping

Véritable outil à tout faire, à connaitre absolument ! Il permet de détecter bon nombre de problèmes concernant votre configuration IP : adressage de votre carte réseau (adresse IP, adresse de réseau, routage, configuration de la résolution de nom).

Exemple : Mon réseau ne fonctionne pas. Je vais donc tester l'adresse avec la commande `ping`
```
root@pingu# ping 192.168.0.3
PING 192.168.0.5 (192.168.0.5) from 192.168.0.3 : 56(84) bytes of data.
From 192.168.0.3 icmp_seq=1 Destination Host Unreachable
```

Plusieurs causes de non-fonctionnement : je vérifie
- que l'adresse IP existe
- que le masque de sous-réseau est cohérent (`ifconfig`)
- que ma résolution de nom est opérationnelle (`/etc/resolv.conf`)
- que le routage est correctement configuré (commande `route` vue plus loin)

Si la commande `ping` fonctionne alors ce n'est ni un problème matériel, ni un problème de configuration IP. Toutefois le réseau peut subir des dysfonctionnements tout simplement parce qu'il est chargé. On augmente alors la taille des paquets envoyés.

Exemple :

```
root@pingu# ping -s 128 192.168.0.1
PING 192.168.0.1 (192.168.0.1) from 192.168.0.3 : 128(156) bytes of data.
136 bytes from 192.168.0.1: icmp_seq=1 ttl=128 time=0.583 ms
136 bytes from 192.168.0.1: icmp_seq=2 ttl=128 time=0.598 ms
136 bytes from 192.168.0.1: icmp_seq=3 ttl=128 time=0.640 ms
136 bytes from 192.168.0.1: icmp_seq=4 ttl=128 time=0.578 ms

--- 192.168.0.1 ping statistics ---
4 packets transmitted, 4 received, 0% loss, time 3008ms
rtt min/avg/max/mdev = 0.578/0.599/0.640/0.038 m
```

Ceci permet d'approfondir le test. On vérifiera :
- les numéros de séquences (`icmp_seq`) : ils sont séquentiels et numérotent les paquets envoyés. Une surcharge réseau ou un câblage de mauvaise qualité par exemple entraine la perte de paquets (voir `% loss`).
- le temps de transmission (`time`) : plus il est long, plus le réseau est chargé.

Attention : si vous lancez un ping sur un serveur et que celui-ci ne répond pas, cela ne signifie pas forcément qu'il y ait un problème. En effet, les firewalls peuvent bloquer toute réponse à un ping et faire échouer toute tentative de ping sur eux.

route

Permet d'afficher la table de routage en cache. Une commande équivalente est `netstat -r`. La commande affiche la table de routage et effectue la résolution de nom dès que possible. Les commandes `route -n` ou `netstat -rn` affichent le table de routage sans résolution de nom.

Exemple :

```
root@pingu# route
Table de routage IP du noyau
Destination Passerelle Genmask Indic Metric Ref Use Iface
loopback1-lns10 * 255.255.255.255 UH 0 0 0 ppp0
thepingu * 255.255.255.0 U 0 0 0 eth1
127.0.0.0 * 255.0.0.0 U 0 0 0 lo
default loopback1-lns10 0.0.0.0 UG 0 0 0 ppp0

root@pingu# route -n
Table de routage IP du noyau
Destination Passerelle Genmask Indic Metric Ref Use Iface
62.4.16.253 0.0.0.0 255.255.255.255 UH 0 0 0 ppp0
192.168.0.0 0.0.0.0 255.255.255.0 U 0 0 0 eth1
127.0.0.0 0.0.0.0 255.0.0.0 U 0 0 0 lo
0.0.0.0 62.4.16.253 0.0.0.0 UG 0 0 0 ppp0
```

Pour info : `U` = routeur installé et opérationnel ; `G` = passerelle distante ; `H` = destination est un hôte et non un réseau.

Un examen de la table de routage vous permet de vérifier par exemple que la machine sur laquelle vous travaillez et qui se situe derrière une passerelle a bien une route par défaut qui lui permet d'aller vers l'extérieur (`Destination : default`).

Vous ne pouvez pas atteindre l'extérieur car vous n'avez pas cette route, 2 solutions :

- vous ajoutez la route pour la session en cours :
  `root@pingu# route add default gw adresseIP_de_la_passerelle`
- vous modifiez le fichier de configuration du routage de manière à ce que la table contienne la route pour toutes les sessions :
  `/etc/sysconfig/network`

Vous ne pouvez pas atteindre l'extérieur car vous avez une route erronée, 2 solutions :

- vous modifiez la route pour la session en cours :
  `root@pingu# route del default gw adresseIP_de_la_passerelle_erronée`
  `root@pingu# route add default gw adresseIP_de_la_passerelle_correcte`
- vous modifiez le fichier de configuration du routage de manière à ce que la table contienne la route correcte pour toutes les sessions :
  `/etc/sysconfig/network`

traceroute

Vous ne parvenez pas à atteindre une URL ou un poste donné... La commande `traceroute`, comme son nom l'indique, vous établit la route suivie par les paquets de données vers la destination. La route est constituée de tous les routeurs traversés pour arriver à destination.

Exemple :

```
root@pingu# traceroute 192.168.0.4 traceroute to 192.168.0.4 (192.168.0.4), 30 hops max, 38 byte
packets 1 192.168.0.4 (192.168.0.4) 0.638 ms 1.016 ms 0.667 ms
```

Il s'agit d'une machine locale. Elle est située sur le même réseau, aucun routeur n'est traversé, l'adresse de destination est donc atteinte directement.

```
root@pingu# traceroute lea-linux.org
traceroute to lea-linux.org (80.67.179.10), 30 hops max, 38 byte packets
 1 loopback1-lns103-tip-telehouse.nerim.net (62.4.16.253) 152.218 ms 51.991 ms 48.179 ms
 2 hsrp1-telehouse.nerim.net (62.4.16.9) 77.410 ms 48.695 ms 66.830 ms
 3 feth0-0-julo.nerim.net (62.4.16.37) 135.323 ms 55.336 ms 47.656 ms
 4 placenet.freeix.net (213.228.3.223) 48.269 ms 144.190 ms 54.287 ms
 5 charon.placenet.org (80.67.178.242) 115.933 ms 58.049 ms 55.553 ms
 6 web.t2.tuxfamily.net (80.67.179.10) 1079.365 ms 949.915 ms 578.333 ms
```

On a ici une adresse située à l'extérieur, tous les routeurs traversés sont indiqués et numérotés.

Là où `traceroute` devient un outil de diagnostic :

Exemple :

```
root@pingu# traceroute 192.168.0.5
traceroute to 192.168.0.5 (192.168.0.5), 30 hops max, 38 byte packets
 1  192.168.0.3 (192.168.0.3)  2999.119 ms !H  2994.720 ms !H  2999.823 ms !H
```

Cette fois−ci la commande nous donne des informations différentes : il lui est impossible de trouver la route et il affiche directement la destination avec un des codes d'erreur suivant :

- **!H** : host unreachable
- **!N** : network unreachable
- **!P** : protocol unreachable

netstat

On a déjà vu la commande netstat dans le cadre de l'affichage de la table de routage. La commande permet également d'obtenir des informations détaillées sur l'**état des interfaces réseau** :

Exemple :

```
root@pingu# netstat −i
Table d'interfaces noyau
Iface MTU Met RX−OK RX−ERR RX−DRP RX−OVR TX−OK TX−ERR TX−DRP TX−OVR Flg
eth0 1500 0 20103 0 0 0 18666 0 0 0 BMRU
eth1 1500 0 18453 0 0 0 19879 0 0 0 BMRU
lo 16436 0 466 0 0 0 466 0 0 0 LRU
ppp0 1492 0 20056 20103 0 0 18619 0 0 0 MOPRU
```

RX	paquets reçus
TX	paquets envoyés (transmis)
OK	paquets reçus/envoyés correctement
ERR	paquets reçus/envoyés avec des erreurs
DRP	paquets reçus/envoyés droppés
OVR	paquets reçus/envoyés retransmis

D'autres options de la commande permettent notamment de visualiser les ports ouverts : `netstat −tu` (tcp et udp).

Exemple :

```
root@pingu# netstat −tu
Connexions Internet actives (sans serveurs)
Proto Recv−Q Send−Q Adresse locale Adresse distante Etat
tcp 0 0 pingu.linuxeries.:32771 132.248.32.28:46503 ESTABLISHED
tcp 0 0 pingu.linuxeries.:32773 64.12.25.127:5190 ESTABLISHED
tcp 0 0 pingu.linuxeries.:33012 217.174.201.37:ircd ESTABLISHED
tcp 0 0 192.168.0.3:netbios−ssn ouaichland:1177 ESTABLISHED
tcp 1231 0 pingu.linuxeries.:34140 193.201.103.96:http ESTABLISHED
```

tcpdump

Il s'agit d'un outil à réserver aux utilisateurs avertis (il nécessite quelques connaissances sur les protocoles) qui va vous permettre de visualiser les paquets qui circulent vers et/ou à partir d'une interface réseau et ce, en temps réel. C'est à la fois un outil de diagnostic sécurité mais aussi un outil de détection d'anomalies de la configuration IP ou matérielle.

Le fonctionnement en est relativement simple : la commande, sans aucun filtre, permet d'afficher le contenu des paquets et leur description qui transitent sur les interfaces réseau. On peut ensuite élaborer des filtres en fonction des informations recherchées. En voici quelques uns :

- `and`, `or` : permet de combiner les filtres
- `src`, `dst` : permet de choisir les paquets provenant de / à destination d'une interface réseau.
- `host`,`net`, `port` : filtrer l'affichage des paquets selon un nom de machine, un réseau et/ou un port

De cette manière, vous pouvez utiliser `tcpdump` si vous essayez de diagnostiquer un problème comme des erreurs de protocole, ou des déconnexions bizarres, ceci car il vous permet de voir en temps réel ce qui arrive sur le réseau. Pour bien utiliser `tcpdump`, vous devez avoir quelques connaissances sur les protocoles et comment ils fonctionnent, mais il est aussi utile pour quelques services simples comme vérifier que les paquets quittent bien votre machine par le bon port si vous essayez de diagnostiquer des problèmes de routage et pour voir si vous recevez des paquets en provenance de destinations éloignées.

Exemple :

```
root@pingu# tcpdump −i eth1 port 23
tcpdump: listening on eth1
00:53:58.230287 192.168.0.4.1497 > pingu.linuxeries.org.telnet: S 1074459360:1074459360(0) win 16384 <mss
1460,nop,nop,sackOK> (DF)
00:53:58.230477 pingu.linuxeries.org.telnet > 192.168.0.4.1497: S 607927502:607927502(0) ack 1074459361 win
5840 <mss 1460,nop,nop,sackOK> (DF)
00:53:58.230820 192.168.0.4.1497 > pingu.linuxeries.org.telnet: . ack 1 win 17520 (DF)
00:53:58.741012 pingu.linuxeries.org.telnet > 192.168.0.4.1497: P 1:13(12) ack 1 win 5840 (DF) [tos 0x10]
00:53:58.741608 192.168.0.4.1497 > pingu.linuxeries.org.telnet: P 1:7(6) ack 13 win 17508 (DF)
.....
```

La commande ici nous permet de visualiser les paquets liés à l'activité `telnet` sur l'interface réseau `eth1`.

nmap / nmapfe

Il s'agit d'un scanner de ports, à utiliser par simple curiosité ou pour s'assurer du niveau de sécurité effectif sur sa machine. Les options de cette commandes sont très nombreuses et je ne les détaillerai pas ici.

La syntaxe : `nmap <options> adresse_IP`

Exemple :

```
# nmap −sS −O 192.168.0.3

Starting nmap V. 2.54BETA31 ( www.insecure.org/nmap/ )
Interesting ports on  (192.168.0.3):
(The 1544 ports scanned but not shown below are in state: closed)
Port       State       Service
21/tcp     open        ftp
53/tcp     open        domain
80/tcp     open        http
3306/tcp   open        mysql

Remote operating system guess: Linux Kernel 2.4.0 − 2.4.17 (X86)
Uptime 2.279 days (since Fri Oct  4 16:24:45 2002)

Nmap run completed −− 1 IP address (1 host up) scanned in 4 seconds:!
```

Le scan réalisé ici permet de vérifier quels sont les ports ouverts et donc potentiellement vulnérables (grâce aux options `−sS`) et le système d'exploitation de la machine testée (grâce à l'option `−O`).

nmapfe est quant à lui une interface graphique (front end) à nmap.

lsof

La commande lsof (list opened files) permet de lister les fichiers ouverts par les processus tournant sur le système. On peut aussi l'utiliser pour ce qui nous concerne et ce de différentes manières. Là encore le traitement ne sera pas exhaustif.

Voir les fichiers utilisés par un processus réseau :

```
root@pingu# lsof|grep proftpd
COMMAND PID USER FD TYPE DEVICE SIZE NODE NAME
proftpd 2136 root cwd DIR 22,2 4096 2 /
proftpd 2136 root rtd DIR 22,2 4096 2 /
proftpd 2136 root txt REG 22,2 236456 244504 /usr/local/sbin/proftpd
proftpd 2136 root mem REG 22,2 89547 1661383 /lib/ld−2.2.5.so
proftpd 2136 root mem REG 22,2 23575 1661394 /lib/libcrypt−2.2.5.so
proftpd 2136 root mem REG 22,2 35340 1661460 /lib/libpam.so.0.75
proftpd 2136 root mem REG 22,2 12102 1661396 /lib/libdl−2.2.5.so
proftpd 2136 root mem REG 22,2 45415 1661416 /lib/libnss_files−2.2.5.so
proftpd 2136 root mem REG 22,2 46117 1661424 /lib/libnss_nisplus−2.2.5.so
proftpd 2136 root mem REG 22,2 89424 1661400 /lib/libnsl−2.2.5.so
proftpd 2136 root mem REG 22,2 16051 1661413 /lib/libnss_dns−2.2.5.so
proftpd 2136 root mem REG 22,2 68925 1661428 /lib/libresolv−2.2.5.so
proftpd 2136 root mem REG 22,21401027 651528 /lib/i686/libc−2.2.5.so
proftpd 2136 root 0u IPv4 19073 TCP *:ftp (LISTEN)
proftpd 2136 root 1u IPv4 19074 TCP *:46000 (LISTEN)
proftpd 2136 root 2u IPv4 19075 TCP *:47000 (LISTEN)
proftpd 2136 root 3u unix 0xc4a6c0a0 16891 socket
proftpd 2136 root 4r REG 22,2 1585 213675 /etc/passwd
proftpd 2136 root 5r REG 22,2 657 212895 /etc/group
```

`COMMAND`	nom du processus
`PID`	numéro de processus (obtenu aussi par la commande `ps`
`USER`	identité sous laquelle est lancé le processus
`FD`	file descriptor − mem : memory−mapped file ; txt : program text (code and data)... plus de détails dans le man de `lsof`
`TYPE`	type de noeud (ou inode) − CHR : fichier spécial en mode caractère ; DIR : répertoire... plus de détails dans le man de `lsof`
`DEVICE`	major et minor number pour un fichier spécial, protocole...
`SIZE`	taille du fichier
`NODE`	numéro d'inode
`NAME`	nom du fichier ou point de montage

Voir si un fichier est utilisé par un processus réseau
Permet éventuellement de surveiller l'utilisation de fichiers dits sensibles pour la sécurité du système.

Exemple : je veux vérifier les éventuels accès au fichier `/etc/passwd`

```
root@pingu# lsof | grep /etc/passwd
proftpd 12586 root 4r REG 22,2 1585 213675 /etc/passwd
```

Lister les processus liés à un protocole et un port
On obtient alors une fonctionnalité similaire à la commande `netstat −a` filtrée en fonction d'un protocole et/ou d'un port.
Exemple : je souhaite lister les informations liées au protocole TCP et au port 80 (donc mon serveur web).

```
# lsof −i tcp:80 COMMAND PID USER FD TYPE DEVICE SIZE NODE NAME httpd 1136 root 16u IPv4 1963
TCP *:http (LISTEN) httpd 1139 root 16u IPv4 1963 TCP *:http (LISTEN) httpd 1140 root 16u IPv4
1963 TCP *:http (LISTEN) httpd 1141 root 16u IPv4 1963 TCP *:http (LISTEN) ....
```

Note
Sur la remarque tout à fait justifiée de Jicé (mais non je ne fayotte pas :), on peut en parallèle à la commande `lsof`, citer `fuser`. Elle est vous donnera moins d'informations mais peut vous permettre d'obtenir le même genre de renseignements.

Une restriction toute fois spécifée dans le man de `fuser` :
" fuser ne dispose de toutes les informations que s'il est exécuté avec les privilèges de root. Ainsi, des fichiers ouverts par des processus appartenant à d'autres utilisateurs n'apparaîtront peut−être pas"

Exemple : je veux savoir quels sont les fichiers utilisés par mon serveur apache :

```
root@pingu# fuser −auv −n tcp 80
USER PID ACCESS COMMAND
80/tcp root 1136 f.... httpd
root 1139 f.... httpd
root 1140 f.... httpd
root 1141 f.... httpd
root 1142 f.... httpd
root 1144 f.... httpd
root 1145 f.... httpd
root 1146 f.... httpd
root 1147 f.... httpd
root 6434 f.... httpd
```

Le problème ici est que httpd fonctionne avec l'identité `apache` et non root. D'où le manque d'informations sur les fichiers utilisés.

resolv.conf

Ce fichier permet la configuration d'un client DNS. C'est lui qui permet l'utilisation de serveurs DNS pour la résolution de noms en adresse IP (ce qui vous permet par exemple de taper une URL dans un navigateur et non une adresse IP). Un fichier mal configuré vous empêchera notamment de surfer.

Rappel de la syntaxe :

```
root@pingu# cat /etc/resolv.conf
domain nom_de_domaine
nameserver adresseIP_DNS_Primaire
nameserver adresseIP_DNS_Secondaire
```

Attention : `nameserver` est un mot−clé à recopier tel que.`domain` contient le nom de domaine du provider qui vous a fourni les DNS ou celui de votre domaine, si vous disposez d'un serveur DNS.

telnet

On connait `telnet` en tant qu'outil prise de contrôle à distance. Ce n'est pas là sa seule utilité, il est aussi très utile pour établir un diagnostic et vérifier le fonctionnement ou non d'un service en se connectant sur son port. Il permet également de vérifier la validité d'une règle de firewall.

La syntaxe : `telnet <adresse IP ou nom de machine> <numéro de port du service à tester>`

Exemple :
Je souhaite vérifier le bon fonctionnement de mon service ftp. Après vérification dans `/etc/services`, je sais que le port correspondant au serveur ftp est le 21.

- 1er cas de figure :

```
root@pingu# telnet 192.168.0.3 21
Trying 192.168.0.3...
Connected to 192.168.0.3.
Escape character is '^]'.
220 ProFTP Linuxerie's Server Ready
```

  La connexion est réussie, le serveur est opérationnel.
- 2e cas de figure :

```
root@pingu# telnet 192.168.0.3 21
Trying 192.168.0.3...
telnet: connect to address 192.168.0.3: Connexion refused
```

La connexion est impossible, le service n'est pas disponible.

whois

Vous avez récupéré des informations sous forme d'IP ou de nom dans vos logs ou sur une sortie écran de `tcpdump`, vous voulez savoir qui se cache derrière cette IP.

Exemple:

```
root@pingu# whois lea-linux.org
Whois Server Version 1.3

Domain names in the .com, .net, and .org domains can now be registered with many different competing
registrars. Go to http://www.internic.net for detailed information.

  Domain Name: LEA-LINUX.ORG
  Registrar: GANDI
  Whois Server: whois.gandi.net
  Referral URL: http://www.gandi.net
  Name Server: NS1.TUXFAMILY.NET
  Name Server: NS2.TUXFAMILY.NET
  Updated Date: 11-feb-2002


Last update of whois database: Tue, 8 Oct 2002 05:00:23 EDT

The Registry database contains ONLY .COM, .NET, .ORG, .EDU domains and Registrars.
.....
```

autres outils d'identification

Ci-dessous, trois autres outils qui vous permettront de faire de la résolution de nom ou de la résolution inverse avec des informations complémentaires sur l'identité du serveur :

- **`nslookup`** : utilisable en mode interactif ou non, à partir d'une adresse IP ou d'un nom.

  Exemple :

  ```
  root@pingu# nslookup
  > lea-linux.org
  Server:         194.149.160.9
  Address:        194.149.160.9#53

  Non-authoritative answer:
  Name:   lea-linux.org
  Address: 80.67.179.10
  ```

  La commande vous donne les infos suivantes :

  - `Server`, `Address` : le serveur DNS qui effectue la résolution
  - `Non-authoritative answer` : si la ligne est présente, cela signifie que l'adresse était présente dans le cache du serveur DNS.
  - `Name`, `Addresse` : résultat de la commande
- **`dig`** : la commande `nslookup` tombant en désuétude, elle est remplacée aujourd'hui de plus en plus par `dig`. Elle dispose de fonctionnalités supplémentaires.

  Exemple :

  ```
  # dig www.lea-linux.org

  ; DiG 9.2.0 www.lea-linux.org
  ;; global options: printcmd
  ;; Got answer:
  ;; -HEADER- opcode: QUERY, status: NOERROR, id: 25724
  ;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 2

  ;; QUESTION SECTION:
  ;www.lea-linux.org.       IN  A

  ;; ANSWER SECTION:
  www.lea-linux.org. 86400  IN  A   80.67.179.10
  ```

```
;; AUTHORITY SECTION:
lea-linux.org.     604800 IN  NS ns1.tuxfamily.net.
lea-linux.org.     604800 IN  NS ns2.tuxfamily.net.

;; ADDITIONAL SECTION:
ns1.tuxfamily.net. 86400  IN  A   80.67.177.2
ns2.tuxfamily.net. 86400  IN  A   80.67.179.2

;; Query time: 325 msec
;; SERVER: 194.149.160.9#53(194.149.160.9)
;; WHEN: Fri Oct 11 00:12:30 2002
;; MSG SIZE rcvd: 132
```

La sortie standard de la commande est sensiblement la même avec quelques précisions supplémentaires (durée de réalisation de la requête, DNS du domaine...)

- **`host`** : sans option, la sortie est extrêmement simple, il s'agit uniquement de la résolution.

Exemple :

```
# host www.lea-linux.org
www.lea-linux.org has address 80.67.179.10
```

Le Mot de la fin

Mon objectif en écrivant cette contribution était de débroussailler quelque peu la jungle des outils de configuration et diagnostic réseau. J'espère qu'il vous donnera envie d'approfondir encore plus ce domaine passionnant.
N'hésitez pas à m'envoyer vos remarques ou ajouts éventuels.

Remerciements

Un grand merci à Fred, Prae et Jice qui ont accepté de prendre du temps pour relire et corriger cet article (Jice : je promets de faire des efforts en html :p)

Paramétrer sa connexion à internet par modem

par Serge et JCC

tout savoir sur la connexion de Linux avec l'Internet.

Installer son modem

Avant tout, il faut paramétrer le modem et/ou vérifier qu'il est bien reconnu.

ATTENTION : **Si vous avez un modem interne**, lisez ceci : si votre modem est un "**WinModem**" (voir glossaire) (son nom sur la boîte de l'emballage), ou un modèle **PCI** (toutes marques sauf *Olitec* qui devrait dans un futur que l'on espère proche faire quelque chose pour Linux, ainsi que quelques modems internes basés sur certains chipsets (voir http://www.linmodems.org/)), votre modem ne pourra pas être installé sous Linux. C'est pas la faute à Linux, mais celle du constructeur qui ne fournit pas de drivers pour Linux, et/ou qui ne donne pas les spécifications du modem. Bref achetez−en un autre ou faites un forcing auprès du constructeur (mailez−leur tous les jours, vous et vos amis aussi).

Pour le reste des modems, la plupart des "VRAIS" modems internes (carte ISA émulant un vrai port série en hardware) et tous les modems externes peuvent être installés sous Linux.

Tout d'abord on va vérifier que votre modem est bien reconnu. Si c'est un modem externe vous devez savoir sur quel port série il est connecté (les ports COM sous windows), souvent indiqué sur votre machine (style sérial 1 ou COM1 ou série 1, etc...). Pour le tester, allez dans une console (ou un terminal si vous êtes sous X) et tapez au prompt:

```
$ echo "ATZ" > /dev/ttySx
$ echo "ATDT3611" > /dev/ttySx
```

en remplaçant `/dev/ttyS`**`x`** ci−dessus (et dans toutes les commandes suivantes) par :

- `/dev/ttyS`**`0`** si le modem est branché sur COM**1**,
- `/dev/ttyS`**`1`** si le modem est branché sur COM**2**,
- `/dev/ttyS`**`2`** si le modem est branché sur COM**3**,
- `/dev/ttyS`**`3`** si le modem est branché sur COM**4**.

Si votre modem fait du bruit, que vous entendez un "BBIIPPP BBBUUPPP", ou autre bruit du même style, que les diodes clignotent (modem externe !), votre modem est bien reconnu, éteignez−le vite (sinon, vous vous connectez au 3611, rassurez−vous, c'est gratuit les 3 premières minutes !). Si votre modem est interne et que vous pouvez pas l'éteindre, debranchez la prise téléphonique, puis tapez :

```
$ echo "+++" > /dev/ttySx
```

Bon maintenant que votre modem est reconnu, on va créer un lien pour le modem, par la commande (là aussi bien mettre le bon `ttyS`**`x`**) :

```
$ ln −s /dev/ttySx /dev/modem
```

Si la réponse est que `/dev/modem` existe déjà, suprimez−le (`rm −f /dev/modem`) et recommencez.

Maintenant appliquons les droits qu'il faut :

```
$ chmod 777 /dev/modem
```

A ce stade on peut dire que votre modem est configuré sous Linux.

Remarque : Si vous avez une erreur de port série, reconfigurez vos ports séries par la commande :

```
setserial /dev/ttySw uart 16450 port 0xyyy irq z
```

pour le port ou se trouve le modem, avec `w`, `yyy` et `z` suivant le tableau :

Port série N° :	1 (/dev/ttyS0 ou COM1)	2 (/dev/ttyS1 ou COM2)	3 (/dev/ttyS2 ou COM3)	4 (/dev/ttyS3 ou COM4)
w :	0	1	2	3
yyy :	3F8	2F8	3E8	2E8
z :	4	3	4	3

Placez cette commande dans un script de démarrage (`/etc/rc.d/rc.local` par exemple).
Pour la Slackware, il vous suffit de décommenter ces lignes dans le fichier `/etc/serial.conf` (je sais pas si ce fichier existe aussi dans les RH et Mandrake).

Paramétrage de la connexion

Bon tout d'abord il faut voir si `pppd` est installé sur votre système (`pppd` est la partie logicielle qui négocie et gère la connexion avec le provider). Vérifiez en tapant la commande "`pppd`" au prompt. Si vous voyez un truc du style "`~é{!{!{!aE.....`" c'est qu'il est installé (attendez un moment, il va s'arrêter tout seul). Autrement, reconfigurez le noyau avec `pppd` soit en module soit en natif dans le kernel (voir la rubrique kernel).

Bon maintenant plusieurs solutions s'offrent à vous :

Connexion en mode graphique

Si vous êtes sous KDE et que dans le menu "Internet" vous avez une application "Kppp" lancez−là, et configurez votre connexion comme indiqué ci−dessous. Si KDE n'est pas votre environement préféré mais qu'il est quand même installé, lancez−le à partir d'un autre WM : dans un terminal, tapez `kppp`, il marche même en dehors de KDE. Il vous reste plus qu'à mettre une icône sur le bureau pour y accéder.
Si `kppp` n'est pas installé, vous pouvez aussi utilisez `ezppp` (prononcez easy ppp) surlequel `kppp` se base. Recherchez ce logiciel sur le CD de votre distribution, ou bien sur internet (Freshmeat par exemple). Les deux logiciels se configurent sensiblement de la même façon.

Configuration :

Créez un **compte** (account ou compte) : ajoutez en un, donnez lui un nom de votre choix, remplissez le numéro de téléphone de votre fournisseur d'accès et laissez le type d'authentification sur PAP. Sur **IP** laissez "dynamic ip" et pour **DNS**, ajoutez les adresses des serveur DNS de votre fournisseur d'accès. Puis sur l'onglet modem, vérifiez que `/dev/modem` (celui qu'on a créé plus haut) soit choisi. Puis remplissez les champs **user** et **password**, puis cliquez sur [**connect**]. Tout devrait bien se passer : la connexion à internet doit s'établir.

ATTENTION : moi−même et d'autres utilisateurs avons eu des problèmes avec `kppp` et la Mandrake 7.0 : la connexion s'arrête toute seule sans raison apparente. Allez voir sur le site de Mandrake et dans les liste d'aide de leur site, ou alors utilisez `linuxconf` pour configurer votre interface `ppp0`.

Connexion en mode console

Bon maintenant la bonne vieille méthode de configuration à la main si vous avez pas d'interface X ou pas d'utilitaire pour configurer la connexion, ou tout simplement si vous voulez pouvoir lancer la connexion internet par un simple script, ce qui est plus dans la philosophie Unix.
On va créer un script à la main, mais comme on est feignant on va s'aider d'un utilitaire : `pppsetup` (livré en défaut sur la slackware) que vous trouverez sur Freshmeat. Eventuellement installez `pppsetup`, puis lancez−le.

Il vous demande (dans l'ordre) : le numéro de téléphone de votre fournisseur d'accès, le port série où est branché le modem, la vitesse du port (prenez 115 KBPS maximum, 57,6 par sécurité) , NO pour le callback, validez l'init sans rien taper, le nom de domaine de votre fournisseur d'accès (du style `free.fr` ou `wanadoo.fr`, etc...) , l'adresse IP du serveur DNS de votre fournisseur d'accès, PAP pour l'authentification, votre user et votre mot de passe, et enfin exit.

Puis pour vous connecter, tapez dans un terminal : "`ppp−go`", et pour vous déconnecter : "`ppp−off`".

Si rien ne marche, regardez les log de syslog (`/var/log/messages`) pour voir déjà si `pppd` se lance, et si oui quel est le message d'erreur (authentication failed: mauvais mot de passe ou mauvais username). Si la connexion se coupe sans de raison vraiment apparente, esayez une vitesse de port série inférieure.

Problème fréquent

Votre connexion passe, mais impossible d'aller sur un site.

Dans ce cas, allez sous la console ou sous un terminal et essayez un ping sur un serveur connu :

```
$ ping  www.netscape.com
```
si ça ne répond pas, essayez alors :
```
$ ping 205.188.247.66
```
si ça répond : alors vous avez mal configuré les serveurs DNS de votre fournisseur d'accès, ouvrez le fichier `/etc/resolv.conf` dans votre éditeur de texte favori et ajoutez autant de lignes que de serveurs DNS de la forme :

```
nameserver   123.456.789.012
```
(où `123.456.789.012` est l'adresse IP du DNS)

Attention de bien avoir au début de ce fichier la ligne :

```
search  nom.domaine
```
(avec `nom.domaine` le nom de domaine de votre fournisseur, par exemple `free.fr`)

Si le ping sur 205.188.147.66 ne passe pas, tapez dans une console la ligne :

```
$ ifconfig
```

vous allez voir une ligne du style :
```
ppp0: xxxxxxxxxxxxxxxx   <− notez bien le ppp0 (ou ppp1, etc...)
     inet adress : yyy.yyy.yyy.yyy <− notez cette adresse
```

tapez alors:

```
$ route add default gw yyy.yyy.yyy.yyy ppp0
```

Retentez le ping et ça devrait passer.

Si lors du `ifconfig` vous n'avez AUCUNE ligne `pppX` , la connexion est mal configurée, revoyez le tout.

Compiler et configurer pengaol 1.0

par Martial

Ce document explique comment compiler la dernière version de pengaol sans support de QTKylix, et la configuration de base de pengaol.

1. Introduction

1.1 Note

Cet article n'a rien d'officiel, et ne décrit pas nécessairement la «bonne méthode» pour installer pengaol. Rapellez vous que des versions précompilées existent déjà dans de nombreux formats (deb, tgz, rpm), et que pour profiter de l'interface graphique, il vaut mieux lire la très bonne documentation officielle sur le site pengaol.org. Le but ici est de compiler pengaol sans avoir besoin de QTkylix pour configurer les options de compilations, et utiliser pengaol uniquement en ligne de commande.

Suite à certains mails, je precise que je n'ai rien à voir dans le développement de pengaol; contactez plutot son auteur: birdy57 chez pengaol point org, et visitez leurs forums qui sont prévus à cet effet.

2. Compilation

2.1 Requis

Comme d'habitude, il vous faut un système prêt à compiler, c'est à dire g++, ainsi que les fichiers d'entête (paquets −devel en rpm) et les utilitaires classiques comme autoconf, make etc... passons, c'est hors−sujet.

J'ai utilisé exactement cette version: peng1.0a01.tar.gz prise sur le site officiel.

2.2 Commençons

On désarchive:

```
tar zxvf peng1.0a01.tar.gz &&
cd peng
```

Puis on va génerer un `configure` plus classique, qui ne cherche pas à lancer une GUI en QTKylix:

```
mv configure configure− &&
mv configure.ori configure
```

On peut alors configurer:

```
./configure \
−−prefix=/usr \
−−mandir=/usr/share/man \
−−infodir=/usr/share/info &&
echo "#define WITH_PPP" >> config.h &&
echo "#define WITH_MODEM" >> config.h &&
echo "#define WITH_TUNTAP" >> config.h &&
echo "#define WITH_CABLE" >> config.h
```

On crée d'avance les répertoires requis:

```
mkdir −p /etc/Pengaol
chmod 770 /etc/Pengaol
mkdir −p /usr/share/sound
```

Et on installe simplement (hum):

```
make INSTALL="/bin/install −c" install &&
cp peng/PengMessages.txt /etc/Pengaol &&
cp peng/Dns /etc/Pengaol &&
chown root.peng /etc/Pengaol/PengMessages.txt /etc/Pengaol/Dns &&
chmod 660 /etc/Pengaol/PengMessages.txt /etc/Pengaol/Dns &&
mv /usr/bin/peng /usr/bin/pengaol &&
chmod 755 /usr/bin/pengaol &&
chmod +s /usr/bin/pengaol &&
cp wavplay−1.4/wavplay /usr/bin &&
cp wavplay−1.4/*.wav /usr/share/sound
```

Au besoin remplacez `/bin/install −c` par `/usr/bin/install −c`. On met les les permissions sur `/etc/Pengaol` comme ceci, pour que, lancé par un utilisateur normal, peng puisse y écrire lors de l'activation du dns.

Pour que cela marche, créez un nouveau groupe `peng` avec `vigrp` ou une autre commande, pour avoir quelquechose dans ce goût là :

```
peng:x:19:martial
```

En modifiant 19 par un numero de groupe libre sur votre système, et `martial` (c'est moi, lol) par une liste d'utilisateurs normaux autorisés à se servir de `pengaol`, séparés par des virgules.

3. Utilisation

3.1 ppp ou tun ?

Ce sont les deux drivers possibles − ppp est sans doute un peu plus rapide. En tous cas, il vous faut soit le module `tun.o`, soit les modules `ppp_generic.o` et `ppp_synctty.o`. Le module `ppp_synctty.o` n'étant pas le plus courant, il se peut que votre distro soit configurée pour charger `ppp_asynctty.o` à la place, il faut alors modifier **/etc/modules.conf**.

Pour ne pas avoir d'erreur de détection des drivers, il faut que les devices soient accesibles par `peng`, donc pensez à ajuster si besoin les permissions de ppp et du modem dans /dev (modules.conf peut automatiser cela avec post−install)

Pour vérifier que tout est bien configuré, tapez `pengaol −ListDriver`, ce qui doit donner quelque chose comme ceci (au moins):

```
Load language : fr
Nom du driver : CPppDriver
Auteur : Birdy57
Message inclus : Vers 0.55 Linux
Guid : 3812
Le pilote fonctionne sur cette configuration :Oui

Nom du driver : CTunTapDriver
Auteur : birdy57
Message inclus : Vers 0.5 Linux
Guid : 4212
Le pilote fonctionne sur cette configuration :Non

Nom du driver : CCableDriver
Auteur : birdy57 & surfufu
Message inclus : Vers 0.5 Multi
Guid : 8301
Le pilote fonctionne sur cette configuration :Non

Nom du driver : CModemDriver
Auteur : Birdy57
Message inclus : Vers 0.5 Linux
Guid : 1711
Le pilote fonctionne sur cette configuration :Oui
```

Il faut que `CModemDriver` et au moins un des `CTunTapDriver CPppDriver` soit à **oui** pour que ca marche. Si ce n'est pas le cas, vérifiez (modprobe) les modules chargés, vérifiez que les devices requis existent dans /dev, et avec les bonnes permissions.

Ensuite, tapez simplement `pengaol −User [login] [Pass]`. Mais il est conseillé de stocker votre mot de passe une fois pour toute avec: `pengaol −AddUser [login] [Pass]`, et de se connecter ensuite avec: `pengaol −Connect [login]`, ceci évite de dévoiler votre mot de passe dans les logs système et avec la commande **top**. Une petite ligne genre `xterm −T AOL −e pengaol −Connect monlogin` dans un menu de gnome, kde, wmaker et on y est.

3.2 Message d'erreur

Load language : fr serveur en attente ! signifie que vous n'avez pas rentré une option valide pour la version en ligne de commande, et cherche donc à lancer la GUI qui dans notre cas n'existe pas. Lire la doc incluse dans le répertoire doc des sources pour toutes les options.

4. Conclusion

Sous linux il y'a plein de manière d'arriver au même résultat, surtout concernant le chargement des modules du noyau et des permissions, donc ne prenez ce qui est dit ici que comme un exemple; essayez, modifiez selon ce que vous souhaitez obtenir.

Connexion à Internet multi−comptes

Partie 1 : Configuration de pppd

Par Fred

Avant de pouvoir nous connecter à internet via tous nos providers, voyons déjà comment établir une connexion à Internet.

Introduction

Cette série de 4 articles va nous permettre d'utiliser une connexion internet avec plusieurs fournisseurs d'accès (on dit aussi ISP, ou FAI), de façon transparente pour la gestion du mail notamment. Les conseils qui sont donnés ici peuvent tout aussi bien s'appliquer à d'autres logiciels que ceux donnés en exemple ; il suffira alors d'adapter les scripts donnés. On peut aussi utiliser ces articles pour configurer sa connexion avec un seul FAI.

Un peu de technique

Vous voulez vous connecter à internet donc vous avez contacté un fournisseur d'accès qui doit vous avoir fourni les renseignements suivants :

- un numéro de téléphone,
- un login (parfois appelé identifiant),
- un mot de passe (parfois appelé code d'accès),
- deux "dns" (aussi appelés nameserver, serveur de noms) sous la forme de deux numéros du type 123.123.123.123,
- un identifiant de messagerie (si vous n'en avez pas c'est que c'est votre login),
- un mot de passe de messagerie (si vous n'en avez pas, c'est le même que le précédant),
- le nom du serveur de messagerie sortant (souvent : smtp.*isp.com* **ou** mail.*isp.com*),
- le nom du serveur de messagerie entrant (souvent : pop.*isp.com*).

Tout ce qui vous a été fourni en plus ne sert à rien ;) avec Linux !

Établir une liaison à internet c'est relier votre ordinateur au réseau des réseaux via votre modem. C'est−à−dire connecter le modem de votre PC à celui de votre ISP par une liaison téléphonique en utilisant un protocole de communication particulier. Le protocole retenu dans ce document se nomme "Point to Point Protocol" ou ppp ; c'est le plus souvent employé... l'exception étant AOL ! Le programme chargé d'établir cette connexion se nomme `pppd` (pour ppp daemon, ppp pour Point to Point Protocol). Mais avant d'établir la liaison avec internet, vous devez contacter votre fournisseur d'accès (j'utiliserai désormais l'abréviation ISP : Internet Service Provider), le programme chargé de communiquer avec votre modem est `chat`.

Voyons comment configurer `chat`, le programme qui discute ("to chat" en anglais) avec le modem :

Configuration de `chat`

Ce n'est pas obligatoire mais nous allons stocker le fichier de configuration de `chat` dans `/etc/ppp/chat`, pour cela vous devez créer ce répertoire (mkdir `/etc/ppp/chat`) car il n'existe pas par défaut.

Les scripts `chat` se composent d'une série de paire de chaînes de caractère. La première chaîne de chaque paire est ce que l'on attend du modem, la seconde ce que l'on doit alors envoyer au modem. Pour configurer très finement votre modem, vous devez posséder la documentation technique de celui−ci mais en général ce n'est pas nécessaire.

Un premier script élémentaire pourrait être :

```
'' 'ATZ'
'OK' 'ATDT0359602000'
'CONNECT' ''
```

Explications :

- on n'attend rien : `''`, on renvoie `'ATZ'` (c'est la chaîne d'initialisation de la plupart des modems)
- en réponse de 'ATZ' on doit recevoir `'OK'`, au quel cas on renvoie `'ATDT0359602000'` (c'est la commande de numérotation − tel : 0359602000 − de la majorité des modems).
- le modem doit répondre `'CONNECT'` (s'il réussi à se connecter !), on ne renvoie rien.

C'est relativement simpliste. Un script plus complexe mais plus sûr devrait être :

```
'ABORT' 'BUSY'
'ABORT' 'ERROR'
'ABORT' 'NO CARRIER'
'ABORT' 'NO DIALTONE'
'ABORT' 'Invalid Login'
'ABORT' 'Login incorrect'
'' 'ATZ'
'OK' 'ATM0L0'
```

```
'OK' 'ATDT0359602000'
'CONNECT' ''
'TIMEOUT' '5'
'~~~' ''
```

Les séquences commençant par `'ABORT'` ne signifient pas que l'on attend `'ABORT'` mais que l'on doit "quitter `chat` en erreur" s'il l'on reçoit la chaîne qui suit. Les séquences commençant par `'TIMEOUT'` indiquent à `chat` que l'on doit attendre la réponse du modem pendant le nombre de secondes qui suit.

Appelons ce script `/etc/ppp/chat/connexion0`. Comme vous pouvez vous en douter, `connexion0` peut être changé en n'importe quoi.
Pour tester ce script, nous devons d'abord configurer `pppd`.

Configuration de pppd

Il existe plusieurs méthodes de configuration de pppd. Nous allons en choisir une et passer sous silence les autres (via kppp, ezppp etc) et créer nous même, avec nos petites mains, les fichiers de configuration. Nous allons créer une connexion appelée `connexion0` (connexion "zéro").

Pour pouvoir initialiser cette connexion, `pppd` a besoin de 3 (de mon point de vue) fichiers de configuration (au moins) :

- `/etc/ppp/options` :

`lock`	# la connexion est exclusive
`ttyS0 115200 crtscts`	# la connexion est sur /dev/ttyS0 (i.e. com1) en 115200 bauds avec contrôle de flux matériel (c'est le modem qui le fait !)
`noauth`	# l' isp ne doit pas s'authentifier pour nous donner une adresse
`defaultroute`	# ppp fourni la route par défaut
`usepeerdns`	# pour que pppd demande les DNS au provider, cela ne marche peut−être pas toujours mais chez moi : si !
`idle 300`	# pour demander à pppd de raccrocher votre téléphone au bout de 300 secondes d'inactivité. (Si vous ne voulez pas utiliser cette option, ne la mettez pas du tout.)

- `/etc/ppp/peers/connexion0`

Si le répertoire `/etc/ppp/peers` n'existe pas : vous devez le créer ! Le nom de ce répertoire est impératif (plus précisément : je ne sais pas comment le changer − autrement qu'en modifiant les sources de pppd).

```
connect '/usr/sbin/chat −v −f /etc/ppp/chat/connexion0'
user <login de connexion>
```

Ce fichier se passe de commentaire. Il n'est nécessaire que parce que c'est celui que `pppd` va chercher quand on lui passe comme argument '`call connexion0`'.

- `/etc/ppp/pap−secrets`

```
<login de connexion> * <mot de passe de connexion>
```

Ce fichier doit contenir votre 'login de connexion' et votre 'mot de passe' fourni par votre ISP. Prenez garde au fait que **le mot de passe est stocké en clair** dans ce fichier. Pour que tout cela fonctionne, il faut que vous seul ayez les droits de lire ces fichiers (mettez par exemple l'utilisateur qui doit activer la connexion internet dans le groupe `pppusers` et faites appartenir ces fichiers à ce groupe, voir l'article sur les permissions).

Ensuite, pour activer la connexion il suffit de faire :

```
/usr/sbin/pppd call connexion0
```

Pour voir comment tout cela ce passe faite : `tail −f /var/log/messages`. Vous devez suivre au fur et à mesure l'établissement de la connexion ainsi que l'affection de votre nouvelle adresse IP.
Si tout fonctionne, vous êtes maintenant connecté à Internet mais vous ne pouvez pas encore vous en servir : Linux ne sait toujours pas comment faire le lien entre les noms (www.trucs.org) et leur adresse IP (111.222.333.444).

Pour cela nous allons configurer la "résolution des noms" :

Configuration de `resolv.conf`

Le programme chargé de la résolution des noms (faire la correspondance entre nom et adresse) s'appelle un serveur de nom. Pour indiquer à Linux quel serveur de nom nous devons utiliser, il faut préciser celui−ci dans `/etc/resolv.conf` :

```
search org
nameserver xxx.xxx.xxx.xxx
nameserver yyy.yyy.yyy.yyy
```

A la place de `xxx.xxx.xxx.xxx` vous mettez l'adresse du premier DNS (name server, serveur de noms) que vous a fourni votre ISP et à la place de `yyy.yyy.yyy.yyy` le second.
On a le droit de mettre des commentaires dans ce fichier, pour peu que ceux−ci soient précédés du caractère '#'.

Testons cela : rétablissez votre liaison : `/usr/sbin/ppp call connexion0` puis lancez `nslookup` (ce que vous devez tapez est en gras, les réponses ne sont pas nécessairement les mêmes) :

```
[root@becane home]# nslookup
Default Server:  ns3.wanadoo.fr
Address:  193.252.19.3

> www.free.fr
Server:  ns3.wanadoo.fr
Address:  193.252.19.3

Non−authoritative answer:
Name:    www.free.fr
Address:  212.27.32.114

> exit
```

A la place de **www.free.fr** vous pouvez taper n'importe quel nom de domaine valide. Si vous obtenez :
`*** gros.tux can't find gros: No response from server`
C'est que :

- ou votre configuration n'est pas valide :
  - ♦ ou vous n'êtes pas connecté
  - ♦ ou vous n'avez pas ajouté le bon DNS dans `/etc/resolv.conf`
- ou le nom de domaine que vous avez tapé n'existe effectivement pas (ce qui est le cas ici : `gros.tux` n'existe pas!).

Établissement et coupure de la connexion

Pour activer la connexion, vous avez déjà pu constater que la commande est :

```
/usr/sbin/pppd call connexion0
```

(évidemment `connexion0` doit être remplacé par le nom du fichier de configuration dans `/etc/ppp/peers`). Pour que n'importe quel utilisateur ait le droit d'activer la connexion, je vous conseille de rendre `pppd` suid (c'est à dire que n'importe quel urilisateur peut le lancer, mais que ce programme possède les droits de root) par :

```
chmod +s /usr/sbin/pppd
```

Couper la connexion n'est pas aussi simple : il n'existe pas de programme s'en chargeant ! Nous devons tuer le processus de `pppd`. Mais heureusement c'est simple car il sauve son 'pid' dans `/var/run/ppp*.pid`. Donc, pour fermer la connexion, la commande est :

```
kill `cat /var/run/ppp*.pid`
```

Si cette commande vous retourne un message d'erreur, c'est sans doute parce que `pppd` n'est pas actif à ce moment !

Dans l'article suivant, nous allons voir comment configurer la messagerie.

Connexion à Internet multi−comptes
Partie 2 : Configuration de la messagerie

par Fred

Avant de pouvoir nous connecter à internet via tous nos providers, voyons déjà comment configurer la messagerie.

Configuration de sendmail et fetchmail

Pour que nous puissions recevoir et envoyer des mails *en local*, il faut configurer sendmail (ou qmail ou autre... mais je ne les connais pas... Si vous utilisez un autre programme que sendmail, adaptez l'article à votre cas).

Pour que notre machine puisse récupérer des mails sur des comptes pop (ceux de votre provider par exemple) il faut configurer fetchmail.

Configurer sendmail et fetchmail avec install−sendmail (conseillé)

Récupérer install−sendmail, décompresser l'archive où vous le souhaitez.
On se place dans le répertoire dans lequel on a décompressé install−sendmail et on lance le programme (attention **il ne faut pas** être root pour lancer ce programme, par compte il faut connaître le mot de passe du root car le script le demande) :

```
$ cd install−sendmail−xxxxx
$ ./install−sendmail −c
```

La première chose que l'on vous demande c'est la langue que vous souhaitez utiliser (parmi 8), choisissez le danois si c'est votre langue maternelle ;). La plupart des questions sont simples (on vous demande le nom de domaine de votre ISP, i.e. internet service provider, etc... toutes choses que vous a fourni votre provider s'il est poli, sinon demandez−les lui !).

- La première question qu'il nous pose : le nom de domaine dont nous souhaitons que tous email en provenance de notre machine aient ? Il faut répondre le nom de domaine que nous préférons, par exemple, moi je préfère `free.fr` donc je mets : **`free.fr`**.
- Ensuite, est−ce que tous les emails sont stockés sur cette machine ? Bien sûr que non, puisqu'ils proviennent de nos deux providers ainsi que de netcourrier qui nous fournit quelques autres adresses. Donc il faut répondre **non** !
- *Voulez vous que votre mail sortant soit expédié par un serveur de mail extérieur ?* Et bien **oui**, nous utiliserons le serveur smtp de notre provider.
- *Quel est votre serveur de mail sortant ?* Il nous faut mettre ici le nom du serveur smtp de notre provider, pour moi c'est : **smtp.free.fr**. *(pour ceux qui ont lu la suite : c'est ici qu'il faut préciser* **`{MAILHOST}`***)*
- *Voulez vous mettre les mails dans la file d'attente ?* Comme nous ne sommes pas connecté en permanent à internet, nous répondons **oui** à cette question.
- *Êtes vous le serveur de mail d'un serveur local ?* Ce pourrait être le cas, mais dans le cadre de cet article, nous supposerons que **non**.
- *Support pour les domaines virtuels, Avez vous besoins de ça ?* A vous de voir, mais je ne détaillerai pas cela, donc : **non**.
- *Voulez vous remapper des adresses locales en d'autres ?* Comme notre réseau ne fait pas partie d'internet, il faut répondre **oui**.
- Pour la section 5 de install sendmail, créer les alias que vous souhaiter, la procédure est suffisamment explicite.
- *Voulez vous utiliser le support de Fetchmail ?* **oui** nous utiliserons fetchmail pour récupérer les mails venant de l'extérieur (de tous nos comptes pop3).
- La configuration des différents compte pop3 est suffisamment simple pour ne pas être explicitée. Attention toutefois, dans la partie login entrez bien l'adresse email (sans le @isp.fr) correspondant au serveur pop !
- Ensuite install−sendmail génère les fichiers nécessaires à sendmail et fetchmail : `.fetchmailrc`, `access`, `access.db`, `sendmail.mc`, `sendmail.cf` et `sendmail.cw`. Puis il vous demande le mot de passe du root pour les copier où il faut (dans `/etc` et `/root`) ainsi que le login du collecteur de mail : vous devez préciser : **root** car nous allons récupérer les mails de tous les utilisateurs en même temps, et seul le root peut faire ça.

Voilà. Normalement à partir de maintenant, sendmail fonctionne en local. Vous pouvez même (après l'avoir démarré par `/etc/rc.d/init.d/sendmail start`) essayer d'envoyer un mail à l'un des utilisateur. Vous devez préciser : **localhost** comme serveur smtp pour votre gestionnaire de courrier.

Configurer sendmail à la main

Pour ceux qui aiment tout faire à la main (les fous ;)), on peut créer soit même tous les fichiers..., voilà le fichier `sendmail.mc` généré par install−sendmail (c'est une bonne base de départ) commenté (les commentaires sont en italique, et commencent par "dnl") :

```
divert(−1)
dnl il faut que le générateur de macro sache où trouver les fichiers de configuration :
include(`/usr/lib/sendmail−cf/m4/cf.m4')
dnl comme identifiant (8:12) vous pouvez sans doute mettre ce que vous voulez :
define(`confDEF_USER_ID',``8:12'')
dnl mon pc s'appelle gros.tux, mettez le nom de votre machine
VERSIONID(`gros.tux nodns')
Cwgros.tux localhost
dnl vous utilisez linux oui ou non ?
OSTYPE(`linux')
dnl on n'utilise pas UUCP (c'est antédiluvien)
undefine(`UUCP_RELAY')
dnl quelqu'un connaît bitnet ?
```

```
undefine(`BITNET_RELAY')
dnl autant ajouter automatiquement le nom de domaine qui va bien (celui d'un de nos providers)
FEATURE(always_add_domain)
dnl on utilise un fichier d'alias
FEATURE(use_cw_file)
dnl ????
FEATURE(nocanonify)
dnl ????
define(`confAUTO_REBUILD')
dnl ????
define(`confTO_QUEUEWARN', `')
dnl on ne relaie que les hôtes ????
FEATURE(relay_hosts_only)
dnl on utilise un fichier d'alias
define(`confCW_FILE', `-o /etc/mail/sendmail.cw')
dnl ????
define(`confCON_EXPENSIVE', `True')
dnl ????
define(SMTP_MAILER_FLAGS, e)
dnl les accès smtp sont restreint par ce fichier
FEATURE(access_db, `hash -o /etc/mail/access.db')
dnl les domaines génériques sont dans :
GENERICS_DOMAIN_FILE(`/etc/mail/genericsdomain')
dnl on va changer les adresses venant de chez nous par celle de notre provider
FEATURE(redirect)
dnl si y'a pas de nom de domaine c'est :
MASQUERADE_AS(`free.fr')
dnl tous les mails en provenance de gros.tux sont changés
MASQUERADE_DOMAIN(`gros.tux')
dnl tous les mails on a dit !
FEATURE(masquerade_entire_domain)
dnl donc il faut changer l'enveloppe :
FEATURE(masquerade_envelope)
dnl on va utiliser procmail pour trier les mails des utilisateurs :
FEATURE(`local_procmail',`/usr/bin/procmail')
dnl on mail en local
MAILER(local)
dnl en smtp (??)
MAILER(smtp)
dnl et via procmail
MAILER(procmail)
dnl notre serveur de mail relais est : smtp.free.fr
define(RELAY_HOST, smtp:smtp.free.fr)
define(SMART_HOST, smtp:smtp.free.fr)
dnl il utilise TCP/IP ;)
define(RELAY_MAILER, TCP)
dnl  cela n'est utile que pour les portables qui ont des adresses parfois insolubles
FEATURE(`accept_unresolvable_domains')
```

Ce fichier n'est pas utilisable par sendmail. Mais la lecture (et la rédaction) d'un fichier de configuration de sendmail ayant, paraît−il, causé la folie de plusieurs spécialistes de sendmail, je ne vais pas vous l'imposer. Ce fichier sert justement à créer le véritable fichier de configuration de sendmail : `sendmail.cf`, qui est créé par :

```
$ m4 sendmail.mc > /etc/sendmail.cf
```

(enfantin), évidemment install−sendmail fait tous les choix convenables pour nous, mais au cas ou certains choix serait inappropriés, vous savez quoi faire : éditez `sendmail.mc` (après avoir lu la doc de sendmail) et régénérez `sendmail.cf`.

Il n'est absolument pas envisageable de créer de toute pièce un fichier `/etc/sendmail.cf` complet, par compte, on peut envisager de modifier légèrement celui−ci.

Cas de `Sendmail 8.11.6`

Cette version de sendmail (et peut−être les précédantes mais ça je n'en suis pas sur) permet de ne pas préciser le relai SMTP en mettant dans `sendmail.mc` la règle :

```
FEATURE(`relay_based_on_MX')dnl
```

et d'enlever les règles :

```
define(RELAY_HOST,...)
define(SMART_HOST,...)
```

puis en régénérant `sendmail.cf` par

```
m4 /usr/share/sendmail-cf/m4/cf.m4 sendmail.mc > sendmail.cf
```

de cette façon, sendmail se basera sur le champ MX de votre DNS (ie: celui de votre FAI:fournisseur d'accès internet) pour le relai des mails : c'est très utile dans le cas ou vous avez plusieurs fournisseurs d'accès ça évite d'utiliser un script perl pour rechercher/modifier la ligne fautive dans `sendmail.mc`.

Configuration de fetchmail à la main

La configuration de fetchmail est beaucoup plus compréhensible. Voilà mon fichier `/root/.fetchmailrc` (commenté, mots de passe remplacés par xxxx) :

```
# Configuration created Sat Mar 11 11:52:47 2000 by fetchmailconf
# le postmaster c'est postmaster ;),
# en fait dans /etc/aliases, il y a un alias : postmaster -> root
set postmaster "postmaster"
# pour que les erreurs de distributions aillent à l'envoyeur :
set bouncemail
set properties ""
# pour récupérer les mails sur pop.free.fr en pop3 :
poll pop.free.fr with proto POP3
# le nom du compte sur pop.free.fr est "sympa",
# son mot de passe est xxxxx il s'appelle fred sur ce pc :
  user "sympa" there with password "xxxxx" is fred here warnings 3600
# on n'accepte pas les mails repérés comme des spams :
    antispam 471 571 550 501 554
# un autre compte sur le même serveur pop :
  user "machin" there with password "xxxxx" is bibi here warnings 3600
    antispam 471 571 550 501 554
# d'autres comptes sur un autre serveur pop
poll mail.netcourrier.com with proto POP3
  user "topsympa" there with password "xxxxx" is fred here warnings 3600
    antispam 471 571 550 501 554
  user "machin" there with password "xxxxx" is bibi here warnings 3600
    antispam 471 571 550 501 554
# d'autres comptes sur un autre serveur pop
poll pop.wanadoo.fr with proto POP3
  user "sympa1" there with password "xxxxx" is fred here warnings 3600
    antispam 471 571 550 501 554
```

Pour pouvoir utiliser un tel fichier de configuration, il faut impérativement avoir le droit d'écrire dans le compte mail (`/var/spool/mail`) de tous les utilisateurs indiquer (ici `bibi` et `fred`) : le mieux est d'être root, et donc que le fichier ci-dessus soit en fait : `/root/.fetchmailrc` !

Il est bien entendu possible à chaque utilisateur de se créer un fichier `/home/user/.fetchmailrc` (même syntaxe) pour récupérer son courrier sur des comptes non connus du root. Mais il devra récupérer lui-même son mail. On pourrait imaginer, à chaque connexion de créer le fichier `/root/.fetchmailrc` (pour le compte de root) avec les fichiers `.fetchmailrc` des différents utilisateurs du système, mais cela permettrait à n'importe qui de mettre des mails sur le compte de n'importe qui d'autre... Une autre solution serait d'utiliser "`su user -c fetchmail`" pour chaque utilisateur, à chaque connexion (sans doute une bonne solution si l'on a beaucoup d'utilisateurs). La solution proposée ici, est pratique si seulement quelques personnes on accès à un serveur (genre : moins de 10).

Pour ceux que install-sendmail rebute, qui veulent configurer fetchmail séparément, ou qui veulent ajouter des comptes pop, il existe un programme de configuration graphique relativement compréhensible : `fetchmailconf` (pensez à installer le paquetage correspondant).

Utilisation

Sendmail

C'est très simple. Tant que vous n'êtes pas connecté à Internet, tous vos mails sont mis en attente sans que vous ayez rien de particulier à faire. A la connexion, par contre, vous devez envoyez tous les mails en attente en utilisant la commande :

```
/usr/sbin/sendmail -q
```

Avant d'utiliser cette commande, vous pouvez vérifier qu'effectivement vous avez des mails en attente par :

```
mailq
```

Fetchmail

Pour recevoir les mails des différents comptes pop que vous avez configuré, depuis le compte de root, faites:

```
fetchmail -v
```

(l'option −v n'est pas obligatoire, mais elle permet de voir que tout va bien). Et c'est tout.

Conclusion

Évidemment, tout cela n'est pas très pratique. Nous allons voir dans l'article suivant comment automatiser ce que nous venons d'expliquer.

Connexion à Internet multi−comptes
Partie 3 : Automatisation partielle (ip−up et ip−down)

par Fred

Jusqu'à maintenant nous avons vu comment nous connecter à Internet et comment recevoir nos mail, mais la connexion reste une procédure complexe pour l'utilisateur moyen.

Rappel des épisodes précédents

Nous savons nous connecter à Internet par la commande :

```
/usr/sbin/pppd call connexion0
```

Nous savons nous déconnecter d'Internet par la commande :

```
kill `cat /var/run/ppp*.pid`
```

qui doit être lancée par l'utilisateur qui a établi la connexion.

Nous savons récupérer nos mails par la commande :

```
fetchmail −v
```

qui, elle, doit être lancée depuis le compte de root.

Nous savons comment envoyer nos mails en attente :

```
/usr/sbin/sendmail −q
```

Comment faire pour automatiser 'tout' cela ?

ip−up

Lorsque la connexion est s'établit, `pppd` lance le script (s'il existe) : `/etc/ppp/ip−up`. Donc, pour que quelque chose se passe lorsque vous vous connectez, il suffit de créer un script ayant ce nom.

Souvant, votre distribution fournit déjà un tel script qui se charge d'executer `/etc/ppp/ip−up.local`. Si c'est le cas (lisez le contenu du script) vous pouvez pour faire ce qui va suivre :

- ou bien effacer le fichier `/etc/ppp/ip−up` préexistant ainsi que `/etc/ppp/ip−up.local` et créer celui que je vais vous indiquer (et que j'espère vous comprendrez) ; plutôt que d'effacer, renommez ces fichiers avec par exemple un suffixe `.bak`, cela vous permettra de revenir en arrière le cas échéant.
- ou bien effacer `/etc/ppp/ip−up.local` et faire ce que je vais décrire pour `/etc/ppp/ip−up` sur `/etc/ppp/ip−up.local` à la place (c'est plus sûr du point de vue de la distribution, mais il y a des chances pour que vous ne sachiez pas exactement ce que fait `/etc/ppp/ip−up` − moi je n'aime pas ça).

Ces considérations philosophiques étant tenues, passons aux choses sérieuses. Que souhaitons−nous qu'il se passe à chaque connexion ? Je vois plusieurs choses :

- que les mails soient automatiquement rapatriés sur tous les comptes,
- que les mails que nous recevrons pendant que nous sommes connectés soient eux aussi rapatriés sur les comptes idoines,
- que le DNS de notre ISP soit activé,
- que les mails en attente soient envoyés,
- prévenir l'utilisateur que la liaison est effectivement active (il se passe toujours un certain temps avant que la liaison ne s'établisse...).

Comme nous utilisons X Window, nous allons faire en sorte d'utiliser des boîtes de dialogues en mode graphique, cela fera plus 'pro'. Comme nous ne sommes pas des programmeurs avertis, nous allons nous contenter du langage `/bin/sh`, c'est à dire le shell `bash` ! Pour lui faire afficher des boites de dialogue, je vous propose d'utiliser l'excellent `Xdialog` : un utilitaire qui permet d'ouvrir une boîte de dialogue depuis un script shell (installation par : `./configure −−prefix=/opt/apps ; make ; make install`).
Je n'expliquerai pas comment il fonctionne (essentiellement comme `dialog`, pour connaître les options tapez : `/opt/apps/bin/Xdialog`).

Voilà ce que l'on peut mettre dans `/etc/ppp/ip−up` :

```
#!/bin/sh

# le paramètre $6 correspond au paramètre
# ipparam du fichier /etc/ppp/peers/script

# −−−Configuration du DNS−−−Début−−−
# ceci nécessite que l'option usepeerdns
# soit activée dans /etc/ppp/peers/script
```

```
# dans ce cas DNS1 et DNS2 reçoivent les
# adresse des deux dns du provider.
# sinon mettez ici vos deux dns :
#DNS1=123.123.123.123
#DNS2=213.213.213.213
echo "# fichier généré par ip−up." > /etc/resolv.conf
echo search org >> /etc/resolv.conf
echo nameserver $DNS1 >> /etc/resolv.conf
echo nameserver $DNS2 >> /etc/resolv.conf
# −−−Configuration du DNS−−−Fin−−−

# récupération des mails :
/usr/bin/fetchmail
# et envoie des mails en attente :
/usr/sbin/sendmail −q

# on récupère automatiquement les mails toutes les 11 minutes
# (11 minutes car netcourrier n'aime pas qu'on consulte
# les comptes pop trop souvent !)
/usr/bin/fetchmail −−daemon 660

# le plus de la semaine :
# mettre son pc à l'heure :
ntpdate ntp−sop.inria.fr

# on est connecté, on le dit :
export DISPLAY=:0
/opt/apps/bin/Xdialog −−title "Information" −−center\
−−infobox "La connexion est maintenant active." 0 0 3000
```

Ce script n'appelle pas de commentaire particulier, sinon que pour que le PC puisse être mis à l'heure il faut que le paquetage `xntp3` soit installé.

Il faut savoir, avant de mettre n'importe quoi dans ce fichier, que celui−ci est toujours exécuté avec les droits de **root**, donc prudence. De plus ce script est exécuté avec un PATH minimal (`PATH=""`), donc vous devez préciser le chemin complet des exécutables.

ip−down

De même que `pppd` lance `ip−up` à la connexion, il lance `/etc/ppp/ip−down` lorsque la connexion vient d'être interrompue. Lorsque la connexion est rompue, nous pouvons stopper le daemon `fetchmail` (en effet `fetchmail` ne peut plus que retourner des messages d'erreur), de plus il faut remettre le fichier `/etc/resolv.conf` à sa valeur par défaut. Voici le script `/etc/ppp/ip−down` que j'utilisais :

```
#!/bin/sh

# comme on n'est plus connecté au net, on quitte fetchmail
/usr/bin/fetchmail −−quit

# −−−Configuration du DNS−−−Début−−−
echo "# fichier généré par ip−down." > /etc/resolv.conf
# mon domaine local est tux !
echo search tux >> /etc/resolv.conf
# −−−Configuration du DNS−−−Fin−−−

# on est déconnecté, on le dit :
export DISPLAY=:0
/opt/apps/bin/Xdialog −−title "Information" −−center\
−−infobox "La connexion est maintenant inactive." 0 0 3000
```

Les mêmes commentaires de prudence que pour `ip−up` s'appliquent à `ip−down`.

Script de connexion/déconnexion

Pour se connecter, il n'est pas nécessaire de créer un script, mais nous allons en créer un, pour que la procédure de connexion soit la même que la procédure de déconnexion. Voilà les deux scripts :

- `/etc/ppp/scripts/pppconnect`

```
#!/bin/sh
/usr/sbin/pppd call connexion0
```

- `/etc/ppp/scripts/pppdisconnect`

```
#!/bin/sh
kill  `cat /var/run/ppp*.pid`
```

Conclusion

Bon, jusqu'à maintenant nous n'avons pas fait grand−chose pour la connexion multicomptes. C'est l'objet de la prochaine partie.

Connexion à Internet multi−comptes
Partie 4 : connexion multi−comptes

par Fred

Nous allons, dans ce chapitre, voir comment il est possible de rendre "simple" pour l'utilisateur la gestion de plusieurs connexions (plusieurs ISP) à internet

Pré requis, Introduction

Pour pouvoir comprendre ce chapitre, vous devez savoir comment établir une connexion à internet, comment récupérer du courrier d'internet sur plusieurs comptes en même temps et savoir quels sont les scripts qui sont lancés lors de l'établissement de la connexion. Tous ces points sont le sujet des articles précédents (voir les parties 1, 2 et 3).

Avec la prolifération actuelle des ISP, notamment les gratuits, nous sommes souvent tentés de nous connecter via un nouvel ISP, ne serait−ce que pour l'essayer (vérifier l'état de sa ligne, sa rapidité, etc.).

En utilisant ce que nous avons vu, il n'est pas très difficile de créer plusieurs scripts de connexion à Internet. Mais cette méthode oblige l'administrateur à modifier ou créer plusieurs scripts (en tout au moins 4 par fournisseur d'accès) pour pouvoir ajouter un nouvel ISP. Ce n'est pas très souple. De plus il nous faut modifier pour chaque ISP le fichier `/etc/sendmail.cf` ; en effet dans celui−ci nous avons mis l'adresse de notre serveur de mail sortant, mais le serveur de mail sortant n'accepte pas en général de connexion depuis un PC qui n'est pas relier localement au réseau "local" de l'ISP. Il est assez évident que la modification de ce fichier devra se faire depuis `/etc/ppp/ip−up`, car elle doit se faire à chaque connexion.

Note : tout ce qui suit, sauf indication contraire, est exécuté avec les droits de root.

Le principe retenu

Voici la solution que j'ai retenue. Toutes les informations relatives à toutes les connexions sont stockées dans le même et unique fichier : `/etc/ppp.conf`. Le fichier est en ASCII pur pour que l'on puisse l'éditer facilement.

L'établissement de la connexion à Internet se fera par un seul script : `/etc/ppp/scripts/pppconnect`. Celui−ci devra ouvrir une boite de dialogue dans laquelle on pourra choisir l'une quelconque des connexions configurées dans `/etc/ppp.conf`.

La rupture de la liaison se fera via le script `/etc/ppp/scripts/pppdisconnect`.

Syntaxe du fichier `/etc/ppp.conf`

Comme un dessin vaut mieux qu'un long discours, voici mon fichier `/etc/ppp.conf` :

```
{free name}Free (max 20h)
{free smtp}smtp.free.fr
{free search}free.fr
{free autodns}
{free login}yyyyyyyyyy
{free password}xxxxxxxx
{free telephone}0359602000

{wanadoo name}wanadoo (max 36h)
{wanadoo smtp}smtp.wanadoo.fr
{wanadoo search}wanadoo.fr
{wanadoo dns}193.252.19.3
{wanadoo dns}193.252.19.4
{wanadoo login}yyyyyyyyyy
{wanadoo password}xxxxxxxx
{wanadoo telephone}0860888080

{offline smtp}
{offline search}tux

{global modeminit}ATM0L0
```

Chaque ligne de ce fichier est de la forme :

```
{ident motclef}valeur
```

`ident` peut être n'importe quoi (du moment que ça ne contient pas d'espace), mais il y a deux valeurs particulières :

- `offline` : pour indiquer que les paramètres qui suivent ne s'appliquent que lorsque la connexion est inactive.
- `global` : pour indiquer que ces paramètres s'appliquent pour toutes les connexions (même `offline`).

`motclef` est une valeur parmi :

Mot clef	Type de la valeur	Utilisation
`name`	chaîne de caractère quelconque	le nom de la connexion `ident`
`dns`	une adresse IP	adresse d'un DNS (serveur de noms de domaine), vous pouvez en mettre autant que vous le souhaitez (sur des lignes distinctes).
`autodns`	rien	si vous précisez cette option pour une connexion (ou pour toutes via `global`), `pppd` demandera à votre ISP les adresses de deux DNS. Dans ce cas (que cela fonctionne ou pas) les DNS que vous précisez par l'option `dns` sont ignorés.
`search`	nom de domaine (valide ?)	un nom de domaine qui sera ajouté automatiquement aux noms qui n'auront pu être résolus sans celui−ci.
`smtp`	nom d'un serveur smtp ou une adresse IP	ce nom doit alors être le nom du serveur de mails *sortant* de l'ISP correspondant à cette connexion.
`login`	chaîne de caractère	le login correspondant à cet ISP.
`password`	chaîne de caractère	le mot de passe de ce login (attention au propriétaire de ce fichier : il doit être en clair !)
`telephone`	numéro de téléphone	de l'ISP
`modeminit`	chaîne de caractère	ce doit alors être une chaîne d'initialisation de votre modem renvoyant 'OK' ; vous pouvez avoir autant de chaîne d'initialisation que vous le souhaitez.

Attention : le parseur distingue les majuscules des minuscules.
Attention : l'ordre des différentes lignes de ce fichiers sont essentielles : elles détermine la validité des diverses options (c'est à dire : si vous préciser deux fois la même option, seule la dernière valeurs est prise en compte).

/etc/ppp/scripts/initcnx

Maintenant, il nous faut un script capable de lire un tel fichier, et de générer les fichiers nécessaires à la connexion à Internet :

- `/etc/ppp/chat/auto` : le script de numérotation de l'isp.
- `/etc/ppp/peers/auto` : le script de commande de pppd.
- `/etc/ppp/pap−secrets` : le fichier contenant les mots de passe pour l'isp.
- `/etc/resolv.conf` : le fichier charger de la résolution des noms (association adresse IP <−> nom de machine).

Voici un tel script (appelez le `/etc/ppp/scripts/initcnx`) :

```
#!/usr/bin/perl

# lecture du fichier de configuration /etc/ppp.conf pour
# la création des paramètres de configurations !
# /etc/ppp/chat/auto
# /etc/ppp/peers/auto
# /etc/ppp/pap−secrets
# /etc/resolv.conf

($cnx,$ipup) = @ARGV ;

open(PPPCONF,"/etc/ppp.conf") ;

@modeminit = () ;
@resolv = () ;
$search = "org" ;
$autodns = 0 ;

while (<PPPCONF>) {
 if (($p) = /{$cnx login}(.*)/) { $login = $p ; }
 if (($p) = /{global login}(.*)/) { $login = $p ; }
 if (($p) = /{$cnx password}(.*)/) { $password = $p ; }
 if (($p) = /{global password}(.*)/) { $password = $p ; }
 if (($p) = /{$cnx smtp}(.*)/) { $smtp = $p ; }
 if (($p) = /{global smtp}(.*)/) { $smtp = $p ; }
 if (($p) = /{$cnx telephone}(.*)/) { $telephone = $p ; }
 if (($p) = /{global telephone}(.*)/) { $telephone = $p ; }
 if (($p) = /{$cnx modeminit}(.*)/) { push(@modeminit,$p) ; }
 if (($p) = /{global modeminit}(.*)/) { push(@modeminit,$p) ; }
 if (($p) = /{$cnx dns}(.*)/) { push(@resolv,$p) ; }
 if (($p) = /{global dns}(.*)/) { push(@resolv,$p) ; }
 if (($p) = /{$cnx search}(.*)/) { $search = $p ; }
 if (($p) = /{global search}(.*)/) { $search = $p ; }
```

```
 if (($p) = /{$cnx autodns}(.*)/) { $autodns = 1 ; }
 if (($p) = /{global autodns}(.*)/) { $autodns = 1 ; }
}

if ($autodns) {
 $resolv = () ;
}
if ($ENV{"DNS1"} ne "") {
 push(@resolv,$ENV{"DNS1"}) ;
}
if ($ENV{"DNS2"} ne "") {
 push(@resolv,$ENV{"DNS2"}) ;
}

open(CHAT,">/etc/ppp/chat/auto") ;

print CHAT "'ABORT' 'BUSY'
'ABORT' 'ERROR'
'ABORT' 'NO CARRIER'
'ABORT' 'NO DIALTONE'
'ABORT' 'Invalid Login'
'ABORT' 'Login incorrect'
'' 'ATZ'
'OK' " ;
foreach (@modeminit) { print CHAT "'$_'\n'OK' " ; }
print CHAT "'ATDT$telephone'
'CONNECT' ''
'TIMEOUT' '5'
'~--' ''
" ;
close CHAT ;

open (PEERS,">/etc/ppp/peers/auto");
print PEERS "connect '/usr/sbin/chat -v -f /etc/ppp/chat/auto'
user $login
ipparam $cnx
" ;
if ($autodns) {
 print PEERS "usepeerdns\n" ;
}
close PEERS ;

open (PAP,"/etc/ppp/pap-secrets") ;
$file = "" ;
while (<PAP>) { $file .= $_ ; }
close PAP ;
open (PAP,">/etc/ppp/pap-secrets") ;
# on supprime les occurrences précédentes :
$file =~ s/#AUTOBEGIN#.*#AUTOEND#//s ;
if ($login ne "") {
 $file .= "#AUTOBEGIN#\n$login * $password\n#AUTOEND#";
}
print PAP $file ;
close PAP ;

open (SENDSRC,"/etc/sendmail.BASE") ;
open (SENDDST,">/etc/sendmail.cf") ;
while (<SENDSRC>) {
 s/{MAILHOST}/$smtp/g ;
 print SENDDST ;
}
close SENDSRC ;
close SENDDST ;

# on modifie le ns :
open (RESOLV,">/etc/resolv.conf") ;
print RESOLV "search $search\n" ;
foreach $nameserver (@resolv) {
 print RESOLV "nameserver $nameserver\n" ;
}
close RESOLV ;

# on relance sendmail (seulement si
# c'est ip-up ou ip-down qui à lancé le
```

```
# script, en effet il faut être root) :
if ($ipup =~ /ip(up|down)/) {
  `/etc/rc.d/init.d/sendmail restart`
} ;
```

Ce script est un compromis : je ne voulais pas écrire plusieurs fois la lecture de `/etc/ppp.conf` (je ne sais pas écrire de bibliothèque en `perl` !) pour des raisons évidentes de maintenance, donc j'ai écrit ce script qui fait, pour chaque opération, bien plus que ce qui est nécessaire. Mais cela ne doit être pénalisant que si vous avez des dizaines de connexions ;).

Le script n'appel pas de commentaire particulier, si ce n'est pour les 12 lignes en gras. Ce sont ces lignes qui vont demander à l'isp des adresses de ses dns. Les 9 premières regardent si les variables d'environnement `DNS1` et `DNS2` existent (quand on est en mode `autodns`). Les 3 dernières ajoutent au script de commande de pppd l'option qui lui réclame ces information.

Pour que ce script puisse fonctionner, il faut que vous créiez un fichier `/etc/sendmail.BASE` dont le nom de serveur smtp est `{MAILHOST}` (à la place du nom du serveur de mail sortant d'un de vos isp). Si votre serveur de sendmail est configuré et fonctionne, il suffit de copier `/etc/sendmail.cf` dans `/etc/sendmail.BASE` et cherchez dans `/etc/sendmail.BASE` la ligne (ou quelque chose de semblable...) :

DSsmtp:**smtp.isp.fr**

et vous la changez en :

DSsmtp:**{MAILHOST}**

Attention : le fichier `/etc/sendmail.cf` est très sensible, ne le modifiez que si vous savez ce que vous faites.

Un fichier /etc/ppp/ip−up à peine plus complexe

(que celui de la partie 3)

```
#!/bin/sh
# le paramètre $6 correspond au paramètre ipparam
# du fichier /etc/ppp/peers/script

# ceci nécessite que l'option usepeerdns
# soit activée dans /etc/ppp/peers/script
# dans ce cas DNS1 et DNS2 reçoivent les
# adresse des deux dns du provider.
# dans le cas où ces variables ne sont pas
# initialisées,  les options {connexion dns}***
# doivent être utilisées pour préciser le dns !
# en lieu et place de autodns
export DNS1
export DNS2
/etc/ppp/scripts/initcnx $6 ipup

# récupération des mail et envoie des mails en attente :
/usr/bin/fetchmail
/usr/sbin/sendmail −q

# on récupère automatiquement les mails toutes les 11 minutes
# (11 minutes car netcourrier n'aime pas qu'on consulte
# les comptes pop trop souvent !)
/usr/bin/fetchmail −−daemon 660

# mettons notre pc à l'heure :
ntpdate ntp−sop.inria.fr
```

Vu ce que l'on a déjà dit, ce script ce passe de commentaire.

Un fichier /etc/ppp/ip−down

```
#!/bin/sh

# comme on n'est plus connecté au net, on quitte fetchmail
/usr/bin/fetchmail −−quit
# y'a plus de dns a priori...
export DNS1=""
export DNS2=""
/etc/ppp/scripts/initcnx offline ipdown
```

```
# on peut effacer les fichiers auto :
rm -f /etc/ppp/chat/auto
rm -f /etc/ppp/peers/auto
```

Idem.

/etc/ppp/scripts/pppconnect

Pour ce script, nous allons un peu compliquer les choses : en effet, c'est lui qui doit permettre le choix de l'isp que nous souhaitons contacter. Comme, dans la partie 3, nous allons utiliser Xdialog pour nous permettre ce choix. Voici le script :

```
#!/usr/bin/perl

open(PPPCONF,"/etc/ppp.conf") ||
`/opt/apps/bin/Xdialog --title Erreur --msgbox\
"impossible d\'ouvrir /etc/ppp.conf\nchangez\
les droits de ce fichier\nou bien créez le."\
0 0` ||
die "/etc/ppp.conf illisible.\n" ;

# on cherche les lignes : {connexion name}isp
while (<PPPCONF>) {
 if (($tag,$name) = /{(.*?) name}(.*)/) {
  $cnx{$tag} = $name ;
 }
}

$cmdline = "" ;
$num = 0 ;

foreach $key (sort keys %cnx) {
 $cmdline .= "\"$key\" \"$cnx{$key}\" false " ;
 $num++ ;
}

$cnx = `/opt/apps/bin/Xdialog --stdout\
--radiolist Connexion 0 0 $num $cmdline` ;
chop($cnx);

if ($cnx ne "") {

 # initialisation (sauf le dns !) qui est fait par
 # ip-up
 `/etc/ppp/scripts/initcnx $cnx` ;

 # connexion !
 `/usr/sbin/pppd call auto` ;

 # ce qui suit n'est pas nécessaire, mais
 # agréable :
 # attendons l'établissement de la liaison
 `sleep 60s`
 # et rapatrions les mails de l'utilisateur
 # pour peu qu'il ai configuré fetchmail
 `fetchmail`
 # ceci permet à un utilisateur de se configurer
 # un compte que le root ne connait pas.
} else {
`/opt/apps/bin/Xdialog --title Information\
--infobox "Aucune connexion n'a été sélectionnée."\
0 0 1500` ;
}
```

(note(pour jcc), je ne me rappelle plus si perl comprend la continuation des lignes par \ ?)
(note : saisissez les lignes terminés par \ sur une seule ligne !)
(note : en lieu et place de /opt/apps/bin précisez le path complet de Xdialog)
Vous pouvez, dès maintenant, vérifier que l'établissement est fonctionnel (si vous avez un fichier `/etc/ppp.conf`) en tapant : `/etc/ppp/scripts/pppconnect`. Si vous aviez des mails en attente et si vous aviez configuré fetchmail, ce script doit établir la connexion, envoyer les mails en attente, et rapatrier vos mails (et ceux de vos utilisateurs).

/etc/ppp/scripts/pppdisconnect

La déconnexion est semblable à ce que nous avons déjà fait :

```
#!/bin/sh
if [ −e /var/run/ppp*.pid ] ; then
 kill `cat /var/run/ppp*.pid` 2>&1 > /dev/null
fi
/opt/apps/bin/Xdialog −−title "Informations:"\
−−infobox "La liaison internet est maintenant\
coupée." 0 0 2000
```

Sans commentaire.

Les permissions

Pour l'instant, cet ensemble de script, ne fonctionne que si vous êtes root. C'est un peu limitatif. Pour arranger les choses, je vous conseille de créer un groupe pppusers auquel vous ferez appartenir les utilisateurs ayant le droit de choisir leur connexion. Puis, dans le répertoire /etc/ppp faites :

pour faire appartenir tout ce que contient /etc/ppp à pppusers :
`chown −R root:pppusers /etc/ppp/*`
pour autoriser les membres du groupe pppusers à modifier les fichiers /etc/ppp/peers/auto etc. :
`chmod −R g+rwx /etc/ppp/scripts/ /etc/ppp/chat /etc/ppp/pap−secrets`
pour que les étrangers le reste :
`chmod −R o−rwx /etc/ppp/* /etc/ppp.conf /etc/sendmail.BASE`

Attention : seuls les utilisateurs ayant établi la connexion auront le droit de la coupée : ce qui est somme toute assez logique. Le root ayant lui, comme toujours, tous les droits.

Utilisation

Chez moi, j'ai mis deux icônes sur mon bureau, une vers pppconnect et l'autre vers pppdisconnect. Comme ça, je peux établir et couper la liaison à Internet de manière très simple.

Voilà, bon surf.

Reste le problème des spammers ! La suppression automatique des mails dits "spam", mails de publicité non sollicités, fera l'objet d'un prochain article.

Configuration d'une connexion ADSL

par Serge (révisions par Jice)

L'accès internet haut débit sous Linux.

Introduction

Il existe trois protocoles différents pour les connexions ADSL : PPTP (Point to Point Tunneling Protocol), PPPOE (PPP Over Ethernet : modems ethernet...) et PPPOA (PPP over ATM : modems USB...). Les kits founis jusqu'au début de l'année 2001 environ (par France Télécom ou autre) étaient basés sur PPTP. Maintenant ils sont généralement basés sur PPPOE ou PPPOA.

Si vous avez une debian, lisez plutot ce document.

Ce document traite principalement des modems ADSL avec interface ethernet, pour les modems USB voir à la fin du document.

Il existe aussi des modems PCI pour se connecter avec ADSL.

- Bewan vend une telle carte, mais les pilotes Linux ne sont pas (encore ?) développés et/ou distribués `:-(.` (décembre 2001 : Bewan a sorti une nouvelle carte PCI qui fonctionne sous Linux, malheureusement pas de nouvelles du driver pour les anciennes cartes, malgré les promesses qui ont été faites... − février 2002 : nouvelle version des drivers Bewan PCI ST, voir cette news sur linuxfr pour les liens)
- D'autre cartes PCI (Olitec...) sont sorties aussi, mais je n'ai pas d'info dessus. Merci de nous contacter si vous pouvez nous aider à compléter cet article à ce sujet.

Ce document s'inspire largement du HOWTO ADSL.

Connexion ADSL via le protocole PPTP

Le protocole pptp permet de créer un "tunnel" ppp au travers d'une connexion ethernet.

Vérification des paramètres du noyau

Tout d'abord vérifions la configuration du noyau : vous devez avoir le support PPP soit dans le noyau, soit en module. Une méthode simple est de lancer en root un :

```
pppd
```

Si vous voyez une réponse du style "%!/:." c'est que c'est bon, arrêtez le alors via un CTRL−C, ou ouvrez un autre terminal (ou changez de console virtuelle) et tapez un :

```
killall pppd
```

Autrement recompilez votre noyau avec le support ppp (en natif ou en module). Voir la section noyau.

Installation de PPTP

Bon il va falloir avant tout ajouter le support pptp. Récupérons donc le package comprenant le client pptp patché :
pptp−linux−1.0.2−patched.tar.gzpptp−linux−1.0.2−patched.tar.gz

Pour installer pptp:

```
tar zxf pptp−linux−1.0.2−patched.tar.gz
cd pptp−linux−1.0.2
make install −b pptp pptp_callmgr /usr/sbin/
```

Et voilà, nous avons le support pptp !

Configuration de ppp

Comme je l'ai indiqué plus haut, pptp se sert de ppp et d'une interface ethernet, donc il est logique de configurer ppp. Nous allons donc modifier les fichiers de configuration de ppp, les fichiers `/etc/ppp/options` et `/etc/ppp/pap−secret` :

Pour `/etc/ppp/options` :

```
noauth
name "login" # mettre ici le login que vous a fourni votre provider
noipdefault
defaultroute
mtu 1492
mru 2400
```

Et pour `/etc/ppp/pap−secret` :

```
 "login" * "password" * # login et password fourni par votre provider, à mettre entre guillemets.
```

Configuration de l'interface ethernet

Il nous reste maintenant à configurer l'interface ethernet qui est reliée au modem ADSL. Je n'explique pas comment la prendre en charge par le noyau, voir la rubrique connexion réseau local de cette même section pour ça. Je suppose que cette interface est `eth0`, adaptez suivant votre configuration bien sûr.

Affectons l'adresse IP **10.0.0.10** à cette interface, attention à **bien mettre une adresse de ce réseau IP privé et pas d'un autre !**

```
ifconfig eth0 10.0.0.10
```

Et voilà !

N'oubliez pas non plus de configurer votre `/etc/resolv.conf` pour qu'il contienne les adresses de serveurs DNS (serveurs de nom) de votre fournisseur d'accès :

`/etc/resolv.conf` :

```
search nom_domain_local.com
nameserver adresse_ip_dns_primaire
nameserver adresse_ip_dns_secondaire
```

Création des scripts de connexion / déconnexion

Nous allons créer maitenant les scripts de connexion / déconnexion.

Script de connexion : appelons le `adsl-start`, il contient :

```
#!/bin/bash
/usr/sbin/pptp 10.0.0.138
sleep 20
echo "Ajout de la route par défaut : "
/sbin/route add -net 0.0.0.0 netmask 0.0.0.0 dev ppp0
```

Et un `adsl-stop`

```
#!/bin/bash
killall pptp
sleep 1
killall pppd
rm -rf /var/run/pptp/
```

Il suffit alors de lancer les scripts pour se connecter / déconnecter. Vous pouvez aussi mettre dans `/etc/rc.local` une ligne appelant `adsl-start` pour lancer la connexion au boot de la machine. Ou alors faites un script pour gérer la connexion/déconnexion en sysVinit.

Connexion ADSL via le protocole PPPOE

Connexion via le logiciel fourni par FT

France Télécom vous fournit normalement un CD contenant le support pppoe pour votre modem sous Linux. Comme ce logiciel n'est ni libre ni gratuit, je ne détaille pas ici comment s'en servir. Sachez juste qu'il faut en gros copier le binaire `pppoed-2.x` (x dépendant de la version de votre kernel, 2.0.x ou 2.2.x) dans `/usr/sbin`, ainsi que les scripts `ppoe-start` et `ppoe-stop`, et le fichier de configuration `ppoe.options` dans `/etc`. Un fichier d'aide est fourni avec ce logiciel.

Installation de rp-pppoe

Comme nous sommes dans le monde du logiciel libre, nous allons voir comment utiliser votre connexion ADSL PPPOE via le support libre sous Linux : `rp-pppoe`.

Téléchargez celui-ci à cette adresse: http://www.roaringpenguin.com/pppoe/ .

J'explique ici son installation à partir des sources tarball (tar.gz) car cela est applicable à toutes les distributions. On détarre / dégzip les sources dans un répertoire de votre choix et on lance le script go qui s'occupe de tout :

```
# tar xvfz rp-pppoe-3.3.tar.gz
# cd rp-pppoe-3.3
# ./go
```

À partir de ce moment, répondez juste aux questions que l'on vous pose.

En cas de problème avec le script go, vous pouvez essayer de compiler / installer "à la main" (comme dans les anciennes versions de rp-pppoe) :

```
# tar xvfz rp-pppoe-2.8.tar.gz
# cd rp-pppoe-2.8
# ./configure
# make && make install
# /usr/sbin/adsl.setup
```

Vérifiez après dans le fichier `/etc/pppoe.conf` les lignes :

```
ETH=ethx # mettre l'interface sur laquelle est reliée votre modem ADSL
USER=login # mettre le login fourni par le FAI
```

Configuration de la carte ethernet

Il ne faut pas cette fois ci configurer votre carte ethernet, c'est à dire que vous devez quand même charger le module de votre carte mais **n'affectez pas d'adresse à cette carte**. C'est les scripts de connexion qui s'en chargeront.

N'oubliez pas non plus de configurer votre `/etc/resolv.conf` pour qu'il contienne les adresses de serveurs DNS (serveurs de nom) de votre fournisseur d'accès :

`/etc/resolv.conf` :

```
search nom_domain_local.com
nameserver adresse_ip_dns_primaire
nameserver adresse_ip_dns_secondaire
```

Configuration de ppp

Il nous reste à configurer ppp. Editez `/etc/ppp/pap-secrets` et ajoutez ces deux lignes :

```
nom_login * password # login et password fournis par le provider
netissimo@netissimo.fr * netissimo
```

Copiez aussi ce fichier en `chap-secrets` (`cp pap-secrets chap-secrets`)

Connexion

Pour lancer alors la connexion : `adsl-start`
Pour la stopper : `adsl-stop`

Vous pouvez là aussi mettre dans votre `/etc/rc.local` un appel a `adsl-start` pour lancer votre connexion ADSL dès le boot du PC, ou ecrire un petit script pour les lancer via l'init sysV (aidez vous des scripts existants !).

Modem ADSL USB Alcatel Speedtouch.

Ce modem fonctionne grâce à un driver opensource. L'installation est extrêmement simple sur une Mandrake 8.x notamment (voir cet article) et doit bien se passer aussi sur les autres distributions (voir l'article sur l'Alcatel Speedtouch USB).

Alcatel fournit aussi des drivers Linux pour ses modems USB : http://www.alcatel.com/consumer/dsl/dvrreg_lx.htm, mais ils sont moins stables que le driver opensource (à préférer). On les récupère juste pour avoir le microcode du modem.

Modem ADSL USB ECI.

Décembre 2001 : un driver opensource pour ce driver est également sorti. Voir http://eciadsl.flashtux.org. Il est maintenant assez stable pour pouvoir être utilisé.

Voici la liste des modems supportés par ce driver (modems basés sur le chipset Globespan).

Un article sur Léa explique maintenant comment installer ce modem.

Modem ADSL Bewan PCI St.

Voir l'article correspondant sur Léa.

Firewall

par Serge (légèrement modifié par Jicé)

Protégez vous derrière un mur de feu.

Ce document a été inspiré du Firewall−HOWTO, de l'IPCHAINS−HOWTO et du IP−Masquerade−HOWTO. C'est la synthèse de ces trois documents pour permettre la mise en place des techniques de Firewalling. Je vous conseille quand même de lire ces HOWTO pour plus de compréhension. Je suppose que vous savez configurer un réseau IP, au cas où jeter un coup d'oeil sur la rubrique réseau local. Il se peut que des erreurs se soient glissées dans cette présentation, de plus je ne pourrai être tenu responsable d'une mauvaise configuration de votre système. La sécurité est un sujet vaste, toujours en évolution et je ne prétends pas vous fournir ici une solution exempte de tout danger. Ce document est à prendre comme une présentation de ces techniques.

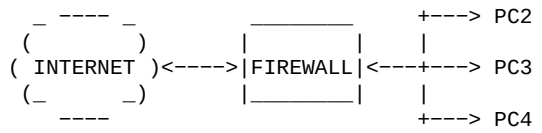
ATTENTION: rubrique non finie, le filtrage plus fin et les techniques de proxy ne sont pas encore décrits.

Explications

Bon qu'est un Firewall ? Si on traduit, cela nous donne « pare−feu » , vous allez me dire « ha bon mon micro me protège du feu ? », bien sûr que non, un Firewall sert en fait à protéger un réseau vis à vis d'un autre ou d'autre réseaux.

Plusieurs types de Firewall existent et ne travaillent pas de la même façon et n'offrent pas les mêmes sécurités.

Les Firewall sont de plus en plus utilisés pour protéger les réseaux locaux d'entreprises vis à vis de l'internet. Il s'agit en fait d'une machine qui est reliée vers l'extérieur (Internet) et vers le réseaux local aussi et qui en quelque sorte analyse le trafic réseau qui la traverse pour savoir si oui ou non elle laisse passer ce trafic que ça soit dans un sens ou dans l'autre :



Les différents types de Firewall

Le Firewall le plus simple est une machine qui possède une connexion vers l'extérieur et une autre sur le réseau local et qui ne transmet pas le trafic réseau d'un réseau à l'autre. Si une machine du réseau local veut accéder vers l'extérieur, elle ouvre en fait une session sur le Firewall, et travaille directement sur cette machine pour accéder vers l'extérieur. De ce fait aucun trafic réseau de l'extérieur ne peut rentrer sur le réseau local.

Vous avez compris les problèmes que cela engendre :

- on travaille directement sur le Firewall donc chaque utilisateur qui va par exemple « naviguer » sur le web vas lancer un nouveau processus « navigateur »,
- on n'a pas accès directement à sa machine avec ses ressources mais à celle du Firewall, et la protection s'arrête juste a une authentification d'utilisateur.

Ce type de Firewall en fait ne filtre absolument rien et peut donc être sujet à la moindre attaque extérieure. En fait il n'est plus utilisé, on va dire que c'est le « Firewall préhistorique » en quelque sorte. Voyons maintenant les types de Firewall utilisés :

Les Firewalls à filtrage de paquets

Ils travaillent sur les paquets réseaux eux−même. Pour les personnes qui ont des connaissances réseaux, ce type de Firewall travaille sur la couche réseau du modèle OSI. Ils analysent les paquets entrants/sortants suivant leur type, leurs adresses source et destination et les ports. Comme ils travaillent directement sur la couche IP, ils sont très peu gourmands en mémoire.

Avec Linux ce style de filtrage au niveau de la couche IP est intégré directement dans le noyau, il suffit donc d'avoir un 486 avec 8 Mo de mémoire et d'une distribution Linux avec juste un kernel de base et la couche IP pour faire un tel Firewall.

De plus ils sont totalement transparents pour les utilisateurs, pas d'authentification pour aller vers l'extérieur et pas de paramétrages spécifiques sur les machines des utilisateurs. Un désavantage, c'est qu'il n'y a pas d'authentification possible par utilisateur mais par adresse IP. C'est à dire que si l'on veut en interne interdire à certaine personnes d'aller vers l'extérieur ce n'est possible que si l'on connaît l'adresse de la machine de cet utilisateur, on ne peut pas empêcher que la personne aille sur une autre machine et il faut de plus que les machines aient toujours la même adresse, ce qui peut poser problème lorsqu'on on utilise du DHCP.

Les Firewalls Proxy

Les Proxy serveur sont utilisés pour contrôler et analyser tout trafic avec l'extérieur. Certains proxies utilisent en plus un cache, c'est à dire qu'ils stockent des données en local, ce qui permet de réduire le trafic réseau, en effet si une même donnée est demandée plusieurs fois, au lieu d'aller la chercher au nouveau vers l'extérieur c'est le proxy lui même qui la fournit. Leur fonctionnement dépend de leur type :

Proxy d'applications

Leur fonctionnement ressemble un peu au fonctionnement du premier Proxy, c'est à dire quand une application d'une machine locale va vers l'extérieur, en fait elle se connecte sur le Proxy et c'est le Proxy lui−même qui va chercher l'information puis la renvoyer vers la machine demandeuse.

Un exemple : vous voulez récupérer un fichier via FTP sur internet, en fait votre client FTP vas se connecter sur le proxy qui va faire serveur FTP, le Proxy va en même temps ouvrir une session FTP sur le serveur distant, il va récupérer le fichier cible et vous le renvoyer via son serveur FTP. En fait c'est toujours le Proxy qui récupère les données et vous les renvoie.

Ça à l'avantage d'être très sécurisé pour la machine cliente, ça fait cache la plupart du temps donc ça réduit le trafic réseau, ça peut permettre l'authentification aussi si l'on oblige une authentification sur les applications du Proxy. Par contre ça demande des configurations spéciales sur les clients, ça demande aussi d'installer sur le proxy les applications serveur de chaque protocole que l'on souhaite fournir aux utilisateurs et ça consomme énormément de ressources.

Proxy « SOCKS »

Il ne travaille pas sur les applications, en fait il « refait » en quelque sorte les connexions. Le client passe par le Proxy qui lui en interne refait la connexion vers l'extérieur. Comme les Firewalls filtrants, ils ne font pas d'authentification, mais peuvent quand même enregistrer l'utilisateur qui a demandé la connexion.

Pré−installation d'un Firewall filtrant sous Linux

On va dans cette partie configurer le kernel pour préparer notre système à devenir un Firewall filtrant.

Comme je l'ai dit plus haut, les Firewall filtrants sont facilement configurables sous Linux car ils sont pris en charge directement dans le noyau de Linux. Ce type de Firewall étant très léger, il vous suffit d'un 486 avec 16 voire 8 Mo de RAM. Vous avez besoin aussi suivant la version du kernel :

- Version 1.x.x : une copie de `ipfwadm` (mais bon je vous conseille vivement de passer a une version supérieure de kernel).
- Version 2.0.x : `ipwadm` est sûrement déjà présent, vérifiez quand même.
- Version 2.2.x : `ipchains` (sûrement présent aussi).
- *Version 2.3.x et 2.4 à venir : le format de firewall a encore changé, il n'est pas encore décrit dans ce document.*

Pour ce document je décris la manière de mettre en place tout ça pour un kernel de version 2.2.X car à mon goût le Firewall est beaucoup plus sûr avec ce kernel (stack IP plus sûre et `ipchains` bien plus puissant que `ipwadm`), de plus il est facile de trouver une distribution à base de kernel 2.2.X, les 2.0.X deviennent rare.

Si vous voulez de plus faire un Proxy, récupérez l'un de ces programmes :

- Squid
- TIS Firewall toolkit (FWTK)
- Socks

Une recherche sur freshmeat vous trouvera ça :) Mais bon pour l'instant ce document ne traite que des Firewall filtrant, j'ajouterais une rubrique Proxy plus tard.

Bon il faut maintenant configurer le kernel pour activer ce filtrage. Cochez les options suivantes :

```
<y> Enable experimental


<y> Enable loadable module support


<*> Packet socket

[ ] Kernel/User netlink socket

[y] Network firewalls

[ ] Socket Filtering

<y> Unix domain sockets

[y] TCP/IP networking

[ ] IP: multicasting

[y] IP: advanced router

[ ] IP: kernel level autoconfiguration

[y] IP: firewalling

[y] IP: always defragment (required for masquerading)

[y] IP: transparent proxy support

[M] IP: masquerading
```

```
--- Protocol-specific masquerading support will be built as modules.

[M] IP: ICMP masquerading

--- Protocol-specific masquerading support will be built as modules.

[Y] IP: masquerading special modules support <- Choisir tout les modules en répondant M

[y] IP: optimize as router not host

< > IP: tunneling

< > IP: GRE tunnels over IP

[y] IP: aliasing support

[y] IP: TCP syncookie support (not enabled per default)

--- (it is safe to leave these untouched)

< > IP: Reverse ARP

[y] IP: Allow large windows (not recommended if <16Mb of memory)

< > The IPv6 protocol (EXPERIMENTAL)
```

Cochez également toutes les autres options nécessaires (voir la rubrique noyau).

Bon vous recompilez le noyau (`make dep; make clean; make bzImage`), déplacez votre ancien répertoire de module (`mv /lib/module/votre_version /lib/module/votre_version.old`), puis compilez et instakllez les modules (`make modules; make modules_install`), copiez alors le nouveau kernel (cp `/usr/src/linux/arch/i386/boot/bzImage /boot`) et si vous utilisez LILO, reconfigurez le pour qu'il pointe sur votre nouveau kernel (editez `/etc/lilo.conf`) et relancez lilo (`/sbin/lilo`), sinon reconfigurez votre loader (CHOS, LOADLIN, etc.). Enfin on reboute (`reboot`). Ouf ! :)

Remarque :

Vérifiez la configuration réseau, assurez−vous que l'adresse de réseau de votre LAN est bien une des adresses réservée (192.168.2.0 par exemple, voir la rubrique réseau local pour ça). Pour la configuration de la carte externe, assurez vous aussi de sa bonne configuration pour accéder au net (testez votre connexion en fait), de même si la connexion se fait par modem. Pour tout ce qui suit je suppose que votre connexion au net se fait via une carte ethernet, en fait si vous utilisez un modem, de toute façon c'est identique. Bon avant de vraiment configurer le firewall on va tout d'abord mettre en place le « masquerading ».

Mise en place du filtrage, du masquerading, routage LAN<−>NET et règles de base

Assurez−vous d'avoir bien la configuration précédente pour votre kernel (validez les lors de la configuration du kernel).

Compilez le kernel puis les modules. Une fois tout ça réalisé, ajouter dans le fichier `/etc/rc.d/rc.local` (vérifiez le chemin, il peut être différent suivant les distributions) le chargement des modules de masquerade :

`/sbin/depmod −a` *(n'ajoutez cette ligne que si elle n'est pas déjà présente)*

`/sbin/modprobe ip_masq_ftp` *(pour ftp)*

`/sbin/modprobe ip_masq_raudio` *(pour real audio)*

`/sbin/modprobe ip_masq_irc` *(pour IRC)*

et tout autre module comme `ip_masq_cuseeme`, `ip_masq_vdolive`, etc. que vous pouvez récupérer sur le net, si vous voulez bien sûr que de tels services soit accessibles par votre réseau local.

Remarque :

Je vous rappelle quand même que plus vous ouvrirez de services, plus votre sécurité baissera. En effet certains services « supplémentaires » peuvent contenir des bugs encore inconnus mais exploitables plus tard. Si vous voulez une sécurité accrue n'autorisez que le web ainsi que le FTP, surtout si vous êtes sur un réseau d'entreprise, je ne vois pas pourquoi l'IRC par exemple doit être activé, on n'en a pas besoin pour travailler dans une entreprise :).

Avant de continuer vérifiez bien que votre LAN (réseau local) est bien sur une adresse réservée privé, c'est a dire du type:

10.0.0.0 − 10.255.255.255 Classe A

172.16.0.0 − 172.31.255.255 Classe B

192.168.0.0 − 192.168.255.255 Classe C

Pour plus de simplicité, je considère que votre LAN est sur l'adresse réseau **192.168.1.0** et que votre « passerelle » (le firewall Linux) a comme adresse **192.168.1.1**

Configurez aussi les autres machines de votre LAN avec bien sûr une adresse IP valide (de 192.168.1.2 jusqu'à 192.168.1.254), l'adresse de passerelle et de DNS, celle du Firewall (192.168.1.1).

Bon maintenant sur le Firewall, on va activer le masquerade avec:

```
# ipchains −P forward DENY

# ipchains −A forward −s yyy.yyy.yyy.yyy/x −j MASQ
```

avec **x** qui correspond au masque et **yyy.yyy.yyy.yyy** l'adresse réseau de votre LAN (ici 192.168.1.0)

Masque	Valeur de x	Classe
255.0.0.0	***8***	***A***
255.255.0.0	***16***	***B***
255.255.255.0	***24***	***C***
255.255.255.255	***32***	***Point à point***

Si pour vous ce style de notation de masque vous gêne vous pouvez aussi utiliser la notation xxx.xxx.xxx.xxx du masque pour remplacer x.

Pour notre exemple on tape alors:

```
# ipchains −P forward DENY

# ipchains −A forward −s 192.168.1.0/24 −j MASQ
```

ou

```
# ipchains −P forward DENY

# ipchains −A forward −s 192.168.1.0/255.255.255.0 −j MASQ
```

Bon on a fait quoi au juste là ?

- La première ligne indique au noyau de ne transmettre AUCUN paquet, donc on bloque TOUT en fait.
- La deuxième ligne elle indique de transmettre les paquets réseaux de notre LAN (192.168.1.0 avec masque 255.255.255.0).

Donc en fait notre Firewall Linux ne laissera passer au travers de lui que les communications LAN<−>Extérieur.

Au lieu de laisser la possibilité à toutes les machines de notre LAN d'avoir accès vers l'extérieur, on aurait pu aussi ne spécifier que certaines machines, par exemple on veut que juste le patron de notre société ainsi que l'administrateur réseau par exemple qui ont des machines avec adresse IP 192.168.1.3 et 192.168.1.10. Pour cela il faut jouer sur le masque, ça nous donne:

```
# ipchains −P forward DENY

# ipchains −A forward −s 192.168.1.3/32 −j MASQ

# ipchains −A forward −s 192.168.1.10/32 −j MASQ
```

Pour rendre ces règles « permanentes » après chaque reboot, ajoutez−les aussi dans un fichier que vous appelez par exemple **`rc.firewall`** que vous placez dans `/etc/rc.d`, sinon vous allez devoir les taper à chaque reboot du système.

On résume alors ce qui peut se trouver dans un tel fichier:

```
#!/bin/bash

#

# Firewall.rc

#
```

```
# Script de démarrage des règles du firewall

#

# Tchesmeli serge , Version 0.2

#

# Lea : http://www.lea−linux.org

#

echo "Démarrage FIREWALL :"


echo "− Activation de l'IP forwarding..."

echo "1" > /proc/sys/net/ipv4/ip_forward

# D'abord on bloque tout

echo −n "− Arrêt total des transmissions réseau..."

if [ −x /sbin/ipchains ]; then

    /sbin/ipchains −P forward DENY

    echo "OK"

else

    echo "Erreur !"

    echo "Votre noyau n'est pas configuré pour\
permettre le filtrage.... veuillez le recompiler."

    exit

fi


# On charge les modules de masquerade

echo −n "− Chargement des modules de masquerade... "

/sbin/modprobe ip_masq_ftp

/sbin/modprobe ip_masq_raudio

/sbin/modprobe ip_masq_irc

echo "OK"
# Vous pouvez y inclure ou enlever
# les modules de votre choix

# Puis on applique les règles

echo −n "− Chargement des règles de filtrage... "

# ipchains −P forward DENY

ipchains −A forward −s 192.168.1.0/255.255.255.0 −j MASQ

echo "OK"
# A vous de bien fixer les règles suivant votre réseau


echo "Firewall prêt."
```

On peut alors appeler ce script depuis `/etc/rc.d/rc.local` en y ajoutant une ligne :

```
        source rc.firewall
```

Voilà ! Votre firewall est configuré ! A vous de jouer sur les règles afin d'optimiser la sécurité sur votre réseau local...

IpTables par l'exemple

Arnaud de Bermingham

IpTables par l'exemple

Contact : duracell chez apinc point org
révision par Jice <jice chez lea−linux point org>
révision par Fred <fred chez lea−linux point org>

Introduction

Cet article présente de façon pratique la mise en place d'un firewall / proxy sur une machine Linux tournant avec un noyau 2.4.
Pour des informations plus théoriques sur les firewall/proxies, vous pouvez lire l'article firewall.

* Présentation d'IpTables

IpTables est une solution complète de firewall (noyau 2.4) remplaçant *ipchains* (noyau 2.2) tournant sous le système GNU/Linux. IpTables permet de faire du firewalling *stateful* (à états), de la translation de port et d'adresse, du filtrage au niveau 2 et beaucoup d'autres choses que nous n'allons pas aborder comme le "Mangle" ou modification des paquets à la volée (atchoum).

IpTables est fiable et dispose de très nombreuses options qui permettent de faire du filtrage très fin.

* Licence de cet article

Cette documentation est sous licence LDP

* Licence de NetFilter

NetFilter est sous licence libre GPL, i.e. gratuit et modifiable du moment que les modifications et améliorations apportées soit rendues publiques.

1/ Installation

1.1/ Prérequis

IpTables est installé en standard sur de nombreuses distributions Linux récentes. En particulier, il est installé sur Linux RedHat 7.1 et sur la plupart des distributions basées sur un kernel 2.4.x.

IpTables a besoin d'un **kernel de génération 2.4** compilé avec des options spéciales. Ceci ne pose pas de problèmes avec les noyaux 2.4 génériques des principales distributions basées sur cette génération de kernel.

1.2/ Options de compilation du kernel

Si vous désirez re−compiler votre kernel, il faut spécifier les options nécessaires au fonctionnement d'iptables.

Les options suivantes doivent êtres activées en module (`M`) ou dans le kernel (`Y`) :

```
CONFIG_PACKET
CONFIG_NETFILTER

CONFIG_IP_NF_CONNTRACK
CONFIG_IP_NF_FTP
CONFIG_IP_NF_IRC
CONFIG_IP_NF_IPTABLES
CONFIG_IP_NF_FILTER
CONFIG_IP_NF_NAT
CONFIG_IP_NF_MATCH_STATE
CONFIG_IP_NF_TARGET_LOG
CONFIG_IP_NF_MATCH_LIMIT
CONFIG_IP_NF_TARGET_MASQUERADE
```

et éventuellement :

`CONFIG_IP_NF_COMPAT_IPCHAINS` pour garder la compatibilité avec ipchains.
`CONFIG_IP_NF_COMPAT_IPFWADM` pour garder la compatibilité avec ipfwadm.
`CONFIG_IP_NF_TARGET_REDIRECT` indispensable, pour les proxies transparents par exemple.
`CONFIG_IP_NF_MATCH_MAC` permet de matcher avec les adresses MAC.

Ne pas oublier le support réseau et TCP/IP et compiler le kernel comme d'habitude :

```
make dep && make clean && make bzImage && make modules && make modules_install
```

1.3/ Installation

Sur une RedHat 7.1, récupérer le package `netfilter` et l'installer comme d'habitude avec `rpm -Uvh netfilter-x.y.z.rpm`.
Sur les versions inférieures a la 7.1, il faut compiler un kernel 2.4 car iptables ne supporte pas les kernels 2.2.x. La compilation de iptables est complexe. Le mieux est de lire le fichier INSTALL du package des sources.

1.4/ Chargement des modules

Dans le cas ou les options iptables du kernel ont étés compilées en modules, il est nécessaire de charger ces modules avant de pouvoir utiliser iptables :

```
# modprobe ip_tables
```

selon les besoins, on peut éventuellement charger les modules suivants :

```
# modprobe ip_nat_ftp
# modprobe ip_nat_irc
# modprobe iptable_filter
# modprobe iptable_mangle
# modprobe iptable_nat
```

Si on a besoin de pouvoir forwarder* les paquets IP (dans la plupart des cas), il sera nécessaire d'exécuter cette commande :

```
# echo 1 > /proc/sys/net/ipv4/ip_forward
```

afin de l'indiquer au noyau.

* *forwarder*
il s'agit de faire passer des paquets IP d'une interface réseau vers une autre. Par exemple, un paquet qui arrive de l'internet via un modem ou une carte réseau sera *redirigé* (ou *forwardé*) vers la carte réseau par laquelle le firewall est attaché au réseau local.

Nota Bene : tous les modprobe semblent inutiles car le kernel les chargent automatiquement si l'on se sert de l'une des fonctionnalités d'iptables.

2/ Présentation

2.1/ Les tables

- **Table NAT** (Network Address Translation) : Table utilisée pour la translation d'adresse ou la translation de port.
  Il a 2 types de chaînes# : *PREROUTING* qui permet de spécifier "à l'arrivée du firewall" et la chaîne *POSTROUTING* qui permet de spécifier "à la sortie du firewall". Il existe 3 targets (ou cibles) : *DNAT**, *SNAT** et *MASQUERADE**.
- **Table FILTER** : C'est la table par défaut lorsque l'on en spécifie pas.
  Cette table contient toutes les règles de filtrage, il existe 3 types de chaînes : *FORWARD* pour les paquets passant par le firewall, *INPUT* pour les paquets entrant et *OUTPUT* pour les paquets sortants. Les cibles disponibles sont : *ACCEPT, DENY, DROP, REJECT* °.
- **Table Mangle** : C'est la table qui contient les règles pour la modification de paquets.
  Elle est peu utilisée et ne sera pas décrite dans cet article.

A noter : Les règles sont matchées dans l'ordre, par défaut la table *FILTER* est vide et donc accepte tout. Aucune règle de translation d'adresse n'est présente par défaut.

\# *chaîne*
une chaîne est une suite de règles, qui sont prises dans l'ordre ; dès qu'une règle matche un paquet, elle est déclenchée, et la suite de la chaîne est ignorée.

* *SNAT*
permet de modifier l'adresse source du paquet.

* *DNAT*
permet de modifier l'adresse destination du paquet.

* *MASQUERADE*
Une gateway transforme les paquets sortants passant par elle pour donner l'illusion qu'ils sortent de la gateway elle-même par un port alloué dynamiquement ; lorsque la gateway recoit une réponse sur ce port, elle utilise une table de correspondance entre le port et les machines du réseau privé qu'elle gère pour lui faire suivre le paquet.

° *policy ACCEPT*
permet d'accepter un paquet grâce à la règle vérifiée.

° *policy DROP*
Rejet d'un paquet sans message d'erreur si la règle est vérifiée ("non ! j'en veux pas mais je dis rien à l'expediteur").

° *policy REJECT*
Rejet avec un retour de paquet d'erreur à l'expediteur si la la règle est verifiée ("un paquet recommandé de La Poste refusé par son destinataire").

2.2/ Les commandes

IpTables n'est pas livré avec une interface graphique ; les commandes et les règles sont passées en ligne de commande. Le mieux est d'écrire des scripts (à rajouter dans `/etc/rc.d/init.d`) qui permettent d'appliquer toutes les règles d'un seul coup, dès le démarrage du Linux.

2.2.1/ Commandes principales

`-A --append` : Ajoute la règle à la fin de la chaîne spécifiée
Exemple :
```
# iptables -A INPUT ...
```

`-D --delete` : Permet de supprimer une chaîne. On peut la matcher de 2 manières, soit en spécifiant le numéros de la chaîne a supprimer, soit en spécifiant la règle à retirer.
Exemples :
```
# iptables -D INPUT --dport 80 -j DROP
# iptables -D INPUT 1
```

`-R --replace` : Permet contrairement à `--delete` de remplacer la chaîne spécifiée.
Exemple :
```
# iptables -R INPUT 1 -s 192.168.0.1 -j DROP
```

`-I --insert` : Permet d'ajouter une chaîne dans un endroit spécifié de la chaîne.
Exemple :
```
# iptables -I INPUT 1 --dport 80 -j ACCEPT
```

`-L --list` : Permet d'afficher les règles.
Exemples :
```
# iptables -L        # Affiche toutes les règles des chaînes de FILTER
# iptables -L INPUT  # Affiche toutes les règles de INPUT (FILTER)
```

`-F --flush` : Permet de vider toutes les règles d'une chaîne.
Exemple :
```
# iptables -F INPUT
```

`-N --new-chain` : Permet de créer une nouvelle chaîne.
Exemple :
```
# iptables -N LOG_DROP
```

`-X --delete-chain` : Permet d'effacer une chaîne.
Exemple :
```
# iptables -X LOG_DROP
```

`-P --policy` : Permet de spécifier au kernel la target par défaut d'une chaîne DENY, ACCEPT, REJECT, DROP ...
Exemple :
```
# iptables -P INPUT DROP
```

2.2.2/ Commandes pour matcher

Remarques :

Le "!" peut être utilisé pour certaines commandes afin de spécifier le contraire (on peut le traduire par "sauf"). Par exemple une commande qui doit refuser tout trafic TCP sauf ce qui provient de l'adresse IP 10.42.42.42 sera traduite par la commande suivante :
Exemple :
```
# iptables -A INPUT -p tcp --source ! 10.42.42.42 -j DENY
```

Les adresses IP peuvent optionnellement être spécifiées avec le masque associé sous la forme [adresse ip]/[masque].

`-p --protocol` : Spécifier un protocole : `tcp`, `udp`, `icmp`, `all` (tous)
Exemple :
```
# iptables -A INPUT -p icmp -j DENY
```

`-s --source` : Spécifier une adresse source à matcher
Exemple :
```
# iptables -A INPUT -p tcp -s 192.168.42.42 -j ACCEPT
```

`-d --destination` : Spécifier une adresse destination
Exemple :
```
# iptables -A FORWARD -p tcp -d 10.1.0.1 -j ACCEPT
```

`-i --in-interface` : Spécifier une interface d'entrée
Exemple :
```
# iptables -A INPUT -p icmp -i eth0 -j DENY
```

`-o --out-interface` : Spécifier une interface de sortie
Exemple :
```
# iptables -A OUTPUT -p icmp -o eth0 -j DENY
```

`-f --fragment` : Paquet fragmenté
Exemple :

```
# iptables -A INPUT -p icmp -f -j DENY
```

`--sport --source-port` : Spécifier le port source ou une plage de ports, fonctionne aussi en udp, `-m multiport` permet de spécifier plusieurs ports à matcher.
Exemples :

```
# iptables -A INPUT -p tcp --sport 80 -j ACCEPT
# iptables -A INPUT -p udp --sport 80 -j DROP
# iptables -A OUTPUT -p tcp -m multiport --sport 3128,21,1000 -j DROP
# iptables -A OUTPUT -p tcp --sport 1024:2042 -j ACCEPT
```

`--dport --destination-port` : Spécifier le port destination ou une plage de ports, fonctionne aussi en udp, `-m multiport` permet de spécifier plusieurs ports a matcher.
Exemples :

```
# iptables -A INPUT -p tcp --dport 110 -j DENY
# iptables -A INPUT -p udp --dport 110 -j DENY
# iptables -A INPUT -p tcp -m multiport --dport 110,4242,119 -j DROP
# iptables -A INPUT -p tcp --sport 4925:4633 -j ACCEPT
```

`--tcp-flags` : Spécifier un flag tcp à matcher : `SYN ACK FIN RST URG PSH ALL NONE`
Exemple :

```
# iptables -A INPUT -p tcp --dport 42 --tcp-flags SYN,ACK -j ACCEPT
```

`--icmp-type` : Spécifier un type de paquet icmp à matcher
Exemple :

```
# iptables -A INPUT -p icmp --icmp-type 8 -j DROP
```

`--mac-source` : Spécifier l'adresse MAC à matcher
Exemple :

```
# iptables -A INPUT --mac-source 42.42.AA.42.42.AA -j DROP
```

`--state` : Permet de spécifier l'état du paquet à matcher parmi les états suivants :
`ESTABLISHED` : paquet associé à une connexion déjà établie
`NEW` : paquet demandant une nouvelle connexion
`INVALID` : paquet associé à une connexion inconnue
`RELATED` : Nouvelle connexion mais liée, idéal pour les connexions FTP
Exemples :

```
# iptables -A INPUT -i eth0 -p tcp --dport 80 -m state --state NEW,ESTABLISHED -j ACCEPT
# iptables -A OUTPUT -o eth0 -p tcp --sport 80 -m state --state ESTABLISHED -j ACCEPT
```

Spécificités NAT :

`--to-destination` : Utilisé en target pour le DNAT, permet de spécifier l'adresse de destination de la translation, on peut également spécifier un port s'il est différent du port source.
Exemples :

```
# iptables -t nat -A PREROUTING -d 42.12.42.12 -p tcp --dport 110 -j DNAT --to-destination 192.168.1.2:6110
# iptables -t nat -A PREROUTING -d ! 42.12.42.12 -p tcp --dport 80 -j DNAT --to-destination 192.168.2.1:3128
```

`--to-source` : Utilisé pour en target pour le SNAT, permet de spécifier l'adresse source de la translation.

Spécificités pour les LOGS :

`--log-level` : Level, niveau de log
Exemple : Cf. chapitre 3

`--log-prefix` : Permet de spécifier un préfixe pour les logs.
Exemple : Cf. chapitre 3

2.2.3/ Quelques exemples :

Les exemples qui suivent supposent que vous êtes reliés à internet par modem via l'interface ppp0 (mais en remplaçant ppp0 par eth0 - par exemple, on peut adapter les exemples pour d'autres type de liaisons) et que votre réseau local est 192.168.1.0/24 (classe C).

- Pour fixer les **politiques par défaut** (cad: ce qui se passe quand aucune règle ne correspond - ne matche pas), ici, on refuse tout (normal, on fait un firewall, oui ou non ?) :

```
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP
```

- Pour **logguer** tout ce qu'on jette :

```
iptables -N LOG_DROP
iptables -A LOG_DROP -j LOG --log-prefix '[IPTABLES DROP] : '
iptables -A LOG_DROP -j DROP
```

Et ensuite, plutôt que de mettre `-j DROP`, il faut mettre `-j LOG_DROP` et les trois dernières règles doivent être :

```
iptables -A FORWARD -j LOG_DROP
iptables -A INPUT -j LOG_DROP
iptables -A OUTPUT -j LOG_DROP
```

- Pour **accepter** tout ce qui se passe sur l'interface `lo` (sinon ce n'est pas la peine d'activer le réseau !) :

```
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT
```

- Pour **accepter** tout ce qui se passe sur le **réseau local** `192.168.1.0` :

```
iptables -A INPUT -s 192.168.1.0/24 -j ACCEPT
iptables -A OUTPUT -d 192.168.1.0/24 -j ACCEPT
iptables -A FORWARD -s 192.168.1.0/24 -j ACCEPT
```

- Pour accepter les résolutions de nom (ie: le **dns**) :

```
iptables -A INPUT -i ppp0 --protocol udp --source-port 53 -j ACCEPT
iptables -A OUTPUT -o ppp0 --protocol udp --destination-port 53 -j ACCEPT
iptables -A INPUT -i ppp0 --protocol tcp --source-port 53 -j ACCEPT
iptables -A OUTPUT -o ppp0 --protocol tcp --destination-port 53 -j ACCEPT
```

- Pour accepter le traffic **web** (on veut surfer!) :

```
iptables -A INPUT -i ppp0 --protocol tcp --source-port 80 -m state --state ESTABLISHED -j
LOG_ACCEPT
iptables -A OUTPUT -o ppp0 --protocol tcp --destination-port 80 -m state --state
NEW,ESTABLISHED -j LOG_ACCEPT
```

La première ligne pour accepter ce qui entre sur notre interface ppp0 sur le port 80 (le port http) si c'est une connexion déjà établie, la seconde pour accepter ce qui sort sur ppp0 sur le port 80 si c'est une nouvelle connexion ou si c'est une connexion déjà établie.

Pour autoriser le **ssh**, il faut préciser le port 22; pour autoriser l'**irc**, le port 6667 (ou celui que vous utilisez pour vous connecter à votre serveur); pour le **smtp** (envoi d'emails), le port 25; pour le **pop3** (réception d'emails), le port 110; pour le **imap** (réception d'emails), les ports 143 et 220 (**imap3**) ; pour le **cvs**, le port 2401 ; pour le **https**, le port 443. De manière générale, le numéros de port se trouvent dans /etc/services.

- Pour le **ftp** c'est un peu plus complexe. D'abord, il faut charger le module : `ip_conntrack_ftp` (c'est lui qui suit − track en anglais − les connexions ftp) et, si vous *nat*ez (en utilisant le masquerading par exemple) vos connexions ftp vers d'autres postes le module : `ip_nat_ftp` :

```
modprobe ip_conntrack_ftp
# éventuellement : modprobe ip_nat_ftp
```

Ensuite, il faut taper les commandes suivantes :

```
iptables -A INPUT -i ppp0 -p tcp --sport 21 -m state --state ESTABLISHED -j ACCEPT
iptables -A OUTPUT -o ppp0 -p tcp --dport 21 -m state --state NEW,ESTABLISHED -j ACCEPT
```

Cela pour que la connexion puisse s'établir. Ensuite (et c'est la qu'on a besoin de `ip_conntrack_ftp`) :

```
iptables -A INPUT -i ppp0 -p tcp --sport 20 -m state --state ESTABLISHED,RELATED -j
ACCEPT
iptables -A OUTPUT -o ppp0 -p tcp --dport 20 -m state --state ESTABLISHED -j ACCEPT
```

Pour que serveur puisse établir la connexion pour les données (en mode actif). Et enfin :

```
iptables -A INPUT -i ppp0 -p tcp --sport 1024:65535 --dport 1024:65535 -m state --state
ESTABLISHED -j ACCEPT
iptables -A OUTPUT -o ppp0 -p tcp --sport 1024:65535 --dport 1024:65535 -m state --state
ESTABLISHED,RELATED -j ACCEPT
```

Pour que le serveur puisse établir la connexion pour les données (en mode passif). Ici aussi `ip_conntrack_ftp` est nécessaire.

- Pour **partager une connexion**, il faut que le forwarding soit activé dans le noyau (`echo 1 > /proc/sys/net/ipv4/ip_forward`), puis il faut autoriser iptable à faire le forwarding :

```
iptables -F FORWARD
iptables -A FORWARD -j ACCEPT
```

et enfin, cacher les machines forward−ées par le firewall :

```
iptables -A POSTROUTING -t nat -o ppp0 -j MASQUERADE
```

Sur chaque machine devant être cachée par le firewall (ou devant partager la connexion avec la machine qui est connectée à internet), il faut ajouter une route par defaut :

```
route add default gw 192.168.1.1
```

Si la machine connectée à internet a comme ip : 192.168.1.1. Il suffit avec une redhat/mandrake d'éditer :

```
/etc/sysconfig/network
```

et d'ajouter dedans :

```
GATEWAY=192.168.1.1
```

puis de redémarer le réseau :

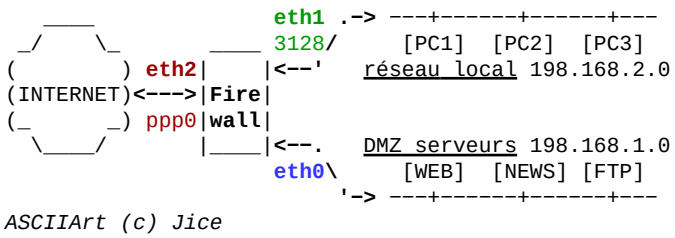
```
/etc/rc.d/init.d/network restart
```

3/ Application par l'exemple

Nous allons mettre en place un firewall / proxy.

Pour cet exemple, le firewall aura la connexion à Internet (interface eth2) et disposera de 2 pattes sur des réseaux privés (eth0 et eth1) :

- il fait office de proxy sur le port 3128 pour un réseau qui aura ainsi un accès internet
- et une DMZ* ("zone démilitarisée") sur laquelle il y a un ensemble de serveurs disponibles de l'extérieur dont un serveur web qui a pour adresse 192.168.1.2 écoutant sur le port 80 en TCP.



La classe d'adresse IP 192.168.2.0 correspond au réseau interne sur l'interface `eth1`.
La classe d'adresse IP 192.168.1.0 correspond a la DMZ sur l'interface `eth0`.
L'interface de la connexion Internet est ppp0 sur l'interface `eth2`.

* *DMZ, ou zone démilitarisée*
Sous−réseau dans lequel des serveurs accessibles depuis internet sont en adressage privé (classe d'adresse IP réservée comme 192.168.x.x) derrière un firewall.

3.1/ Le script init.d

Nous allons écrire un script qui permettra de charger automatiquement au démarrage de la machine ou sur demande les règles du firewall qui seront stockées dans le fichier `/etc/firewall.sh`.
Le script de démarrage sera nommé `/etc/init.d/firewall`. Bien sûr, on n'oubliera pas d'exécuter un `chmod +x` sur les 2 scripts que nous allons créer au long de ce chapitre.

Go !
Fichier de chargement `/etc/init.d/firewall` :

```
#!/bin/bash
#
# Lancement du script de Firewall
# Arnaud de Bermingham

. /etc/init.d/functions

RETVAL=0

# Fonction pour le lancement du firewall
start() {
  echo −n "Application des règles IpTables: "
  /etc/firewall.sh
  RETVAL=?
  [ −eq 0 ] && touch /var/lock/subsys/firewall
  echo
}

# Fonction pour arrêter le firewall (on flush)
stop() {
```

```
  echo -n "Flush des règles IpTables: "
  /etc/flush_iptables.sh
  RETVAL=?
  [ -eq 0 ] && rm -f /var/lock/subsys/firewall
  echo
}

case "" in
  start)
    start
    ;;
  stop)
    stop
    ;;
  restart)
    stop
    start
    ;;
  status)
    /sbin/iptables -L
    /sbin/iptables -t nat -L
    RETVAL=?
    ;;
  *)
    echo "Usage: firewall {start|stop|restart|status}"
    RETVAL=1
esac

exit
```

C'est tout simple non ?

3.2/ Le script pour flusher (vider) les règles

Et maintenant : `/etc/flush_iptables.sh`.

```
#!/bin/sh
#
# Script pour vider les règles iptables
# Arnaud de Bermingham
# duracell@apinc.org

#
# On remet la police par défaut à ACCEPT
#
iptables -P INPUT ACCEPT
iptables -P FORWARD ACCEPT
iptables -P OUTPUT ACCEPT

#
# On remet les polices par défaut pour la table NAT
#
iptables -t nat -P PREROUTING ACCEPT
iptables -t nat -P POSTROUTING ACCEPT
iptables -t nat -P OUTPUT ACCEPT

#
# On vide (flush) toutes les règles existantes
#
iptables -F
iptables -t nat -F

#
# Et enfin, on efface toutes les chaînes qui ne
# sont pas à defaut dans la table filter et nat

iptables -X
iptables -t nat -X

# Message de fin
echo " [termine]"
```

Bon, on va enfin commencer les choses sérieuses : le script du firewall :)

3.3/ Les prérequis pour le script du firewall et création des tables pour les logs

Le script sera commenté au fur et à mesure, afin de décrire chaque étape.

```
#!/bin/sh

# script /etc/firewall.sh

# Firewall d'exemple a but pédagogique
# Arnaud de Bermingham
# duracell@apinc.org

# Activation du forwarding
# C'est pas pour faire joli, on aura des règles
# de forward et il faut bien que les paquets
# traversent la machine, donc on met ce fichier à 1

echo 1 > /proc/sys/net/ipv4/ip_forward

# Alors la, on va appliquer quelques astuces
# pour empêcher les attaques de type spoofing
# et bloquer les réponses ICMP du firewall,
# comme ça c'est très propre. Attention, le
# fait de bloquer le trafic ICMP sur
# le firewall bloque les pings.

# Je veux pas de spoofing

if [ -e /proc/sys/net/ipv4/conf/all/rp_filter ]
then
  for filtre in /proc/sys/net/ipv4/conf/*/rp_filter
  do
    echo 1 > $filtre
  done
fi

# pas de icmp

echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_all
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts

# On va utiliser iptables. Si on l'a compilé en module
# dans le kernel, il faut charger le module ip_tables.

modprobe ip_tables

# on va charger quelques modules supplémentaires pour
# gérer la translation d'adresse, l'IRC et le FTP
# Tu me fait 4 pompes. Chef oui chef !

modprobe ip_nat_ftp
modprobe ip_nat_irc
modprobe iptable_filter
modprobe iptable_nat

# Pour faire bien, on va vider toutes les règles
# avant d'appliquer les nouvelles règles de firewall

iptables -F
iptables -X

# On va rajouter 2 nouvelles chaînes.
# Ceci permettra d'ajouter des nouvelles cibles qui
# auront la possibilité de loguer ce qui se passe.

# La on logue et on refuse le paquet,
# on rajoute un préfixe pour pouvoir
# s'y retrouver dans les logs
iptables -N LOG_DROP
iptables -A LOG_DROP -j LOG \
  --log-prefix '[IPTABLES DROP] : '
iptables -A LOG_DROP -j DROP
```

```
# ici, on logue et on accepte le paquet,
# on rajoute un préfixe pour pouvoir
# s'y retrouver dans les logs
iptables -N LOG_ACCEPT
iptables -A LOG_ACCEPT -j LOG \
  --log-prefix '[IPTABLES ACCEPT] : '
iptables -A LOG_ACCEPT -j ACCEPT

# On veut faire un firewall efficace,
# donc la politique a appliquer est de tout
# refuser par défaut et rajouter une a une
# les règles que l'on autorise.
# Bien sur, on a RTFM un peu et on a vu que
# l'option -P permet de définir
# la cible par défaut

iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP

# Pour éviter les problèmes, on va tout accepter sur
# la machine en local (interface lo).
# Je déconseille de retirer cette règle car
# ça pose pas mal de problèmes et ça peut
# faire perdre la main sur la machine

iptables -A INPUT  -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT

# Bon, la partie initialisation et préparation est
# terminée, passons aux choses sérieuses

# Comme on l'a dit dans la présentation de
# l'architecture réseau, le firewall fait
# également office de proxy grâce par exemple
# à un squid installé dessus. On va donc
# accepter que le proxy ait une connexion
# internet directe. Tant qu'à faire, on va
# mettre des états pour que ça soit bien sécurisé

iptables -A OUTPUT -o ppp0 -m state \
  --state NEW,ESTABLISHED -p tcp --dport 80 -j ACCEPT
iptables -A INPUT  -i ppp0 -m state \
  --state ESTABLISHED     -p tcp --sport 80 -j ACCEPT

# Maintenant, on va faire en sorte que le
# proxy soit totalement transparent pour le LAN
# bénéficiant de la connexion internet.
# L'astuce consiste a rediriger toutes les
# requêtes ayant un port de destination 80
# vers le port 3128 du proxy, ici c'est le
# firewall (qui est sur le firewall et qui
# a l'adresse IP 192.168.2.1 ).
# Une règle de NAT suffira largement pour faire ça.

iptables -t nat -A PREROUTING -i eth1 -p tcp \
  --dport 80 -j DNAT --to-destination 192.168.2.1:3128

# Bon, c'est pas trop compliqué ! Maintenant
# on sait que l'on a un serveur web sur la DMZ
# (la machine d'adresse IP 192.168.1.2) sur
# le port 80. On souhaite que toutes les requêtes
# provenant d'internet arrivant sur l'adresse IP
# publique du serveur ( ici 42.42.42.42 ) soit
# redirigées sur le serveur web de la DMZ.
# Rien de bien compliqué. Dans l'exemple,
# on peut retirer le :80 de la target
# --to-destination

iptables -t nat -A PREROUTING -d 42.42.42.42 \
  -p tcp --dport 80 -j DNAT --to-destination 192.168.1.2:80

# C'est bien tout ça ! mais le problème c'est
# que les chaînes de la table FILTER sont toutes
```

```
# à DENY, donc tout ceci ne fait rien du tout.
# On va donc passer a la configuration du
# firewall proprement dit.

# On va quand même accepter les connexions ssh
# (port 22) provenant d'une machine (la votre en
# l'occurrence, on va dire 192.168.2.42) vers le
# firewall pour pouvoir modifier les règles
# facilement pour bien surveiller, on vas quand
# même loguer les connexions provenant de mon IP
# et à destination du ssh du firewall

iptables -A INPUT -i eth1 -s 192.168.2.42 -m state \
  --state NEW,ESTABLISHED -p tcp --dport 22 -j LOG_ACCEPT
iptables -A OUTPUT -o eth1 -d 192.168.2.42 -m state \
  --state ESTABLISHED     -p tcp --sport 22 -j LOG_ACCEPT

# On veut que le LAN connecté à l'interface
# eth1 ait un accès complet à internet.
# La règle de NAT qui permettait d'avoir
# un proxy transparent sera automatiquement
# appliqué. L'interface correspondant
# à la connexion internet est ici ppp0

iptables -A FORWARD -i eth1 -o ppp0 -j ACCEPT
iptables -A FORWARD -o eth1 -i ppp0 -j ACCEPT

# Maintenant on donne le droit au LAN de
# consulter les pages web du serveur de la DMZ

iptables -A FORWARD -i eth1 -o eth0 -p tcp \
  --dport 80 -state --state NEW,ESTABLISHED -j ACCEPT
iptables -A FORWARD -i eth0 -o eth1 -p tcp \
  --sport 80 -state --state ESTABLISHED -j ACCEPT

# Maintenant il n'y a plus qu'à dire au firewall
# d'autoriser à transmettre des paquets TCP à
# destination du port 80 provenant de l'adresse
# IP publique (i.e. d'internet) vers le serveur
# web de la DMZ que nous avons naté précédemment.

iptables -A FORWARD -i ppp0 -o eth0 -p tcp \
  --destination-port 80 -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A FORWARD -o ppp0 -i eth0 -p tcp \
  --source-port 80      -m state --state ESTABLISHED     -j ACCEPT

# Maintenant il ne reste plus grand chose à faire !

# Il faut permettre à l'ensemble du LAN de dialoguer
# sur internet avec la même adresse IP sinon, bien
# évidemment ça ne marchera pas (à moins que vous
# ayez 30 adresses ip !).
# Une petite règle de NAT avec un -j MASQUERADE
# suffira (masquerade = dialoguer avec l'adresse
# IP publique sur firewall)

iptables -t nat -A POSTROUTING \
  -s 192.168.2.0/24 -j MASQUERADE

# Il faut également que le serveur web de la DMZ
# soit masqueradé sinon, le serveur
# dialoguera sur internet avec son IP privée

iptables -t nat -A POSTROUTING \
  -s 192.168.1.0/24 -j MASQUERADE

# Toutes les règles qui n'ont pas passé les
# règles du firewall seront refusées et loguées...
# facile :

iptables -A FORWARD -j LOG_DROP
iptables -A INPUT -j LOG_DROP
iptables -A OUTPUT -j LOG_DROP
```

```
# Pour faire zoli
echo " [Termine]"

# c'est enfin fini
```

Et voilà ! le firewall de compèt' est prêt et fonctionnel.

Ceci était bien évidemment un exemple, vous pouvez dès à présent préparer votre propre firewall personnalisé !
Vous pouvez l'adapter à vos besoins, votre connexion vers internet (par ADSL par exemple), etc.

SmoothWall

par Laurent DUBETTIER−GRENIER

Installation d'un firewall basé sur SmoothWall

Introduction

Surfer sur Internet de manière sûre ? C'est possible, en réalisant avec un vieux PC un "mur de feu" (Firewall). Et c'est gratuit ! SmoothWall est en effet un firewall distribué sous licence GNU/GPL, avec son code source. Et c'est bien plus ! Interface web de management à distance (sécurisée : https), fichiers de log, NAT, DNS, Proxy, VPN, SSH, IPSEC sont aussi au menu... Le tout avec une abondante documentation, essentiellement en anglais...

Et si cela ne vous suffit pas, il existe aussi une version commerciale professionnelle de SmoothWall, avec plus d'options et un support professionnel, disponible auprès de SmoothWall Ltd.

A noter : lors de sa première connexion, ce firewall enverra quelques informations non−personnelles comme le processeur utilisé, la taille de la RAM, la date et l'heure, la version de SmoothWall installée sur votre système aux auteurs. Les auteurs précisent que ce n'est pas un Spyware.

SmoothWall est manageable à distance, donc, une fois installé, il n'y a vraiment besoin que d'un PC, sans écran, sans souris, sans clavier... SmoothWall supporte les cartes réseaux les plus courantes (3com, Realtek, NE 2000...) et les modes de connexion internet les plus courants (Modem, ISDN, ADSL, USB ADSL, Ethernet).

Nota :
une liste des cartes réseaux supportées (fichiers avec extensions .o) est disponible dans le fichier lib.tar.gz du cdrom SmoothWall (il faut connaître le chipset équipant votre carte réseau : rlt8139.o est présent dans l'archive lib.tar.gz et indique que les cartes équipées d'un chipset Realtek 8139 sont supportées).

Il est possible de définir trois zones réseaux différentes : verte (Zone totalement sûre : votre réseau local), orange (Zone partiellement sûre : DMZ, De−Militarised Zone, c'est sur cette branche du réseau que sera branché votre serveur HTTP par exemple) et rouge (Internet et ses pirates...). Ceci veut dire que vous pouvez avoir jusqu'à trois cartes réseaux sur votre Firewall : une verte sur laquelle seront reliés votre réseau local et votre serveur local de fichier par exemple, une orange sur laquelle sera relié votre serveur web Apache ou votre serveur de jeu en réseau, et la dernière (rouge) sur laquelle sera relié votre modem ethernet ADSL par exemple...

La configuration requise est vraiment minimale :

- Processeur 486DX4
- 16 Mo de RAM
- Disque dur de 200 Mo (SmoothWall n'occupe qu'environ 60 Mo)
- Carte réseau 10 Mb

Ceci est le minimum pour faire fonctionner une connexion par Modem. Pour une configuration multi−utilisateurs (partage de connexion) et l'ADSL, il faudra étoffer un peu cette configuration...

Enfin, SmoothWall est basé sur un noyau Linux 2.2.19, et sa mascotte est un ours polaire du nom de Smoothie...

Installation

!!! ATTENTION !!!
L'INSTALLATION DE SMOOTHWALL PARTITIONNE ET FORMATE COMPLETEMENT LE DISQUE DUR DU SYSTEME SUR LEQUEL IL S'INSTALLE.
TOUTE LES DONNEES PRESENTES SUR CE SYSTEME SERONT DONC PERDUES IRREMEDIABLEMENT.
!!! ATTENTION !!!

Il est bien sûr conseillé de lire les documentations officielles de SmoothWall pour avoir plus d'informations. Notamment la FAQ... (j'ai bien aimé l'humour british des chapitres "On Not Reacting Like A Looser" et "Questions Not To Ask"...). C'est essentiel pour ne pas reposer aux auteurs ou à la communauté SmoothWall des questions dont la réponse figure déjà dans la documentation...

Nota :
Il est judicieux de brancher votre firewall sur un onduleur : le système de fichier du noyau linux 2.2.19 est ext2 et n'est pas journalisé. Il supporte très mal les reboot brutal...

Il faut d'abord télécharger l'image Iso (21 Mo environ) sur le site de SmoothWall, puis graver cette image sur un cdrom.

Nota :
Si votre bios n'accepte pas le boot sur cdrom, il est possible de créer une disquette de boot (avec l'utilitaire rawritewin figurant dans le répertoire dosutils du cdrom, sous Windows, et en sélectionnant l'image boot−1.0.img). En fait c'est comme pour n'importe quelle distribution Linux... Il est même possible de l'installer depuis un serveur ftp/http distant... Que demander de plus !

Pour installer Smoothwall, Il faut insérer le cdrom dans votre lecteur puis rebooter la machine.Les étapes sont :

- Choix du langage
- Fenêtre de bienvenue
- Choix du mode d'installation (cdrom)

- Préparation du disque dur (partition et formatage : toutes les données présentes sur le disque sont effacées !)
- Configuration réseau (carte verte : IP statique 192.168.1.1)
- Installation de SmoothWall
- Configuration clavier (latin) et zone horaire
- Hostname (le nom donné à votre firewall)
- Configuration ISDN (Choisir disable si vous n'utilisez pas ce mode de connexion à Internet)
- Configuration USB ADSL (idem)
- Configuration Ethernet

  Nota :
  Vous pouvez choisir la configuration de votre firewall : vert + modem, vert + orange + modem, vert + rouge, vert + orange + rouge... puis configurer chaque carte réseau : pour une configuration constituée d'un modem relié au port série du firewall et une carte réseau reliée par un câble croisé à votre PC, choisir la configuration verte (GREEN)

  Conseils de configuration :

  - **Carte Verte** : adresse IP statique 192.168.1.1
    Etendue DHCP : 192.168.1.100 à 192.168.1.200
    Les clients de votre reseau local (vert) doivent être configuré pour obtenir une adresse IP automatiquement par DHCP. Leur paramètre DNS doit être 192.168.1.1
  - **Carte Orange** : adresse IP statique 192.168.0.1 (différente de la verte)
    Les serveurs et machines en zone orange doivent utiliser une adresse IP fixe comprise entre 192.168.0.2 et 192.168.0.254
  - **Carte rouge** : à régler sur DHCP si l'adresse IP et l'adresse DNS sont fournies par votre fournisseur d'accès
- Mot de passe (3 utilisateurs : root, setup, admin)

Nota :
L'utilisateur "root" a le control total du firewall
L'utilisateur "setup" peut réaliser la mise à jour du firewall ou toute configuration post−installation
L'utilisateur "admin" peut changer les paramètres du firewall, effectuer le contrôle journalier du firewall et des fichiers de log, changer le mot de passe de l'utilisateur "dial", démarrer ou arrêter un service
L'utilisateur "dial" a l'autorisation de lancer une connexion à Internet mais est configuré (ainsi que son mot de passe) par l'interface web de management du firewall.

Une fois l'installation terminée, il faut redémarrer le firewall (après avoir retiré cdrom et disquette d'installation).

Administration

Votre firewall est alors autonome et peut être administré à distance par une interface web, à partir d'un poste relié au réseau vert, en vous connectant comme utilisateur "admin" :

```
https://192.168.1.1:445 (accès sécurisé)
http://192.168.1.1:81 (non sécurisé)
```

Il est alors nécessaire de télécharger les patchs disponibles, et cela peut se faire à partir de l'interface d'administration.

Et voilà ! Maintenant, vous pouvez surfer tranquille... ou presque ! D'après ses auteurs, SmoothWall n'a pas de failles de sécurité connues...

SSh, la sécurisation par le chiffrement

par Jean−philippe Gaulier

Préambule

Il y a aujourd'hui deux possibilités de travail, en local (local work) ou à distance (remote work). Ce dernier est permis par l'utilisation du réseau, qu'il soit local (LAN) ou réellement éloigné, comme par Internet (WAN). De nombreux outils ont été fournis pour utiliser la capacité du réseau. Échanger, copier, utiliser des shells à distance. Les noms de ces outils sont respectivement FTP, rcp, telnet, etc... Bien que ces outils, utilisés pendant des années et même encore aujourd'hui dans certaines entreprises, soient très pratiques, ils comportent une faiblesse importante. Leurs transactions sont transmises en clair via le réseau. De ce fait, l'informatique devenant ouvert à un grand nombre, n'importe quelle personne mal intentionnée peut être en mesure d'observer ce que vous faîtes, allant même jusqu'à subtiliser vos données personnelles et mots de passe.

Le problème par la pratique

Pour pouvoir appuyer cette notion de non−confidentialité, nous allons regarder ce qui se passe lors de l'établissement d'une connexion d'une session telnet. Le logiciel utilisé pour tracer les connexions est **ethereal**.
Établissons la connexion sur un serveur *telnetd* fonctionnant sous GNU/Linux. Après quelques échanges de paquets, nous obtenons la bannière d'accueil :

```
0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 10   ........ ......E.
0010  00 5f 7d 65 40 00 40 06  bf 21 7f 00 00 01 7f 00   ._}e@.@. ¿!......
0020  00 01 00 17 86 10 2c fc  06 1a 2c 1a b3 30 80 18   ......,ü ..,.³0..
0030  7f ff f7 2c 00 00 01 01  08 0a 00 23 78 ca 00 23   .ÿ÷,.... ...#xÊ.#
0040  78 ca 44 65 62 69 61 6e  20 47 4e 55 2f 4c 69 6e   xÊDebian  GNU/Lin
0050  75 78 20 74 65 73 74 69  6e 67 2f 75 6e 73 74 61   ux testi ng/unsta
0060  62 6c 65 20 63 65 6e 74  61 75 72 0d 0a            ble cent aur..
```

on constate que la connexion s'effectue sur le port 23 (en 22 et 23 dans la trame).
Les données quant à elles commencent clairement en 42, avec la bannière d'accueil :
Debian GNU/Linux testing/unstable centaur

Les données ont été envoyées en clair et la bannière permet de définir le système d'exploitation sur lequel nous sommes accueillis. Continuons notre étude, la demande d'identification est proposée par le système distant :

```
0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 10   ........ ......E.
0010  00 43 7d 66 40 00 40 06  bf 3c 7f 00 00 01 7f 00   .C}f@.@. ¿<......
0020  00 01 00 17 86 10 2c fc  06 45 2c 1a b3 30 80 18   ......,ü .E,.³0..
0030  7f ff 66 8b 00 00 01 01  08 0a 00 23 78 ce 00 23   .ÿf..... ...#xÎ.#
0040  78 ce 63 65 6e 74 61 75  72 20 6c 6f 67 69 6e 3a   xÎcentau r login:
0050  20
```

Il faut maintenant se faire reconnaître auprès du système. Nous supprimerons tout echo local doublant l'envoi de caractères, un pour le système distant et un pour voir ce que nous écrivons. Il en résulte une suite d'échange avec notre interlocuteur :

```
0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 10   ........ ......E.
0010  00 35 ca b7 40 00 40 06  71 f9 7f 00 00 01 7f 00   .5Ê·@.@. qù......
0020  00 01 86 10 00 17 2c 1a  b3 4f 2c fc 06 c2 80 18   ......,. ³O,ü.Â..
0030  7f ff e7 e2 00 00 01 01  08 0a 00 23 86 d6 00 23   .ÿçâ.... ...#.Ö.#
0040  86 63 6a                                          .cj

0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 10   ........ ......E.
0010  00 35 ca b9 40 00 40 06  71 f7 7f 00 00 01 7f 00   .5Ê¹@.@. q÷......
0020  00 01 86 10 00 17 2c 1a  b3 50 2c fc 06 c3 80 18   ......,. ³P,ü.Ã..
0030  7f ff e2 65 00 00 01 01  08 0a 00 23 86 de 00 23   .ÿâe.... ...#.Þ.#
0040  86 d6 6f                                          .Öo

0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 10   ........ ......E.
0010  00 35 ca bb 40 00 40 06  71 f5 7f 00 00 01 7f 00   .5Ê»@.@. qõ......
0020  00 01 86 10 00 17 2c 1a  b3 51 2c fc 06 c4 80 18   ......,. ³Q,ü.Ä..
0030  7f ff e1 55 00 00 01 01  08 0a 00 23 86 e4 00 23   .ÿáU.... ...#.ä.#
0040  86 de 70                                          .Þp
```

Nous envoyons notre identifiant à la machine distante, les trames ne sont pas utilisées à l'économie puisque chaque paquet contient seulement un et un seul caractère comme données (data). On retrouve ici notre identifiant, jop, envoyé pour nous loguer. Ce qui est inquiétant, c'est que la même chose va également se produire pour notre mot de passe censé être privé et inviolable.

```
0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 10   ........ ......E.
0010  00 35 94 19 40 00 40 06  a8 97 7f 00 00 01 7f 00   .5..@.@. ".......
0020  00 01 87 20 00 17 e3 5e  dc 53 e3 2d 4b 8a 80 18   ... ..ã^ ÜSã-K...
0030  7f ff 75 08 00 00 01 01  08 0a 00 34 5a b6 00 34   .ÿu..... ...4Z¶.4
0040  50 e9 62                                          Péb

0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 10   ........ ......E.
0010  00 35 94 1a 40 00 40 06  a8 96 7f 00 00 01 7f 00   .5..@.@. ".......
0020  00 01 87 20 00 17 e3 5e  dc 54 e3 2d 4b 8a 80 18   ... ..ã^ ÜTã-K...
0030  7f ff 5e 23 00 00 01 01  08 0a 00 34 5a c9 00 34   .ÿ^#.... ...4ZÉ.4
0040  5a ba 6f                                          Zºo

0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 10   ........ ......E.
0010  00 35 94 1b 40 00 40 06  a8 95 7f 00 00 01 7f 00   .5..@.@. ".......
0020  00 01 87 20 00 17 e3 5e  dc 55 e3 2d 4b 8a 80 18   ... ..ã^ ÜUã-K...
0030  7f ff 6a f1 00 00 01 01  08 0a 00 34 5a eb 00 34   .ÿjñ.... ...4Zë.4
0040  5a c9 62                                          ZÉb

0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 10   ........ ......E.
0010  00 35 94 1c 40 00 40 06  a8 94 7f 00 00 01 7f 00   .5..@.@. ".......
0020  00 01 87 20 00 17 e3 5e  dc 56 e3 2d 4b 8a 80 18   ... ..ã^ ÜVã-K...
0030  7f ff 5d a9 00 00 01 01  08 0a 00 34 5b 10 00 34   .ÿ]@.... ...4[..4
0040  5a eb 6f                                          Zëo

0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 10   ........ ......E.
0010  00 36 94 1d 40 00 40 06  a8 92 7f 00 00 01 7f 00   .6..@.@. ".......
0020  00 01 87 20 00 17 e3 5e  dc 57 e3 2d 4b 8a 80 18   ... ..ã^ ÜWã-K...
0030  7f ff be 45 00 00 01 01  08 0a 00 34 5c 4d 00 34   .ÿ¾E.... ...4\M.4
0040  5b 10 0d 00                                       [...
```

Le mot de passe est ici clairement exprimé. On peut lire bobo, la dernière trame transmet un retour chariot.

Nota : Notre étude ne se portant pas sur le service telnet, nous ne rentrerons pas sur le détail des transmissions et considérons que ces quelques remarques suffisent pour notre étude.

De la même manière, tout comme on vient de vous dérober votre mot de passe, on peut récupérer vos conversations, vos fichiers. La confidentialité est rompue, la porte est ouverte à l'insertion non−chalente de personnes non autorisées dans le système...

La solution proposée

Afin de ne plus être ouvert au monde entier, nous avons la possibilité de chiffrer nos échanges. Pour ce faire, nous allons installer un client ssh, **Openssh**. Client libre et gratuit. Par la même occasion, vous pourrez installer le serveur, afin que votre ordinateur puisse aussi être contacté par d'autres utilisateurs, vous permettant de délivrer des services sécurisés.

Nous allons continuer notre étude dans le cadre de ce logiciel, libre à vous d'adapter à vos préférences par la suite, sachant que le principal est ici transcrit.

Afin de mieux comprendre le comportement et l'utilité de chaque programme, décomposons les différents paquets ainsi que l'usage fait des exécutables.

Mémo

sshd : Serveur ssh

scp : Copie distante sécurisée

ssh−keygen : génération de clefs d'authentification, management et conversion

sftp : Transfert sécurisé de fichiers

slogin/ssh : Client ssh

ssh−add : Ajoute les identités DSA ou RSA à l'agent d'authentification

ssh−agent : Agent d'authentification

ssh−keyscan : Recueille les clefs publiques ssh

Nous traiterons chacune de ces applications afin de comprendre dans quels conetxte elles peuvent être utilisés.

Connexion à un hôte

Le client ssh opère selon un ordre défini :

1. les paramètres en ligne de commande
2. les informations spécifiques à l'utilisateur du fichier **~/.ssh/config**
3. la configuration globale du système dans le fichier **/usr/local/etc/ssh_config** ou **/etc/ssh/ssh_config** (cela dépendant de la distribution que vous utilisez)

Le fichier ssh_config :
Ce fichier représente la configuration du client ssh.

```
#$OpenBSD: ssh_config,v 1.15 2002/06/20 20:03:34 stevesk Exp $

# Ceci est le fichier de configuration par défaut des clients ssh du système. Voir
# ssh_config(5) pour de plus amples informations. Ce fichier fournit les paramètres par défaut
# pour les utilisateurs, ces valeurs peuvent être changées pour chaque utilisateur dans leur propre
# fichier de configuration ou sur la ligne de commande.

# Les données de configuration sont analysées comme ci-dessous :
# 1. Les options de la ligne de commande
# 2. Le fichier spécifique de l'utilisateur
# 3. Le fichier par défaut
# N'importe quelle valeur de configuration est uniquement changée la première fois qu'elle est mise.
# Ainsi, les définitions spécifique d'hôte doivent être au début du fichier de configuration
# et les options par défaut à la fin.

# Options par défaut

# Pour que les options soient prises en compte, il vous faut enlever le dièse précédent la ligne.

# Host permet de spécifier que la configuration qui suit concerne un ou plusieurs hôtes précis.
# Il est possible d'employer des caractères joker tels que * ou ?.

# Host *
# Specifie si la connexion à l'agent d'authentification doit être renvoyé vers la machine distante

# ForwardAgent no

# Autorise ou non la redirection du serveur graphique

# ForwardX11 no

# Autorise la méthode d'authentification par Rhosts. Peu sûr.

# RhostsAuthentication no

# Autorise la méthode d'authentification par Rhosts sur du RSA. Ne fonctionne que dans la version
# première du protocole et nécessite un setuid root. De préférence à ne pas utiliser.

# RhostsRSAAuthentication no
# Méthode d'authentification par RSA, ne fonctionne qu'avec la première version du protocole. Il
# faut que le fichier identity existe.

# RSAAuthentication yes

# Autorise la connexion par mot de passe, comme on pouvait la trouver dans rlogin ou telnet.

# PasswordAuthentication yes

# Si l'indicateur est positionné à yes, la demande de passephrase ou de mot de passe sera annulée.
#Cette option est utilisée dans le cas où seul des scripts interviennent et où il n'y a pas d'utilisateur
# présent pour insérer le mot de passe.

# BatchMode no

#Vérification de l'adresse IP de l'hôte qui se connecte dans le fichier known_hosts

# CheckHostIP yes

#Cette option permet de gérer l'ajout et le changement des clefs hôtes dans known_hosts.
#Si la clef a changé, la connexion vous sera refusée, vous indiquant le motif.

# StrictHostKeyChecking ask

#Localisation des fichiers identity, id_rsa, id_dsa

# IdentityFile ~/.ssh/identity
# IdentityFile ~/.ssh/id_rsa
# IdentityFile ~/.ssh/id_dsa

#Port de connexion à l'hôte distant.
```

```
# Port 22

#Version des protocoles supportés et désirés. Ici, on préfèrera employer la deuxième version si
#elle est disponible.

# Protocol 2,1

#Protocole de chiffrement (ssh v1) et protocoles disponibles par ordre de préférerance (ssh v2).

# Cipher 3des
# Ciphers aes128-cbc,3des-cbc,blowfish-cbc,cast128-cbc,arcfour,aes192-cbc,aes256-cbc

#Caractère d'échappement.

# EscapeChar ~
```

Il existe différentes options pour se connecter à un hôte via ssh. Libre à vous de les utiliser ou non.

-l login : Identifiant de l'utilisateur.
-v -vv -vvv : Mode verbeux, permet d'obtenir les messages de debugage plus ou moins complets (plus il y a de v, plus vous obtenez d'information, le nombre maximum étant 3).
-1 ou -2 : version de ssh employé. Il est déconseillé d'employer la version 1 du protocole. Bien qu'aucun exploit public ne circule, les faiblesses cryptographique du protocole ont été prouvée. De plus, la version 2 est reconnue par l'IETF
-p port : Numéro du port distant

Exemple d'utilisation :
```
ssh -vv -l utilisateur -p port -(1|2) hôte
```
ou
```
ssh utilisateur@hôte
```
L'hôte distant doit avoir un serveur ssh, nommé **sshd**, qui permet la connexion.

Création de paire de clefs

Ssh s'appuie sur des algorithmes à paire de clefs, ce qui signifie que vous disposez d'une clef publique, disponible pour tout un chacun et une clef privée dont vous gardez jalousement l'entrée. Ce système va nous permettre de nous identifier auprès des hôtes que nous désirons contacter. Il nous faut au préalable créer le trousseau.

```
$ ssh-keygen -t dsa
Generating public/private dsa key pair.
Enter file in which to save the key (/home/jop/.ssh/id_dsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/jop/.ssh/id_dsa.
Your public key has been saved in /home/jop/.ssh/id_dsa.pub.
The key fingerprint is:
4a:0b:3b:eb:ed:05:47:56:cb:23:28:d3:d7:81:69:08

$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/jop/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/jop/.ssh/id_rsa.
Your public key has been saved in /home/jop/.ssh/id_rsa.pub.
The key fingerprint is:
52:65:28:9a:8b:64:cb:b7:6e:70:75:10:d9:0a:01:d9
```

Pour les deux algorithmes (dsa, rsa), le système nous demande dans quel fichier nous désirons sauvegarder la clef. Les fichiers par défaut semblent une bonne solution. Par la suite, une passphrase nous est demandée. Celle-ci est un « mot de passe amélioré », car non limité à un mot ou une petite suite de caractères. Il faut cependant prendre des précautions, car en cas de perte de la passphrase, vous ne pourriez plus vous authentifier en tant que propriétaire authentique.

Nous allons maintenant voir **trois méthodes de connexion via ssh**.

Connexion par mot de passe

La première des méthodes, la plus connue et la plus usitée, reposant sur le modèle employé par rlogin ou rsh; l'hôte distant demande un mot de passe pour s'assurer de votre identité.

```
$ ssh -p 222 -l root 192.168.0.1
The authenticity of host '192.168.0.1 (192.168.0.1)' can't be established.
RSA key fingerprint is 74:4c:57:fd:c2:2c:0d:c3:3f:09:01:7d:e8:b7:21:24.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.0.1' (RSA) to the list of known hosts.
```

```
root@192.168.0.1's password:
Last login: Thu Dec 26 03:37:03 2002 from centaur
$
```

Je me connecte au port 222 sur une machine de mon réseau local, demandant de m'identifier comme root. La machine n'est pas connue dans mon fichier **known_hosts**. Ce fichier contient les clés hôte DSA des serveurs SSH auxquels l'utilisateur s'est connecté. Cette méthode de connexion est intéressante, mais minimise les capacités que vous avez avec ssh. Pour des connexions telles que vers un serveur CVS, un tunnel pour votre courrier, il serait fastidieux de s'authentifier à chaque fois par ce moyen.

Connexion par paires de clef

Puisque nous utilisons des algorithmes à paire de clefs (rsa, dsa), c'est à dire composée d'une clef secrète et d'une clef publique, il faut bien que cela nous serve à quelque chose. Nous allons donc automatiser la connexion. Pour ce faire, votre hôte contient un fichier **authorized_keys** dans le répertoire **.ssh** où vous vous connectez (en général un home directory). Il suffit de copier l'identifiant ou les identifiants de vos clefs publiques pour que vous soyez reconnu du serveur.

Attention, il faut que *PubkeyAuthentication* soit positionné à *yes* dans vos fichiers de configuration.
Pour insérer ma clef, j'ai plusieurs méthodes à ma disposition. Je peux employer des moyens conventionnels tels que le ftp ou le mail (à l'administrateur par exemple), ou je peux le faire au moyen d'outil sécurisé. Puisque c'est notre sujet, profitons en pour donner un exemple de *scp* sur lequel nous reviendrons ultérieurement.

```
$scp .ssh/id_dsa.pub jop@scipc-jpg:/home/jop/.ssh/dsa2connex
Warning: Permanently added 'scipc-jpg' (RSA) to the list of known hosts.
jop@scipc-jpg's password:
id_dsa.pub 100% |*****************************| 613 00:00
Mon fichier est maintenant copié sur l'hôte distant, il me reste à inclure la clef dant le fichier
authorized_keys. Une simple commande suffira :

$ cat dsa2connex >>authorized_keys
```

Je peux maintenant me connecter, l'hôte distant me reconnaît. Je peux me connecter sans mot de passe et avec une passphrase si j'en ai désiré une.

L'agent d'authentification

Nous savons maintenant nous connecter à un hôte distant connaîssant notre identité par l'intermédiaire de notre clef publique. Nous avons vu précédement que nous étions libre de mettre ou non une passphrase afin de chiffrer notre identification et compliquer l'usurpation qui pourrait en être faite [mode parano on]. S'il paraît fastidieux d'insérer un mot de passe à chaque connexion, cela l'est d'autant plus lorsque l'on doit rentrer une phrase entière. L'agent ssh est là pour nous simplifier la vie. Lancé au début de la session (terminal ou graphique), il retient la passphrase le temps où **ssh-agent** sera actif (le temps d'une session).

Pour une session en mode terminal :
```
$ssh-agent screen
$ssh-add
```

Après avoir lancé l'agent ssh, on ajoute la passphrase à l'agent par l'intermédiaire de **ssh-add** ; tous les hôtes distants disposant de votre clef publique ne vous demanderons alors plus la passphrase puisque gérée par l'agent ssh.

De même, en mode graphique :
```
$ssh-agent startx
```

Il vous suffit ensuite d'ouvrir un terminal et de taper
```
$ssh-add
```
L'agent sera actif pour toutes les applications utilisées en mode graphique.

Une dernière méthode consiste à **lancer l'agent-ssh** qui fournit un certains nombre de variables à exporter, puis demander l'action de ssh-add :

```
$ssh-agent
```

On récupère les variables fournies et on les exporte :

```
$SSH_AUTH_SOCK=/tmp/ssh-XXy0hiXW/agent.2019; export SSH_AUTH_SOCK;
$SSH_AGENT_PID=2020; export SSH_AGENT_PID;
$echo Agent pid 2020;
```

On ajoute :

```
$ssh-add
```

Pour tuer l'agent ssh, il suffit de faire :

```
$ ssh-agent -k
unset SSH_AUTH_SOCK;
unset SSH_AGENT_PID;
echo Agent pid 2020 killed;
```

Invalidité des clefs d'un hôte connu

Comme nous l'avons vu précédement, lors de votre connexion, il vous est possible d'enregistrer l'emprunt (fingerprint) fournit par le serveur distant. Dans le cas où celle−ci serait modifiée, vous seriez immédiatement prévenu. Cela peut découler de quatre choses :

- Vous vous êtes fait pirater et l'on a modifié les clefs présentes dans votre known_hosts.
- Le serveur distant c'est fait pirater et la valeur de ses clefs a changé.
- La troisième peut correspondre à une attaque en bon et due forme d'un man−in−the−middle.
- La dernière et néanmoins plus plausible des solution nécessite que l'administrateur du serveur distant ait changé les clefs.

La seule solution fiable demande de contacter l'administrateur en question et lui demander ce qu'il en est. Dans la pratique, peu de personnes usent de cette assurance de sécurité.

```
$ ssh 127.0.0.1
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@ WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED! @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man−in−the−middle attack)!
It is also possible that the RSA host key has just been changed.
The fingerprint for the RSA key sent by the remote host is
8b:d5:26:a3:79:ed:25:0f:3b:6f:fe:30:1f:ed:89:ae.
Please contact your system administrator.
Add correct host key in /home/jop/.ssh/known_hosts to get rid of this message.
Offending key in /home/jop/.ssh/known_hosts:13
RSA host key for 127.0.0.1 has changed and you have requested strict checking.
Host key verification failed.
$
```

La copie sécurisée

Comme nous avons pu le voir plus haut, ssh fournit un outil de copie sécurisée en standard, sous le nom de scp pour **SecureCoPy**. Il remplace son ancêtre rcp. Son usage est très simple :

```
scp hôte_d_ou_je_veux_copier:source_copie hôte_destination:cible
```

Lorsque l'hôte correspond à la machine où vous vous trouvez, il n'est pas nécessaire de l'inscrire.

```
$scp iptables scipc−jpg:/home/jop/download
iptables 100% |*****************************| 27654 00:00
```

Vous pouvez également faire des copies récursives, comme nous le ferions avec n'importe quel autre utilitaire de copie.

```
$scp −r download scipc−jpg:/home/jpg/dl
img2.png 100% |*****************************| 277 00:00
internals.pl 100% |*****************************| 188 00:00
labels.pl 100% |*****************************| 271 00:00
images.pl 100% |*****************************| 624 00:00
portscanning.css 100% |*****************************| 891 00:00
index.html 100% |*****************************| 56753 00:00
```

Le transfert de fichier sécurisé

Tout comme on peut copier des fichiers à distance par l'intermédiaire de scp, il est également possible de transférer des fichiers par l'intermédiaire d'un ftp sécurisé nommé **SecureFTP**.
Bien que l'outil soit encore trivial (pas de reprise de chargement en cas de coupure, pas d'indication du temps restant, pas de telechargement ou de chargement réccursif, ...), il vous permettra de faire toutes les manipulations basiques disponibles sur un ftp non sécurisé.

Commandes disponibles sur **sftp** :

`cd path`	Change le répertoire distant vers 'path'
`lcd path`	Change le répertoire local vers 'path'
`chgrp grp path`	Change le groupe de fichier 'path' par 'grp'
`chmod mode path`	Change les permissions du fichier 'path' à 'mode'
`chown own path`	Change le propriétaire du fichier 'path' par 'own'
`help`	Affiche ce message d'aide
`get remote−path [local−path]`	Télécharge le fichier
`lls [ls−options [path]]`	Affiche le listing du répertoire local
`ln oldpath newpath`	Crée un lien symbolique du fichier distant
`lmkdir path`	Crée un répertoire local
`lpwd`	Affiche le répertoire courant
`ls [path]`	Affiche le listing du répertoire distant
`lumask umask`	Positionne l'umask local à 'umask'

`mkdir path`	Crée le répertoire distant
`put local-path [remote-path]`	Charge le fichier
`pwd`	Affiche le répertoire courant distant
`exit`	Quitte sftp
`quit`	Quitte sftp
`rename oldpath newpath`	Renomme le fichier distant
`rmdir path`	Supprime le répertoire distant
`rm path`	Supprime le fichier distant
`symlink oldpath newpath`	Crée un lien symbolique du fichier distant
`version`	Affiche la version de sftp
`!command`	Exécute la 'commande' dans un shell local
`!`	Sort vers un shell local
`?`	Affiche ce message d'aide

```
$ sftp jop@scipc-jpg
Connecting to scipc-jpg...
$sftp> lpwd
Local working directory: /home/jop
$sftp> lcd download
$sftp> lpwd
Local working directory: /home/jop/download
$sftp> put iptables
Uploading iptables to /home/jop/iptables
$sftp> quit
$
```

Le tunnel et le Xforwarding

Tout au long de cet article, vous avez pu découvrir des applications connues, il nous manque cependant l'exportation des applications distantes. En effet, à l'aide de telnet, vous pouviez utiliser un logiciel non présent sur votre machine, mais présent sur l'ordinateur distant. Il suffisait de taper :

```
$export DISPLAY=mon_adresse:0.0
```

Comme par magie, l'application arrivait plus ou moins rapidement selon votre connexion sur votre écran. C'est ce qu'on appelle du **Xforwarding**. L'avantage une fois de plus d'utiliser ssh réside dans la connexion chiffrée et l'impossibilité à un agresseur éventuel de lire ce que vous faîtes via le réseau.

```
$ssh jop@scipc-jpg gedit
```

Dans l'exemple donné, nous demandons d'ouvrir la connexion ssh pour inscrire Gedit à l'intérieur (bloc-notes de Gnome). En fermant Gedit, nous fermerons la connexion ssh.
Pour que cela soit réalisable, n'oubliez cependant pas d'**activer l'option X11Forwarding** dans les fichiers de configuration.

Toujours dans la même idée de tunneler des applications, ssh vous permet d'encapsuler des protocoles. Cela est très intéressant lorsque vous avez recours à des protocoles tels que smtp, pop, ... Vous courriers, pour l'exemple, ne serons plus à l'indiscrétion de vos fournisseurs, ni des fouines du réseau.

`ssh -L port_local:hôte_distant:port_distant nom_utilisateur@nom_hôte` Mettons que je veuille encapsuler les ports 25 (smtp) et 110 (pop) `$ssh -L 2025:scipc-jpg:25 jop@scipc-jpg`
`$ssh -L 2110:scipc-jpg:110 jop@scipc-jpg`

Il vous suffit de configurer votre logiciel de messagerie favori aux ports 2025 pour le smtp et 2110 pour le pop pour recevoir correctement votre courrier, et ce de manière sécurisée.

Où se trouve quoi ?

Le paquet **OpenSSH** fournit : /etc/ssh
/etc/ssh/moduli
/usr/bin/scp
/usr/bin/ssh-keygen
/usr/libexec/openssh
/usr/libexec/openssh/ssh-keysign
/usr/share/doc/openssh-3.5p1
/usr/share/doc/openssh-3.5p1/CREDITS
/usr/share/doc/openssh-3.5p1/ChangeLog
/usr/share/doc/openssh-3.5p1/INSTALL
/usr/share/doc/openssh-3.5p1/LICENCE
/usr/share/doc/openssh-3.5p1/OVERVIEW
/usr/share/doc/openssh-3.5p1/README
/usr/share/doc/openssh-3.5p1/README.privsep
/usr/share/doc/openssh-3.5p1/README.smartcard
/usr/share/doc/openssh-3.5p1/RFC.nroff
/usr/share/doc/openssh-3.5p1/TODO
/usr/share/doc/openssh-3.5p1/WARNING.RNG

/usr/share/man/man1/scp.1.gz
/usr/share/man/man1/ssh−keygen.1.gz

Le paquet **OpenSSH−client** fournit :
/etc/ssh/ssh_config
/usr/bin/sftp
/usr/bin/slogin
/usr/bin/ssh
/usr/bin/ssh−add
/usr/bin/ssh−agent
/usr/bin/ssh−keyscan
/usr/share/man/man1/sftp.1.gz
/usr/share/man/man1/slogin.1.gz
/usr/share/man/man1/ssh−add.1.gz
/usr/share/man/man1/ssh−agent.1.gz
/usr/share/man/man1/ssh−keyscan.1.gz
/usr/share/man/man1/ssh.1.gz
/usr/share/man/man5/ssh_config.5.gz

Le paquet **OpenSSH−server** fournit :
/etc/pam.d/sshd
/etc/rc.d/init.d/sshd
/etc/ssh
/etc/ssh/sshd_config
/usr/libexec/openssh/sftp−server
/usr/sbin/sshd
/usr/share/man/man5/sshd_config.5.gz
/usr/share/man/man8/sftp−server.8.gz
/usr/share/man/man8/sshd.8.gz
/var/empty/sshd
Les documentations sont répertoriées en bordeaux Les fichiers de configuration en vert Les exécutables en bleu

Conclusion

Georges Orwell écrivait 1984 bien des années plus tôt et nommait un ennemi invisible Big Brother en avançant le slogan « *Big Brother is watching you* ». Non loin de cette réalité, nous avons atteint avec Internet l'ouverture de l'outil informatique aux masses. Ne pouvant être en mesure de s'assurer que personne ne s'introduit dans votre intimité, le conseil reste d'être prudent avec vos données sensibles et personnelles. Oubliez donc les vieux protocoles non sécurisés et travaillez en toute sécurité...

Bibliographie

RFCs

Telnet − numéro 854
SSH − Groupe de travail IETF
SSH − SCP − SFTP − Alix Mascret
Secure Shell − Denis Bodor
Formation Linux Debian − Alexis de lattre

Logiciel

OpenSSH − Secure Shell
OpenSSL −− Secure Socket Layer
Ethereal − sniffeur
OpenOffice.org − suite bureautique
The gimp − Traitement d'image
Ksnapshot − Impression d'écran

Je remercie tout particulièrement Anne pour le temps qu'elle a passé à la relecture de cet article ainsi qu'à sa correction et sa mise en forme.

Les ponts filtrants

Tchesmeli serge

Un firewall totalement invisible

Introduction

Un pont est un équipement réseau qui permet de relier deux sous−réseaux de manière totalement transparente. Pour ceux qui connaissent un peu le modèle OSI, il effectue une interconnexion au niveau 2, c'est−à−dire qu'il travaille au niveau des trames.
En gros, un pont écoute toute l'activité de chaque sous−réseau auquel il est connecté, les stocke en mémoire et les redirige vers le (ou les) sous−réseaux concernés.
Il n'a pas besoin d'adresse MAC (donc pas besoin d'adresse IP non plus) pour fonctionner, et est donc totalement indétectable.

Schématiquement, on peut représenter un pont reliant deux réseaux de cette facon (le pont est la machine servant de pont, `eth0` et `eth1` ses deux cartes réseau):

```
Réseau 1 −−−−−−−| eth0 Pont eth1 |−−−−−−−− Réseau 2
```

Un pont filtrant, lui, est un pont comme décrit plus haut, mais va en plus appliquer des régles de filtrage (firewalling) sur les trames qui vont le "traverser". On vas donc obtenir un firewall qui ne posséde pas d'adresse réseau, donc indétectable. Difficile pour un "pirate" d'attaquer une machine qui n'est pas visible sur un réseau, et qui ne permet aucune connexion réseau car elle n'as pas d'adresse réseau.

Un pont filtrant a aussi un autre avantage : il s'interconnecte de façon totalement transparente sur votre réseau, vous n'avez absolument pas besoin de modifier la topologie de votre réseau (routage, passerelle,...) pour l'ajouter. Il se comporte en gros comme un simple cable réseau intelligent qui filtrerait le traffic qui le traverse.

Pré−requis

Tout d'abord vous devez avoir un ordinateur avec deux cartes réseaux pour pouvoir interconnecter /filtrer les deux réseaux. Le support du "pont" est intégré à partir du kernel 2.4. , mais vous devez prendre le dernier kernel à jour (2.4.19 lors de la rédaction de ce document) pour avoir le meilleur support du pont. Vous devez aussi appliquer un patch kernel, pour ajouter le support du pont à iptable (pour pouvoir filtrer les paquets qui vont traverser le pont):
bridge−nf−0.0.7−against−2.4.19.diff
Ceci est le patch pourle kernel 2.4.19. Si lors de la lecture de ce document, le dernier kernel n'est plus le 2.4.19, récupérez alors le patch pour la version du kernel actuelle que vous trouverez ici:
Site officiel "pont filtrant" pour Linux

Appliquez alors le patch à votre kernel, pour le 2.4.19 par exemple, allez dans le répertoire des sources du kernel et faites un :

```
patch −p1 < bridge−nf−0.0.7−against−2.4.19.diff
```

Recompilez alors votre kernel, en activant dans la partie `Network Options` :
`Network Packet Filtering (remplace ipchains)` (em module ou en natif)
Passez en module (ou en natif) toutes les options de :
`IP: Netfilter Configuration`

et activer aussi le support du pont :

```
[*] 802.1d Ethernet Bridging
```

A partir de ce moment, notre machine peut faire office de pont, et filtrer les paquets du pont.
Il vous faut aussi les utilitaires de configuration, c'est à dire les programmes pour configurer le pont lui−même, que vous téléchargez ici:
bridge−utils−0.9.6.tar.gz

Comme avec n'importe quelle source de programmes, vous "détarballez" et compilez le tout :

```
tar zxvf bridge−utils−0.9.6.tar.gz
cd bridge−utils
./configure
make
make install
```

Configuration du pont lui même

Comme nous l'avons dit plus haut, le pont en lui−même n'a pas besoin d'adresse IP (ni même d'adresse MAC) pour fonctionner. Donc dans un premier temps, nous enlevons toute adresse IP des deux cartes réseaux qui constituent notre pont, et les passons en mode "promisc" pour qu'elles puissent écouter tout le réseau et donc transférer les trames réseaux d'un réseau à l'autre.
ATTENTION Si vous etes en telnet ou ssh sur cette machine, la connexion va être coupée, faites donc ces manipulations en console sur la machine elle−même.

```
ifconfig eth0 0.0.0.0 promisc
ifconfig eth1 0.0.0.0 promisc
```
Maitenant nous mettons en place le pont lui−même, c'est−à−dire nous donnons un nom au bridge lui−même (mon−pont) et déclarons les interface réseaux qui vont servir au pont (eth0 et eth1):

```
brctl addbr mon−pont

brctl addif mon−pont eth0
brctl addif mon−pont eth1
```
Pour certains équipement actifs (switch, routeur, etc) le pont doit se "déclarer" pour que tout fonctionne bien. Si la connexion entre les deux réseaux ne fonctionne pas, essayez ces commandes (mais dans 90% des cas cela n'est pas nécessaire) :

```
brctl sethello mon−pont 1
brctl setfd mon−pont 4
```

Pour vérifier que tout marche, normalement toute machine du réseau1 doit pouvoir pinguer toutes les machines du réseau2 et inversement. Et si vous faites un "traceroute" d'une machine du réseau1 vers le réseau2 (ou inversement), vous ne verrez PAS le pont. Il se comporte là comme un simple "fil" reliant les deux réseaux.

Par contre, vous avez peut−être besoin d'accéder en réseau à la machine qui sert de pont (pour faire du ssh ou autre), il faut donc lui donner une adresse IP. Mais comment faire étant donné que les deux interfaces réseaux servent au pont ? Il y a deux solutions :

− Soit vous ajoutez une 3éme carte réseau a cette machine, que vous configurez normalement, avec une adresse IP etc....
− Soit vous configurez le pont lui−même AVEC une adresse Ip. Le pont se comporte donc exactement comme une interface réseau et pour lui affecter une adresse IP, on le fait de la même manière que pour une interface "normale", par exemple :

```
ifconfig mon−pont 192.168.1.50 netmask 255.255.255.0
```

Et si vous refaites des traceroute, vous ne le verrez toujours PAS sur le résultat du traceroute.

Placez toute ces commandes dans un fichier de démarrage de votre machine bien sur, pour ne pas avoir à tout retaper à chaque reboot.

Mise en place du Filtrage

Maitenant que notre pont est opérationnel, il ne reste plus qu'à lui appliquer les règles de filtrage que l'on souhaite. Comme le traffic réseau ne fait QUE le traverser, les règles de filtrage ne sont que des règles de forwarding bien sûr (pas de INPUT ni OUPUT). On applique les règles comme pour un firewall de type "gateway" sauf la chaîne "FORWARD".

Je vous donne ici un exemple de filtrage (à vous d'adapter, suivant votre réseau, vos applications − voir la rubrique iptable pour cela).
Appeler ce fichier `rc.firewall` par exemple et lancez−le au boot de la machine.

```
#!/bin/bash
# Script Firewalling exemple

PATH="/usr/sbin:$PATH"

# On vide les chaines
iptables −F

#On efface toute les chaines utilisateurs
iptables −X

#Notre réseau que l'on sécurise:
LANS="192.168.0.0/255.255.255.0"

# Adresse de notre serveur web et ftp
WEB="192.168.0.10"
FTP="192.168.0.20"

#On créé la chaine KEEP_STATE pour suivre l'état des connexions
iptables −N KEEP_STATE
iptables −F KEEP_STATE

#On drop les paquets non valides
iptables −A KEEP_STATE −m state −−state INVALID −j DROP

#On accepte les paquets provenant bien
#d'une connexion et dont l'etat est correct
iptables −A KEEP_STATE −m state −−state RELATED,ESTABLISHED −j ACCEPT


#On drop et log tout paquet dans un état incorect et
#qui permet des scans/Denial of service
iptables −A FORWARD −p tcp −−tcp−flags ALL FIN,URG,PSH \
  −m limit −−limit 5/minute \
  −j LOG −−log−level notice −−log−prefix "NMAP−XMAS: "
iptables −A FORWARD −p tcp −−tcp−flags ALL FIN,URG,PSH −j DROP
iptables −A FORWARD −p tcp −−tcp−flags SYN,FIN SYN,FIN \
```

```
  -m limit --limit 5/minute \
  -j LOG --log-level notice --log-prefix "SYN/FIN: "
iptables -A FORWARD -p tcp --tcp-flags SYN,FIN SYN,FIN -j DROP
iptables -A FORWARD -p tcp --tcp-flags SYN,RST SYN,RST \
  -m limit --limit 5/minute \
  -j LOG --log-level notice --log-prefix "SYN/RST: "
iptables -A FORWARD -p tcp --tcp-flags SYN,RST SYN,RST -j DROP
iptables -A FORWARD -p tcp --tcp-flags RST RST,ACK \
  -m limit --limit 5/minute \
  -j LOG --log-level notice --log-prefix "RST/ACK: "
iptables -A FORWARD -p tcp --tcp-flags RST RST,ACK -j DROP

#On passe les paquets valides dans notre chaine KEEP_STATE
iptables -A FORWARD -j KEEP_STATE

#On permet le controle de "time exceeded"
#et de "port not found" sur notre réseau
iptables -A FORWARD -p udp -s $LANS --dport 11 -j ACCEPT
iptables -A FORWARD -p udp -s $LANS --dport 3 -j ACCEPT

#On permet le http sur notre serveur web
iptables -A FORWARD -p tcp -d $WEB --dport 80 -j ACCEPT

#On permet le Ftp sur notre serveur FTP
iptables -A FORWARD -p tcp -d $FTP --dport 21 -j ACCEPT

# Tout notre réseau a le droit de "sortir" vers l'internet
iptables -A FORWARD -s $LANS -j ACCEPT

# Tout ce qui n'est pas déclaré doit etre bloqué
iptables -A FORWARD -j DROP
```

XINETD sous toutes ses coutures

par Anne

Où comment utiliser xinetd pour sécuriser encore plus votre pingouin préféré

Avant propos

Cet article a pour objectif d'expliquer le fonctionnement et la configuration du super démon `xinetd`. Pourquoi particulièrement celui−ci ? Parce qu'il est utilisé fréquemment pour l'accès à nombre de services réseau et devient un outil non négligeable de sécurisation de celui−ci (en plus d'un bon firewall bien sûr). Cet article n'a pas pour objectif d'être complètement exhaustif mais de présenter les configurations les plus courantes de `xinetd`. Pour plus d'infos, consulter le man (on s'en serait douté ;))

Inetd ou Xinetd ?

Selon la distribution vous trouverez soit un démon `inetd`, soit un démon `xinetd`. La tendance est tout de même d'utiliser de plus en plus ce dernier.

Définition

Dans l'absolu, `inetd` et `xinetd` ont le même rôle, à savoir de piloter l'accès à un ou plusieurs services réseaux. Ils agissent comme une standardiste. Ils reçoivent des requêtes de clients, extérieurs pour la plupart, qui demandent un accès à un service réseau déterminé (ex : ftp, telnet, ssh...). Le super démon va, en fonction des instructions qu'on lui aura données (fichiers de configuration) transmettre ou rejeter l'appel.

Ce qu'apporte xinetd

`inetd`, jusque là utilisé, permettait, grâce au fichier `/etc/inetd.conf`et au wrapper `tcpd`, de paramétrer l'accès aux services en l'autorisant/interdisant totalement ou partiellement. (cf. les fichiers `/etc/hosts.allow` et `/etc/hosts.deny`).

`xinetd` apporte des fonctionnalités bien plus importantes et permet d'affiner les paramétrages d'accès aux services. On citera dans le désordre :

- possibilité d'affiner les logs des services gérés
- paramétrage d'accès par service et non global
- paramétrage des plages horaires de disponibilité des services
- possibilité de chrooter les services (ex : ftp)
- possibilité de limiter les attaques de type deny of service (contrôle de la priorité d'un serveur, contrôle de la charge CPU, contrôle du nombre de connexions par service, ...)
- redirection de ports

Passer de inetd à xinetd

Si votre distribution préférée utilise `inetd` et qu'après avoir eu le courage de lire cet article, vous souhaitez utiliser `xinetd`, il existe un script qui vous permettra de transformer le fichier `inetd.conf` en fichier `xinetd.conf`, utilisable par `xinetd`

Ce script, `xconv.pl`, est un script perl fourni avec `xinetd`. Attention si ce script peut vous donner la structure générale du fichier de configuration, il ne vous permettra pas de profiter des apports de `xinetd`. Rien de tel qu'un bon éditeur de texte et ce qui suit ci−dessous.

configuration générale de xinetd

On entre dans le vif du sujet :). Pour configurer `xinetd`, vous aurez à connaître la syntaxe, commune de `/etc/xinetd.conf` et, selon les cas de figure, les fichiers situés dans le répertoire `/etc/xinetd.d`.

L'arborescence de xinetd

L'arborescence de la configuration de `xinetd` est relativement simple. On en rencontre 2 types :

- **un seul fichier de configuration** : `/etc/xinetd.conf` qui comprendra la configuration générale de `xinetd` **et** la configuration des services gérés par xinetd. (exemple plus loin dans l'article)
- un fichier réservé à la **configuration générale** de `xinetd`, nommé aussi `/etc/xinetd.conf`. La **configuration des services** est déportée dans des fichiers situés dans le répertoire `/etc/xinetd.d`. Ce répertoire comprend un fichier de configuration par service géré par `xinetd`. Le fichier porte le nom du service.
  Pour utiliser ce deuxième cas de figure, le fichier `/etc/xinetd.conf` doit contenir la ligne suivante :
  `includedir /etc/xinetd.d`

C'est ce deuxième cas de figure qui est le plus couramment utilisé dans les distributions.

Syntaxe générale d'un fichier de configuration

Le fichier de configuration de `xinetd` est un ensemble d'une ou plusieurs directives dont la syntaxe est la suivante :

```
service nom_du_service
   {
    ...
   }
```

Le nom de la directive est soit `defaults` et la configuration porte alors sur l'ensemble des services gérés par `xinetd`, soit le nom d'un des services géré par `xinetd`. A l'intérieur de chacune de ces directives on trouvera des attributs, un par ligne, écrits de la manière suivante :
`<attribut> <assignement> <valeur> <valeur> ...`
Les attributs seront détaillés plus loin. Les assignements peuvent prendre différentes valeurs : `=, −=, +=`.

Exemple :

```
root@pingu# cat /etc/xinetd.d/telnet
service telnet
{
   ...
   server  = /usr/sbin/in.telnetd
   ...
}
```

avec :
attribut : server
assignement : "="
valeur : /usr/sbin/in.telnetd

Pour reprendre l'arborescence de `xinetd` et les 2 cas de figure exposés ci−dessus, on obtiendra les fichiers suivants :

un seul fichier de configuration	un fichier de configuration par service

/etc/xinetd.conf
defaults
{
...
}
service service1
{
...
}
service service2
{
...
}
	/etc/xinetd.conf
	defaults
	{
	...
	}
	/etc/xinetd.d/service1
	service service1
	{
	...
	}
	/etc/xinetd.d/service2
	service service2
	{
	...
	}

Écriture des fichiers : attributs obligatoires

Nous allons lister les différents attributs utilisables pour configurer `xinetd`. Certains sont facultatifs et vous permettent d'affiner son rôle. Toutefois d'autres sont obligatoires et s'ils ne sont pas présents, empêcheront tout ou partie des services de fonctionner.

Avant de les lister, petit point de vocabulaire : on distingue les services dits internes et externes. A quoi ? A `xinetd` bien sur :). Les services internes, comme `servers, services, xadmin` sont des services propres à `xinetd` qui fournissent des informations sur le fonctionnement du super−démon. Il vaut mieux ne pas utiliser ces services car ils exposent la machine inutilement et les fichiers de log de xinetd vous fourniront les mêmes informations.

Voilà donc les attributs que vous devrez utiliser :

Attribut	Définition
socket−type	type de socket utilisé pour le service : `dgram` s'il utilise le protocole UDP, `stream` s'il utilise le protocole TCP − consulter le fichier `/etc/services` pour avoir l'information.
user	identité sous laquelle le service sera lancé
server	chemin et nom du serveur
wait	Définit le comportement du service dans le traitement des threads : `yes` pour un service mono−thread (une connexion simultanée par service et une seule), `no` pour un service multithread (possibilité d'avoir plusieurs connexions simultanées au service)
protocol	Protocole utilisé par le service. Si rien n'est précisé, c'est le protocole spécifié dans le fichier `/etc/services` qui sera utilisé.
rpc_version **rpc_number**	Ne concerne que les services basés sur les RPC (exemple : NFS)
port	Port associé au service. Là encore, s'il n'est pas précisé, c'est le port spécifié pour le service dans le fichier `/etc/services`.

Affiner les logs avec xinetd

Une des fonctionnalités de `xinetd` est de permettre d'affiner ce que vous voulez loguer et dans quel(s) fichier(s) vous voulez le loguer.

Localisation des logs

C'est l'attribut **`log_type`** qui va donner cette localisation. Il peut prendre 2 valeurs :

- **`SYSLOG syslog_facility [syslog level]`** : les logs seront alors gérés par le démon `syslogd`. Vous pourrez préciser le *niveau* de log (voir man syslog) : avec les valeurs `emerg, alert, crit, err, warning, notice, info, debug`. L'ordre de ces valeurs indique une quantité croissante d'informations récupérées par syslogd. Par défaut, c'est le niveau `info` qui est utilisé. Vous retrouverez ensuite les logs généralement dans le fichier `/var/log/messages` (cf. `/etc/syslog.conf`).
  Inconvénient de cette méthode : toutes les informations sont stockées dans un seul fichier et difficiles à lire étant donné la quantité d'éléments recueillis pour le noyau.

- **`FILE file [soft limit [hard limit]]`** : vous allez pouvoir complètement configurer la destination de vos logs et dans l'absolu, prévoir un fichier de log par service géré par xinetd. Le mot−clé `FILE` sera suivi du nom du fichier. Si celui−ci n'existe pas, il sera créé.
  De manière facultative, vous pouvez également préciser 2 types de limites à la taille du fichier de log : une *limite soft* qui lorsqu'elle est franchie provoque l'envoi d'un message d'alerte (mais les logs continuent d'alimenter le fichier) et une *limite hard*, qui lorsqu'elle est franchie bloque l'envoi de logs supplémentaires dans le fichier concerné. L'utilisation de ces deux limites est recommandée pour éviter de saturer le système de fichiers.
  Les limites sont à donner en octets (1000b), en kilo−octets (1000k) ou en mega−octets (1000m).

Contenu des logs

Outre la localisation des logs, il est possible également de paramétrer le contenu des logs. On utilisera les attributs `log_on_success` et `log_on_failure` qui, comme l'intitulé l'indique, listent ce qui sera logué en cas de succès et d'échec de l'accès au service. Les valeurs sont :

- `PID` : numéro de process du serveur lancé
- `HOST` : adresse distante cliente du serveur
- `USERID` : user id de l'utilisateur distant
- `DURATION` : durée de la session

Les 4 valeurs sont utilisables avec l'attribut `log_on_success`. Seuls `HOST` et `USERID` sont utilisables avec `log_on_failure`.

Exemple :

```
service trucmuche
{
   ...
   log_type = FILE /var/log/xinetd/trucmuche.log
   log_on_success  = PID USERID HOST DURATION
   log_on_failures = HOST USERID
   ...
}
```

Dans ce cas de figure, le fichier de log du service trucmuche sera `/var/log/xinetd/trucmuche.log`. Les informations recueillies dans tous les cas seront l'adresse du client et son identité et en cas de succès, on aura également le PID du serveur et la durée de la session

Xinetd pour contrôler les accès à votre machine

Xinetd dispose de nombreux attributs complémentaires, facultatifs mais qui vont permettre d'en faire un outil de sécurisation des services réseaux et du système dans son ensemble. Ci−dessous, les principales fonctionnalités proposées.

Contrôler l'origine des accès

Avec des attributs supplémentaires, vous allez pouvoir filtrer les clients qui vont pouvoir ou non se connecter à vos serveurs.

Filtrer des adresses : `only_from = valeur [valeurs...]`.
La connexion au service ne sera possible qu'à partir de la liste fournie à cet attribut. Elle peut contenir :

- des adresses IP : facile à comprendre je ne m'étends pas.
- des adresses réseau : 192.168.0.0 par exemple. Seules les adresses IP de ce réseau pourront accéder au service
- des hostnames : Ce sont des noms de machine. A utiliser si et seulement si le fichier `/etc/hosts` est correctement renseigné. La résolution de nom se fait au moment de l'accès.
- des noms de domaine : lea−linux.org par exemple. Seul le domaine de Léa pourra alors accéder à votre service.

Vous pouvez bien sûr panacher les valeurs.

Exemple : configuration des accès ftp sur ma machine :

```
service proftpd
 {
   ...
   only_from = citrouille 192.168.1.0 lea−linux.org
   ...
 {
```

Les seuls autorisés à utiliser mon serveur ftp seront : la machine "citrouille", les machines du réseau 192.168.1.0 et celles du domaine lea−linux.org.

Un attribut qui a également pour effet de filtrer les accès est **`no_access = valeur [valeurs...]`**. Il fonctionne exactement de la même façon que `only_from` sauf qu'il détermine les machines, adresses IP, hostnames, adresses réseaux et/ou noms de domaine pour lesquels vous voulez interdire l'accès à votre service.

Attention, encore une fois, `xinetd` utilisé seul ne vous garantira pas la sécurité de votre système. Il est essentiel d'y adjoindre un bon firewall.

Contrôler le moment des accès

Vous pouvez choisir le moment auquel vous autoriserez les accès à tout ou partie de vos services réseaux. On utilisera l'attribut `access_times`. Il vous permet de définir une ou plusieurs plages horaires pendant lesquelles la connexion sera possible.

Syntaxe : `access_times = interval [interval...]`
L'intervalle de temps s'écrit : heures:minutes−heures:minutes

Exemple : Je veux limiter l'accès de mon ftp de 9h à 12h et de 14h à 16h.
```
service proftpd
 {
   ...
   access_time = 9:00−12:00 14:00−16:00
   ...
 {
```

Contrôler l'exposition du système pendant l'accès : chroot

Véritable couteau suisse de la configuration de services réseau, `xinetd` vous permet de "chrooter" un service. *Rappel* : la commande `chroot` permet de lancer un programme en restreignant ses accès disques à une sous arborescence. En fait pour le processus, la racine du disque est la racine de l'arborescence dans laquelle il a été restreint.

L'attribut `server_args` va nous permettre d'automatiser le chroot : la commande `chroot` est considérée comme le serveur et le service est passé en argument.

Exemple : Je veux chrooter mon serveur ftp.
```
service proftp
 {
   ...
   server = /usr/sbin/chroot
   server_args = /opt/proftpd/proftpd
   ...
 {
```

Lorsqu'un client tente un accès ftp, `chroot` est exécuté en tant que serveur et `proftpd` en tant qu'argument, ce qui revient à la commande connue : `/usr/bin/chroot opt/proftpd/proftpd`.

Autoriser / interdire un service

Dans la plupart des distributions, les services installés et gérés par `xinetd` sont désactivés d'office, pour des raisons de sécurité. D'autre part vous pouvez, pour un temps, choisir de désactiver complètement un service. Tout ceci est lié à l'utilisation de l'attribut `disable`.

Syntaxe : `disable = yes|no`

Exemple : Je veux désactiver telnet.
```
service telnet
 {
   disable = yes
   ...
 {
```

Xinetd pour limiter les attaques de type Deny of Service

- **Contrôle de la charge CPU : `rlimit_cpu = seconds`**. Cet attribut vous permet de limiter le temps CPU utilisé par un ou plusieurs services. Un des effets induits par une attaque de type Deny Of Service.
- **Priorité accordée au processus serveur : `nice = level`**. Fonctionnant comme avec la commande `nice`, l'attribut permet de fixer une priorité d'ordonnancement pour le serveur. Le `level` peut prendre les valeurs de −20 (le plus prioritaire) à 19 (le moins prioritaire). Cela vous permet par exemple de passer en process moins prioritaire un serveur ftp par rapport aux process de vos applications courantes. Après tout c'est votre machine ;).
- **Limite du nombre de connexions par service : `instances = value`**. L'attribut détermine le nombre d'instances simultanées du serveur qui seront autorisées. Préciser un nombre. Sans précision, le nombre d'instances pourra être illimité.
- **Limite du nombre de connexions ayant la même origine : `per_source = value`**. Non seulement vous pouvez filtrer les adresses IP clientes, le nombre d'instances du serveur mais vous pouvez aussi limiter le nombre de connexions à un serveur donné provenant d'une même adresse IP. Ceci est une limite non négligeable aux attaques dues à des connexions multiples sur vos serveurs accessibles.

- **Blacklister des adresses IP** : Il vous est possible blacklister des adresses IP qui tenteraient des connexions sur des services que vous avez désactivés mais qui constituent la cible préférée des hackers (exemple : telnet). On utilisera 2 attributs de manière combinée :

```
flags = SENSOR
deny_time = minutes
```

SENSOR empêche toute connexion au service concerné et stocke l'adresse IP pendant un temps déterminé par l'attribut `deny_time`. Si cette même adresse tente de se connecter à **n'importe quel service géré par xinetd**, il sera automatiquement bloqué. Le temps est déterminé en minutes, mais vous pouvez utiliser également la valeur FOREVER : l'IP restera blacklistée jusqu'au prochain redémarrage de `xinetd`.

Autres fonctionnalités de xinetd

Je ne verrai pas ici toutes les possibilités offertes par xinetd mais 2 en particulier : la redirection de port et l'attribution d'un service à une interface réseau.

Redirection de ports

Même si ce n'est pas sa fonction principale, `xinetd` peut vous permettre de faire de la redirection de port, si vous ne souhaitez pas approfondir `iptables`. On utilisera pour cela l'attribut **`redirect`**.

Syntaxe : `redirect = adresseIP port`

Exemple : Je dispose d'une passerelle vers Internet et une machine sur le même réseau local dont l'adresse IP est 192.168.0.3. Je souhaite installer un serveur telnet sur cette dernière machine. Je vais donc faire en sorte que les requêtes qui arrivent de l'extérieur pour mon serveur telnet soient redirigées vers cette machine.

```
root@pingu# cat /etc/xinetd.d/telnet
service telnet
{
   ...
   server = /usr/sbin/in.telnetd
   redirect = 192.168.0.4 23
}
```

On teste maintenant la connexion :

```
root@pingu# telnet 217.11.12.13
Trying 217.11.12.13...
Connected to 217.11.12.13.
Escape character is '^]'.
Red Hat Linux release 8.0 (Psyche)
Kernel 2.4.18-14 on an i586
login: anne
Password:
Last login: Fri Nov 8 12:26:44 on :0
[anne@citrouille anne]$ hostname
citrouille
```

La connexion telnet est établie et la commande `hostname` nous confirme que je ne suis pas sur la passerelle mais sur la machine citrouille dont l'adresse IP est 192.168.0.4

Attribution d'un service à une adresse IP

`xinetd` va vous permettre de lier un service à une adresse IP grâce à l'attribut `bind`.

Pour éclaircir la description de cet attribut, nous allons nous appuyer sur un exemple. Je vais reprendre mes 2 machines de l'exemple précédent. Mon objectif : je veux construire 2 serveurs ftp bien différenciés. L'un sera réservé à mes machines en local (serveur de fichiers interne) l'autre mettra à disposition d'autres fichiers sur un serveur ftp réservé aux connexions externes. (Après tout je ne partage pas tout :)).

Ma machine `pingu` (ma passerelle) a 2 interfaces réseau avec les adresses IP respectives : 192.168.0.3 et 217.11.12.13. Je vais donc effectuer les attributions grâce au fichier suivant :

```
root@pingu# cat /etc/xinetd.d/proftpd
service proftpd
 {
   id = ftp_public
   ...
   server = /opt/proftpd/proftpd
   bind = 217.11.12.13
}
service proftpd
 {
   id = ftp_privé
   ...
```

```
   server = /opt/proftpd/proftpd
   bind = 192.168.0.3
 }
```

L'attribut `id` sert uniquement à différencier les 2 configurations du service. Tout ceci bien sûr doit s'accompagner d'une configuration de votre serveur ftp.

xinetd parle à vos visiteurs

Pour finir, `xinetd` vous permet également d'afficher des messages lors de la connexion à un service grâce aux attributs suivants :

- **`banner = fichier`** − affiche le contenu de `fichier` si l'accès est autorisé, et avant même authentification.
- **`banner_succes = fichier`** − affiche le contenu de `fichier` en cas de réussite de l'authentification
- **`banner_fail = fichier`** − affiche le contenu de `fichier` en cas d' échec de l'authentification

Exemples de configuration

Ci−dessous quelques exemples de fichiers de configuration de services, issus entre autres du man de xinetd.conf

Exemple 1

Le 1er exemple est un fichier de configuration du service `ftp`.

```
root@pingu# cat /etc/xinetd.d/ftp
service ftp
 {
   socket_type = stream
   wait = no
   nice = 10
   user = root
   server = /usr/etc/in.ftpd
   server_args = −l
   instances = 4
   log_type = FILE /var/log/ftp.log
   log_on_success = DURATION HOST USERID
   access_times = 2:00−9:00 12:00−24:00
 }
```

Le serveur ftp, hormis les paramètres génériques de configuration, autorise des connexions entre 2h et 9h et entre midi et minuit. Son fichier de log est `/var/log/ftp.log` et contient, en cas de connexions réussies, la durée de la connexion, le nom de la machine cliente et l'UID de l'utilisateur connecté. Seules 4 instances simultanées sont autorisées

.

Exemple 2

Le 2e exemple est un fichier de configuration du service `telnet`.

```
root@pingu# cat /etc/xinetd.d/telnet
service telnet
 {
   socket_type = stream
   wait = no
   nice = 10
   user = root
   server = /usr/etc/in.telnetd
   rlimit_as = 8M
   rlimit_cpu = 20
 }
```

Exemple 3

Le dernier exemple se compose de 2 fichiers de configuration pour la gestion de `samba`. par xinetd. En effet, le service samba lance 2 démons, `nmbd` et `smbd`

```
root@pingu# cat /etc/xinetd.d/netbios−ssn
service netbios−ssn
{
   socket_type = stream
   protocol = tcp
   wait = no
   user = root
   server = /usr/sbin/smbd
   only_from = 192.168.0.0
```

```
   disable = no
}

root@pingu# cat /etc/xinetd.d/netbios-ns
service netbios-ns
{
   socket_type = dgram
   protocol = udp
   wait = no
   user = root
   server = /usr/sbin/nmbd
   only_from = 192.168.0.0
   disable = no
}
```

Hormis les paramètres génériques de configuration de smbd et nmbd, on a configuré samba de manière à ce qu'il ne soit utilisable que sur le réseau interne 192.168.0.0.

Le mot de la fin

Je n'ai pas exploré ici l'intégralité des possibilités de xinetd. La meilleur des choses à faire pour cela est de se reporter au man. Si vous disposez encore une distribution utilisant inetd, et au vu de toutes les fonctionnalités de xinetd, il semble opportun d'abandonner le premier pour le deuxième. Bien exploité, il devient véritablement un outil de sécurisation de votre système, dès lors que vous mettez en place un certains nombres de serveurs qui, ouvrant votre machine sur l'extérieur, ouvre également la voie aux vilains hackers ;).

N'hésitez pas à me faire parvenir vos remarques et ajouts divers sur le contenu de cet article.

Installation Apache, PHP, MySQL

Jean−Marc LICHTLE
Mise à jour par Anne

Installation d'un serveur Apache/PHP/MySQL en environnement Mandrake

1 Objectif de ce document

L'objet de ce document est de guider les premiers pas d'un utilisateur débutant dans l'utilisation de la trilogie Apache, PHP, MySQL. On suppose que les bases de LINUX sont connues, en particulier que l'utilisateur est familiarisé avec l'arborescence des fichiers, la manipulation d'un logiciel de décompression des paquetages, la notion d'utilisateur, d'administrateur, etc.

2 Introduction

La configuration de l'ensemble Apache, PHP, MySQL n'était pas spécialement aisée avec les distributions plus anciennes. Un article (très intéressant) de la revue Linux pratique décrit dans le numéro 12 comment s'y prendre pour réaliser cette opération, 9 pages d'instructions, de mises à jour de fichiers .conf etc.. Passionnant (et pénible).

Depuis un certain nombre de versions de Mandrake, il est possible de tout faire fonctionner sans retoucher le moindre fichier de configuration, simplement en installant les **rpm** ! Et Mandrake 10 ne déroge pas à ce constat.

Rappelons brièvement que:

- Apache est un serveur Web (le logiciel qui ``envoie'' des pages html à un navigateur). Il s'agit du serveur le plus utilisé actuellement sur le Web puisqu'il représente environ 2/3 des serveurs installés.
- PHP est un langage de programmation interprété. Correctement interfacé avec Apache il permet au serveur de fournir des pages dynamiques gérées en fonction des besoins du client. En clair, la page n'est plus un document statique mais peut évoluer, afficher des informations différentes selon les souhaits de l'utilisateur.
- MySQL est un gestionnaire de bases de données assez puissant et rapide. Il peut très bien fonctionner en utilisant son propre client en mode texte sans l'utilisation d'une quelconque interface graphique. L'utilisateur doit alors maîtriser le langage SQL de MySQL (proche de la norme ANSI) et se servir de la ligne de commande pour adresser des requêtes au serveur. Spartiate, on se croirait revenus à la grande époque de DBase sous DOS! (nostalgie).

3 Rappel: Utilisation de la commande rpm.

Il existe de nombreux utilitaires tels que Kpackage, GnoRPM, Midnight Commander etc. qui permettent d'installer et de gérer les packages au format Red Hat (RPM).

Pour ma part je trouve plus rapide et au moins aussi efficace de travailler en mode ligne de commande en utilisant directement la commande rpm. Voir l'article rpm sur Léa. Encore plus simple, URPMI permet aujourd'hui une gestion complète des dépendances. Je vous laisse le soin de consulter la documentation.

4 Installation d'Apache

Note prélable : la plupart des distributions proposent aujourd'hui les versions 1.3.x et 2.x de Apache. Si Apache 2.x présente plus d'option pour la configuration avancée du serveur et d'autres caractéristiques liées à sa performance, le groupe de développement PHP n'a toujours pas validé la plate−forme sous cette version. On peut raisonnablement dire que pour un serveur personnel ou ne nécessitant pas de fonctionnalités particulières, Apache 1.3.x fait parfaitement l'affaire. C'est ce que nous utiliserons.

4.1 Mise en place des rpm

L'installation d'Apache ne pose pas de problèmes. L'objectif est de disposer en fin d'installation des packages suivants :

- `apache2−common` : fichiers d'installation pour Apache et apache−mod_perl
- `apache2−modules` : ensemble des modules pour Apache
- `apache−conf` : fichiers de configuration de Apache
- `apache2` : le démon Apache
- `apache−manuel` (pas obligatoire mais peut se révéler utile).

```
# urpmi apache2
Pour satisfaire les dépendances, les paquetages suivants vont être installés (1 Mo):
apache−conf−2.0.48−2mdk.i586
apache2−2.0.48−6mdk.i586
apache2−common−2.0.48−6mdk.i586
apache2−modules−2.0.48−6mdk.i586
Est−ce correct ? (O/n)
Préparation...              ##################################################
   1:apache−conf            ##################################################
   2:apache2−modules        ##################################################
   3:apache2−common         ##################################################
   4:apache2                ##################################################
```

Le package `apache-conf` contient en particulier les fichiers de configuration `/etc/httpd/conf/httpd.conf` et `/etc/httpd/conf/commonhttpd.conf` qui définissent les paramètres de fonctionnement du serveur[Note 1]

4.2 Premier test d'Apache

Le fonctionnement d'Apache est matérialisé par la mise en route d'un démon nommé **httpd**. Pour mettre en route, arrêter, vérifier etc. ce service tapez /etc/rc.d/init/httpd option ou option peut prendre les valeurs start, stop, restart, status ou extendedstatus selon votre besoin. En clair, pour démarrer Apache tapez `/etc/rc.d/init.d/httpd start`
Note 2 Sur Mandrake, vous pouvez également utiliser la commande`service`: `service httpd start` par exemple.

.

En principe l'installation d'Apache a modifié les fichiers de configuration du lancement de LINUX. Le démon httpd devrait donc démarrer systématiquement et automatiquement à chaque boot. L'installation a par ailleurs démarré immédiatement ce démon, il n'est donc pas nécessaire de rebooter comme on le ferait avec un autre système d'exploitation très connu avec lequel le moindre changement de mulot passe par un reboot...

Nous pouvons par ailleurs passer immédiatement au test du serveur sans passer par la moindre phase de configuration. Pour cela chargez votre navigateur préféré (pour ma part j'utilise Konqueror). Saisir ``http://localhost'' dans la barre d'URL. Vous voyez apparaître après quelques instant une page de présentation d'Apache. En fait le serveur Apache vient d'envoyer la page `/var/www/html/index.shtml`, affirmation que vous pourrez facilement vérifier en tapant ``file:/var/www/html/index.shtml'' dans la barre d'URL de votre navigateur. Attention: il n'y a qu'un slash après `file:` alors que par habitude vous pourriez être tentés (comme moi!) d'en mettre deux.

La page affichée dans ce cas est bien la même que la première. Les différences d'affichage dans les parties graphiques s'expliquent par le fait que l'affichage a été demandé en direct au lieu de passer par le serveur Apache.

Si vous préférez utiliser Lynx (un navigateur en mode texte ultra rapide puisqu'il ne charge pas les parties graphiques) vous pouvez même vous contenter de taper ``lynx localhost''. Lynx ajoutera automatiquement ce qui manque. Merveilleux de simplicité non ?

5 Installation de PHP

5.1 Mise en place des paquetages rpm

L'installation de PHP doit contenir :

- `libphp-common432`
- `php.ini`
- `mod_php`

L'installation de ces packages ne devrait pas poser de problème particulier. Sauf erreur ils s'installent automatiquement avec l'option d'installation ``recommandée'' de Mdk. Vérifiez toutefois la présence de `mod_php` (`rpm -qa | grep php`) !

```
# urpmi php
Un des paquetages suivants est nécessaire :
 1- apache2-mod_php-2.0.48_4.3.4-1mdk.i586
 2- mod_php-4.3.4-1mdk.i586
 3- php-cli-4.3.4-4mdk.i586
 4- php-cgi-4.3.4-4mdk.i586
Que choisissez-vous ? (1-4)1
Préparation...              ##################################################
   1:apache2-mod_php        ##################################################
Shutting down httpd2: [  OK  ]
Checking configuration sanity for Apache 2.0:  [  OK  ]
Starting httpd2: [  OK  ]
```

L'installation des rpm se termine par un redémarrage de Apache afin de prendre en compte le module PHP.

5.2 Premier script PHP

Il importe maintenant de réaliser le premier test. Créez un sous répertoire `/var/www/html/test/` puis créez dans ce sous répertoire un fichier `test.php` contenant :

```
<html>
<?
echo 'premier test php<br><br>';
phpinfo();
?>
</html>
```

Sauvegardez et chargez dans un navigateur (URL `http://localhost/test/test.php`).

Voilà votre première page web dynamique ! Vous avez, en quelques lignes, créé du code html (les balises <html> et </html>), faite générer du html par PHP (`echo 'premier test php<br><br>';`), les
 sont des retours chariots, et fait appel à l'une des nombreuses fonctions de PHP, phpinfo() qui donne les principales informations sur le configuration actuelle de PHP.

Vous pourrez réaliser d'autres tests, l'aide nécessaire est à votre disposition dans les bouquins, les revues et sur le net.

Lorsque vous aurez constitué un petit groupe de fichiers de test vous pourrez alors faire l'expérience suivante :

- tapez simplement `http://localhost/test/` dans la barre d'URL de votre navigateur, celui−ci affiche la liste des fichiers créés, en fait le contenu du sous répertoire `/var/www/html/test/`. Si vous cliquez sur l'un des fichiers vous lancez le serveur Apache qui va interpréter le code PHP contenu dans le fichier choisi.
- copiez l'un des fichiers (ou renommez le) vers index.php. La même URL que précédemment affiche alors directement les contenu de index.php! Le contenu du sous répertoire n'est plus visible! L'accès aux différents fichiers reste toutefois possible si vous tapez directement leurs noms respectifs.
- effacez le ficher `index.php`, l'affichage de la liste des fichiers est à nouveau possible. C'est pratique pour éviter qu'un visteur de passage puisse consulter le contenu d'un sous−répertoire. Bien sûr c'est un simple truc, pas un rideau de fer contre un kracker expérimenté, mal intentionné et patient !

6 Installation de MySQL

6.1 Mise en place des rpm

L'installation de **mysql** est elle aussi très simple. Les packages suivants sont nécessaires (ils devraient avoir été installés par défaut, en cas de besoin les installer par la suite):

- `mysql−common`
- `mysql−client`
- `mysql`

MySQL−client n'est pas indispensable. Il s'agit en fait de la partie client de l'ensemble client−serveur. Ce client permet toutefois (attention, en mode texte c'est plutôt spartiate !) de faire quelques requêtes au serveur.

```
'# urpmi mysql
Un des paquetages suivants est nécessaire :
 1− MySQL−4.0.18−1.1.100mdk.i586
 2− MySQL−Max−4.0.18−1.1.100mdk.i586
Que choisissez−vous ? (1−2)1
Pour satisfaire les dépendances, les paquetages suivants vont être installés (18 Mo):
MySQL−4.0.18−1.1.100mdk.i586
MySQL−client−4.0.18−1.1.100mdk.i586
MySQL−common−4.0.18−1.1.100mdk.i586
libmysql12−4.0.18−1.1.100mdk.i586
perl−Mysql−1.22_19−9mdk.i586
Est−ce correct ? (O/n)
Préparation...              ##################################################
   1:libmysql12             ##################################################
   2:MySQL−client           ##################################################
   3:perl−Mysql             ##################################################
   4:MySQL−common           ##################################################
   5:MySQL                  ##################################################
040512 12:40:46  /usr/sbin/mysqld: Shutdown Complete
# service mysql start
Lancement du serveur MySQL                                 [  OK  ]
```

En principe après l'installation de mysql vous devriez disposer d'un serveur MySQL opérationnel (et en marche). Pour le vérifier tapez `ps −ax | grep mysql`. Vous devriez voir s'afficher quelques lignes relatives à autant de serveurs en attente de requêtes. Si tel n'est pas le cas tapez (comme root) "`safe_mysql &`" dans une fenètre texte. Cette commande devrait démarrer le serveur MySQL.

Attention : l'installation par défaut utilise root comme super administrateur du serveur et le mot de passe root. Faites donc un essai de fonctionnement du serveur MySQL en utilisant simplement le client MySQL en mode texte.

mysql −p
Enter password:
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 4 to server version: 4.0.18−log

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql>

Tapez alors quit à l'invite mysql et vous voilà sorti.

Il va falloir faire fonctionner tout ce petit monde ensemble! Apache et PHP sont déjà opérationnels. Ils communiquent ensemble, l'essai a été fait. MySQL est en route également. Pour que PHP puisse entrer en contact avec MySQL il lui faut un minimum de connaissances qui lui sont apportées par le module **php−mysql**.

```
# urpmi php−mysql
Préparation...              ##################################################
   1:php−mysql              ##################################################
```

Pensez donc à l'installer lui aussi sinon pas de langue commune! Redémarrez Apache.

6.2 Installation du script phpMyAdmin

Plutôt que de vous lancer dans la rédaction d'un premier script php qui va s'adresser au serveur MySQL (pour lui dire quoi ? Rien n'existe encore, ou si peu !) je vous suggère de récupérer sur le net un ensemble de scripts nommé phpMyAdmin. Il s'agit tout simplement d'un ensemble de scripts PHP qui va vous permettre, via un navigateur, d'administrer vos premières bases de données, de lire les bases de données système MySQL (celles qui définissent les droits des utilisateurs), d'en créer d'autres.

```
# urpmi phpmyadmin
Préparation... ##################################################
1:phpMyAdmin ##################################################
```

6.3 Premier test

Une fois l'installation faite utilisez votre navigateur préféré et tapez `http://localhost/admin/phpMyAdmin/` dans la barre d'URL. Vous devriez voire apparaître une page html "Bienvenue à phpMyAdmin 2.5.4 − Connexion". Il vous reste à modifier le fichier /var/www/html/admin/phpMyAdmin/config.inc.php et le paramètre `$cfg['blowfish_secret'] = '';`.
Exemple : `$cfg['blowfish_secret'] = 'ma phrase secrete';`

Rechargez la page de votre navigateur, voilà vous êtes prêts à l'utiliser !

7 Synthèse

Vous venez de créer deux ensembles de données, les répertoires contenant les pages HTML/PHP, et les répertoires contenant les bases de données MySQL.

- Les pages PHP/HTML sont rangées dans `/var/www/html/`. La page d'accueil Apache est à l'adresse `/var/www/html/index.shtml` (rappel !). Le sous−répertoire `/var/www/html/admin/phpMyAdmin` contient le script de connexion à MySQL. Vous pourrez créer d'autres sous répertoires contenant vos premiers essais, par exemple `/var/www/html/test/` qui sera accessible à l'URL `http://localhost/test/`.
- Les données MySQL sont rangées dans `/var/lib/mysql/`. Chaque base de données correspond à un sous−répertoire. Exemple : `/var/lib/mysql/mysql/` contient les tables qui définissent les droits d'accès à MySQL. Ce sous−répertoire n'est évidemment accessible que pour l'administrateur système. Si vous créez une nouvelle base de données avec phpMyAdmin ou directement en utilisant le client MySQL, le serveur créera un nouveau sous répertoire /var/lib/mysql/test par exemple si votre nouvelle base s'appelle test.

Conclusion

La description ci−dessus avait pour but de vous conduire à disposer d'un serveur Apache qui soit opérationnel et qui puisse mettre en oeuvre des bases de données MySQL via PHP. Ce but étant atteint il convient maintenant d'être modeste. Le vrai travail commence juste. Lisez les docs, les man pages. Il existe aussi d'excellents bouquins sur la question. Pour ma part j'étudie actuellement ``Pratique de MySQL et PHP'' édition O'Reilly. Je le trouve très bien fait.

Enfin, lisez l'article suivant, concernant la sécurisation de l'environnement que l'on vient de créer.

Remerciement

Lors d'une installation il est courant de progresser un peu à tâtons, par essais successifs. Après quelques heures de bagarre (qui peuvent être réparties sur plusieurs jours) il faudrait être un sur−homme pour se souvenir de tout ce qu'on a fait. La mise au point d'un document comme celui−ci ne peut donc se faire qu'après plusieurs installations réussies. Idéalement la bêta−version du document doit elle−même être testée par un autre personne qui suit scrupuleusement la description et commente les erreurs, inversions, oublis etc. Je remercie donc M. Guy ASSFELD d'avoir bien voulu apporter sa contribution à ce travail en déboguant la première version.

L'auteur

JML dit Jean−Marc LICHTLE, email jean−marc.lichtle@gadz.org, ingénieur Arts et Métiers promo CH173 (rigolez pas, à l'époque le hi−tech c'était les cartes perforées sur IBM 1130) !

Notes

... serveur[1]
De nombreuses distributions ne se contente plus du seul fichier httpd.conf. La version contenue dans LINUX Mandrake 10.0 utilise un fichier supplémentaire, commonhttpd.conf qui est appelé par httpd.conf par un ``include''.

... start[2]
Je ne précise pas, il va sans dire que certaines commandes ne peuvent être lancées que par l'administrateur système (root). L'appel à httpd fait partie de ces commandes réservées. Donc si LINUX vous fait un bras d'honneur commencez par vous poser la question de votre habilitation à lancer une commande.

Configuration d'apache: httpd.conf

par serge

Mettez un serveur Web dans votre LinBox.

Introduction

Apache est un serveur Web libre, c'est le standard comme serveur Web sous linux, mais aussi le serveur Web le plus utilisé sur Internet avec plus de 60% des sites d'Internet (contre environ 20% pour IIS). Donc cela ne peut que prouver sa stabilité et ses performances.

S'il est très souvent disponible en standard sur les distributions mais on doit souvent le configurer pour ses propes besoins.
Je vais donc essayer de vous détailler le fichier **httpd.conf**, fichier de configuration d'apache.

Attention: Utilisez une version récente d'apache car:
− De nombreux bugs de sécurité ont été corrigé
− Cette documentation traite de la configuration des derniers serveurs apaches. Dans les anciennes versions, il y avait trois fichiers (`httpd.conf`, `srm.conf`, `access.conf`) mais maitenant tous sont regroupés dans un même fichier: `httpd.conf`.
Même si la compabilité avec les anciennes versions a été conservée (on peut toujours utiliser les anciens fichiers) je ne traite ici QUE du fichier `httpd.conf`.

De plus cette documentation n'est pas encore tout à fait complète, il manque encore quelques options à expliquer, mais certaines me semblent obscures (je ne connais pas tout) ou difficiles à expliquer. J'espère compléter ce document assez vite (de même si vous voulez ajouter vos connaissances à ce document, n'hésitez pas à me mailer

Je ne traite pas de l'installation d'apache, car il est présent dans toutes les distributions. De plus son installation et sa compilation ne sont pas aisées du tout suivant le support que l'on veut lui intégrer. Il existe en effet plusieurs dizaines de facons de le compiler suivant les options que l'on veut lui passer. Mais dans la plupart des distributions, apache est configuré de telle facon que l'on peut y inclure des fonctionalités facilement grâce à des modules "DSO".

Configuration du fichier httpd.conf

Pour cela on va éditer le fichier `/chemin_d'install_apache/conf/httpd.conf`. Si ce fichier n'existe pas vous devez alors avoir un fichier appelé `httpd.conf−dist`. Copiez le en `httpd.conf` et éditez−le.
Détaillons ce fichier, en cherchant les lignes enoncées ci dessous (si une ou plusieurs de ces options ne vous servent pas, commentez−les avec un # devant la ligne)

```
ServerType          standalone
```

Cette ligne indique si apache se lance en '*autonome*' (**standalone**) ou via **inetd**.
Pour la plupart des configurations, c'est en standalone. Si vous ne l'avez pas réinstallé, qu'apache était déja installé, ne modifiez pas cette ligne. Ceux qui ont créé votre distrib ont bien configuré cette ligne normalement. Si vous n'êtes pas sur, éditez le fichier **/etc/inted.conf** , si aucune ligne ne comporte **httpd** et que par contre vous avez un fichier du style **rc.httpd** ou **Sxxhttpd** ou **Sxxapchectl** dans un sous−répertoire de **/etc/rc.d** alors c'est qu'il se lance en autonome. Inversement si une ligne httpd existe dans `/etc/inetd.conf`, alors il se lance en **inetd**.

```
ServerRoot              "/var/lib/apache"
```

Vous indiquez ici le répertoire d'installation d'apache. Normalement les scripts d'installation ont bien renseigné cette ligne. Vérifiez quand même.

```
LockFile             /var/run/httpd.lock
```

Laissez cette ligne comme elle est , c'est−à−dire commentée, pour 90% des cas (# devant).

```
PidFile                 /var/run/httpd.pid
```

Vérifiez bien que cette ligne soit décommentée. Elle indique au script de démarrage d'enregistrer le numéro de process d'apache pour que, lors de l'arrêt du sytème apache soit stoppé correctement.

```
ScoreBoardFile       /var/run/httpd.scoreboard
```

Idem. Cette ligne doit exister, ce fichier stocke des informations pour le bon fonctionnement d'apache−lui même.

```
RessourceConfig     /conf/srm.conf
AcessConfig         /conf/acess.conf
```

Les anciens fichiers de configuration d'apache, qui ne sont plus utilisés maitenant (en fait tout est maintenant intégré dans `httpd.conf` pour simplifier). Ces options sont normalement commentées (# devant).

```
Timeout     300
```

Temps en millisecondes avant que le serveur n'envoie ou recoive un "timeout".
En fait quand le serveur attend une réponse d'un programme externe (parseur PHP, script CGI, etc..), si au bout de 3s il ne recoit pas cette réponse, il

va envoyer un timeout au programme pour l'arrêter et renvoyer un timeout à l'utilisateur pour le prévenir d'une erreur. Laissez cette valeur par défaut à moins que vous vous aperceviez que des traitements prennent plus de temps que 3s et que le serveur n'attend pas assez longtemps. Ne montez pas trop haut cette valeur non plus car si le programme externe a "planté" ou si une erreur s'est produite, vous risquez de rendre inacessible le serveur apache pour trop de temps (toujours désagréable d'attendre 30s pour rien).

KeepAlive on

Autorise ou pas les connexions persistantes (plusieurs requêtes par connexion). En fait cela permet aux utilisateurs de votre serveur de lancer plusieurs requêtes à la fois. Cela permet d'accélérer les réponses du serveur. Laissez cette valeur par défaut la plupart du temps. Mais dans un environement très sollicité (serveur Web trés fréquenté) cela peut rendre le serveur indisponible pour d'autres utilisateurs ou faire monter le système en ressource. En gros pour de petits serveurs, laissez cette option sur "on". Pour un serveur très sollicité dont le système ralenti énormément ou devient indisponible asez souvent, essayez avec la valeur "off". Mais avant, essayez de baisser la valeur de l'option suivante.

MaxKeepAliveRequests 100

En combinaison avec l'option précédente, indique le nombre de requêtes pour une connexion. Laissez cette valeur assez haute pour de très bonnes performances. Si vous mettez **0** comme valeur, **vous autorisez en fait un nombre illimité** (attention donc). Laissez la valeur par défaut là aussi.

KeppAliveTimeout 15

Valeur en seconde d'attente pour la requette suivante d'un même client sur une même connexion avant de renvoyer un timeout. Là aussi, laissez la valeur par défaut.

```
MinSpareServers    5
MaxSpareServer     10
```

Ces valeurs servent à l'autorégulation de charge du serveur.
En fait apache contrôle lui−même sa charge, suivant le nombre de clients qu'il sert et le nombre de requêtes envoyées par chaque client. Il fait en sorte que tout le monde puisse être servi et ajoute tout seul un certain nombre d'instances apaches pour servir de nouveaux clients qui se connecteraient. Si ce nombre est inférieur à **MinSpareServers**, il en ajoute un. Si ce nombre dépasse la valeur de **MaxSpareServer**, il en arrête. Ces valeurs par défaut conviennent à la plupart des sites.

StartServers 5

Nombre de serveurs à démarrer au lancement d'apache.

MaxClients 150

Contrôle le nombre maximum de serveurs pouvant etre lancés simultanément. Laissez donc cette valeur assez haute. Toutefois attention à son niveau pour ne pas qu'en cas de trop forte sollicitation, le serveur apache fasse "tomber" en ressource votre système. **Cette valeur ne modifie pas la valeur de MaxKeepAliveRequests car ici Maxclients indique le nombre de clients maximum alors que MaxKeepAliveRequests indique le nombre de requêtes pour UN client**.

Remarque: Apache par défaut ne peut servir plus de 254 clients à la fois (c'est a dire si au même moment 254 clients se connectent simultanément, le serveur sature et certains clients doivent attendre avant de pouvoir accéder à votre site), donc la valeur de cette directive ne peut excéder 254 (apache vous signale un warning à son démarrage si vous dépassez cette valeur). Pour pouvoir servir plus de 254 clients simultanément, il faut recompiler apache et modifier dans les sources d'apache la valeur max de clients simultanés. Voir la documentation officielle d'apache pour cela.

```
Listen    3000
Listen    12.34.56.78
```

Indique au serveur des ports ou des adresses où il doit "écouter" pour des connexions **EN PLUS** de l'adresse et port par défaut. Voir l'option `VirtualHost` plus loin.

**BindAdress *

Vous pouvez ici inclure un hôte virtuel en indiquant une adresse, toutes les adresses (*) ou un nom de domaine où le serveur va attendre des connexions. Voir là aussi l'option `VirtualHost` .

```
LoadModule    xxxxxx.mod libexec/yyyyyy.so
.....
AddModule  zzzz.c
```

Support pour les modules DSO. Si vous devez ajouter des modules à prendre en compte (par exemple pour ceux qui installent PHP en module dynamique), attention à l'ordre de chargement des modules qui a son importance car souvent un module dépend d'un autre module. Voir les documentations des modules que vous souhaitez ajouter.

ExtendedStatus on

Indique si le serveur doit renvoyer des informations complètes de status (**on**) ou des informations réduites (**off**). `off` par défaut. Laissez cette valeur par défaut sauf en cas de développement et de débuggage.

```
Port     80
```

Port d'écoute du serveur par défaut. Vous pouvez le changer si vous désirez mais attention les clients cherchent normalement sur le port 80.

```
User nobody
Group nobody
```

User et group system avec lesquels le serveur est démarré. Pour des questions de sécurité laissez toujours ces options! **Ne jamais spécifier root!** En fait si quelqu'un arrive à "exploiter" votre serveur (par exemple il arrive à faire exécuter du code par le serveur apache), il hérite des droits du serveur lui−même. Donc si c'est "nobody", il n'a aucun droit spécifique. Si c'est root ou un user réel, il aura alors les droits de ce user/group et pourra endommager grandement votre système.

```
ServerAdmin   root@localhost.domainname
```

Adresse E−mail réelle de l'administrateur du site. Cette adresse est affichée par le serveur en cas d'erreur par exemple pour que les utilisateurs puissent en avertir l'administrateur. Mettez votre E−mail donc.

```
ServerName    www.domainname
```

Adresse que le serveur va renvoyer au client Web. Il est astucieux de mettre www au lieu du nom de la machine réelle, comme cela les visiteurs ne voient pas le nom réel de votre machine (utile pour la sécurité aussi).

```
DocumentRoot    "/var/lib/apache/htdocs"
```

Répertoire racine où se trouvent vos pages Web.

```
<Directory />
    Options FollowSymLinks
    AllowOverride None
</Directory>
```

Pour des questions de sécurité, laissez cela par défaut. Cela laisse les permissions de tous les répertoires par défaut et n'autorise pas d'accès spéciaux même si un .htaccess existe (AllowOverride None). De ce fait, personne ne peux modifier les droits que vous avez imposés sur les répertoires où apache a accès. Voir plus bas les explications de chaque option passée.

```
<Directory "/var/lib/apache/htdocs">
  Options Indexes FollowSymlinks Multiviews
  AllowOverride None
  Order allow,deny
  allow from all
</Directory>
```

Bon là, ce qu'on a fait, c'est qu'on définit cette fois les règles pour le répertoire `/var/lib/apache/htdocs` (pour notre exemple c'est le repertoire racine des pages Web), on autorise le serveur apache à suivre les liens symboliques (FollowSymlinks) , on empêche quiconque de changer ces règles (AllowOverride None) , puis on définit dans quel ordre appliquer les règles d'autorisation/refus de connexion (order allow,deny) et on autorise les connexions venant de n'importe quel hôte (Allow from all).

Bon détaillons un peu comment ça marche. On commence toujours par:

```
<Directory xx>
```

Pour définir sur quel répertoire (**xx**) on applique ces règles.

`Options` : on définit les options pour ce répertoire qui sont pour les plus importantes :

None	Désactive toutes les options.
All	Active toutes les options SAUF Multiviews.
Indexes	Permet aux utilisateurs d'avoir des indexes générés par le serveur. C'est à dire si l'index du répertoire (index.htm le + souvent) est manquant, cela autorise le serveur à lister le contenu du répertoire (dangereux suivant les fichiers contenus dans ce répertoire).
FollowSymLinks	Autorise à suivre les liens symboliques.
ExecCGI	Autorise à exécuter des scripts CGI à partir de ce répertoire.
Includes	Autorise des fichiers include pour le serveur.
IncludesNOEXEC	Permet les includes mais empêche la commande EXEC (qui permet d'exécuter du code).
Multiviews	Autorise les vue multiples suivant un contexte. Cela permet par exemple d'afficher les pages dans un language suivant la

	configuration du language du client.
SymLinksIfOwnerMatch	Autorise à suivre les liens seulement si le user ID du fichier (ou répertoire) sur lequel le lien pointe est le même que celui du lien.

`AllowOverride` : définit comment sont gérés les fichiers .htaccess de ce répertoire:

All	Gère tout ce qui est dans .htaccess
AuthConfig	Active les directives d'autorisations AuthDBMGroupFile, AuthDBMUserFile, AuthGroupFile, AuthName, AuthType, AuthUserFile, require
FileInfo	Active les directives d'autorisations AddEncoding, AddLanguage, AddType, DefaultType, ErrorDocument, LanguagePriority
Limit	Active la directive d'autorisation Limit
None	Ne lit pas le fichier .htaccess et laisse les droits "Linux" de ce répertoire.
Options	Active la directive Option

`Order` : Donne l'ordre d'application des règles Allow Deny:

deny,allow	Applique les règles deny puis allow
allow,deny	Applique les règles allow puis deny

`Allow` (ou **`deny`**):

Nom d'hôte	Autorise les hôtes spécifié, les adresses IP, le nom de domaine, etc..(ou les refuse si la règle est deny)
All	Autorise tout le monde (ou refuse)

`</Directory>` : Indique la fin des règles pour ce répertoire.

A vous de placer vos règles suivant le contenu de vos répertoire accessibles par le Web. Il existe les mêmes règles pour les fichiers **(<Files> </Files>)** et les locations **(<Location> </Location>)**. Voir un exemple pour les fichiers **(file)** plus bas.

`AccesFileName .htaccess`

Nom du fichier des règles d'accès pour les règles AllowOverride.
Un conseil : placez comme vu précédemment une règle file du style:

```
<Files ~ "^\.ht">
  Order allow,deny
   Deny from all
  </Files>
```

pour interdire aux visiteurs la possibilité voir le contenu des fichiers .ht qui contiennent les règles de sécurité.

`#CacheNegotiatedDocs`

Autorise ou pas les proxies à mettre en cache les documents (pour autoriser, enlevez le commentaire # en début de ligne)

`UseCanonicalName On`

Sur **on**, remet l'url par rapport aux valeurs "Server" et "Port" spécifié plus haut dans le fichier `httpd.conf`, sur **off** utilise l'url que le client a utilisée. Attention , mettre sur *on* si vous utilisez des CGI avec des variables SERVER_NAME car si l'url du client n'est pas la même que celle du CGI, votre script CGI ne marchera pas.

`DefaultType text/plain`

Type mime par défaut que le serveur renvoie aux clients. Convient dans la plupart des cas. Si par−contre votre serveur web envoie principalement des images, du streaming, etc... modifiez la valeur par
"**application/octet−stream**" pour éviter que les clients essaient de visualiser du binaire (ce qui va leur donner comme résultat des "**ù*$kfdPmm%ù**" au lieu d'afficher une video par exemple. Le client doit pouvoir prendre en compte alors le fichier grâce à un plugin et il doit avoir ses types mimes

configurés pour lancer l'application associée.

HostNameLookups off

Indique si le serveur enregistre le nom (sur on) du client ou l'adresse IP (sur off)

ErrorLog /var/log/error_log

Chemin complet du fichier où les erreurs apaches seront enregistrées

LogLevel warn

Niveau d'enregistrement des erreurs avec comme valeurs possibles:

emerg	Enregistre seulement les erreurs qui rendent le serveur inutilisable
Alert	"emerg" + erreurs nécessitant une intervention.
Crit	"emerg" + "Alert" + erreurs critiques (accès réseau impossible par exemple)
error	"ermerg" + "Alert" + "Crit" + erreurs dans les pages, les scripts
warn	"ermerg" + "Alert" + "Crit" + "error" + erreurs non bloquantes (pages mal codées, scripts comportant des erreurs non bloquantes...)
info	"ermerg" + "Alert" + "Crit" + "error" + "Warn" + toutes les informations générées
debug	Enregistre TOUT ce qui peut se passer sur le serveur (un client demande une page: on enregistre)

La plupart du temps, le niveau warn suffit (info et debug fournissent trop d'informations)

ServerSignature on

Si on ajoute la signature (version, OS...) du serveur lorsqu'il génère des pages lui même (index manquant, erreur de script, etc....) , off ne nontre que l'erreur, et sur Email ajoute un lien vers l'email définit plus haut dans ce fichier.

Alias faux_nom nom_réel

permet de faire des alias de répertoires (des liens en quelque sorte)

ScriptAlias /cgi-bin/ chemin_complet_des_cgi

Alias pour les scripts CGI

AddType type extensions

Permet de définir des extensions de fichiers pour des applications (mime type). Si on veut ajouter le support PHP4, on doit ajouter ici:

```
  AddType application/x-httpd-php .php
AddType application/x-httpd-php-source .phps
```

Ou pour du PHP3:

```
  AddType application/x-httpd-php3 .php3
AddType application/x-httpd-php3-source .phps
```

Il faut en plus que vous ayez compiler PHP en module DSO (module dynaique) et ajouter les lignes **modules** comme décrit plus haut pour activer le support PHP (voir la doc d'installation de PHP).

```
AddHandler cgi-script .cgi
```

Pour utiliser les scripts CGI.

Et enfin les définitions de "virtual host" pour faire des serveurs virtuels suivant des répertoires, etc...

```
<VirtualHost ip_ou_adresse_virtuelle>
    ServerAdmin email_webmaster@hoste_virtuel
    DocumentRoot racine_hote_virtuel
```

```
    ServerName nom_server_virtuel
    ErrorLog chmein_complet_errorlog
 </VirtualHost>
```

Explication sur les VirtualHost:
Les virtualhosts permettent de mettre plus d'un site web par IP, en les appelant par nom de domaine ou par IP. Ce qui permet d'avoir des centaines de sites sur un serveur n'ayant qu'une seule IP (c'est comme cela que font tous les hébergeurs, ils ne prennent pas une nouvelle IP pour chaque nouveau client hébergé).

Par exemple, votre serveur web héberge deux sites web d'url: www.domaine1.com et www.domaine2.com, et a comme adresse IP: 80.10.20.30, voici comment définir les deux VirtualHost:

```
 NameVirtualHost 80.10.20.30

<VirtualHost 80.10.20.30>
ServerAdmin webmaster@domain1.com
DocumentRoot /home/domain1/www
ServerName www.domain1.com
ErrorLog /var/log/apache/domain1-error.log
CustomLog /var/log/apache/domain1-access.log combined
ServerAlias domain1.com
</Virtualhost>

<VirtualHost 80.10.20.30>
ServerAdmin webmaster@domain2.com
DocumentRoot /home/domain2/www
ServerName www.domain2.com
ErrorLog /var/log/apache/domain2-error.log
CustomLog /var/log/apache/domain2-access.log combined
ServerAlias domain2.com
</Virtualhost>
```

`NameServer` permet de définir sur quelle IP les virtualhosts sont définis.
`<VirtualHost>` permet de définir un nouvel hôte virtuel apache, avec son adresse IP associée. En combinaison avec la directive `ServerName` il définit aussi le nom avec lequel le serveur doit etre appelé. Cela signifie que si le serveur reçoit une requête sur son IP 80.10.20.30 avec le nom "www.domain1.com", le serveur va donc fournir les pages webs contenues dans `/home/domain1/www` (grâce à la directive `DocumentRoot`).
De la même facon, il va fournir les pages de `/home/domain2/www` si on appelle celle-ci avec l'url www.domain2.com.

Pour que cela fonctionne, il faut bien sur qu'un serveur DNS soit configuré pour faire pointer www.domain1.com et www.domain2.com sur l'IP 80.10.20.30.

Les directives `ErrorLog` et `Customlog` permettent de définir les fichier logs de chaque hôte virtuel (autrement les logs s'ajouteront aux logs principaux du serveur défini dans les directives en dehors de la directive `<VirtualHost>`.

`ServerAlias` permet d'indiquer sous quel autre nom l'hôte virtuel peut etre appelé (ici domainX.com en plus de www.domainX.com), bien sur là aussi le serveur DNS doit être configuré pour faire pointer domainX.com sur 80.10.20.30

Une autre facon de faire des hôtes virtuels, si on n'a pas plusieurs noms de domaine, est de la faire par IP ou par IP et Port.

Par IP, il suffit de faire des `<VirtualHost ip1>` et `<VirtualHost ip2>` etc... , et par port il suffit de faire des `<VirtualHost ip:port1>` `<VirtualHost ip:port2>`

Il faut aussi des directives de `Bind` et `Listen` sur chaque IP et port supplémentaire (voir les directives Listen et Bind plus haut dans ce document).

Dans chaque virtualhost, vous pouvez aussi ajouter d'autres directives, comme les directives de `ScriptAlias` (pour l'exécution de cgi), de `Directory` (option des répertoires du site), ... En gros, toutes les directives peuvent être redéfinies dans chaque VirtualHost pour configurer le site web virtuel. Voir pour cela la documentation officielle d'apache. Pour chaque directive, il est spécifié si elle peut être utilisé dans un contexte de VirtualHost ou pas.

Voilà donc pour les principales directives utilisées dans `httpd.conf`. D'autre directives existent (voir la doc officielle), et les modules chargés par apache ajoutent eux aussi des directives spécifiques (voir la documentation du module ajouté).

Protection d'un serveur Apache PHP MySQL

Jean−Marc LICHTLE

Protection d'un serveur Apache PHP MySQL contre les visites non souhaitées

1 Objectif de ce document

L'objectif est de décrire comment protéger un serveur constitué de la trilogie Apache + PHP + MySQL des ``visites inopportunes''. La description est relative à une installation LINUX, plus spécifiquement la version 8.0 de MANDRAKE.

Exemple de problème de sécurité posé : vous avez installé phpMyAdmin. Il est souhaitable d'interdire qu'un visiteur non autorisé accède à ce script ce qui lui donnerait des droits dangereux sur toutes les bases de données installées.

Il n'entre pas dans les vues de l'auteur de transformer le lecteur en spécialiste de la sécurité informatique. Ce document veut être :

- Un pied à l'étrier pour tout utilisateur qui voudrait étudier la question de la sécurité d'un serveur Web Apache + PHP + MySQL.
- La description du minimum de précautions à mettre en place sur un serveur d'entreprise ne contenant aucune information stratégique mais qui mettrait à la disposition des utilisateurs des informations relativement banalisées. L'objectif de sécurité est relativement sommaire :
  - ♦ − éviter que des manipulations non souhaitées ne détruisent une partie ou la totalité des informations,
  - ♦ − limiter les droits d'accès à certaines informations plus sensibles à un nombre limité d'utilisateurs identifiés.

2 Introduction

Ce texte est la suite naturelle d'un document décrivant l'installation de la trilogie Apache PHP MySQL en environnement LINUX MANDRAKE 8.0 (voir l'article). L'installation décrite dans ce premier document est la plus simple possible, son seul objectif était d'aboutir à un système qui soit fonctionnel. Dans l'installation de MySQL par exemple j'avais précisé de faire l'impasse sur la mise en place d'un mot de passe administrateur afin de ne pas bloquer le fonctionnement ultérieur de phpMyAdmin dans son installation de base. Il est temps maintenant de faire le pas supplémentaire qui verrouillera un peu mieux le site.

Pour plus d'informations on pourra se reporter à l'importante littérature qui existe sur la question, notamment ``Pratique de MySQL et PHP'', édition O'REILLY.

Il importe de comprendre que la protection d'un site Apache + PHP + MySQL prend des facettes multiples, chaque logiciel ayant ses ``protections'' propres. De plus comme nous le verrons bientôt avec phpMyAdmin, ces systèmes de protections dialoguent entre eux.

3 Protection du serveur Apache

Il s'agit dans un premier temps de protéger le serveur http, c'est à dire le logiciel qui va adresser les pages html au navigateur qui en fait la demande. A ce stade il importe peu que PHP et MySQL soient exploités ou non. Cette protection est très générale et s'applique aussi bien à un serveur simple ne mettant en oeuvre que des pages html pures qu'à un serveur plus évolué faisant appel à un langage de programmation et à un gestionnaire de bases de données (PHP et MySQL ou une autre combinaison). Les exemples qui suivent portent sur la protection d'un sous répertoire `test/` situé au premier niveau de l'arborescence du serveur, plus précisément `/var/www/html/test/`. Attention, toutes les informations de chemins d'accès sont relatives à la distribution Mdk 8.0 !

3.1 Protection par .htaccess

Le principe consiste à créer dans le sous répertoire à protéger un fichier nommé `.htaccess` (commençant par un point donc caché) et qui va en limiter les droits d'accès. En fait `.htaccess` va renvoyer vers un fichier contenant les logins et mots de passe des utilisateurs autorisés.

Ce fichier de mots de passe peut porter un nom quelconque. Habituellement ce dernier est rangé dans `/etc/httpd/auth/`, répertoire qu'il convient éventuellement de créer puisqu'il n'existe pas après l'installation de base. Il est assez logique que tous les fichiers d'authentification seront rangés dans le même sous−répertoire. Il est de ce fait judicieux de choisir des noms de fichiers qui soient parlant et qui rappellent l'objet de la protection, par exemple test.users pour le fichier qui définira les droits d'accès des diféfrents utilisateurs au sous répertoire test. L'emploi de `/etc/httpd/auth/` pour ranger les fichiers d'autorisation est une pure question de convention. Ces fichiers pourraient aussi bien être rangés n'importe ou ailleurs. Il va toutefois sans dire (mais mieux en le disant) que ces fichiers doivent être hors de portée des visiteurs, en clair ne doivent pas faire partie de l'arborescence `/var/www/html/`...

3.1.1 Création de .htaccess

Exemple simple :

Avec un éditeur quelconque (de préférence emacs =;−) créer le fichier minimum suivant:

```
AuthUserFile /etc/httpd/auth/test.users

AuthName ``Accès restreint''

AuthType Basic

require valid−user
```

Sauvegardez...

`AuthUserFile` définit quel est le fichier chargé de contenir les autorisations. La Mandrake 8.0 suppose par défaut que les fichiers d'autorisation sont rangés dans `/etc/httpd/`. La première partie de l'adresse `AuthUserFile` peut donc être omise éventuellement.

`AuthName` précise le message qui sera affiché dans la boite de dialogue, dans ce cas ``Accès restreint à localhost'' si la machine est nommée localhost. Nous verrons plus loin l'intérêt de choisir un message aussi explicite que possible.

3.1.2 Création du fichier des autorisations d'accès

La première étape consiste à créer, s'il n'existe pas encore, le sous répertoire `/etc/httpd/auth/` qui contiendra les autorisations (md... sous compte root). Le fichier d'autorisation lui−même se crée (et s'entretient) avec la commande `htpasswd`. Attention: la commande htpasswd peut être lancée aussi bien par un utilisateur de base que par l'administrateur. Toutefois, les fichiers contenus dans `/etc` étant tous la propriété de root je préconise, dans un but de cohérence, de lancer cette commande uniquement sous compte root. De toute façon, sauf à imaginer que `/etc/httpd/auth/` soit propriété d'un utilisateur non root, le lancement de `htpasswd` depuis un login non root conduirait à un échec à l'enregistrement des données.

Syntaxes:

- Pour la création (avec un premier utilisateur lambda): `htpasswd −c /etc/ httpd/auth/ test.users lambda`. htpasswd va alors vous demander le mot de passe de lambda (avec confirmation).
- Pour l'ajout d'un utilisateur supplémentaire: `htpasswd /etc/httpd/auth/test.users lambda2`
- Pour supprimer un utilisateur et ben le plus simple est de supprimer la ligne correspondante dans le fichier d'autorisation !

La consultation de .htaccess se fait à chaque accès du serveur Apache au sous−répertoire concerné. Avantage : la nouvelle configuration entre en action immédiatement sans qu'il soit nécessaire de relancer le serveur Apache pour recharger la nouvelle configuration.

3.2 Protection par modification de httpd.conf

Le principe est assez similaire à ce qui a été exposé ci−dessus si ce n'est que le renvoi vers le fichier des autorisations est réalisé dans `/etc/httpd/conf/httpd.conf`.

Pour mettre en place ce renvoi éditez ce fichier et ajoutez (par exemple tout à la fin) les lignes suivantes :

```
<Directory /var/www/html/test>

AuthName "Accès limité"

AuthUserFile /etc/httpd/auth/test.users

AuthType Basic

require valid−user

</Directory>
```

Pour protéger plusieurs répertoires créez plusieurs paragraphes `<Directory> </Directory>`.

Sauvegardez puis relancez le démon Apache (comme root : `/etc/rc.d/init.d/httpd restart`). En effet dans ce cas vous venez de modifier un des fichiers de configuration du serveur, fichier qui est lu au moment du chargement de Apache. Il convient donc de forcer Apache à relire ce fichier pour prendre en compte les modifications.

3.3 Différence entre les deux méthodes .htaccess et httpd.conf

Votre attention aura certainement été attirée par le similitude de rédaction entre les deux solutions. Cette similitude n'est absolument pas fortuite et provient simplement de la logique interne du fonctionnement d'Apache. Le serveur lit au lancement les paramètres de configuration stockés dans httpd.conf. Il ne les relira ensuite qu'en cas de demande explicite (restart).

Par contre, il cherchera à chaque requête d'un navigateur, à lire un éventuel fichier .htaccess contenu dans le sous−répertoire concerné par la demande. S'il le trouve les informations contenues dans ce fichier remplaceront, pour la durée de la requête, celles qui sont contenues dans httpd.conf. En clair vous pouvez ``masquer'' la configuration officielle contenue dans httpd.conf en proposant une nouvelle configuration dans un .htaccess. Faites l'essai en mettant en place une protection double :

- Par httpd.conf en mettant ``Accès contrôlé par httpd.conf'' dans la variable AuthName.
- Par .htaccess en mettant ``Accès contrôlé par htaccess'' dans AuthName.

Vous pourrez vérifier simplement (en renommant .htaccess par exemple) que la protection par .htaccess, lorsqu'elle est en place, remplace bien la protection induite par httpd.conf.

L'intérêt de cette petite démonstration ne semble pas évident. Trois éléments méritent toutefois d'être retenus :

- Une protection par .htaccess nécessite un travail supplémentaire d'analyse de la part du serveur (à prendre en compte pour les serveurs très chargés ou installés sur des machines poussives).
- Certaines directives de httpd.conf peuvent être neutralisées, modifiées...
- Une protection par .htaccess est prise en compte immédiatement sans nécessité d'intervenir sur le serveur.

3.4 Protection de fichiers particuliers dans un sous répertoire

La protection par .htaccess décrite ci−dessus s'applique à l'ensemble d'un sous−répertoire. En fait vous pouvez différencier, si vous le souhaitez, les accès aux fichiers. La syntaxe de .htaccess s'en trouve légèrement modifiée, par exemple pour protéger un fichier nommé `common.php` :

```
<Files common.php>

AuthName "Accès limité au fichier common.php par .htaccess"

AuthUserFile /etc/httpd/auth/common.users

AuthType Basic

require valid−user

</Files>
```

L'intérêt de libeller `AuthName` de façon explicite apparaît ici de façon nette. Lors des essais vous pourrez en effet savoir immédiatement quels sont les contrôles d'accès qui sont en place.

Vous pouvez souhaiter protéger plus spécifiquement l'accès à un fichier particulier. La solution consiste à créer un fichier .htaccess libellé comme suit :

Exemple pour protéger un fichier `common.php` :

```
<Files common.php>

Order Deny,Allow

Deny from All

</Files>
```

Pour protéger plusieurs fichiers il suffit de créer plusieurs rubriques `<Files> </Files>`.

Attention: cette méthode interdit l'accès via le serveur Web pour tout le monde, y compris le propriétaire du fichier, l'administrateur, etc.

3.5 Rendre le contenu d'un sous−répertoire invisible

L'astuce qui suit consiste simplement à faire afficher par le serveur Web une page html qui signale au visiteur qu'il n'a rien à faire dans ce sous−répertoire.

Lorsque Apache accède à un sous−répertoire pour satisfaire à la demande d'un navigateur client il commence par vérifier les autorisations d'accès (voir ci−dessus). S'il ne rencontre pas de veto alors il part à la recherche d'un éventuel fichier index.html ou index.php (si PHP est installé). S'il ne trouve pas ce fichier alors il affiche le contenu du sous−répertoire qui apparaît donc en clair pour le visiteur.

Pour éviter que le contenu du sous−répertoire ne s'affiche il suffit donc de créer un fichier index.html ou index.php. Concevoir le contenu de ce fichier de telle sorte qu'il réponde à votre attente, du simple message d'avertissement au script PHP qui va renvoyer de force le visiteur dans le droit chemin. C'est simple et relativement efficace pour un environnement non critique. La solution la plus simple consiste simplement à créer un fichier index.html vide. Le navigateur recevra donc une page blanche. Pas très explicite mais simple et efficace !

Attention, il s'agit là simplement de créer une dissuasion. Nous ne sommes plus dans le domaine de la protection gérée comme ci−dessus. Un utilisateur qui connaîtrait un ou plusieurs noms de fichiers contenus dans ce sous−répertoire pourrait y accéder sans le moindre problème en tapant simplement leurs URL complets ! De grâce, ne construisez pas un site central de banque avec de telles méthodes !

4 Protection du gestionnaire de bases de données MySQL

Un peu comme dans le cas d'Apache cette section est spécifique à MySQL. Il importe peu pour ce qui va suivre qu'Apache soit installé ou non (idem pour PHP). Nous supposerons que les notions de bases du langage SQL sont connues (ben voyons !). La première partie de ce chapitre va en effet faire largement appel à ce langage pour régler la configuration. Attention, nous allons utiliser le client texte MySQL ! Efficace mais pour le moins spartiate !

Pour commencer ouvrez une fenêtre texte (xterm, rxvt, terminal ou autre) et lancez le client MySQL par :

`# mysql −u root` (on suppose que vous êtes logué comme root).

La réponse devrait être immédiate ``Welcome to the MySQL Monitor ....'' et se terminer par l'invite de commande de MySQL qui est modestement mysql>. Je reproduirais cette invite pour tous les exemples de syntaxe SQL donnés ci−dessous.

Attention, vous êtes logué comme administrateur, tout ce que vous allez faire pourra devenir dramatique en cas d'erreur !

tapez `mysql>use mysql;` pour préciser au serveur d'utiliser dans ce qui suit la base de données nommée mysql.

La réponse devrait se terminer par ``Database changed''.

4.1 Configuration des fichiers d'autorisations d'accès

La protection de MySQL s'articule autour des fichiers d'autorisations (qui constituent eux−même une base de données MySQL) qui définissent très précisément les droits de chaque utilisateur, des informations les plus globales (telle base de donnée est accessible à tel utilisateur) aux plus pointues (la n−ième colonne de telle table est accessible pour tel utilisateur mais seulement pour telle opération).

Ces droits sont différenciés, consultation, mise à jour, etc.

La base de données qui décrit la structure des autorisations d'accès est contenue dans `/var/lib/mysql/mysql/` (dans l'installation Mdk8.0). Cette BDD est créée automatiquement par le script d'installation du package MySQL. Vous pouvez consulter la liste des tables de cette base par `mysql>show tables;`

Cette commande affiche la liste des tables constituant la BDD.

L'objet du présent document étant de constituer une aide de départ (et non de remplacer la documentation spécialisée) nous ne nous intéresseront dans ce qui suit qu'à une seule table de cette base de données, la table user qui défini les droits d'accès globaux des utilisateurs à l'ensemble des BDD. De même nous n'étudierons que le cas d'une machine isolée servant à la fois de serveur et de client. La configuration en réseau n'est guère plus complexe mais dépasserait le cadre de la présente étude.

4.1.1 Structure de la table user

La structure de la table user est assez simple. Elle contient les champs suivants:

`Host char(60)`	nom de la machine depuis laquelle est fait l'appel
`User char(16)`	login de l'utilisateur
`Password char(16)`	mot de passe codé
`Select_priv`	droits d'effectuer des requêtes sélect (valeur Yes ou No)
`Insert_priv`	...
`Update_priv`	...
`Delete_priv`	...
`Create_priv`	droits d'effectuer une création de table
`Drop_priv`	droits de suppression d'une table
`Reload_priv`	droits de relancer le serveur MySQL
`Shutdown_priv`	droits d'arrêter le serveur MySQL
`Process_priv`	
`File_priv`	
`Grant_priv`	
`References_priv`	
`Index_priv`	
`Alter_priv`	

Cette structure peut être consultée très simplement par `mysql>desc user;` Son contenu peut être affiché par `mysql>select * from user;`

4.1.2 Mise à jour de la table user

Cette table contient d'origine (après l'installation décrite dans le document précédent) 4 enregistrements qui définissent les droits de l'administrateur depuis localhost et localhost.localdomain et ceux d'un utilisateur non nommé depuis les mêmes hôtes. L'administrateur root (dont vous utilisez le compte actuellement) a tous les droits, l'utilisateur non nommé aucun, si ce n'est celui d'accéder à la base de données mais sans pouvoir faire la moindre opération. Vous constaterez également que root n'a pas de mot de passe (voir plus haut, cette partie du script a été sautée au moment de l'installation de façon à ne pas bloquer le fonctionnement de phpMyAdmin).

La première étape va consister à supprimer les lignes qui sont inutiles de telle sorte à ne conserver qu'une seule ligne, celle qui correspond à l'administrateur root depuis localhost.

Tapez donc :

```
mysql>delete from user where Host='localhost.localdomain';

mysql>delete from user where User='';

mysql>select * from user;
```

Cette fois la liste devrait contenir le seul enregistrement relatif à root depuis localhost.

Pour ajouter un utilisateur lambda procédez comme suit :

```
mysql>insert into user values('localhost','lambda',password('mdp_lambda'),
'Y','Y','Y','Y','N','N','N','N','N','N','N','N','N','N');
```

Ne comptez pas, il y a 4 'Y' et 10 'N' !

Vous venez de créer un nouvel utilisateur nommé lambda et dont le mot de passe est mdp_lambda (vous mettez bien sûr ce qui vous convient).

La syntaxe de cette ligne mérite quelques commentaires :

- Insert into user ... donne l'ordre de créer une nouvelle ligne dans la table user.
- Une ligne insert complète (avec toutes les options SQL) préciserait la liste des champs à compléter, dans ce cas particulier nous donnons exactement autant de valeurs (values) qu'il y a de champs dans la table, l'affectation des valeurs aux champs devient donc implicite (et la ligne plus simple !).
- La liste values() contient exactement 17 valeurs séparées par des virgules. Les chaînes de caractères sont placées entre guillemets simples (').
- La composition de cette liste reflète précisément la structure de la table user mise en évidence précédemment avec `mysql>desc user;`
- Le nouvel utilisateur dispose de 4 droits simples, sélection, insertion de nouvelles lignes, modifications de lignes existantes et effacement de lignes. Il ne peut ni créer une nouvelle table, ni en effacer une existante. Toutes les fonctions d'administration lui sont interdites.
- Password() est une fonction texte qui transforme un mot de passe donné en clair en une chaîne codée.

Vous pouvez maintenant quittez le client mysql en tapant `mysql>quit;`

4.1.3 Rechargement de la table des autorisations par MySQL

Il reste à vérifier la validité de ce nouvel utilisateur en essayant de se connecter avec ce nouveau login. Mais avant il faut que MySQL actualise la table des autorisations conservée en mémoire et chargée au lancement initial du serveur.

Pour cela tapez dans un terminal :

```
# mysqladmin flush−privileges
```

Cette commande demande à MySQL de recharger les tables définissant les autorisations d'accès depuis le disque sans pour autant arrêter le serveur.

Une autre solution plus brutale serait d'arrêter le serveur et de le redémarrer en tapant (compte root) `# /etc/rc.d/init.d/mysql stop` puis `# /etc/rc.d/init.d/mysql start`.

On peut enfin arrêter complètement LINUX et relancer mais là c'est la honte, n'est−ce pas ? Nous ne sommes pas sous Window$ que diable !

4.1.4 Login avec le nouveau compte

Vous pouvez maintenant vous loguer avec le compte nouvellement créé. Plusieurs possibilités s'offrent à vous :

- `$ mysql −u lambda −pmdp_lambda`
  Cette syntaxe vous donne l'accès direct. Attention: vous laissez un espace après −u mais pas après −p, collez directement le mot de passe à −p !Inconvénient : le mot de passe apparaît en clair, sur l'écran mais aussi dans l'historique des frappes au clavier.
- `$ mysql −u lambda −p`
  Avec cette syntaxe le serveur MySQL vous demandera immédiatement le mot de passe associé au login lambda. Vous devrez taper ce mot de passe en aveugle. Il n'apparaîtra donc ni à l'écran, ni dans un historique de frappes.
- `$ mysql −u lambda −p mysql`
  Variante de la précédente, l'espace après −p va provoquer une demande de mot de passe comme ci−dessus, l'indication de mysql va indiquer au serveur d'utiliser la base de donner mysql (équivalent à `mysql>use mysql;`).

Ce qui ne marche pas:

- `$ mysql −u lambda`
  MySQL va protester parce que l'utilisateur lambda est connu comme étant validé par un mot de passe. Le message sera assez explicite puisqu'il se termine par Using password: YES. Nota : voir plus loin le chapitre Automatisation des connexions, il sera possible d'automatiser l'indication du mot de passe et donc de se connecter en utilisant cette syntaxe.

Voilà, vous êtes maintenant en mesure de modifier efficacement le fichier user pour accorder et retirer des droits à chacun des utilisateurs de votre système. Ce chapitre n'a simplement fait que soulever un coin du voile sur la question. L'étude de l'utilisation des autres fichiers de `/var/lib/mysql/mysql/` vous permettra de définir encore mieux ces droits et de créer des droits différenciés par BDD et par utilisateur, voire même par table ou par champ. L'étape suivante s'articulerait autour de la table db. Cette étude de détail sortirait toutefois du cadre fixé ici.

4.2 Automatisation des connexions

A ce stade nous pouvons définir autant d'utilisateurs que souhaité. Chacun devra se connecter à MySQL avec la syntaxe $ mysql −u utilisateur −p (utilisateur étant à remplacer par le login). En réponse MySQL demandera le mot de passe correct.

Il peut paraître lourd, alors que LINUX vous a déjà demandé de vous identifier correctement au login, de se re−identifier à nouveau à chaque lancement d'un client MySQL.

Cette procédure est heureusement contournable.

A chaque lancement d'un client MySQL celui−ci va en effet vérifier la présence des fichiers suivants:

- /etc/my.conf

- /var/lib/mysql/my.conf
- /.my.conf, étant le répertoire home de l'utilisateur LINUX dûment logué et identifié.

Dès qu'il trouvera un fichier le client MySQL lira les données de configuration qui y sont rangées et ira à la recherche du fichier suivant dans la liste et dans cet ordre.

Les données les plus récentes iront à chaque fois écraser les plus anciennes ce qui permet de personnaliser très finement le comportement du client MySQL en fonction des besoins de l'utilisateur. Et parmi ces données, mais vous l'aurez déjà deviné, figurent le mot de passe de connexion au serveur MySQL !

Dans une installation de base aucun de ces fichiers n'existe. Par contre le sous−répertoire /usr/share/mysql/ contient un certain nombre de fichiers exemples nommés my−small.cnf, my−medium.cnf, etc. qui correspondent à des propositions de fichiers de configuration pour différents cas d'utilisation. Les premières lignes de chacun de ses fichiers précisent les domaines d'application.

Pour automatiser le login MySQL pour un utilisateur lambda en se basant sur le modèle medium procéder comme suit :

- Copier /usr/share/mysql/my−medium.cnf vers /.my.cnf (fichier caché).
- Editer /.my.cnf et modifiez la rubrique [client], la première qui soit proposée après l'entête, de telle sorte à faire apparaître une ligne password = mot_de_passe (remplacer mot_de_passe par le mot de passe MySQL en clair de l'utilisateur). Dans le fichier exemple cette ligne est affectée d'un signe commentaire (#) qu'il faudra effacer. Pour simplifier vous pouvez même supprimer toutes les autres ligne de ce fichier et ne laisser que le paragraphe [client]. De la sorte vous évitez d'écraser des configurations ajustées qui auraient été mises en place, ou pourraient l'être ultérieurement, dans /etc/my.conf ou /var/lib/mysql/my.conf.
- Sauvegardez et modifiez les droits d'accès du fichier sauvegardé en tapant $ chmod 700 .my.cnf pour limiter les droits d'accès. Il serait en effet détestable qu'on puisse lire votre mot de passe MySQL !

Le tour est joué ! A la prochaine connexion avec le client MySQL celui−ci trouvera le mot de passe et établira une connexion directe en utilisant le login LINUX de l'utilisateur et le mot de passe mémorisé dans `/.my.cnf`. Pour vérifier tapez simplement $ mysql pour lancer un client MySQL. La connexion devrait être immédiate. Vérifiez sous quel login vous êtes entré en tapant `mysql> select user();` ce qui devrait provoquer l'affichage d'un tableau à une seule cellule titrée user() et contenant une information du genre lambda@localhost.

5 Contrôle des accès MySQL initiés par des scripts PHP.

Un script PHP qui accède à une base de données MySQL le fait dans des conditions fixées par la syntaxe de la commande de connexion. En PHP on obtient une connexion à un serveur MySQL au moyen de la commande `mysql_connect()`. Celle−ci utilise trois arguments, $host, $user, $password et renvoie une valeur TRUE, 1 si la connexion est établie, FALSE, 0 si celle−ci a échoué.

Le login de connexion ainsi que le mot de passe apparaissent donc en clair dans le script de connexion ce qui est moyennement satisfaisant. Aspect positif le script lui−même n'est jamais envoyé au client puisque PHP s'exécute entièrement sur le serveur (contrairement à JAVA). Il n'y a donc aucune raison pour qu'un visiteur puisse lire ce mot de passe. Oui, mais... il n'y a pas que des visiteurs ``normaux'' !

Une protection supplémentaire (et aussi une simplification si on rédige beaucoup d'applications PHP) est de définir ses variables $host, $user et $password dans un fichier extérieur au script lui−même et d'incorporer ces valeurs au moyen d'une directive include. Avantage : le fichier qui contient les variables peut être stocké en dehors des sous−répertoires consultés normalement par Apache (hors de l'arborescence /var/www/html/), et protégé par un fichier .htaccess.

Finalement le contrôle d'accès à MySQL via un script PHP découle directement de ce qui a été exposé au chapitre précédent et s'appuie directement sur les règles d'accès à MySQL définies dans la base de données /var/lib/mysql/mysql/. Il peut donc s'avérer souhaitable de créer un compte utilisateur MySQL spécifique aux applications PHP qui tournent sur le serveur. On pourrait aussi imaginer d'en créer plusieurs, 1 par famille de scripts par exemple.

6 Cas particulier de phpMyAdmin.

PhpMyAdmin est une petite merveille, un ensemble de scripts PHP, qui permet d'accéder au serveur MySQL en utilisant une interface graphique plus avenante que le spartiate client MySQL en mode texte.

L'installation par défaut conduit à faire de phpMyAdmin un client MySQL avec un login root sans mot de passe (d'où l'intérêt de ne pas mettre, dans un premier temps, de mot de passe au compte root faute de quoi phpMyAdmin ne serait plus en mesure d'accéder au serveur MySQL).

En fait le fonctionnement de phpMyAdmin est réglé par le fichier `/var/www/html/phpMyAdmin/config.inc.php3`

Ce fichier, assez court, contient notamment les lignes suivantes :

```
// The $cfgServers array starts with $cfgServers[1]. Do not use $cfgServers[0].

// You can disable a server config entry by setting host to ''.

$cfgServers[1]['host'] = 'localhost'; // MySQL hostname

$cfgServers[1]['port'] = ''; // MySQL port − leave blank for default port
```

```
$cfgServers[1]['adv_auth'] = false; // Use advanced authentication?

$cfgServers[1]['stduser'] = 'root'; // MySQL standard user (only needed with advanced
auth)

$cfgServers[1]['stdpass'] = ''; // MySQL standard password (only needed with advanced
auth)

$cfgServers[1]['user'] = 'root'; // MySQL user (only needed with basic auth)

$cfgServers[1]['password'] = ''; // MySQL password (only needed with basic auth)

$cfgServers[1]['only_db'] = ''; // If set to a db-name, only this db is accessible

$cfgServers[1]['verbose'] = ''; // Verbose name for this host - leave blank to show the
hostname
```

Le texte de ce fichier semble assez explicite. Il nécessite toutefois une analyse plus détaillée pour bien en comprendre certaines finesses.

$cfgServers[1] désigne le premier serveur MySQL, le seul qui nous intéresse ici (il semblerait qu'il y ait des vicieux qui démarrent plusieurs serveurs.. imaginez un peu ;-)).

Le paramètre important est [adv_auth]. Il peut prendre deux valeurs, false (défaut à l'installation) et true.

6.1 Configuration avec adv_auth = false

Avec false l'authentification se fait selon une ancienne méthode qui utilise les valeurs de $cfgServers[1]['user'] et $cfgServers[1]['password'] pour s'identifier auprès de MySQL. L'installation de base résumée ci-dessus reflète cette authentification. En clair phpMyAdmin s'enregistre sous compte root et sans mot de passe. C'est précisément la raison pour laquelle il ne faut surtout pas, en cours d'installation de MySQL, donner suite à la proposition du script de définir un mot de passe administrateur (voir http://jeanmarc.lichtle.free.fr/Apache_PHP_MySQL.html). Définir un mot de passe à ce stade de l'installation bloquerait le fonctionnement ultérieur de phpMyAdmin. Pour le vérifier faites simplement l'essai de fixer un mot de passe pour root dans MySQL. La syntaxe SQL est la suivante (mais vous l'aurez déjà deviné) :

```
mysql> update user set Password=password('mdp_root') where user = 'root';
```

Un petit coup de `# mysqladmin flush-privileges` (y'en a qui avaient oublié !?) et voilà le mot de passe administrateur pris en compte par le serveur MySQL. Pour la petite histoire c'est la dernière fois que vous tapez cette commande avec cette syntaxe simple. Maintenant que root a un mot de passe dans MySQL il faudra utiliser la syntaxe `# mysqladmin -u root -p flush-privileges`, ce qui va provoquer la demande du mot de passe par le serveur.

Et maintenant adieu phpMyAdmin, ça ne marche plus ! Seulement voilà, arrivé à ce stade de notre étude nous savons comment faire prendre en compte ce mot de passe par phpMyAdmin. Il suffit de l'incorporer au fichier de configuration `/var/www/html/phpMyAdmin/config.inc.php3` à la ligne $cfgServers[1]['password'] = ''.

Au prochain lancement de phpMyAdmin le fichier de configuration va être pris en compte et tout va rentrer dans l'ordre, la connexion s'effectuant à nouveau de façon directe...

Bon, ça ne marche pas ? Ne vous affolez pas ! Je fais le pari que vous avez fait des essais de .htaccess ou de httpd.conf sur le sous répertoire de phpMyAdmin et qu'il reste des traces des essais effectués précédemment. Supprimez (ou renommez) l'éventuel fichier .htaccess situé dans le sous répertoire `/var/www/html/phpMyAdmin/` et/ou mettez des commentaires (#) aux éventuelles lignes ajoutées à `/etc/httpd/conf/httpd.conf`. Attention de relancer Apache si vous modifiez httpd.conf !

6.2 Configuration avec adv_auth=true

L'authentification avancée diffère légèrement du cas précédent en ce que le login et le mot de passe de l'utilisateur qui lance phpMyAdmin sont demandés au lancement. Il n'y a plus un login de connexion unique, celui-ci peut changer en fonction des informations données par l'utilisateur qui lance phpMyAdmin. Ces informations sont ensuite comparées à la base de données des utilisateurs autorisés par MySQL. La connexion est établie si les informations sont correctes.

Il faut toutefois établir une première connexion de façon à avoir un accès temporaire à la table user. C'est à cette fin que la configuration prévoit un compte défini par $cfgServers[1]['stduser'] et $cfgServers[1]['stdpass'], ces deux variables devant permettre la vérification des droits d'accès. Il va sans dire que le [stduser] désigné ici devra avoir des droits suffisants pour lire la table user.

Cette authentification offre au minimum deux avantages :

- Les accès à MySQL via phpMyAdmin sont différenciés suivant les utilisateurs et suivent exactement les règles d'autorisation d'accès à MySQL.
- Le mot de passe administrateur n'a plus à figurer dans le fichier de configuration de phpMyAdmin. Il suffit de créer dans la table user de MySQL un utilisateur, nommé par exemple phpMyAdmin, dont les droits sont strictement limités à la consultation des tables (et donc de la table user). Il suffit de mettre un seul 'Y' et 13 'N' dans la fameuse requête de création d'un utilisateur. N'oubliez pas de relancer `mysqladmin -u root -p flush-privileges` hein !

Pour le fun, et c'est une excursion vers l'étude de la configuration avancée des droits d'accès MySQL on peut même limiter encore plus les droits de cet utilisateur spécial. Il suffit de ne lui donner aucun droit dans la table user (14 fois 'N') et de lui créer une ligne dans la table db. Cherchez bien, la

syntaxe est extrêmement proche de celle qui ajoute un utilisateur dans user, il suffit de préciser cette fois que ces droits s'applique spécifiquement à la base de données mysql. L'utilisateur 'phpMyAdmin' aura donc uniquement le droit de lire le contenu des tables user, db etc.. qui constituent la base mysql.

Vous pouvez même aller plus loin dans le raisonnement et limiter les droits strictement à la table user. Mais là c'est une autre affaire, je sens que vous allez passer quelques heures sur l'étude détaillée des droits d'accès MySQL.

7 Le petit bréviaire

Je ne sais pas si vous êtes comme moi mais j'ai besoin de prendre des notes pour retrouver rapidement les informations vitales dans un document aussi touffu que celui−ci.

Parmi les informations qu'il me semble intéressant de résumer ici je dresse à toutes fins utiles la liste des fichiers et répertoires qui ont été évoqués ci−dessus (dans leur ordre d'apparition en scène):

Répertoire /fichier	Fonction
`/var/www/html/`	Base arborscence site Web
`/var/www/html/phpMyAdmin/config.inc.php3`	Configuration de phpMyAdmin
`/var/lib/mysql/mysql/`	Base de données des droits d'accès
`/var/lib/mysql/my.conf`	Configuration de base de MySQL (optionnel)
`/etc/httpd/auth/`	Rangement des fichiers d'authentification par .htaccess
`/etc/httpd/conf/httpd.conf`	Configuration serveur Apache
`/etc/rc.d/init.d/httpd`	Lancement du serveur Apache option restart
`/etc/rc.d/init.d/mysql`	Lancement du serveur MySQL opt stop ou start
`/etc/my.conf`	Configuration de base de MySQL (optionnel)
`/.my.conf`	Configuration de base de MySQL (opt. spécif. util.)
`/usr/share/mysql/`	Contient des exemples de fichier my.cnf
`/var/log/httpd/`	Contient les fichiers log

Je précise à nouveau que ce document s'applique au cas d'une installation en environnement LINUX Mandrake 8.0 telle que décrite dans un document identifié plus haut. Un système qui serait basé sur une autre distribution et/ou qui utiliserait des versions d'Apache, PHP ou MySQL qui auraient été compilées par l'utilisateur pourrait utiliser d'autre répertoires pour stocker les différentes informations. L'essentiel est d'obtenir un ensemble cohérent.

8 Conclusion

L'exposé ci−dessus devait vous mettre en situation d'appréhender sérieusement la question assez complexe (mais oh combien stratégique) du contrôle des accès sur un serveur Apache PHP MySQL. J'espère avoir atteint cet objectif en montrant comment structurer la réflexion :

- protection du serveur Apache
- protection du serveur MySQL
- mise en place de contrôles d'accès via le langage PHP et plus précisément avec le script phpMyAdmin.

Il est évident qu'un sujet aussi vaste ne peut être qu'effleuré en quelques pages aussi je vous renvoie vers la littérature plus complète (livres, HowTo, pages man etc..) sur les différents points abordés.

J'espère sincèrement que ce document servira un jour à quelqu'un. Sa rédaction est de toute façon justifiée par le simple fait que l'exercice m'a obligé à mettre mes connaissances et mes notes au propre, à tester pas à pas toutes mes propositions. J'en suis donc le premier lecteur. N'hésitez pas à me faire part de votre avis sur l'intérêt de cette étude. Toute suggestion d'amélioration sera bienvenue.

L'auteur

JML dit Jean−Marc LICHTLE, email jean−marc.lichtle@gadz.org, ingénieur Arts et Métiers promo CH173 (rigolez pas, à l'époque les écrans n'existaient pas, le conversationnel c'était à grands renforts de télétypes et de rouleaux de papier! Pire encore, 16 clés, 16 voyants et un bouton run! Beurk.....)

PostgreSQL: installation

par serge, Mise à jour par Guillaume LELARGE et Anne

Une petite description de l'installation de PostgreSQL.

Introduction

PostgreSQL est un système de gestion de base de données relationelles, c'est−à−dire un SGDBR. C'est un logiciel libre. Sous Linux, les deux SGDBR les plus utilisés sont PostgreSQL et MySQL. MySQL est très utilisé pour de petites applications ou pour générer des pages Web dynamique (comme les forums de ce site par exemple) mais comporte quelques lacunes par rapport à PostgreSQL. De plus, PostgreSQL est mieux adapté pour de plus grosses bases, il est plus robuste en quelques sortes.
Cette rubrique n'est pas un apprentissage de PostgreSQL mais juste un "manuel d'installation".

Création de l'administrateur PostgreSQL

PostgreSQL gère lui aussi des utilisateurs pour attribuer des droits aux bases de données comme linux gère les droits pour la gestion des fichiers et du système. Donc, comme pour linux, il faut un "root", autrement dit un administrateur, pour PostgreSQL. Pour des questions de sécurité, **utilisez toujours un compte utilisateur qui n'a aucun droit d'administration linux pour le compte "root" de PostgreSQL.** On va donc créer cet utilisateur, avec par exemple comme nom **postgres**. Tapez en tant qu'utilisateur root :

```
# adduser postgres
```

Validez toutes les questions, et si le système vous demande un mot de passe, saisissez−en un. Au cas où aucun mot de passe ne vous est demandé, lancez la commande suivante :

```
# passwd postgres
```

Saisissez le mot de passe une première fois, puis confirmez−le.

Profitons d'être root pour créer le répertoire où nous déballerons les sources et pour donner les droits sur ce répertoire à l'utilisateur postgres :

```
# mkdir /usr/src/pgsql
# chown −R postgres /usr/src/pgsql
```

Maintenant, connectez−vous en tant qu'utilisateur postgres pour toutes les commandes ci−dessous, sauf si je vous indique clairement de vous connecter de nouveau en tant qu'utilisateur root. Pour tout le reste du document, je suppose que votre administrateur PostgreSQL est "postgres", à vous d'adapter suivant le nom que vous avez donné à ce compte.

Récupération des sources et compilation

Bon, tout d'abord, récupérez les sources sur un des miroirs. Au moment où j'écris cet article, la dernière version stable est la 7.4.2. Récupérez l'archive tar postgresql−7.4.2.tar.gz (ou une autre version si une mise à jour est disponible). Une fois les sources récupérées, placez−vous dans le répertoire que nous venons de créer, placez−y l'archive tar compressée et lancez la décompression avec la commande :

```
$ cd /usr/src/pgsql
$ tar xvfj postgresql−7.4.2.tar.bz2
```

Entrez dans le répertoire créé et lancez l'opération de configuration :

```
$ cd /usr/src/pgsql/postgresql−7.4.2
$ ./configure "options"
```

Dans cette commande, "options" doit être remplacé par une ou plusieurs options de configuration. En voici quelques−unes parmi les plus intéressantes :

- **`--prefix=chemin`**, chemin où vous voulez installer PostgreSQL, sachant que celui−ci s'installe par défaut dans le répertoire /usr/local/pgsql ;
- **`--enable−nls`**, ajoute le support des locales (support multi−langues) ;
- **`--with−odbc`**, compile le module ODBC ;
- **`--with−jdbc`**, compile le module JDBC ;
- **`--with−perl`**, ajoute le support de Perl pour PostgreSQL ainsi que les modules Perl (utile si vous voulez utiliser Perl pour vos applications avec PostgreSQL) ;
- **`--with−tcl`**, ajoute le support de Tcl/Tk ;
- **`--with−pam`**, ajoute le support de PAM ;
- **`--with−openssl`**, ajoute le support d'OpenSSL.

Remarque : pour les développeurs qui souhaitent ajouter d'autres supports ou fonctions, tapez la commande ./configure −−help pour connaître toutes les options disponibles.

Une fois que le **`./configure`** a terminé son travail sans erreurs, lancez la compilation des sources avec :

```
$ make
```

Une étape optionnelle, mais néanmoins intéressante, est de lancer une vérification avec la commande :

$ make check

PostgreSQL va créer une base de données temporaire et tester un certain nombre d'opérations, comme la création d'une table, l'utilisation de champs de type divers, etc. Normalement, vous devriez avoir un taux de réussite de 100 %.

Passez ensuite en super−utilisateur root pour lancer l'installation :

make install

puis toujours en tant qu'utilisateur root, lancez la commande :

chown postgres /usr/local/pgsql

Remarque : remplacez /usr/local/pgsql par le répertoire d'installation spécifié avec l'option −−prefix si vous en avez spécifiez un.

Installation à partir des RPMS

L'installation est plutôt simple :

```
# urpmi postgresql−server
Pour satisfaire les dépendances, les paquetages suivants vont être installés (9 Mo):
postgresql−7.4.1−2mdk.i586
postgresql−server−7.4.1−2mdk.i586
Est−ce correct ? (O/n)

Préparation...              ##################################################
   1:postgresql             ##################################################
   2:postgresql−server      ##################################################
```

Configuration du système

Passez en utilisateur **root** pour ces commandes.

On va ajouter dans la variable d'environnement PATH les binaires de PostgreSQL, les pages man, les données et ajouter au système les bibliothèques PostgreSQL pour activer le démarrage de PostgreSQL en automatique. Pour cela, éditez le fichier **/etc/profile** et ajoutez les lignes suivantes :

PATH=$PATH:/usr/local/pgsql/bin
MANPATH=$MANPATH:/usr/local/pgsql/man
PGLIB=/usr/local/pgsql/lib
PGDATA=/usr/local/pgsql/data
export MANPATH PGLIB PGDATA

Remarque : là aussi, remplacez /usr/local/pgsql par le répertoire d'installation spécifié avec l'option −−prefix si vous en avez spécifiez un.

Editez le fichier **/etc/ld.so.conf** et ajoutez la ligne :

/usr/local/pgsql/lib

Pour que cette modification soit prise en compte, lancez alors la commande :

ldconfig

Puis, copiez le fichier **/usr/src/pgsql/postgresql−7.4.2/contrib/start−scripts/linux** dans le répertoire **/etc/init.d** sous le nom de **postgresql** :

cp /usr/src/pgsql/postgresql−7.4.2/contrib/start−scripts/linux /etc/init.d/postgresql

Il vous sera possible de lancer le serveur avec ce script.

Mais avant de lancer PostgreSQL, nous devons créer le répertoire des données grâce à la commande initdb. Pour cela, connectez−vous en tant qu'utilisateur postgres et lancer cette commande :

/usr/local/pgsql/bin/initdb −D /usr/local/pgsql/data

Plusieurs messages s'afficheront pour vous indiquer la progression.

Voila, notre système est complètement configuré. Il ne nous reste plus qu'à lancer le serveur en tant qu'utilisateur root :

/etc/init.d/postgresql start

Vérifiez que postmaster est bien en cours d'exécution par un :

ps −aux | grep postmaster

Si quelque chose ne va pas, relisez bien les indications ci−dessus, ou envoyez moi un courrier électronique.

Contributions à PostgreSQL

Dans le répertoire des sources se trouve un répertoire nommé **contrib**. Il contient un ensemble d'outils qui ne sont pas compilés par défaut mais dont l'utilité est particulièrement intéressante... au moins pour certains d'entre eux que voici :

- dbsize ajoute une procédure stockée permettant le calcul de la taille d'un objet, que ce soit une base de données ou une table ;
- mysql permet la conversion d'une base MySQL en une base PostgreSQL ;
- en indiquant à oid2name un OID, celui−ci vous indique le nom de l'objet ;
- oracle permet la conversion d'une base Oracle en base PostgreSQL ;
- pg_autovacuum est un démon analysant en permanence l'état des tables de vos bases de données... lorsqu'il s'aperçoit qu'une opération de VACUUM devrait être lancée sur une table, il s'en occupe automatiquement, cela évite d'avoir à automatiser ce processus sur une simple base de temps (tous les jours, toutes les trois heures, etc.) ;
- tsearch2 est un outil de recherche de mots contenus dans des champs d'une ou plusieurs tables.

Beaucoup d'autres outils existent dans ce répertoire, n'hésitez pas à les tester !

Profitons−en pour rappeller l'existence de deux sites de devéloppement d'outils spécifiques à PostgreSQL : l'ancien, appelé GBorg et son remplaçant, nommé PGFoundry.

Configuration de PostgreSQL

La configuration par défaut de PostgreSQL n'est pas optimale. Elle est simplement étudiée pour correspondre aux machines les moins puissantes. Des améliorations ont été apportées, notamment pendant la création du dépôt des bases de données. En effet, la commande initdb va essayer de lancer le serveur avec plusieurs valeurs pour voir lesquelles sont les plus intéressantes pour *votre* serveur. Ceci dit, beaucoup d'autres options de configuration sont à vérifier, tester, voire modifier.

Les développeurs de PostgreSQL vous encouragent vivement à modifier les valeurs par défaut et à tester vos modifications pour obtenir des performances optimales.

Ce fichier de configuration se trouve dans le répertoire des données et a pour nom **postgresql.conf**. Le chemin complet par défaut est donc **/usr/local/pgsql/data/postgresql.conf**. Il est bien commenté, ce qui permet de faire une première configuration assez rapidement. Voici quelques points à prendre en considération rapidement :

- tcpip_socket doit être vrai pour permettre une connexion TCP/IP ;
- max_connections détermine le nombre maximum de clients acceptés en même temps ;
- shared_buffers précise la mémoire partagée utilisée par PostgreSQL ;
- syslog est compris entre 0 et 2... avec 0, les traces sont envoyées sur la sortie standard... avec 1, elles arrivent sur la sortie standard et dans syslog alors qu'avec 2, elles ne vont que dans syslog.

Il existe beaucoup d'autres options. En dehors des listes de diffusion, un document très important dans le cadre de la modification de ce fichier se trouve sur le site d'Elein Mustain. Il s'agit tout simplement d'un immense tableau récapitulant toutes les options pour le fichier de configuration, leur équivalent en ligne de commande ou en requête SQL si cet équivalent existe et leur explications. Malheureusement, aucune traduction de ce document n'est encore disponible.

Informations supplémentaires

Vous pouvez aussi contacter l'une des nombreuses listes de diffusion dont la liste française.

Installation d'un serveur SAMBA

par Fred, mis à jour par Anne

Comme moi, vous avez une belle machine et vous aimeriez bien que celle−ci puisse partager ces ressources avec d'autres machines dans un réseau hétérogène. Mais, laisser cette tâche à Win$$$ vous déplait, ne le faites pas. Une nouvelle fois, les logiciels libres viennent à votre secours, ils sont plus souples et gratuits.

Les informations contenues dans cette page ne sont aucunement garanties. J'apprécierai grandement toutes les critiques constructives, en particulier, celles liées à mon orthographe déplorable, à ma syntaxe difficile à suivre et aux erreurs qui se sont certainement glissées subrepticement au sein de ce texte. Si vous voyez quelque chose à ajouter, je vous serais reconnaissant de bien vouloir me mailer les modifications que vous voulez apporter à ce document.

Ce document concerne l'installation d'un serveur SAMBA au sein d'un réseau TCP/IP hétérogène (des machines Linux/UN*X et Win$$$). Il décrit uniquement les paramètrages élémentaires ! Pour ce qui est de la sécurité : allez sur les newsgroup. Nous supposerons ici que vous n'êtes pas reliés à l'internet.

Pour installer un serveur SAMBA sous Linux vous aurez besoin de :

- Des ordinateurs relié entre eux au sein d'un réseau TCP/IP fonctionnel (ce document ne décrit pas cela : allez voir linuxenrezo), je supposerai que toutes les machines sont sur le même réseau (bien que je ne sache, faute de l'avoir expérimenté, si cela a une importance : c'est le cas chez moi).
- Une distribution contenant les packages SAMBA, ou, à défaut Linux + les packages SAMBA (je ne décrirai que la procédure pour la Mandrake, pour les autres distributions la procédure est similaire).

***Note** : l'ensemble de la procédure d'installation décrite se déroule avec les droits de root donc faites attention à ce que vous taperez. Certaines parties de la procédure ne nécessitent pas réellement ces droits, mais ce sera plus simple pour moi de décrire l'ensemble de la procédure avec ces droits−ci. Si le coeur vous en dit essayez vous−même de vous passer de ces droits, linux vous préviendra quand vous essayerez de les outrepasser..*

Avant propos : Pour que tout ceci puisse fonctionner, il faut que tout votre réseau (même − surtout − les machines Win$$$) soit configuré de manière à utiliser TCP/IP comme protocole réseau : SAMBA fonctionne en effet au dessus de TCP/IP. Pour ma part, j'ai désactivé tous les autres protocoles (sauf IPX/SPX pour pouvoir jouer à Warcraft...) des machines Win$$$ connectées à mon réseau (4 machines : 1 PII 350 configuré en serveur SAMBA − il peut fonctionner en dualboot Win/Linux−, 2 P75 sous Linux (éléments d'une étude de cluster), 1 P166 en dualboot Win/Linux, les ressources du serveur sont accessible au P166 sans aucun changement que le serveur soit sous Linux ou Win$$$ vu que j'ai donné les mêmes noms aux ressources sous Win$$$ et Linux). Normalement ce ne doit pas être la peine, mais si ce que j'écris ici ne fonctionne pas chez vous : faites comme moi avant de m'écrire ;−).

Récupération des packages SAMBA.

Si vous avez une distribution décente, tout le nécéssaire est fourni, mais pas nécessairement installé. Comment le vérifier ? Si votre distribution de Linux utilise RPM, c'est très simple :

```
[root@etoile−noire home]# rpm −qa | grep samba
samba−2.0.5a−3mdk
```

Si vous obtenez quelque chose de similaire, c'est que tout est déjà près, passez à la section suivante. Sinon il faut installer les packages depuis le CD de votre distribution (ou tout autre support contenant votre distribution) :

```
[root@etoile−noire home]# mount /mnt/cdrom
[root@etoile−noire home]# find /mnt/cdrom/ −name "samba*rpm"
/mnt/cdrom/Mandrake/RPMS/samba−2.0.5a−3mdk.i586.rpm
[root@etoile−noire home]# cd /mnt/cdrom/Mandrake/RPMS/
[root@etoile−noire RPMS]# rpm −i samba−2.0.5a−3mdk.i586.rpm
```

Normalement, si vous avez fait ce qui est écrit (et si je ne me suis pas trompé ;−) vous ne devez avoir aucun message d'erreur (peut−être quelques avertissements... mais rien d'autre !). Si `rpm` vous signale que des dépendances manquent : à vous de les résoudre (en installant les rpms qui manquent).

Remarque : Si vous avez perdu le CD de votre distribution ou si vous voulez installez la toute dernière version allez :

- sur le site SAMBA, pour les dernier sources
- sur le site RUFUS, pour les derniers RPMS

Lancement "test" de SAMBA.

Maintenant que SAMBA est installé, on va vérifier que tout va bien :

```
[root@etoile−noire home]# /etc/rc.d/init.d/smb start
Starting SMB services:
Starting NMB services:
```

On lance les deux démons nécessaire au fonctionnement de SAMBA. (Même si SAMBA est déja lancée, normalement, il n'y a pas d'erreur...)

```
[root@etoile−noire home]# /etc/rc.d/init.d/smb status
smbd (pid 970) is running...
```

```
nmbd (pid 972) is running...
```

On vérifie que SAMBA est bien en train de fonctionner. Vous devez voir apparaitre les deux dernières lignes. Si tout va bien (ce qui devrait être le cas) vous pouvez passer à la suite ; sinon vous pouvez tout recommencer.

A partir de maintenant SAMBA fonctionne sur votre machine.

Lancement définitif de SAMBA.

En utilisant init Sys V :

SAMBA est maintenant correctement installé, autant le lancer à chaque démarrage de Linux. Pour la Mandrake (et la RedHat aussi) qui utilise les fichiers `/etc/rc.d/init.d/*` pour lancer des programmes au démarrage de Linux, il suffit de créer les liens adéquats :

SAMBA **ne doit pas** fonctionner dans les "runlevels" 0, 1, 2 et 6 (en effet dans ces "runlevels" le réseau n'est pas activé) :

```
# ln −s /etc/rc.d/init.d/smb /etc/rc.d/rc0.d/K35smb
# ln −s /etc/rc.d/init.d/smb /etc/rc.d/rc1.d/K35smb
# ln −s /etc/rc.d/init.d/smb /etc/rc.d/rc2.d/K35smb
# ln −s /etc/rc.d/init.d/smb /etc/rc.d/rc6.d/K35smb
```

SAMBA **peut** (et devrait) fonctionner dans les "runlevels" 3, 4 et 5 :

```
# ln −s /etc/rc.d/init.d/smb /etc/rc.d/rc3.d/S91smb
# ln −s /etc/rc.d/init.d/smb /etc/rc.d/rc4.d/S91smb
# ln −s /etc/rc.d/init.d/smb /etc/rc.d/rc5.d/S91smb
```

A partir de maintenant, dès que vous démarrerez Linux, SAMBA sera démarré par init. C'est la méthode à préférer si votre serveur SAMBA doit etre toujours actif.

Pour ceux qui utilisent une distribution Mandrake (voire peut−être d'autres, c'est à tester chez vous), vous pouvez remplacer toutes ces commandes par :

```
# chkconfig −−level 0126 smb off
# chkconfig −−level 345 smb on
```

C'est un peu plus simple.

En utilisant inetd :

On peut aussi faire en sorte que ce soit le super démon inetd qui lance smbd et nmbd quand il faut, pour cela, editez le fichier /etc/service et vérifiez qu'il contient les lignes :

```
        netbios−ns 137/tcp     # NETBIOS Name Service
        netbios−ns 137/udp
        netbios−dgm 138/tcp    # NETBIOS Datagram Service
        netbios−dgm 138/udp
        netbios−ssn 139/tcp    # NETBIOS session service
        netbios−ssn 139/udp
```

ainsi que le fichier /etc/inetd.conf qui doit contenir :

```
        netbios−ssn stream tcp nowait root /usr/sbin/smbd smbd
        netbios−ns  dgram  udp wait   root /usr/sbin/nmbd nmbd
```

puis vous redémarrez le super démon inetd par :

```
        /etc/rc.d/init.d/inet restart
```

Et c'est tout.

Le problème de cette méthode, c'est que le serveur SAMBA à très peu de chance de devenir le MASTER BROWSER du réseau, si ce n'est pas nécessaire pour vous, cette méthode convient très bien. Si votre serveur SAMBA n'a pas besoin d'etre actif en permanence, préférez cette méthode.

En utilisant xinetd :

De plus en plus, les distributions remplacent `inetd` par `xinetd`. Il vous faut donc créer deux fichiers appelés `netbios−ssn` et `netbios−ns` dans le répertoire `/etc/xinetd.d`. Ci−dessous un exemple de configuration, à adapter à votre cas :

```
root@pingu# cat /etc/xinetd.d/netbios−ssn
service netbios−ssn
{
   socket_type = stream
   protocol = tcp
   wait = no
```

```
   user = root
   server = /usr/sbin/smbd
   disable = no
}

root@pingu# cat /etc/xinetd.d/netbios-ns
service netbios-ns
{
   socket_type = dgram
   protocol = udp
   wait = no
   user = root
   server = /usr/sbin/nmbd
   disable = no
}
```

Puis pour prendre en compte les modifications, relancer `xinetd`.

```
root@pingu# /etc/rc.d/init.d/xinetd restart
```

Gestion des utilisateurs de SAMBA.

On peut certainement mieux protéger son réseau, mais ce qui suit est simple et compréhensible. Si la sécurité est un problème sur votre réseau − par exemple s'il est relié à Internet − ce document n'est pas pour vous.
Nous allons créer un utilisateur de SAMBA et un seul ! Tous les utilisateurs de SAMBA seront connectés sur votre ordinateur sous ce nom. Ce qui ne veut pas dire qu'ils autont tous les mêmes droits. Seulement tout ce que pourra faire SAMBA sur votre serveur sera déterminé pas les droits en lecture/écriture de cet utilisateur virtuel. Ne perdez pas de vue que ceci n'a rien d'obligatoire. Si vous le souhaitez chaque utilisateur, ou seulement certains, peuvent se connecter à SAMBA avec leurs propres droits.

donc, on crée un groupe spécifique :

```
# groupadd smbusers
```

Puis, on peut crée cet utilisateur spécifique (appartenant au groupe `smbusers`):

```
# useradd -g smbusers -p "passwd" -s /bin/false smbuser
```

Comme le shell de cet utilisateur est `/bin/false` : personne ne pourra pas se connecter sur votre Linux sous ce nom (même en connaissant le mot de passe : je vous déconseille quand même d'utiliser *password* comme mot de passe ;−)).

Gérer la configuration de SAMBA avec SWAT

SAMBA est livré avec un outil de configuration assez puissant : SWAT. Son rôle est l'édition du fichier de configuration de SAMBA : `/etc/smb.conf` et le contrôle de SAMBA. Il fonctionne via un browser de page WEB (par exemple netscape).

Utilisation de SWAT avec inetd

Pour cela il faut autoriser les transactions de SWAT et donc, modifier le fichier `/etc/inetd.conf` de façon que celui−ci contienne la ligne :

```
        swat      stream  tcp     nowait.400      root /usr/sbin/swat swat
```

Il est judicieux de modifier cette ligne pour utiliser `tcpd` de façon à n'autoriser les accès au serveur SAMBA que depuis certains endroits (en modifiant les fichiers `/etc/hosts.allow` et `/etc/hosts.deny`), vous pouvez par exemple remplacer la ligne précédante par :

```
        swat      stream  tcp     nowait.400      root /usr/sbin/tcpd /usr/sbin/swat
```

Il faut aussi modifier `/etc/services` de façon que celui−ci contienne la ligne :

```
        swat    901/tcp
```

Maintenant que SWAT est correctement configuré, il faut redémarrer le super daemon `inetd` pour recencer ses services.

Note : SWAT est correctement configuré parce que l'installation via paquetage RPM copie un fichier /etc/smb.conf de base. Si ça ne marche pas c'est que celui−ci est corrompu, réinstallez le paquetage...

Pour rédemarrer inetd avec la Mandrake il faut taper :

```
[root@etoile-noire home]# /etc/rc.d/init.d/inet restart
```

Utilisation de SWAT avec xinetd

Tout comme mentionné plus haut, il est fréquent que votre distribution fournisse plutôt `xinetd`. Il vous faudra donc modifier `/etc/xinetd.d/swat` pour activer le service. 2 étapes à suivre :

1. modification de `/etc/xinetd.d/swat` :

```
root@pingu# cat /etc/xinetd.d/swat
```

```
service swat
{
   port = 901
   socket_type = stream
   wait = no
   only_from = 127.0.0.1
   user = root
   server = /usr/sbin/swat
   log_on_failure += USERID
   disable = yes
}
```

2. Redémarrer `xinetd` :

```
root@pingu# /etc/rc.d/init.d/xinetd restart   ou
root@pingu# service xinetd restart
```

Générer et modifier smb.conf avec SWAT

Maintenant on peut utiliser SWAT. Pour cela, lancer un navigateur (`Netscape` fonctionne, mais `Lynx` aussi, ainsi que `w3m`) et dans la zone adresse tapez (si vous utilisez Lynx enlevez le `<root@>` car lynx ne le reconnait pas !) :

```
http://root@localhost:901
```

Votre browser va vous demander un mot de passe (le login sera déja rempli), tapez votre mot de passe root habituel. SWAT est maintenant lancé. Ce que vous voyez est la page d'accueil de SWAT, elle s'appelle Home : c'est une page d'aide (en fait ce sont des renvois aux manpages relatives à SAMBA).
Note : Il peut etre astucieux de créer un utilisateur (genre : smbadmin) qui aura les droits du root concernant la configuration de SAMBA.

Nous voulons autoriser certains utiliateurs à se connecter à notre serveur SAMBA. C'est assez simple puisqu'il suffit de cliquer sur l'icône <`Password Management`>.

Seule, la partie `<Server Password Management>` nous intéresse. Supposons que vous souhaitiez que `machin` puisse se connecter sur votre serveur :

Zone de saisie :	Valeur :	Commentaires :
`User Name`	`machin`	n'importe quel nom valide sur votre système.
`New Password`	`*******`	Un mot de passe est impératif (pour l'instant).
`Re-type New Password`	`*******`	Le même mot de passe (évidemment)

Quand vous avez renseigné tous les champs de cette partie (ne touchez pas à ce qui concerne `<Client/Server Password Management>` qui ne nous concerne pas... encore), cliquez sur `<Add New User>`. Vérifiez que SAMBA accepte votre nouvel utilisateur. Si ce n'est pas le cas, cela peut−être pour trois raisons (au moins) :

- l'utilisateur que vous souhaitez ajouter n'existe pas sur votre système.
- vous n'avez pas tapez deux fois le même mot de passe.
- vous avez tapez un mot de passe invalide.

Corrigez cela et tout devrait rentrer dans l'ordre.

Et c'est tout. Recommencez pour tous les utilisateurs auxquels vous souhaiter autoriser les connexions SAMBA sur votre serveur.

A partir de maintenant les utilisateurs que vous avez ajouté seront connus de SAMBA et ils ont donc la possibilité (dès que nous les y autoriserons) de se connecter au serveur.

Ne quittez pas SWAT (pas encore...).

Configuration de SAMBA en tant que serveur de fichiers.

Configurations des partages de types HOMES :

Lancez SWAT. Cliquez sur <`GLOBALS`>. Puis remplisser les zones saisie en suivant les conseils de ce tableau :

Zone de saisie :	Valeur :	Commentaire :
`workgroup`	`WORKGROUP`	mettez ce que vous voulez. Sachez simplement que c'est plus agréable de mettre un nom connu de vos machines Win$$$. Par exemple vous pouvez laisser `WORKGROUP`
`serveur string`	`Serveur Libre Linux/Samba`	mettez ce que vous voulez. C'est un commentaire, un truc du style `Serveur Libre Linux/Samba` sera le bien venu.
`interfaces`	(rien)	il faut indiquer les adresses IP des interfaces qui serviront à établir la liaison entre SAMBA et les machines clientes ainsi que le masque réseau (par exemple, si vous avez deux cartes ethernet − adresses 192.168.1.1 et 192.168.2.1 connectées respectivement aux réseaux 192.168.1.0 et 192.168.2.0 − mettez

		192.168.1.1/24 192.168.2.1/24. Si vous n'avez qu'une seule carte ethernet, vous pouvez ne rien mettre, SAMBA la trouvera tout seul comme un grand)
security	SHARE	sélectionnez le mode SHARE : c'est le mode de fonctionnement de Win$$$ 9$
encrypt password	Yes	sélectionnez <Yes> (sauf cas très particuliers... Win$$$ 95 avant l'OSR2 et ses bogs multiple de sécurités etc...)
guest account	ftp	mettez ftp ou un nom d'utilisateur valide sur votre système : ce sera le compte sur lequel seront connectés les invités si vous les autorisez. Il faut donc mettre un compte qui possède très peu de droit comme ftp ou nobody
hosts allow	192.168.1.	mettez les adresses IP des machines ou des réseau (séparée par des espaces) que vous souhaitez autoriser à se connecter à votre serveur (essayez de faire cela avec Win$$$ 9$...) par exemple pour autoriser les machines du réseau 192.168.1.0 ttapez **192.168.1.** (le point final est important). Vous pouvez autoriser toutes les machines de ce réseau sauf 192.168.1.10 en tapant : 192.168.1. EXCEPT 192.168.1.10 **Note**: avec SAMBA **2.0.7** j'ai des problème à faire fonctionner cette option avec SWAT, je la rajoute à la main dans /etc/smb.conf. Je ne sais pas encore pourquoi, mais dès que cette ligne est activée, SWAT refuse de se lancer prétextant que l'hote n'est pas autorisé, ce qui n'est manifestement pas le cas.
hosts deny	ALL	mettez les adresses IP des machines auxquelles vous souhaitez interdire l'accès à votre serveur SAMBA par exemple : **ALL pour interdire tout le monde *sauf* les machines autorisées par <hosts allow>. (Ici aussi le mot clef EXCEPT fonctionne).** **Note**: cf note précédante.
character set	ISO8859-1	Si vous utilisez Win$$$ en France (avec des accents par exemple), mettez ISO8859-1. Note : Normalement, dans ce cas vous devez aussi fixer la valeur de client code page à 850 (la code page par défaut de Win$$$ en france) dans la Advanced view. Mais, chez moi ce n'est pas la peine, c'est la valeur par défaut.

Dans toutes les autres zones vous pouvez laissez les valeurs par défaut, sinon lisez l'aide disponible.

Cliquez alors sur <Commit Changes> pour activez ces changements.

Maintenant on va pouvoir activer des partages (au sens de Win$$$). Cliquez sur <SHARES>. Nous allons autoriser chacun des utilisateurs que nous avons enregistré, à se connecter à son répertoire home. Dans la boîte déroulante située entre les boutons <Choose Share> et <Delete Share>, sélectionnez <homes> et cliquez sur <Choose Share>.

Les seules zones que vous devez fixer sont :

Zones de saisie :	Valeur :	Commentaires :
guest ok	No	les invités ne sont pas autorisés dans les répertoires maison (home). Si vous souhaitez les autoriser ... mettez Yes, mais ce n'est pas très logique.
create mask	0750	la meilleur valeur est 0750 qui signifie que seul l'utilisateur aura le droit de lire et écrire et que seuls les utilisateurs appartenant au groupe pourront lire les fichiers qui seront créés dans ce répertoire par le biais de SAMBA.
available	Yes	sinon au quoi sert que fred il se décarcasse !

A partir de maintenant, on peut vérifier que SAMBA nous autorise à nous connecter à notre home ! (plus précisément, si l'on demande la ressource \\\\etoile-noire\\machin et que l'on n'est pas machin, SAMBA s'en fiche, **il fera comme si** !) Pour cela nous allons utiliser smbclient pour nous connecter à notre serveur. Supposons que l'utilisateur fred souhaite se connecter sur le compte home de machin (dont il connait le mot de passe SAMBA), il devra taper :

```
[fred@awing /home/fred]$ smbclient \\\\etoile-noire\\machin -U machin
Added interface ip=192.168.1.1 bcast=192.168.1.255 nmask=255.255.255.0
Password: ******* (rien ne s'affiche, mes mots passe sont perso!)
Domain=[WORKGROUP] OS=[Unix] Server=[Samba 2.0.5a]
smb: \> exit (pour quitter smbclient)
```

Hourra ! Ca marche !

Mais, il n'y a pas que les maisons dans la vie. On peut souhaiter partager autre chose que sa maison :

Configurations des autres types de partage :

La méthode est globalement la même, sauf qu'il faut créer le partage. On se place donc dans la page <Shares> et on remplit la zone suivant <Create Share> avec un joli nom bien significatif (par exemple partage_samba), puis on clique sur <Create Share>.

On obtient une page à remplir. Voici les valeurs que je suggère :

Zone de saisie :	Valeur :	Commentaire :
Comment	Le beau partage	tapez ce qu'il vous plaira

path	/c:	ce doit être le chemin complet d'un répertoire accessible par l'utilisateur smbuser et/ou le groupe smbusers
guest account	ftp	si vous souhaitez autotoriser les accès aux invités, vous devez spécifier ici un utilisateur ayant très peu de droit comme l'utilisateur ftp sur un système Linux de base.
valid users	machin bidule truc	entrez la liste (séparée par des espaces) des utilisateurs qui auront le droit de se connecter à cette ressource.
force user	smbuser	entrez le nom sous lequel nous nous connectons au ressources partagé par SAMBA (vous pouvez ne rien mettre, en ce cas l'utilisateur qui se connectera le sera avec tous ses droits).
force group	smbusers	idem précédemment.
read only	No	Mettez Yes si vous voulez empêcher l'accès en écriture. (Je vous conseille de créer deux noms par ressource : un premier interdisant les écritures, dont beaucoup d'utilisateurs auront les droits d'accès − ie: par valid users − , et un second autorisant les écritures mais à un minimum d'utilisateur − ayant des mots de passe !)
guest ok	No	Mettez Yes si vous souhaitez autoriser des invités à accéder à cette ressource.
browseable	Yes	Mettez No si vous souhaitez cacher cette ressource lors de l'affichage par le voisinage réseau de Win$$$
available	Yes	pour partager une ressource, celle−ci doit être disponible.

Puis validez, en cliquant sur <Commit Changes>. On peut alors tester notre partage :

- Si l'on est loggé en tant que fred qui n'a pas de droit sur cette ressource, on se connecte comme si l'on était truc qui lui a des droits sur cette ressource :

```
[fred@bwing /home/fred]$ smbclient \\\\etoile-noire\\partage_samba -U truc
Added interface ip=192.168.1.1 bcast=192.168.1.255 nmask=255.255.255.0
Password: ******* (en fait rien ne s'affiche)
Domain=[WORKGROUP] OS=[Unix] Server=[Samba 2.0.5a]
smb: \> exit (pour quitter smbclient)
```

- Si l'on est loggé en tant que machin qui a des droit sur cette ressource :

```
[machin@bwing /home/machin]$ smbclient \\\\etoile-noire\\partage_samba
Added interface ip=192.168.1.1 bcast=192.168.1.255 nmask=255.255.255.0
Password: ******* (en fait rien ne s'affiche)
Domain=[WORKGROUP] OS=[Unix] Server=[Samba 2.0.5a]
smb: \> exit (pour quitter smbclient)
```

Houra bis ! Ca remarche !

Pour tester plus avant votre serveur SAMBA il faut alllez voir les commandes de smbclient.

Configuration de SAMBA en tant que serveur d'impression.

Et maintenant, configurons notre serveur SAMBA pour pouvoir imprimer grace à lui. Toujour grâce à SWAT nous allons faire une configuration de base : nous allons permettre l'impression sur toutes les imprimantes reconnues par Linux. Pour cela lancez SWAT. Sélectionnez Printers. Dans la section <Choose Printer> sélectionnez <Printers> pour modifier les propriétés de toutes vous imprimantes en même temps (chez moi je n'ai qu'une imprimante, donc je n'ai pas pu vérifier ce qui ce passe quand plusieurs imprimantes sont connectées, mais je pense que tout fonctionne correctement, c'est du moins ce que prétend la documentation de SAMBA). Puis validez votre choix en cliquant sur Choose Printer. Vous devez obtenir une nouvelle page à remplir.

Zone de saisie :	**Valeur :**	**Commentaires :**
comment	Les belles imprimantes	sans commentaires ;−)
path	/tmp	vous devez mettre un répertoire qui devra être accessible à l'utilisateur connecté à cette ressource : par exemple vous pouvez mettre n'importe quel répertoire qui est accessible en écriture par tout le monde − chez moi c'est le cas de /tmp)
guest account	(rien)	ne mettez rien ou un utilisateur qui a le droit d'imprimer (sinon mettez guest ok à false).
guest ok	True	on va imprimer en tant qu'invité.
available	True	sauf si vous ne souhaitez pas imprimer !

Puis validez vos changement en cliquant sur Commit Changes. Pour voir si cela fonctionne tapez, depuis n'importe compte utilisateur :

```
[darkvador@etoile-noire home]$ smbclient -L etoile-noire
Added interface ip=192.168.1.1 bcast=192.168.1.255 nmask=255.255.255.0
Password: (Aucun mot de passe : validez par <Entrée> !)
Domain=[WORKGROUP] OS=[Unix] Server=[Samba 2.0.5a]

        Sharename      Type      Comment
        ---------      ----      -------
        zipdrive       Disk      Lecteur Zip
        samba-public   Disk      Repertoire public
        IPC$           IPC       IPC Service (Serveur Linux/Samba)
        deskjet        Printer   lp
```

```
Server               Comment
---------            -------
ETOILE-NOIRE         Serveur Linux/Samba
WINDAUBE98

Workgroup            Master
---------            -------

WORKGROUP            WINDAUBE98
```

Hourra ! Hourra ! Et dire que certains pensent encore que Linux est plus compliqué que Win$$$! ;-)
Le nom de l'imprimante que vous devez voir apparaitre est celui qui est dans `/etc/printcap` qui doit être configuré de façon que depuis Linux les commandes `lpr`, `lpq`, `lprm` fonctionnent correctement (vous pouvez par exemple utiliser l'utilitaire `printtool` pour ce faire). Il n'est pas important que Linux sache gérer votre imprimante, il suffit que Linux sache comment envoyer un fichier sur cette imprimante, même s'il ne connait rien à cette imprimante (si ça se trouve ce pourrait même être une imprimante Win$$$ Printing System, à vérifier... une bonne âme, heureux possesseur d'une telle imprimante ?).

Test de votre installation depuis Win$$$.

Maintenant que tout semble fonctionner, il faut passer à l'épreuve du feu : démarrons une machine Win$$$ (appelons-la WINDAUBE) sur notre réseau (prions pour que cette pollution ne déteriore pas les cables... ;-). Tout est plus simple si vous vous connectez sur WINDAUBE en mettant un nom d'utilisateur reconnu par votre serveur SAMBA, donc : faites cela (nous verrons plus tard comment nous affranchir de cette limitation).
Allez dans le voisinage réseau, puis dans le Workgroup correspondant à votre serveur SAMBA, puis sur votre serveur et, magie : toutes les ressources que vous avez partagée aparaissent ! (chez moi, avant que le serveur aparaisse, je suis parfois obligé d'ouvrir et fermer la fenêtre de voisinage réseau de *multiple* fois... et quand je déconnecte le serveur, parfois Win$$$ continue de l'afficher dans le voisinage réseau, c'est bien les *caches*, mais parfois ça ralentit... Je suis même parfois obliger de redémarrer WINDAUBE quand je modifie le fichier `/etc/smb/conf`...).
Note : Ce problème semble du à l'election d'un MASTER BROWSER pour le Workgroup en question, en fait il suffit d'attendre suffisamment longtemps, genre une demi-heure ;-).

Si vous ne voyez pas votre machine : c'est que WINDAUBE est mal configurée : vous devez l'avoir configurée pour qu'elle utilise TCP/IP comme protocole réseau ! Il faut par exemple que le masque de réseau soit le même que celui de votre serveur SAMBA (Ce n'est peut-être pas une obligation, mais je n'ai pas assez de PC pour faire le test!)

Voilà c'est tout ! Lisez des fichiers, essayez d'écrire là où vous avez le droit, installez l'imprimante comme d'habitude (en cliquant avec le bouton droit sur la ressource imprimante), imprimez des fichiers depuis votre traitement de texte favoris, des images depuis votre programme de dessin chéri : tout doit fonctionner !

Attention : Les ressources d'impression de SAMBA n'effectue aucune traduction, elle envoie directement à l'imprimante les données reçues du client. Si vous souhaitez que cela change, je vous conseille de rajouter une entrée dans le fichier `/etc/printcap` de façon cette nouvelle entrée fasse systèmatiquement une traduction (par exemple passe par ghostscript), et d'imprimer sur cette imprimante.

Quelques commandes utiles.

smbclient.

Le programme `smbclient` permet d'établir des connexions avec un serveur SAMBA. Pour savoir quelles sont les ressources disponible sur un serveur faites :

```
smbclient -L serveur-samba
```

En ce qui concerne le mot de passe qui va vous être demandé, vous pouvez mettre n'importe quoi : il ne sera pas vérifié ! Cette connexion semble avoir lieu en tant qu'invité.

Pour vous connecter à la ressource `ressource-samba` du serveur `serveur-samba` tapez :

```
smbclient \\\\serveur-samba\\resource-samba -U utilisateur
```

ou

```
smbclient "\\serveur-samba\resource-samba" -U utilisateur
```

Ne précisez pas d'utilisateur si vous lancez cette commande avec les droits nécessaires. Après vous être connecté à cette ressource, vous pouvez envoyer des fichier par `put`, en récupérer par `get`. Si la ressource est une imprimante, vous pouvez imprimer en utilisant `print`.

Attention : quand vous imprimer sur une ressource Win$$$ ou SAMBA, le fichier n'est pas interprété par le serveur ! Donc, si vous voulez imprimez un document, il faut le transformer pour le rendre compatible avec votre imprimante ! Avec Linux, c'est habituellement le travail de `ghostscript`. Supposons, par exemple, que vous souhaitiez imprimer depuis votre machine qui tourne sous Linux sur une imprimante qui est partagé par un serveur smb (SAMBA ou Win$$$), il faut que vous sachiez quel est le type de l'imprimante (marque, et numéro etc...), puis vous devez transformer les données que vous souhaitez imprimer pour cette machine, puis envoyer le résultat à la ressource partagée par smbclient. Comme ce n'est pas d'un grand intérêt (pour nous autres fans de Linux) d'imprimer sur un serveur NT (d'autant que je ne possède pas NT), je n'ai jamais fait l'expérience, mais la documentation de SAMBA prétend que ce n'est pas très compliqué (voir la HOWTO)

On quitte `smbclient` par `quit`.

Evidemment elle ne marche pas seulement avec les serveurs SAMBA, vous pouvez, avec cette commande rapatrier un fichier depuis une machine Win$$$ sans installer sur cette dernière un serveur ftp (ce qui est problèmatique si vous n'avez que le CD de Win$$$, puisque Win$$$ ne fourni pas en standard de serveur ftp, les rats...)

smbpasswd.

Le programme `smbpasswd` sert à modifier les mots de passe SAMBA en ligne de commande. Vu que l'on à SWAT, à quoi cela peut−il bien servir me direz−vous ? Effectivement cela peut paraitre sans objet, mais imaginons que nous souhaitions, en tant qu'utilisateur modifier notre mot de passe, et bien, point de SWAT ! Donc, il n'y a qu'une solution : `smbpasswd`. Cette commande fonctionne de la même façon que `passwd` (que, d'ailleurs, elle appelle...).

Une autre utilisation de cette commande : vous êtes `root`, et souhaitez qu'un utilisateur particulier ait un mot de passe vide : SWAT ne l'acceptera pas ! En tant que `root`, tapez :

```
smbpasswd −n utilisateur
```

Et modifiez le fichier `/etc/smb.conf` de façon que la section `[global]` contienne la ligne :

```
null passwords = Yes
```

Maintenant, `utilisateur` a un mot de passe vide !

A propos de mot de passe, si vous vous êtes connecté sous Win$$$ avec un nom d'utilisateur qui n'est pas reconnu par Linux, mais que vous connaissez un login/password du serveur SAMBA vous pouvez demander de vous connecter à la ressource `"\\serveur\ressource%utilisateur"`. (Attention à <"> et <">) S'il y a un mot de passe, il vous sera demandé, et dans tous les cas vous serez connecté à cette ressource comme si vous étiez `utilisateur`.

montage des ressources smb dans un système de fichier Linux

Il est possible d'utiliser une ressource smb (SAMBA, Win$$$) comme faisant parti du système de fichier Linux : il suffit de monter la ressource. Pour que cela fonctionne, il faut que le noyau de Linux ainsi que SAMBA aient été compilés avec le support du système de fichiers smbfs (c'est la cas du noyau et des packages SAMBA de la Mandrake). Sur le compte de root, il suffit d'entrer :

```
# mount \\\\etoile−noire\\ressource /mnt −o username=truc password=**
```

Puis vous pouvez utiliser votre ressource smb comme si c'était une partie normal de votre système de fichier. Evidemment, le système des permissions UN*X ne fonctionne pas sur ce type de montage (ne perdons pas de vue que, à la base, on utilise un protocole destiné à Win$$$), préférez leur un montage nfs qui permet de monter un vrai système de fichiers Linux.

testparm

Le programme testparm sert à vérifier la validité des entrées du fichier /etc/smb.conf. Lorsque vous éditez à la main ce fichier, n'hésitez pas à l'utiliser : il vous dira les erreurs que vous avez peut−être commises en l'éditant. La syntaxe d'appelle est simple :

```
testparm
```

S'il vous dit que tout est Ok, vous pouvez relancer SAMBA pour prendre vos modifications en compte.
Ce programme sert aussi à vérifier les deux entrées `hosts allow` et `hosts deny`, si vous voulez savoir si la connexion à une ressource particulière sera accepté ou pas par SAMBA depuis la machine `machine` il suffit de taper :

```
testparm /etc/smb.conf machine
```

Et vous saurez tout.

Note: Avec SAMBA **2.0.7** j'ai un problème, `tesparm` me répond que l'hote depuis lequel j'utilise SWAT est autorisé, mais SWAT refuse la connexion, alors je ne comprends pas ... si quelqu'un connait la réponse à ce mistère.

Autres programmes utiles en vrac :

- `findsmb` : permet de savoir quelles sont les machines accessible par `smbclient` (proposant des partages de ressources via `smb`)
- `smbstatus` : permet de connaître l'état du serveur à ce moment précis (état disponible dans la page `status` de SWAT).

Quelques adresses utiles :

Je vous les livre en vrac :

- Vous pouvez toujours vous procurer la dernière version de SAMBA sur le site principal : http://www.samba.org. Pour avoir d'autres informations (plus techniques) sur SAMBA voyez la Samba−HOWTO .
- Il existe plusieurs utilitaires pour configurer SAMBA. Pour les mots de passes allez voir du côté de gsmb. Pour configurer les ressources partagées par SAMBA, allez voir KSamba.
- En remplacement de SWAT, vous pouvez utiliser linuxconf. (si linuxconf est installé sur votre machine et si vous avez autorisé l'accès à linuxconf en réseau, cliquez là)
- Pour ce qui est de la configuration d'un réseau TCP/IP avec linux, allez voir linuxenrezo. Une aide succinte concernant SAMBA y est disponible.

- Le livre Using SAMBA. C'est un livre en Open Content (comme ça on peut être content)! (**Note** : le livre est inclu dans SAMBA à partir de la version 2.0.7)
- Le site personnel de G. Blanchet qui explique comment configurer d'un autre point de vue un serveur SAMBA.

Se connecter à un ordinateur utilisant le protocole smb.

Pour ce connecter à un ordinateur utilisant le protocole smb (un pc Linux/SAMBA ou un pc Win$$$), il existe la méthode brutale : `smbmount` ! Evidemment ça marche, mais ce n'est pas très convivial, de plus il est bien dommage que l'on ne fasse pas mieux ni plus simple avec Linux qu'avec Win$$$. Pour ce qui est du plus simple, c'est très discutable. Par contre, il existe un programme, `LinNeighborhood`, qui fait tout et même plus que l'explorateur de Win$$$ pour ce qui est des connexions via `smb`.

Récupération et installation du paquetage LinNeighborhood.

La version que j'utilise est la 0.4.4. On peut la trouver sur le site de Hans Schmid.
Récupérez la version que vous voulez. Je vais décrire la procédure d'installation depuis les sources. On récupère le paquetage : `LinNeighborhood-0.4.4.src.tar.gz`. Puis on le déconpresse et l'installe par :

```
tar xzf LinNeighborhood-0.4.4.src.tar.gz
cd LinNeighborhood-0.4.4.src/src
make
cp LinNeighborhood /quelque/part/dans/le/path
```

Et voilà c'est tout, ce n'est pas la mer à boire et la version que vous avez maintenant sur votre machine est compilé spécialement pour elle (ça lui fait une belle jambe...). Evidemment, vous pouvez préférer installer une version précompilée, en ce cas reportez vous à l'installation d'un paquetage pour votre distribution.

Configuration de LinNeighborhood.

La configuration de LinNeighborhood en elle-meme reste triviale, mais pour une utilisation confortable, il faut lui apporter quelques petits raffinements. Pour commencer, lancez LinNeighborhood par la commande :

```
LinNeighborhood
```

La configuration se fait en utilisant le bouton : `<Prefs>`. Le plus simple est de préciser `WORKGROUP` dans la zone : `Scan\Workgroup` (où WORKGROUP est le nom du group de travail de votre serveur SAMBA). Les autres options de la boite de dialogue s'expliquent toutes seules. Puis sauvez la configuration, cliquez sur l'onglet `<Programs>` et choisissez la version de SAMBA que vous utilisez. Sauvez la configuration. Cliquez sur l'onglet `<Post Mount>` , choisissez un gestionnaire de fichier prédéfini (en cliquant sur la flèche noire) ou bien définissez le votre. Sauvez une dernière fois la configuration. Et quittez la boite de dialogue de configuration.

A partir de là, vous pouvez, depuis la fenetre de `LinNeighborhood`, parcourir le réseau comme vous le feriez avec `Win$$$` (ou presque). Mais, vous allez etre confronté à un problème : pour accéder à une ressource réellement, `LinNeighborhood` vous envoie le message : `"smbmount not found"`, la raison est simple : smbmount ne peut fonctionner qu'avec les privilèges de root. Donc, pour que cela fonctionne depuis un compte utilisateur il faut rendre `SUID` les programmes : `smbmount`, `smbumount` et `smbmnt`.

```
chmod +s /usr/sbin/smbmount
chmod +s /usr/sbin/smbmnt
chmod +s /usr/sbin/smbumount
```

Mais cela ne suffit pas, le programme `smbmount` essaie de lancer le programme `smbmnt` mais celui-ci ne ce trouve pas dans le `PATH` d'un utilisateur standard (il est dans `/usr/sbin` !). Solution créer un petit script que vous metterez dans un répertoire de votre PATH :

```
#!/bin/sh
export PATH=$PATH:/usr/sbin
smbmount $*
```

que vous appelerez : `smbmount-user`. Et un autre :

```
#!/bin/sh
export PATH=$PATH:/usr/sbin
smbumount $*
```

que vous appelerez : `smbumount-user`. Puis vous cliquez à nouveau sur le bouton `<Prefs>` de LinNeighborhood et dans l'onglet `<Programs>`, vous modifiez les entrées concernant `smbmount` et `smbumount` de façon à ce qu'elles deviennent respectivement `smbmount-user` et `smbumount-user`. Et là tout fonctionnera comme sur des roulettes.

Utilisation de Konqueror

Konqueror est maintenant un bon moyen d'accéder à un serveur smb (et donc Windows), il suffit pour cela dans la zone URL :

```
smb://user@serveur/
ou
smb://user:password@serveur/
```

pour accéder au serveur "serveur" en tant que "user".

Conclusion :

Ainsi s'achève la configuration de notre serveur. Les possibilités de SAMBA sont beaucoup plus grandes que celles que j'ai survolées dans cette indroduction. D'après ce que j'ai lu SAMBA peut émuler l'ensemble des possibilités de Win$$$ NT et bien plus. Si vous souhaitez en savoir plus, la lecture des divers manpages ainsi que de l'ensemble de la documentation de SAMBA est un point de passage obligé. Sachez qu'on peut faire en sorte que SAMBA crée les utilisateurs Linux à la demande, que SAMBA utilise un serveur de mot de passe etc... Alors plutôt de que de se battre avec celui qu'il est chargé de remplacer : utilisez SAMBA (les nombreuses heures que vous aurez passez à comprendre comment il fonctionne vous auront permis de mieux maîtriser votre système en entier, alors que les quelques heures qui auraient été nécessaire à la configuration de NT ne vous auraient rien apris sur le fonctionnement opaque de celui−ci!).

Installation d'un serveur NFS

Guillaume Pierronnet

NFS, ou le partage des ressources selon Unix.

Introduction

NFS signifie Network File System. C'est, comme son nom l'indique, un système de fichiers en réseau qui permet de partager ses données principalement entre systèmes UNIX. À la différence de SAMBA, NFS gère les permissions sur les fichiers et on peut donc l'utiliser de manière totalement transparente dans son arborescence Linux.

ATTENTION !

Comme toute application réseau, NFS ouvre des trous dans la sécurité du système. Je t'invite donc à consulter les liens à la fin de cet article pour des précisions sur la sécurité.

Les softs

Les modules du noyau

Dans la configuration du noyau, on va dans la section "**File systems −−−> Network File Systems**"

- Pour le client:

```
NFS file system support et Provide NFSv3 client support

  CONFIG_NFS_FS=y ou m
  CONFIG_NFS_V3=y
```

- Pour le serveur:

```
NFS server support et Provide NFSv3 server support

  CONFIG_NFSD=y ou m
  CONFIG_NFSD_V3=y
```

À partir du noyau 2.2.18, les modules supportent entièrement la version 3 du protocole ainsi que différentes corrections de bug.
Il serait temps d'upgrader si tu ne l'as pas déjà fait! (profites−en pour passer au 2.4, ce sera réglé :)

Les packages

Les packages (sur ma Debian) sont :

- `nfs−common`
- `nfs−user−server` pour le serveur.

Tu peux toujours récupérer les sources qui se trouvent sur http://nfs.sourceforge.net/.

On va aussi installer le *wrapper* TCP pour un minimum de sécurité. Toujours sur ma Debian, le paquet s'appelle `tcpd`. Les sources se trouvent ici.

Le serveur

Les 3 fichiers de configuration principaux sont `/etc/exports`, `/etc/hosts.deny` et `/etc/hosts.allow`.

/etc/exports

Le fichier `/etc/exports` est très simple :

```
répertoire machine1(option11,option12) machine2(option21,option22)
```

par exemple :

```
/home 192.168.0.10(rw) 192.168.0.25(ro)
```

signifie que l'on autorisera la machine *192.168.0.10* à accéder à notre répertoire `/home` en lecture et écriture (`rw`) ainsi que la machine *192.168.0.25* mais uniquement en lecture (`ro`).

- `répertoire` :
  le répertoire du serveur à partager.

- machine :
  Une liste de machines séparée par des virgules et autorisées à monter ce répertoire (utilisez des adresses IP plutôt que des noms à cause des problèmes de "*dns spoofing*").
- options :
  - **ro** :
    C'est la valeur par défaut, lecture seule.
  - **rw** :
    La machine à un accès en lecture/écriture au répertoire.
  - **no_root_squash** :
    Les accès par l'utilisateur root sur le serveur se font sous l'identité root, au contraire de nobody (par défaut)
    À UTILISER AVEC PRÉCAUTION
  - **sync** : uniquement NFS v2
    Ne diffère pas les écritures physiques au volume, augmente la fiabilité en cas de mauvais démontage. La version 3 dispose d'un mécanisme de *commit−rollback* donc cette option n'est pas utile.

Un point important, pour un bon fonctionnement : tu dois avoir les mêmes numéros de groupes et d'utilisateurs sur les deux machines.
Des systèmes permettent de gérer çà, NIS (assez ancien) ou LDAP (plus récent). Avec peu d'utilisateurs, tu peux tout simplement éditer `/etc/group` et `/etc/passwd` pour synchroniser ces numéros.

Il n'est pas recommandé d'exporter un système DOS ou VFAT à cause de leurs absences de gestion multi−utilisateurs ; ils ne sont pas fait pour être partagés avec NFS.

/etc/hosts.deny

On va interdire toutes les machines qui ne sont pas autorisées explicitement dans le `/etc/hosts.allow`.
Un bon vieux "`ALL: ALL`" interdira l'accès à tous les services à partir de toutes les machines. On peut cependant être plus précis en écrivant :

```
portmap:ALL
lockd:ALL
mountd:ALL
rquotad:ALL
statd:ALL
```

/etc/hosts.allow

Dans le même esprit que pour le `/etc/hosts.deny`, ce fichier a l'architecture suivante :

```
  [service]: [IP de la machine client]
```

Donc pour autoriser 192.168.1.34 à se connecter à un partage NFS, on écrira :

```
  portmap:192.168.1.34
  lockd:192.168.1.34
  mountd:192.168.1.34
  rquotad:192.168.1.34
  statd:192.168.1.34
```

On va pouvoir lancer les services ; sur ma Debian, je lance :

```
# /etc/init.d/nfs−server start
```

La commande `rpcinfo −p` permet de vérifier que les services fonctionnent. Elle devrait produire un résultat dans cet esprit :

```
cacahuete:~# rpcinfo −p
 program vers proto   port
  100000    2   tcp    111  portmapper
  100000    2   udp    111  portmapper
  100024    1   udp    737  status
  100024    1   tcp    739  status
  100011    1   udp    851  rquotad
  100011    2   udp    851  rquotad
  100003    2   udp   2049  nfs
  100003    2   tcp   2049  nfs
  100005    1   udp    872  mountd
  100005    2   udp    872  mountd
  100005    1   tcp    875  mountd
  100005    2   tcp    875  mountd
cacahuete:~#
```

Pour recharger les services NFS (par exemple après une modification du fichier de config) :

```
cacahuete:~# killall −HUP nfsd
```

le serveur est prêt !

Le client

Pour utiliser NFS v3, il faut au minimum la version 2.10m du programme `mount`. Pour voir sa version, taper :

```
cacahuete:~# mount -V
mount: mount-2.11h
cacahuete:~#
```

On va maintenant pouvoir monter notre partage!

```
cacahuete:~# mount mon.serveur.nfs:/home /mnt/home
cacahuete:~#
```

En principe tout devrait bien se dérouler.

Pour monter ce partage définitivement à chaque démarrage de la machine, éditons notre `/etc/fstab`:

```
# device       mountpoint  fs-type options dump fsckorder
...
master.foo.com:/home  /mnt nfs     rw      0    0
...
```

Liens

- Cet article est principalement basé sur le NFS howto
- Les sources des logiciels NFS sur http://nfs.sourceforge.net/
- LDAP howto
- NIS howto
- Va te faire peur sur security focus

Introduction à proftpd

par DuF

Un exemple de configuration de proftpd

Introduction

Tout d'abord, si je rédige cet article, c'est tout simplement que j'ai moi−même eu des difficultés à me servir de ce logiciel et j'ai eu énormément de problèmes pour trouver une documentation précise, claire et en français pour ce logiciel. J'en écris donc une qui même si elle n'est pas aussi claire ou précise que voulue, a au moins le mérite d'être en français, soyez conscient que ce n'est pas si mal :). Je tiens aussi à préciser que dans cet article je vais parler de proftpd et de sa configuration seulement par rapport aux besoins que j'ai eu à un moment donné, qui étaient avant tout liés à une utilisation personnelle et non professionnelle, afin de remplacer `wu−ftpd` qui est par défaut avec la plupart des distributions, si ce n'est tout les distributions (c'est proftpd avec la Mandrake 8.2 que j'avais ;−). Donc pour moi, le besoin par rapport à un serveur FTP, c'était de pouvoir partager des ressources que j'ai sur mon ordinateur et qui sont sur une partition FAT32 (ce fut un besoin à un instant T) mais aussi de donner la possibilité à un utilisateur de pouvoir écrire dans un endroit bien précis, restreint en essayant de suivre quelques règles de sécurité de base (qui sont vraiment de base)...

Je reprends cette documentation aujourd'hui pour prendre en compte quelques changements, remplacements de certains directives, pour corriger des erreurs (si si il y en avait :)) et ajouter une partie concernant le mod_tls pour faire des transferts chiffrés (à ne pas confondre avec du transfert FTP over SSH). La version actuelle de proftpd servant de support à cette mise à jour, est la version 1.2.8.

Installation

Bon et bien là c'est vraiment simple, il faut une connexion internet (disons que c'est mieux pour récupérer la dernière version de proftpd sur le site www.proftpd.org) et sinon si jamais il vous manque quelque chose, lors de l'installation tout ce qui vous manque sera indiqué et donc vous pourrez allez récuperer tout ce dont vous avez besoin sur internet. Pour ma part il me manquait 2 choses, comme il indique à la fin précisément ce qui manque, avec même un lien internet de la ressource manquante, il faut juste quelques minutes pour répondre aux besoins de proftpd. Si je ne rentre pas plus dans le détail pour cette partie c'est tout simplement que les ressources nécessaires à proftpd dépendent totalement de votre configuration et de votre distribution, donc je pourrai en mentionner alors que vous n'en aurez pas besoin et en oublier alors que vous en aurez besoin... Donc pas la peine de vous envoyer sur de mauvaises pistes, lisez juste le message d'erreur lors de l'installation et il n'y aura pas de souci.

Je vous conseille de prendre la dernière version (actuellement la 1.2.8 est la dernière version stable), pour l'installer, c'est comme d'habitude :

tar zxvf proftpd−1.2.8.tar.gz

Ensuite on se place dans le répertoire nouvellement créé, on lit bien le fichier d'installation (INSTALL ou README) et on lance les commandes habituelles.
Il est possible de récuperer le package proftpd sous la forme de package, il n'est pas la peine d'expliquer cela.

Configuration

Tout d'abord dans la configuration de proftpd il n'y a qu'un seul fichier qui rentre en ligne de compte, c'est : `/etc/proftpd.conf` (le chemin peut être différent suivant votre distribution et/ou distribution).

C'est dans ce fichier que vous allez "tout" définir (enfin presque) concernant la configuration du serveur. Mais attention, cela concerne la configuration du serveur, ce n'est pas ici que vous allez définir les utilisateurs qui ont accès ou non au serveur (du moins en parti). Occupons nous tout d'abord des utilisateurs !

Les utilisateurs

Attention, tous les utilisateurs se connectant sur le serveur proftpd doivent exister réellement sur le système (avec un uid). Il est cependant de faire un alias d'un utilisateur n'existant vers un utilisateur existant, pour cela regarder l'exemple concernant le contexte de configuration anonyme. avec l'explication sur le `UserAlias`. Noter aussi que pour ma part j'ai fait le choix de créer les utilisateurs avec un "faux shell" plutôt que de faire des alias, mais j'explique cela juste après.

Avec proftpd on a le choix dans sa stratégie. J'ai choisi d'utiliser le fichier `/etc/ftpusers` pour définir tous les utilisateurs qui n'ont pas accès au service FTP. Tous les utilisateurs présent dans ce fichier ne pourront donc en aucun cas se connecter au service FTP.

Vous pouvez indiquer à proftpd d'utiliser le fichier /etc/ftpusers avec la directive :

UseFtpUsers

Mais par défaut il le fait, donc vous pouvez vous servir de `/etc/ftpusers` sans dire explicitement dans le fichier `proftpd.conf` que vous souhaitez vous en servir, sans utiliser la directive `UseFtpUsers` donc (j'espere que vous suivez, toute façon j'y reviendrai après).

Pour ma part j'ai laissé le fichier `/etc/ftpusers` comme il était en ajoutant tous les utilisateurs qui ont un shell sur ma machine, voici un exemple de fichier `/etc/ftpusers` :

Exemple de fichiers ftpusers

root
bin

daemon
adm
lp
sync
shutdown
halt
mail
news
uucp
operator
games
gopher
postgres
squid
gdm
htdig
dhcpd
...
...
...
nobody
anonymous
DuF
ftp

Vous remarquerez que l'utilisateur `anonymous` est indiqué, c'est tout simplement que je ne souhaite pas donner un accès anonyme, si vous n'avez pas utilité d'un accés anonyme, je vous conseille d'ajouter, vous aussi, `anonymous` dans le fichier `/etc/ftpusers` .

Sinon comme je l'ai dit plus haut, j'ai choisi que tous mes utilisateurs ayant un shell (ksh, bash...) n'aient pas accès à mon serveur FTP, je vais donc créer des utilisateurs spécifiquement pour l'utilisation du FTP. Ces utilisateurs que je vais créer n'auront pas de shell leur permettant de se connecter en telnet, etc... Pour cela, j'indique le shell suivant :

/bin/false

Pour que ce shell soit "valide", il faut indiquer dans le fichier `/etc/shells` la ligne `/bin/false` si cela n'a pas déjà été fait. Car proftpd par défaut n'accepte pas la connexion si le shell de l'utilisateur n'est pas "valide" donc indiqué dans `/etc/shells` (bien sûr on se sera assuré que le shell en question existe effectivement dans l'arborescence). Si le fichier `/etc/shells` n'existe pas, par contre il accordera la connexion (personnellement j'ai trouvé ça étonnant, mais il doit y avoir des raisons à cela, il faut donc en tenir compte).

De plus vous pouvez spécifiez lors dans la configuration de `proftpd.conf` si vous souhaitez que proftpd vérifie que les utilisateurs aient oui ou non un shell valide. Cela se fait par la directive : `RequireValidShell` (qui prend comme argument on ou off !)

Sinon concernant la création des utilisateurs, pour cela je vous renvoie aux tutoriaux déjà présents sur léa, si je devais faire une recommandation, c'est de créer un groupe unique qui servira à placer tous les utilisateurs avec le shell `/bin/false` dedans, cela peut s'avérer utile par la suite, et de toute façon il n'est pas utile de créer un groupe pour chaque nouvel utilisateur.

Pour la création pur et dur des utilisateurs, vous pouvez faire un adduser ou useradd, pour plus d'options c'est très simple, il faut faire man adduser :) ou sinon maintenant, il existe des outils graphiques qui le font aussi, donc en récapitulant, vous créez les "users" par exemple user1 et user2 et vous les mettez dans le groupe ftptest.

Voilà déjà pour la partie spécifiques aux utilisateurs (users), je vais en reparler dans la partie du fichier `proftpd.conf` .

Le fichier Proftpd.conf

Contexte de configuration Server Config

On entre dans le vif du sujet.

Tout d'abord le fichier `proftpd.conf` se divise en plusieurs parties, qui ne sont pas toutes nécessaires (vitales). Je vais essayer, pour garder une cohérence avec la documentation en ligne du site www.proftpd.org de reprendre les mêmes terminologies, on a donc plusieurs contextes de configuration :

server config, <Global>, <Anonymous>, <VirtualHost>, <Limit>, <Directory>, .ftpaccess

Donc pour résumé sur les contextes de configuration, on peut avoir un fichier proftpd.conf avec seulement le contexte : `server config.`

Si le contexte "server config" n'est pas entre <> c'est tout simplement qu'il est implicite, ce n'est pas utile de le mentionner. Notez aussi que les options qui sont à l'intérieur des contextes de configuration sont les directives !

Les directives les plus courantes

On commence par un fichier proftpd.conf ne contenant qu'une directive et le contexte de configuration obligatoire qu'est "server config".

#début du fichier profptd.conf

```
UseFtpUsers             off

#fin du fichier proftpd.conf d'exemple, pour info ce fichier en l'état n'est pas valide
```

On peut dire que tous les contextes de configuration sont forcément inclus dans le contexte "server config" et que l'on peut avoir un contexte comme <Directory> qui soit dans un "sous" contexte comme <Anonymous>. C'est un principe de configuration imbriquée.

Les premieres directives que l'on va avoir dans le fichier `proftpd.conf` vont concerner le mode de lancement du serveur et les infos le concernant.

```
ServerName          "Server FTP du DuF"
ServerType          standalone
DefaultServer       on
```

`ServerName` => je ne reviens pas dessus

`ServerType` => est important (indispensable) car il indique si le serveur est démarré par vos soins en faisant : `/etc/init.d/proftpd start` (ou `/etc/rc.d/init.d/proftpd start`). Ou alors si il a la valeur `inetd` il est démarré par le meta−daemon du même nom (qui peut selon les distributions être `xinetd` au lieu d'`inetd`, mais il faut quand même laisser `inetd` dans le fichier `proftpd.conf`, car `xinetd` sera interprété comme un mauvais paramètre).

`DefaultServer` => cela est utile si vous faites des virtualhost, si vous n'utilisez qu'un "server config" sans virtualhost alors ce n'est pas utile de l'indiquer. En fait cette directive vérifie qu'elle configuration du serveur sera prise en compte (soit server config ou un des virtual hosts, ou anonyme....) Si il n'y aucune configuration de prévue, "l'inconnu" aura le message suivant : "no server available to service your request" et déconnecté.

Ensuite nous passons à des options dont l'utilité est très simple à comprendre :

```
AllowStoreRestart       on
Port                    45000
Umask                   022
MaxInstances            30
```

`AllowStoreRestart` => permet d'autoriser les clients à reprendre les uploads vers vous, ce n'est pas cette directive qui leur permet de reprendre quand ils téléchargent depuis votre serveur. Donc cette option, AllowStoreRestart n'est utile que si vous autorisez au moins une personne à écrire chez vous.

`Port` => je passe, simple à comprendre sinon je me demandes ce que vous faites là :) c'est bien évidemment le numéro de port sur lequel le client se connecte.

`Umask` => c'est comme dans unix en général, 022 est une valeur qui est bien, donc laissez−là à 022, si vous voulez en savoir plus sur umask en général, consultez notre ami à tous : www.google.fr/linux

`MaxInstances` => comme c'est indiqué dans le fichier par défaut de `proftpd.conf`, cela sert à spécifier le nombre de processus fils maximum que va gérer (utiliser) proftpd , et comme indiqué, au delà d'une valeur de 30 vous être vulnérable à des attaques de type Ddos, donc laissez à 30, pour une utilisation, sincèrement c'est largement suffisant.

Ensuite viennent 2 paramètres très important, on va indiquer sous quel utilisateur le serveur FTP est lancé, il ne s'agit donc pas de mettre root :)

Le user `nobody` et le group `nogroup` sont les paramètres par défaut et à mon avis ils sont très bien, donc on n'y touche pas car il faut que l'utilisateur qui lance le serveur n'ai pas trop de privilèges.

Ensuite il est question de l'option `PersistentPasswd`, dont l'utilité ici est pas évidente à expliquer étant donner que je n'ai pas compris l'explication sur le site de proftpd, je l'ai laissé à `off`. En fait si je me trompes pas, en le mettant à on cela permet à proftpd de chercher lui même dans `/etc/passwd` la validité des mots de passe, mais c'est à vérifier.

Ensuite vient une option qui a mon avis est très importante pour les personnes qui comme moi veulent partager une petite connexion adsl, il faut limiter le nombre de tentatives de logins (l'adsl ça sature vite :p) :

```
MaxLoginAttempts    3
```

Ensuite nous avons une option pas du tout vitale mais sympa je trouve, cela concerne la personnalisation de votre serveur, tout du moins le message d'accueil :

```
AccessGrantMsg              "Bienvenue %u chez DuF...."
```

Vous remarquerez le %u, c'est un paramètre qui recupère le user qui se connecte et le remplace en lieu et place de %u. Cette option indique le message de bienvenue quand l'utilisateur a réussi à se connecter.

Pour ne pas donner d'info précise sur le serveur, je conseil de mettre à on l'option suivante :

```
DeferWelcome                on
```

Contexte de configuration <Limit> appliqué à notre cas

Maintenant on va indiquer un contexte de configuration Limit qui va s'appliquer au contexte de configuration server config (partie générale du serveur) plus concrètement, c'est à dire tous ceux qui vont se connecter et qui ne seront pas concernés par un virtualhost. En fait ce contexte de configuration est utile (très important) pour tous ceux qui souhaitent partager des données sur des partitions FAT32 par exemple tout en limitant les possibilités (écriture, création de repertoires...)

Pour cela j'ai utilisé le contexte de configuration Limit avec les directives suivantes :

```
<Limit MKD RNFR RNTO DELE RMD STOR CHMOD SITE_CHMOD SITE XCUP WRITE XRMD XPWD>
  DenyAll
</Limit>
```

Bon moi j'en ai mis beaucoup pour l'exemple, je vais juste en expliquer certaines :

- `MKD` : création de répertoire
- `RNFR` : (rename from) empêche de pouvoir renommer
- `RNTO` : (rename to) c'est la suite de RNFR en fait, donc si RNFR est interdit, ce n'est pas util de le mettre, mais bon
- `DELE` : suppression de fichiers
- `STOR` : écriture de fichiers depuis un client vers notre serveur proftpd
- `CHMOD` : changement de permission sur les fichiers (et répertoires)
- `RMD` : suppression de répertoire

Il est aussi possible d'utiliser des mots clefs comme `READ` et `WRITE` qui englobent plusieurs commandes, et vont limiter l'accès en lecture et en écriture. Pour le reste des options vous pouvez consultez les commandes de la section Limit sur le site de www.proftpd.org

Contexte de configuration Global

Là on arrive à une section relativement importante, le contexte de configuration <Global>

En effet ce contexte de configuration peut être utilisé à l'intérieur de la "server config" et du contexte de configuration <VirtualHost>

Tout ce qui va être défini dans <Global> va être appliquer à l'ensemble du contexte de configuration dans laquelle <Global> se trouve. Cela est donc très pratique lorsque l'on a défini des <VirtualHost> car nous n'aurons pas à redéfinir plusieurs fois les mêmes paramètres.

Le mieux est de passer directement à un exemple de contexte de configuration <Global> :

Exemple de contexte de configuration Global

```
<Global>
  DefaultRoot              ~
  AllowOverwrite             yes
  MaxClients               3
  MaxClientsPerHost          1
  UseFtpUsers              on
  AllowForeignAddress        on
  ServerIdent              on      "ProFTP DuF's Server Ready"
  AccessGrantMsg             "Bienvenue %u sur le serveur du DuF"
</Global>
```

Explications de ce qui est à l'intérieur de <Global>

`DefaultRoot` => Limite le user à son home directory, si son home directory est par exemple /home/user, il pourra se ballader dedans, mais ne pourra remonter plus haut, il ne pourra pas aller dans /home par exemple et quand il se connecte, le user voit comme path dans son client FTP le chemin /

`AllowOverwrite` => Cela permet de remplacer d'anciens fichiers par les nouveaux, option inutile si vous interdisez l'écriture. J'indique différentes possibilités pour l'option, mais c'est à vous d'être cohérent. De toutes façons, si vous interdisez l'écriture, cette option ne prendra pas le dessus, vous ne pourrez pas écraser les fichiers.

`MaxClients` => C'est pour dire le nombre de clients différents qui peuvent se connecter en même temps sur le serveur, si vous avez une connexion ADSL, pas la peine de mettre 50...

`MaxClientsPerHost` => Option que je trouve très utile, elle limite le nombre de clients pour la même personne, si vous utilisez l'option MaxClients, il faut forcément que MaxClientsPerHost soit strictement inférieur (ou <=) à MaxClients sinon cela ne sert à rien.

`UseFtpUsers` => C'est dire que l'on utilise ou non le fichier `/etc/ftpusers` pour savoir qui a le droit d'utiliser le service FTP. Par défaut proftpd utilise le fichier, donc l'option n'est pas utile si on la met à on, mais moi j'ai préféré la mettre, c'est un choix ;−)

`AllowForeignAdress` => Alors cette option sert à autoriser ou non le fait que quelqu'un envoie ou télécharge des fichiers sur notre serveur FTP depuis un autre ordinateur que le sien. Pour faire simple, on va dire que la personne A veut transférer des fichiers entre le serveur B et notre serveur C car A n'a pas de serveur mais il a accès à B. Sans cette option mise à on cela n'est pas possible que A puisse passer les commandes.

`ServerIdent` => Cette option permet d'indiquer quel sera le premier message affiché quand quelqu'un essaiera de se connecter sur notre serveur, et cela même si sa connexion échoue. Si vous mettez cette option à "off" le client verra le message suivant : "[hostname] FTP server ready.". Le hostname sera souvent localhost.localdomain si vous ne l'avez pas modifié. Moi je vous invite à mettre cette option à on et mettre la chaine de

caractere que vous souhaitez mais qui ne donne pas trop d'indication non plus sur votre serveur. Dans mon exemple j'ai mis un message explicite, mais c'est juste un exemple, un message comme "Server Ready" sera tout aussi bien.

`AccessGrantMsg` => C'est là que vous définissez le message d'accueil lorsque la connexion a réussie, donc si vous le mettez dans un contexte de configuration <Global> pas la peine de le mettre à un autre endroit (server config, virtual host....)

Voilà pour la partie Globale, avec déjà toutes ces infos, vous êtes en mesure de partager grâce à votre serveur FTP des données en ayant bien le contrôle de ce qui se passe. Et surtout vous pouvez donnez l'accès à des données qui sont sur des partitions FAT32 (mais aussi n'importe quel type de patition ext2, reiserfs etc...), partitions qui normalement vous empêche de définir une stratégie utilisateur, car si vous avez besoin d'écrire sur des partitions FAT32, et donc que vous les montez en lecture/écriture, vous seriez embêté lors de l'accès par FTP car tout le monde pourrait écrire, supprimer, créer, faire ce qu'il veut en somme sur ces partitions, ce que pas grand monde souhaite. Donc grâce au contexte de configuration <Limit> des commandes, vous empêchez que l'on puisse toucher à vos données (autrement qu'en lecture) ce qui est intérêssant pour ceux qui ont encore un multiboot.

Maintenant vous vous dites mais j'aimerai quand même qu'une personne puisse accéder en écriture chez moi, même sur une partition ext2, mais vous dites que maintenant ce n'est plus possible, car on ne peut plus passer les commandes comme MKD, STOR, DELE.... Et bien trompez vous, nous allons créer un VirtualHost, terme que certains doivent connaître car c'est le même principe pour le serveur Web Apache.

Contexte de configuration VirtualHost

Maintenant on sait comment marche le fichier, donc dans <VirtualHost> le principe va être le même que pour Global etc... On va définir des options à l'intérieur du contexte de configuration <VirtualHost>.

Premièrement il faut savoir que <VirtualHost> à la base est prévu pour un serveur par exemple qui pourra être accessible d'un côté par des personnes s'y connectant depuis internet et d'autres qui s'y connecteront depuis le réseau local et l'on souhaite que ceux qui s"y connectent depuis internet n'aient pas accès aux même données que ceux qui s'y connectent depuis le réseau local. Donc ces différents personnes vont se connecter sur le serveur en indiquant une adresse différente. Par exemple l'utilisateur A est chez lui sur internet, il veut se connecter, pour cela il indique l'adresse suivante : ftp.serveur_proftpd_internet.com

Une autre personne, B, va elle se connecter sur ce serveur depuis le lieu de travail, donc depuis le réseau local, elle va utiliser l'adresse interne qui sera l'adresse : ftp.serveur_proftpd_local.com

On a donc ftp.serveur_proftpd_local.com qui dirige vers l'adresse IP 192.168.10.10 et le ftp.serveur_proftpd_internet.com qui dirige vers l'adresse IP 159.159.159.159. Ces 2 adresses IP dirigeant vers la même machine. Il est possible de spécifier le même port (ou de ne pas le spécifier, le port sera celui défini dans le contexte de configuration `server config`, cela ne posant pas de problème car proftpd va regarder dans son <VirtualHost> et tout se fera par rapport à l'adresse qui aura été indiquée pour la connexion.

Il est aussi possible de faire plusieurs <VirtualHost> qui travaillent sur la même adresse IP mais qui ont un port différent ce qui peut être très pratique aussi. Mais attention, cela d'après le site www.proftpd.org est incompatible avec "ServerType inetd" c'est à dire lorsque l'on lance le serveur par le démont inetd, personnellement je n'ai pas testé donc je ne pourrai pas vous en dire plus, moi je l'ai juste testé en "ServerType standalone"

Exemple de <VirtualHost> :

```
# Serveur Virtuel pour écriture
<VirtualHost    ftp.serveur_proftpd.com>
ServerName                  "Mon serveur FTP virtuel"
Port                        46000
Maxclients                  3
MaxClientsPerHost               1
DefaultRoot                 ~
AccessGrantMsg                  "Bienvenue %u sur le serveur virtuel du DuF"
 <Limit LOGIN>
   AllowUser                ToTo
   DenyAll
 </Limit>
</VirtualHost>
```

Bon alors les options vous les connaissez toutes, par contre je vais expliquer le contexte de configuration <Limit LOGIN> :

Avec ce contexte de configuration on va indiquer quel(s) utilisateur(s) va(ont) pouvoir se connecter dans ce virtual Host, c'est à dire que ceux qui ont accès par exemple au service FTP en lecture par une connexion sur le port 45000 ne peuvent pas se connecter sur ce virtualhost, dans notre exemple, il n'y a que l'utilisateur `ToTo` qui puisse le faire.

Donc pour que `ToTo` se connecte il doit indiquer l'adresse IP ftp.serveur_proftpd.com et le port 46000, ce sont les deux conditions à remplir, sinon cela ne marchera pas. Il faut savoir donc qu'il y a un ordre entre `AllowUser` et `DenyAll` à ne pas négliger. Mais il est possible de l'inverser.

En fait ProFTPD va examiner les autorisations explicites, puis les interdictions. Si une connexion ne correspond à aucun des critères, elle est **autorisée**. Il est possible d'inverser cela en utilisant la commande Order deny,allow. Si elle est présente, le serveur va d'abord prendre en compte les commandes Deny, suivi des commandes Allow, et interdire toute autre connexion ce qui peut être pratique, car dans ce cas là, si jamais votre règle n'est pas infaillible ou que vous avez oubliez une possibilité, et bien vous êtes sûrs que de toute façon la connexion sera refusée.

A noter aussi qu'il existe des options comme `AllowGroup` etc.... je vous invite à visiter le site de proftpd pour plus de renseignements.

Sinon juste pour l'exemple, un autre virtualhost, dans le cas ou le serveur est en type `standalone` et qu'on a donc sur la même IP plusieurs virtuelhost

mais sur des ports différents.

```
<VirtualHost ftp.serveur_proftpd.com>
  ServerName          "Mon serveur FTP virtuel"
  Port                47000
  MaxClients          3
  MaxClientsPerHost   1
  DefaultRoot         ~
  <Limit LOGIN>
    AllowUser         Foo
    DenyAll
  </Limit>
</VirtualHost>
```

Les autres contextes de configuration

Contextes de configuration : `<Anonymous>` et `<Directory>`

Le contexte de configuration `<Anonymous>` comme son nom l'indique sert à configurer un accès anonyme au service FTP et le contexte de configuration `<Directory>`, permet de définir un contexte pour les répertoires, il est possible de les utiliser comme suit :

```
<Anonymous /home/ftp>
MaxClients        5    "Nombre de clients maximum atteints : 5"
User              ftp
Group             ftp
  <Limit WRITE>
    DenyAll
  </Limit>
 <Directory uploads/*
  <Limit READ>
    DenyAll
  </Limit>
  <Limit STOR>
    AllowAll
  </Limit>
 </Directory>
</Anonymous>
```

Suite à une remarque de Drinou, voici une précision sur le contexte de configuration `<Anonymous>` et surtout sur la directive `UserAlias`, si jamais vous voulez que la connexion anonymous se fasse avec un user qui n'existe pas sur votre système, vous devez utiliser les UserAlias, vous pouvez vérifier votre configuration avec l'exemple suivant :

```
<Anonymous /home/e−smith/files/primary/html/download>
Group public
User public
UserAlias anonymous public
UserAlias ftp public
AnonRequirePassword off
MaxClients 10
  <Limit WRITE>
    DenyAll
  </Limit>
  <Directory upload/*>
    <Limit READ>
      DenyAll
    </Limit>
    <Limit STOR>
      Allow All
    </Limit>
  </Directory>
</Anonymous>
```

Donc dans cette exemple nous voyons que les users anonymous et ftp n'existant pas sous notre OS sont "aliasés" avec le user public qui lui existe réellement. Cela permet donc aux clients de se connecter avec le compte anonymous sans que celui−ci n'existe réellement sur le système.

Contexte de configuration : `.ftpaccess`

Pour ce dernier je n'aurai pas d'info à vous donner n'ayant pas trouvé son utilité et la manière de s'en servir (surtout que je n'en ai pas eu besoin...)

Complément sur la configuration de proftpd

Filtrage par Adresse IP

Il faut savoir qu'il est possible de filtrer pas adresse IP, cela est pratique dans un réseau local à IP fixe ou lorsque le client a une IP fixe, mais je ne saurai que trop vous déconseillez de mettre une filtre sur un nom de domaine, ou un redirecteur pour des raisons évidentes de sécurité même si cela peut paraitre une solution de facilité.

Voici un exemple de filtrage par adresse IP sur une IP (172.16.18.5) et une classe d'adresse IP (192.168.10.x) :

```
<Limit LOGIN>
  Allow 172.16.18.5 192.168.10.
  Deny all
</Limit>
```

Gestion de la Bande Passante

Depuis la version 1.2.8 de proftpd, la gestion de la Bande Passante n'est plus la même. Auparavant on utilisant des directives comme `RateReadBPS` et `ReadWriteBPS` notamment (il y en avait d'autres), maintenant il existe en fait une seule directive (`TransferRate`) qui sert à la fois à définir l'upload et le download par exemple.

Voici un exemple de gestion de la Bande Passante avant la version 1.2.8 :

```
(.....)
MaxClientsPerHost 1
RateReadBPS 12000
RateWriteBPS 63000
(.....)
```

A noter que la valeur était défini en octets, mais maintenant cela a changé, depuis la version 1.2.8 c'est TransferRate qu'il faut utiliser. Plutôt que de parler longuement, voici un exemple comment l'utiliser :

```
(.....)
MaxClientsPerHost 1
TransferRate RETR 12
TransferRate APPE,STOR 63
(.....)
```

Pour essayer de faire clair, en fait les 2 exemples font la même chose, le premier dans le cas des versions strictement inférieures à proftpd−1.2.8 et dans le second exemple, c'est pour les versions supérieures ou égales à la version proftpd−1.2.8. Donc RETR signifie Retrieve, ce qui correspond au fait de "récupérer" un fichier depuis le serveur, donc c'est le cas lorsqu'un utilisateur download. Pour APPE et STOR cela correspond à append et store, ce qui correspond au fait de "résumer" et "enregistrer" un fichier sur le serveur. Vous remarquez aussi que maintenant la valeur est en KiloOctets, et sachez que cette directive est valable dans tous les contextes de configuration.

Il faut noter que TransferRate ajoute d'autres options très intéressantes, comme le fait de d'allouer un seuil d'octets transférés avant que le contrôle du taux de transfert soit appliqué. Cela permet pour les clients transfèrant de petit fichier de ne pas être touché, mais ceux qui transfèrent de gros fichiers d'être limités pour donner la priorité à ceux qui transfèrent les petits fichiers. En gros ceux qui vont transfèrer des fichiers textes ne seront pas contrôlés à l'inverse de ceux transfèrant des fichiers de type iso par exemple. N'ayant pas testé je ne peux vous dire concrêtement si le transfert est stoppé ou seulement limité. Il est aussi possible de créer des groupes d'utilisateurs et de définir des limites de transferts pour ces groupes seulement. Cela permet de limite la BP pour certains mais pas par exemple l'administrateur, ce qui évite de faire plusieurs contextes de configuration.

Chiffrement des transferts

Cette partie est totalement optionnelle et non nécessaire pour la plupart d'entre nous. Donc ici il va être question de chiffrer les communications (commandes, transferts...) lors de la session FTP, mais il ne s'agit pas réellement de crypter de bout en bout la connexion avec SSL, car dans l'exemple suivant il n'y aura pas d'échanges de certificats (clés) entre le client et le serveur, mais seulement authentifier, lister et transférer le tout de manière chiffré.

Première étape, créer les certificats, pour cela je vous renvoi vers d'autres articles que vous pourrez trouver en cherchant avec google ou alors directement en lisant le How−TO SSL Certificates. Une fois que vous avez créer votre certificat (le fichier .pem) nous pouvons configurer proftpd.

Tout d'abord il faut indiquer dans le contexte de configuration "Server Config" le type du protocole TLS ainsi que le chemin vers le certificat et aussi ajouter la section concernant le module `mod_tls`. Voici en exemple le type de configuration que vous devez avoir :

```
(...)
TLSProtocol                SSLv23
TLSRSACertificateFile      /home/proftpd/certs/cert−rsa−duf.pem
TLSRSACertificateKeyFile   /home/proftpd/certs/cert−rsa−duf.pem
TLSCACertificateFile       /home/proftpd/certs/cert−rsa−duf.pem
(...)
<IfModule mod_tls.c>
 TLSEngine             on
 TLSLog                /var/log/proftpd−tls.log

 # Are clients required to use FTP over TLS when talking to this server?
 TLSRequired           on
</IfModule>
```

Là nous avons donc spécifier les différents chemins nécessaires vers le certificat, le protocole utilisé, l'activation du moteur TLS, le chemin vers le log spécifique au mod_tls et à quel type de requête le client doit répondre.

- `TLSProtocol` : Cela sert à définir quelle version de protocole mod−tls doit utiliser. A noter que cette directive ne peut être utiliser que dans le contexte de configuration 'Server Config'. Les choix possibles ici sont TLSv1 (autorise seulement TLSv1), SSLv3 (autorise seulement SSLv3) et SSLv23 qui autorise les deux (SSLv3 et TLSv1).
- `TLSEngine` : Cela permet d'activé ou non le `mod_tls` avec "on" pour "actif" et "off" pour "inactif" naturellement. Par défaut il est désactivé à la fois pour le serveur principal et les virtualhosts.
- `TLSLog` : Chemin vers le fichier log spécifique au mod−tls.
- `TLSRequired` : Cela sert à définir une politique sécuritaire basique, si il est à "ctrl" alors SSL/TLS est requis sur le canal de contrôle, si il est à "data" alors SSL/TLS est requis sur le canal de données (transfert des données), si il est à "on" alors SSL/TLS est requis sur les 2 canaux (données et contrôle) et si il est à "off" alors SSL/TLS n'est requis sur aucun des 2 canaux.

Voilà pour la partie configuration par défaut dans le contexte `Server Config`, maintenant si vous avez en plus des virtualhosts et que vous voulez faire des configurations spécifiques, par admettons que le `mod_tls` soit désactivé pour l'ensemble du serveur, mais que vous voulez juste l'activer pour un virtualhost précis, vous aurez alors une configuration proche de celle qui suit :

```
(...)
<VirtualHost ftp.duf_tls.com>
 ServerName            "FTP duf"
 Port                  6240
 MaxClients            2
 MaxClientsPerHost     1
 TransferRate          APPE,STOR 63
 AllowForeignAddress   on
 AllowStoreRestart     on
 DefaultRoot           ~

 TLSRSACertificateFile    /home/proftpd/certs/cert−rsa−duf.pem

 <IfModule mod_tls.c>
 TLSEngine             off
 TLSLog                /var/log/proftpd−tls.log

 </IfModule>

 <Limit LOGIN>
  AllowUser            user_tls
  DenyAll
 </Limit>
 <LIMIT MKD RNFR RNTO DELE RMD STOR WRITE SITE XCUP>
  AllowAll
 </Limit>
</VirtualHost>
```

Vous remarquez donc que la valeur de TLSProtocol n'est pas indiqué dans le contexte "virtualhost" car comme j'ai dit sa place est exclusivement dans le contexte "Server Config". Sinon comme vous pouvez le constater, afin de chiffrer une communication FTP, cela ne demande pas une grosse configuration dans le fichier proftpd.conf.

Attention, cette partie est très très succinte sur le sujet, et à moins que vous ne soyez pas déjà familier avec la création de certificats, l'utilisation de mod_tls avec apache par exemple, cette partie sera très grandement insuffisante. Donc ne vous aventurez dans cette configuration que si vous vous y connaissez déjà un minimum et documentez−vous au maximum avec les liens fournis tout en bas de cette page. De même un lien est fourni pour obtenir une liste de client permettant de se connecter sur un serveur avec le `mod_tls` activé.

Exemple de fichier proftpd.conf

Pour finir sur le fichier `proftpd.conf` voici en grande partie le mien modifié :).

Je rappelles que moi je cherchai à la base (il y a longtemps maintenant :)) pouvoir donner accès par service FTP à des ressources sur partition de type FAT32 et cela à la fois en lecture pour certains et en écriture pour d'autres et que je me suis vu confronté au problème qu'il n'est pas possible de définir de droits pour les utilisateurs sur les partitions FAT32.

Exemple de fichier `proftpd.conf` :

This is a basic ProFTPD configuration file (rename it to # 'proftpd.conf' for actual use. It establishes a single server # It assumes that you have a user/group # "nobody" for normal operation.

```
ServerName          "ProFTPD Linux DuF Service"
ServerType          standalone
DefaultServer       on

# Pour autoriser les clients à résumer les téléchargements, très utile. # Remember to set to off if you have an incoming ftp for upload.
AllowStoreRestart     on
```

```
# Port 21 is the standard FTP port.
Port                 45000

# Umask 022 is a good standard umask to prevent new dirs and files # from being group and world writable.
Umask                022

# Limitation de la bande passante en lecture :
TransferRate         RETR 11

# To prevent DoS attacks, set the maximum number of child processes # to 30. If you need to allow more than 30 concurrent connexions # at once, simply increase this value. Note that this ONLY works # in standalone mode, in inetd mode you should use an inetd server # that allows you to limit maximum number of processes per service # (such as xinetd)
MaxInstances         30

# Set the user and group that the server normally runs at.
User                 nobody
Group                nogroup

# Nombre maximum de clients
#MaxClients          3

# Number of Max Clients per host
# MaxClientsPerHost    1

# Nombre maximums de tentatives de login
MaxLoginAttempts     3

# Message d'acceuil après une connexion réussie
AccessGrantMsg       "Bienvenue %u chez moi !"

# Pour ne pas donner d'info sur le serveur
DeferWelcome         on

#Regles pour limiter les commandes...
<Limit MKD RNFR RNTO DELE RMD STOR CHMOD SITE_CHMOD SITE XCUP WRITE XRMD PWD XPWD>
  DenyAll
</Limit>

<Global>
DefaultRoot          ~
AllowOverwrite       yes
MaxClients           3
MaxClientsPerHost    1
UseFtpUsers          on
AllowForeignAddress  on
ServerIdent          on   "ProFTP DuF's Server Ready"
AccessGrantMsg       "Bienvenue %u sur le serveur du DuF"
</Global>

# Serveur Virtuel pour écriture
<VirtualHost ftp.duf.com>
ServerName           "Mon serveur FTP virtuel numero 1"
Port                 46000
Maxclients           3
MaxClientsPerHost    1
DefaultRoot          ~
AccessGrantMsg       "Bienvenue %u sur le serveur virtuel du DuF"
  <Limit LOGIN>
    AllowUser        ToTo
    DenyAll
  </Limit>
</VirtualHost>

<VirtualHost ftp.duf.com>
ServerName           "Mon serveur FTP virtuel numero 2"
Port                 47000
MaxClients           3
MaxClientsPerHost    1
DefaultRoot          ~
  <Limit LOGIN>
    AllowUser        Foo
    DenyAll
  </Limit>
</VirtualHost>
```

Voilà pour ce qui est du fichier `proftpd.conf`.

Utilisation de proftpd

Alors là c'est plutot simple, deux cas de figure ! Soit vous avez indiqué dans le fichier `proftpd.conf` que le serveur démarre en `standalone` ou par `inetd`. Si c'est en inetd il faut relancer le démon inetd ou xinetd suivant votre distribution. Pour le redémarrer faites :

```
/etc/rc.d/init.d/xinetd restart ; ftp localhost
```

Si votre serveur est en standalone vous faites simplement :

```
/etc/init.d/proftpd start
```

Sachez que vous pouvez passer les commandes start, restart, stop, status... à /etc/init.d/proftpd.

Sinon pour ceux qui veulent utiliser proftpd avec xinetd, voilà une marche à suivre :

il faut copier le fichier ftp.d vers proftpd en faisant :

cp /etc/xinet.d/wu−ftpd /etc/xinet.d/proftpd

Ensuite il faut éditer le fichier /etc/xinet.d/proftpd, comme l'exemple suivant :

service ftp
{
disable = no
flags = REUSE
socket_type = stream
instances = 10
wait = no
protocol = tcp
user = root
server = /usr/local/sbin/in.proftpd
}

Cette partie là je ne l'ai pas testé, donc si vous avez un problème renseignez−vous sur l'utilisation de `xinetd`, personnellement je n'en sais pas plus.

Problèmes rencontrés

Lors de l'installation si vous rencontrez un problème de librairie, avant de vous jeter sur votre navigateur web et de les chercher, il se peut que cela vienne juste du fait de ne pas avoir `/usr/local/lib` dans votre chemin des librairies, ce qui fut mon cas. Dans ce cas ajoutez−le donc dans votre `/etc/ld.so.conf` et une fois ld.so.conf editez faites un `ldconfig −v`.

Si vous n'avez pas pris proftpd en package rpm/debian mais en source,et bien les chemins sont différents , le chemin d'installation est `/usr/local` et le fichier de configuration se retrouve dans `/usr/local/etc/proftpd.conf`.

"J'ai des erreurs quand je veux installé proftpd, il me manque plein de choses..." Comme je l'ai dit, lisez les messages, ils sont très explicites, allez récuperer ce qui vous manque, n'essayez pas de forcer quoi que ce soit, ce ne sera jamais bon pour votre système de faire ça, et vous risqueriez de mordre la queue en somme.

Vous êtes sûr d'avoir correctement créé vos utilistateurs mais ils ne peuvent pas se connecter, l'accès au service FTP leur est interdit : vérifiez qu'ils ne soient pas dans la liste du fichier `/etc/ftpusers`.

Vos utilisateurs ne sont pas dans le fichier `/etc/ftpusers` mais ils ne peuvent toujours pas se connecter, ils ont une erreur d'accès refusé au password, cela peut venir soit d'un mot de passe erroné, ou de leur répertoire home (leur path où ils sont censés se connecter et arriver) qui est soit invalide, ou même non existant sur votre système.

Conclusion

Je rappelle que cet article n'est pas la solution unique à l'utilisation de proftpd, surtout pas. Si vous voulez donner accès à des centaines d'utilisateurs, ayez une stratégie d'ensemble et utilisez les paramètres lié aux systèmes de fichiers, les permissions etc... Ce sera beaucoup plus simple à gérer et à mettre en place. Je le répète, cet article vous donne une idée des informations nécessaire à la configuration de proftpd et donne à mon avis toutes les informations nécessaires à la mise en place d'un serveur personnel pour le partage de ressources qui ne sont pas forcément toutes sur des partitions ext2 (linux). Voilà j'espère que cet article vous aura aidé !

Si jamais vous avez des précisions, des corrections à ajouter sur la partie concernant le `mod_tlsn`'hésitez pas car j'ai juste fait une présentation très très légère du sujet, très incomplète et qui mérite d'être améliorer, alors si vous êtes expert en la matière, n'hésitez pas à corriger, améliorer cette partie.

Ressources

http://www.proftpd.org (Vous y trouverez toutes les informations nécessaires et surtout toutes les directives y sont listées et expliquées)
http://www.webring−adsl.com

http://www.ze-linux.org/
http://frlinux.net/index.php

Ressources concernant le `mod-tls` :

http://www.castaglia.org/proftpd/modules/mod_tls.html
http://www.tldp.org/HOWTO/SSL-Certificates-HOWTO/index.html
http://www.castaglia.org/proftpd/doc/contrib/ProFTPD-mini-HOWTO-TLS.html Clients ftp ayant le support SSL/TLS

vsFTPD, serveur FTP

par Acid

vsFTPd : un serveur ftp ultra−sécurisé et simple

Il existe de nombreux serveurs ftp, dont les plus connus sont wu−ftp, proftp, mais qui sont souvent soit de véritables trous de sécurité, comme l'on en voit quasiment tous les jours sous wu−ftp, soit très gourmands en mémoire comme c'est le cas pour ProFTP.

Mais une solution existe, elle est rapide, sécurisée et fait la joie d'Alan cox, d'IBM et de bien d'autres...

Et il se paye même le luxe d'être plus sécurisé qu'ftpd−BSD le serveur ftp d'OpenBSD !

C'est vsFTPd (Very Secure File Transfert Protocol deamon)

En effet l'auteur, Chris Evans, est parti de la constatation suivante : il n'y a pas de programmeur parfait, et faire un applicatif répondant à toutes les normes de sécurité est difficile, surtout quand cela n'a pas été pensé à la création du projet.

Donc, Chris a pris en compte tous ces problèmes avant même de commencer à coder son application. Et il en est ressorti ceci :

- le respect de la gestion des droits et processus en respectant les règles d'Unix
- La mise en cage des éléments (cf. chroot) indispensable
- la lutte nécessaire contre les "buffer overflows"

Pour cette raison également, de nombreuses commandes utilisées par vsFTPd font partie de l'application et ne font pas appel au système (pour le "ls" par exemple). ceci a pour conséquence de sécuriser encore un peu plus le système et d'accélérer le traitement des informations.

Pour cette raison, sa conception est modulaire, chaque partie étant traitée de manière indépendante, réduisant ainsi le nombre d'erreurs de programmation possible, et son optimisation importante. Pour cette raison aussi, il ne fonctionne pas en mode autonome comme c'est le cas de ProFTPd, mais doit être appelé à partir d'inetd ou de xinetd.

Exemple de configuration xinetd :

```
service ftp
{
        socket_type             = stream
        wait                    = no
        user                    = root
        server                  = /usr/local/sbin/vsftpd
       server_args             =
       log_on_success          += DURATION USERID
       log_on_failure          += USERID
        nice                    = 10
        disable                 = no
}
```

Ceci permet en outre de pouvoir réguler la bande passante utilisée.

Par défaut vsFTPd la configurera en standard (interdit tout ou presque), c'est à vous et seulement avec quelques lignes de configuration à définir quelles seront les capacités et les accès autorisés ou interdits à l'utilisateur.

pour vous en convaincre voici mon fichier "standard" de configuration de vsFTPd :

anonymous_enable=NO	connexion en tant qu'anonyme interdite
local_enable=YES	connexion pour les utilisateurs locaux auorisés
write_enable=NO	écriture interdite
anon_upload_enable=NO	Upoad pour anonyme interdit
anon_mkdir_write_enable=NO	création de répertoire pour les anonymes interdit
anon_other_write_enable=NO	écriture de fichier pour anonyme interdit
chroot_local_user=YES	enferme l'utilisateur dans le répertoire ftp (chrooté)
per_source = 5	nombre de connexions maximales autorisées par une même adresse IP
no_access = 192.168.1.3	interdiction de se connecter au ftp à partir de cette adresse
guest_enable=YES	Rend possible la connexion d'utilisateurs virtuels
guest_username=virtual	souhaitable, vu que la connexion anonyme est refusée
pasv_min_port=30000	cette option permet de limiter l'accès des ports
pasv_max_port=30999	disponibles, utile derrière un firewall
xferlog_enable=YES	enregistrement des logs...

simple et court, n'est−ce pas ?

Mais les options sont bien plus nombreuses, à vous de les découvir.

vsFTPd mérite vraiment d'être reconnu à sa juste valeur, alors essayez−le, vous n'en reviendrez pas.

vsFTPd, le site...minimaliste

Téléchargement

PureFTPd

par Laurent DUBETTIER−GRENIER

Installer un serveur FTP basé sur PureFTPd

Introduction

Avec l'avènement du P2P (peer to peer) et de BitTorrent, les serveurs FTP semblent un peu moins à la mode. Néanmoins, ils rendent encore, et pour longtemps, de fiers services. Les principales distributions sont par exemples disponibles en téléchargement depuis des serveurs FTP.

Cet article explique comment installer un serveur FTP (File Transfer Protocol) sur un serveur Mandrake Linux 10 sur lequel sont installés uniquement les paquetages de base, sans interface graphiqe. Il sera nécessaire d'installer gcc, pour compiler les sources.

Pour tester si votre serveur FTP fonctionne correctement, il faudra un poste client relié à votre serveur FTP, équipé d'un logiciel client FTP comme FileZilla par exemple.

PureFTPd est basé sur le package original Troll−FTPd, et est distribué sous license BSD. Un module webmin existe et permet d'en assurer le controle à distance. PureFTPd est compatible IPv6, et supporte MySQL, PostgreSQL PAM et LDAP pour le stockage des paramètres utilisateurs, et SSL/TLS (échange de login, mot de passe et des commandes FTP cryptées, mais echange de données non cryptées, entre clients et serveur) ou SSH (encryption totale). Il est disponible pour de nombreuses distributions : OpenBSD, NetBSD, FreeBSD, Solaris, MacOSX, QNX, AIX, HPUX...

Nota : La sécurité de votre serveur surtout s'il est connecté à Internet n'est pas l'objet de cet article, elle n'est donc pas forcément assurée...

Installation

A partir des sources

Télécharger les sources de ce logiciel sur le site de PureFTPd (version 1.0.18 au moment de la publication de cet article).
Décompresser le paquetage :
```
tar zxvf pure−ftpd−1.0.18.tar.gz
```
Se placer dans le répertoire nouvellement créé :
```
cd pure−ftpd−1.0.18
```

Il faut maintenant compiler et installer le programme. De nombreuses options de configuration sont disponibles, permettant d'adapter finement le programme à vos souhaits. Pour les découvrir toutes :
```
./configure −−help
```
(./configure se lit : point slash configure)

Quelques−unes de ces options de compilation sont décrites ci−dessous :

−−with−language=french : installation de la langue française
−−with−cookie : permet, en installant le paquetage fortune−mod, d'obtenir un petit message lors de la connexion d'un client au serveur FTP
−−with−ftpwho : autorise la commande 'pure−ftpwho'
−−with−paranoidmsg : les messages d'erreurs sont minimaux, quelque soit l'erreur
−−with−puredb : autorise le support des utilisateurs virtuels
−−with−privsep : autorisation des privilèges de séparation (améliore la sécurité)
−−with−quotas : autorisation des quotas (limitation de l'espace disque alloué à chaque compte)
−−with−throttling : autorisation de l'ajustement de la bande passante
−−with−tls : pour compiler PureFTPd avec le support SSL/TLS
−−without−banner : pour ne pas diffuser la bannière initiale
...

Nous allons configurer notre serveur de la manière suivante :
```
./configure −−with−ftpwho −−with−puredb −−with−language=french −−with−privsep −−with−cookie
```

Attention, pour une meilleure sécurité, si votre serveur est connecté à Internet, les options −−with−paranoidmsg, with−−tls et −−without−banner sont conseillées !

puis
```
make
make install
```

Voilà, le serveur FTP est installé !

A partir des RPMS

L'opération est simple avec urpmi. Attention vous ne pourrez pas installer pure−ftpd si vous disposez déjà d'un serveur ftp, celui−ci sera alors désinstallé.

urpmi pure−ftpd
Préparation... ##
1:pure−ftpd ##

Vous disposez également d'un package complémentaire qui permet d'ajouter les éléments de l'arborescence nécessaires pour réaliser un ftp anonyme.

```
# urpmi pure-ftpd-anonymous
réparation... ##################################################
1:pure-ftpd-anonymous ##################################################
```

Configuration

PureFTPd dispose aussi d'options encore plus nombreuses au moment de son démarrage ! Leur liste est disponible de la manière suivante :
`pure-ftpd --help`

Voici une description de quelques options de démarrage (Attention, Pure-FTPd est sensible à la casse !)

-A : "chroot" tout les utilisateurs sauf root.
-b : ignore certains standards pour permettre à certains clients (Internet Explorer !!!) de fonctionner malgré leurs bogues...
-B : Ordonne au serveur de démarrer en arrière plan
-c 5 : limitation du nombre de client simultané sur votre serveur à 5 (défaut = 50)
-C 2 : limite le nombre de connexion simultanées de chaque utilisateur (provenant de la même adresse IP) à 2
-E : autorise seulement la connection d'utilisateurs enregistrés (pas d'anonymes)
-e : autorise seulement la connection d'utilisateurs anonymes
-j : création automatique d'un répertoire utilisateur si celui qui se connecte n'en a pas déjà un.
-l puredb:/etc/pureftpd.pdb : specifie le chemin vers la base de données des utilisatuers virtuels.
-R : interdit l'utilisation de la commande chmod par les clients
-u 1 : n'accepte pas les uids inférieur à 1 (root en l'occurence)
-X : interdit aux utilisateurs l'accès aux fichiers ou répertoires cachés (ceux qui commencent par un point)
-H : pas de résolution DNS (à utiliser si connexion très lente)

Pour notre exemple, nous allons utiliser la commande suivante :

```
/usr/local/sbin/pure-ftpdd -A -b -B -c 15 -C 2 -E -j -l puredb:/etc/pureftpd.pdb -R -u 1 -X -F
/usr/share/games/fortunes/fr/amusantes -H
```

Pour avoir un fonctionnement correct des petits messages lors de la connexion des clients, il faut auparavant avoir chargé le programme fortune-mod :
`urpmi fortune-mod`
Il y a un choix assez vaste de messages. Ils sont situés dans le répertoire :
`/usr/share/games/fortunes/`

S'il n'y a pas d'erreur, pour éviter de retaper à chaque démarrage du serveur cette longue ligne de commande, il suffit de l'insérer à la fin du fichier `/etc/rc.local`, en utilisant votre éditeur de texte favori (vi par exemple)

Utilisateurs virtuels

Avec PureFTPd, les utilisateurs enregistrés n'existent pas en tant qu'utilisateurs systèmes : ils n'ont pas à être créé avec la commande adduser.
PureFTPd gère en effet les utilisatuers virtuels : ce sont des utilisateurs créés et connus uniquement par PureFTPd.

La première chose à faire est de ***créer un groupe d'utilisateur*** dédié au serveur FTP, nommé ftpgroup, et un utilisateur FTP, nommé ftpuser, auquel seront rattachés tous les utilisateurs de votre serveur FTP, chacun ayant son propre répertoire de stockage.
```
groupadd ftpgroup
useradd -g ftpgroup -d /dev/null -s /etc ftpuser
```
Ceci crée un utilisateur ftpuser, appartenant au groupe ftpgroup, ne disposant d'aucun répertoire sur le système, et d'aucun shell.

Ensuite, l'essentiel de la ***maintenance*** est réalisé par l'instruction pure-pw, et comme d'habitude avec PureFTPd, de nombreuses options sont disponibles. Pour les visionner :
`pure-pw --help`

Créons un premier utilisateur virtuel (nommé ici 'lea'):
`pure-pw useradd lea -u ftpuser -d /home/ftpusers/lea`
Un mot de passe alloué à ce compte vous est alors demandé, ainsi que sa confirmation.
Ceci crée un utilisateur virtuel nommé lea, connu de PureFTPd seul, rattaché à l'utilisateur ftpuser, ayant pour répertoire personnel chrooté `/home/ftpusers/lea`.
Inutile de créer le répertoire utilisateur de lea, il sera créé automatiquement lors de sa première connexion (c'est le rôle du paramètre -j vu au paragraphe précédent).

Pour enregistrer définitivement cet utilisateur, il faut mettre à jour la base de données PureFTPd (fichier `/etc/pureftpd.pdb`) :
`pure-pw mkdb`
Nul besoin de redémarrer le service, la mise à jour est prise en compte immédiatement.

Quelques commandes utiles :

- Changer le mot de passe oublié d'un de vos utilisateur virtuel :
  ```
  pure-pw passwd nom_du_compte_utilisateur
  pure-pw mkdb
  ```
- Détruire un compte utilisateur virtuel :
  `pure-pw userdel nom_du_compte_utilisateur`

- Visionner les paramètres de configuration d'un compte utilisateur virtuel :
  `pure-pw show nom_du_compte_utilisateur`
- Lister l'ensemble des personnes connectées à votre serveur FTP :
  `pure-ftpwho`

Serveur anonyme

Ce serveur PureFTPd, démarré avec les options décrites ci-dessus, n'autorise pas un accès aux utilisateurs anonymes. Si vous souhaitez avoir un fonctionnement permettant à chaque utilisateur virtuel enregistré de se connecter à votre serveur, mais aussi posséder une zone en accès libre anonyme, la démarche à suivre est la suivante :

- Créer un utilisateur système "ftp", disposant d'un répertoire personnel, mais sans shell de connexion.
  `useradd -g ftpgroup -d /home/ftp -s /bin/false ftp`
- Créer le répertoire /home/ftp, qui sera le répertoire contenant les fichiers accessibles de manière anonyme :
  `mkdir /home/ftp`
- Modifier les droits sur ce répertoire :
  `chmod 500 ftp`
- Arrêter le serveur PureFTPd :
  `ps ax | grep pure-ftpd`
  Le premier nombre obtenu dans la réponse est le numéro de process PureFtpd. il faut le tuer :
  `kill numero_de_process`
- Relancer le serveur PureFTPd en supprimant l'option de démarrage -E :
  `/usr/local/sbin/pure-ftpdd -A -b -B -c 15 -C 2 -j -l puredb:/etc/pureftpd.pdb -R -u 1 -X -F /usr/share/games/fortunes/fr/amusantes -H`
  N'oubliez pas de modifier aussi ceci dans le fichier `/etc/rc.local`, en prévision du prochain redémarrage de votre serveur !

Vous avez alors un serveur acceptant les connexions avec utilisateurs virtuels enregistrés et avec utilisateurs anonymes.

Introduction à profxp

par DuF

Présentation rapide de profxp

Introduction

Pour ceux qui ne connaissent pas "FXP", je vais faire une présentation rapide. On peut dire que FXP est un protocole, même si en fait ce n'est qu'une application spécifique du protocole FTP faisant partie de la suite de protocoles TCP/IP. Grâce à FXP il est possible d'effectuer des transferts de fichiers de serveur FTP à serveur FTP depuis un poste tier. Cela peut être très utile pour transférer depuis chez soi des fichiers à un ami alors qu'ils sont sur un autre serveur. Ainsi il ne nous est pas obligatoire de les télécharger pour ensuite les renvoyer à notre ami. De plus notre bande passante pendant le transfert n'est à aucun moment réellement utilisé (juste quelques petites infos, mais c'est infime). Par contre il faut savoir que pendant le transfert il n'est pas possible de suivre l'état du transfert, on peut avoir des estimations, mais c'est tout, on sait juste lorsque cela commence, et lorsque cela fini. Il existe plusieurs logiciels, je ne vais parler que d'un seul, profxp, car en fait c'est le seul qui ait marché chez moi. Par contre il est en mode texte, mais une fois habitué, ça va très très bien.

Installation

Alors l'installation est très simple...
Il faut d'abord récupérer le tarball profxp−v3−pre1−linux−glibc.tar.gz (c'est la dernière version au moment où j'ecris ces lignes) sur le site : http://duncanthrax.net/profxp/.
Vous pouvez télécharger aussi les sources, mais cela n'est pas du tout utile pour faire marcher le programme, c'est en perl, ce n'est pas trop compliqué, mais pour pouvoir vous servir du programme perl il vous faudra récupérer des modules non inclus (lire le README fourni avec).

Pour installer et lancer le programme il suffit de faire un

```
tar zxvf profxp−v3−pre1−linux−glibc.tar.gz
cd profxpv3
./profxp
```

Il est possible que le nom du répertoire et/ou le nom de l'exécutable ne soit pas tout à fait le même, cela n'a aucune importance.

Utilisation

Bon et bien première chose à faire, c'est de modifier les droits du répertoire ~/.profxpv3 car il va contenir les fichiers "sites" que vous allez enregistrer. Par exemple votre copain Robert vous a ouvert un compte avec adresse IP, port, login et mot de passe et comme vous y allez tous les jours, vous vous dites qu'il serait bienvenue de pouvoir enregistrer les informations de connexion pour ne pas avoir à les retaper à chaque fois. Et bien cela est possible via une commande, mais les informations de connexion chez votre copain Robert vont être enregistrées en clair dans un simple fichier texte .site. Alors je vous conseil donc de faire un
`chmod −R 700 .profxpv3`

bon maintenant l'utilisation du programme en lui même. Pour le lancer il suffit donc de se mettre dans le répertoire où vous l'avez mis et de faire un ./profxp (ou ./profxpv3 suivant le nom qu'il a).
On arrive sur un bel écran texte comme le suivant :

```
::
:: −/−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−/−
::  / :: profxp v3.0 − the final rewrite ::                     /
::  /                                                           /
::  / (c) duncanthrax aka Tom Kistner < tm@duncanthrax.net >      /
:: −/−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−/−
::
```

Je vous rassure il y a de la couleur :−) mais je l'ai pas mise là !
L'utilisation est très simple et surtout lors de la première utilisation lisez bien tout ce qui est marqué !

Lors du premier lancement, vous avez un écran qui correspond au suivant :

::
::
:: −/− Welcome ! −/−
::
:: It looks like this is your first time starting profxp. I have some
:: very helpful hints for you:
::
:: 1. READ THE DOCS ! (this is a pretty hardcore program)
:: 2. start commands with CAPS to send them RAW, e.g. 'Site who'.
:: 3. profxp uses quoteless command lines, DO NOT USE QUOTES FOR PATHS
:: 4. get familiar with the alias function, it rocks.
::
:: Type 'he' for help. Please use 'cf' to see a list of configuration

:: flags, set your local interface IP ('cf localif<ipaddr>') if you
:: want to use active transfer modes.
::
[disconnected|no−path]
[disconnected|no−path]

Comme vous le remarquez vous avez deux fois la ligne : **[disconnected|no−path]**
En fait cela est très simple, pour pouvoir FXP, il faut être connecté à 2 serveurs FTP en même temps, lorsque vous allez vous connecter au premier serveur, la ligne du bas va changer pour devenir quelque chose du genre : [ftp.dufbusiness.com|/Shares]
Là vous me dites, mais comment je me connectes au deuxième site ? Toute l'astuce réside là, il suffit d'appuyer sur la touche entrée pour basculer sur l'autre prompt (si on peut dire). Donc en ne tapant aucune commande dans la zone prévue à cet effet et en appuyant seulement sur la touche entrée, on bascule du serveur 1 au serveur 2 et vice versa. Enfantin me direz vous...

Commandes de base

on

La toute première commandes, celle qui permet de se connecter à un serveur FTP :
`on <hostname/IP> [port] [user] [pass]`
Exemple : `on ftp.serveurftp.com 21 nom_user mot_de_passe`
pour se connecter sur un serveur ftp anonyme qui utilise le port 21 standard il suffit de faire :
`on ftp.serveur.com`
et vous vous connectez directement sans spécifier le port ou le mot de passe.

cl

Seconde commande, la commande `cl` : cette commande permet de fermer la connexion avec le serveur en cours. Pas la peine d'épiloguer dessus.

ss

La commande `ss` permet d'enregistrer une connexion sous un nom plus parlant, en gros vous tapez votre commande `on <hostname/IP> [port] [user] [pass]` et une fois connecté vous faites `ss <sitename> [slotname]` et cela va enregistrer votre connexion sous le nom que vous aurez donné dans `<sitename>`. Il est possible aussi de classer les sites que vous enregistrez en définissant un [slotname], cela vous permet par exemple de regrouper un ensemble de sites ayant un lien dans un même `[slotname]`.

sl

Si vous commencez à avoir beaucoup de sites d'enregistrés, la commande `sl` est là pour vous aider, elle affiche la liste de tous les sites que vous aurez enregistré.

op

La commande op sert à rendre la commande ss très utile, en effet, admettons que vous ayez enregistré votre site ftp sous le nom "bidule" en faisant un `ss bidule` auparavant, et bien maintenant lorsque vous lancez profxp, pour vous connectez à votre site bidule il suffit de faire :
`op bidule`
Il n'y a pas plus enfantin !

sd

La commande `sd` permet de supprimer un site de votre liste de sites, exemple :
`sd bidule`

Vous le remarquez l'utilisation des commandes est relativement simple.... Passons aux commandes liées à la navigation une fois connecté à un site.

ls

Et bien c'est en gros la même commande que sous votre shell favori sauf que si vous y regarder de plus près, le `ls` ici est en fait un `ls −la` ce qui permet de voir les fichiers cachés (commençant par un . notamment).

ld

Tout simplement la même commande que `ls` sauf que là les fichiers et répertoires sont rangés par ordre chronologique (d = date).

lr

Pour ma part, de ce que j'en ai constaté, la commande `lr` revient à faire un `ls −l` sur le site distant comme si vous étiez en local purement et simplement.

cd

Equivaut à un `cd`, pas la peine d'épiloguer non plus.

vf

La commande `vf` (view file) permet d'éditer très simplement tout type de fichier texte, cela évite de télécharger le fichier puis de le visualiser par commandes successives.

rm

La même commande qu'en shell, mais de manière recursive. C'est à dire que tout ce qui est dans l'entrée fournit à `rm` sera supprimé, les sous−répertoires, etc....

cf

Cette commande est un peu particulière, je ne vais pas rentrer dans les détails car je m'en suis très peu servi, par défaut profxp étant pour moi rêglé sur les bonnes options. En fait la commande `cf` permet de configurer des options, créer des alias, pouvant s'avérer très utile lorsque l'on utilise les mêmes commandes très souvent ou lorsqu'on a des problèmes pour se connecter sur un site FTP, ou pour transférer un fichier, car il va être possible de changer le mode de transfert de fichier FXP par exemple. Pour connaître les possibilités de commande `cf` il suffit de taper cette commande sans argument et vous obtenez un écran équivalent au suivant :

```
[> cf
:: −/−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−/−
:: profxp configuration options
:: −−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
::   aliases [ aliases to abbreviate commands ]
::      |− chmod = 'Site chmod'
::      |− del = 'Dele'
::      |− new = 'Site new'
::      |− mkdir = 'Mkd'
::      |− who = 'Site who'
::      |− rmdir = 'Rmd'
::   cmode = '0' [ connexion mode (0−direct_passive 1−direct_active 2−proxy_passive 3−proxy_active) ]
::   fmode = '0' [ FXP mode (0−normal 1−alt) ]
::   listing [ display options for directory listings ]
::      |− 0 = '1'
::      |− showdot = '0'
::      |− pauselines = '30'
::   localif = '0.0.0.0' [ the local interface to use for outgoing connexions ]
::   noopcmd = 'NOOP' [ anti−idle command to send (use NOOP or PWD) ]
::   noopdelay = '60' [ anti−idle interval in seconds ]
::   skipdot = '1' [ skip dot files ]
::   skipempty = '1' [ skip empty directories ]
::   skipext [ file extensions to skip ]
::      |− 'diz'
::      |− 'core'
::   skipsame = '1' [ skip files present with same size ]
::   skipzero = '0' [ skip 0−byte files ]
::   socks [ set SOCKS parameters ]
::      |− ip = '0.0.0.0'
::      |− pass = ''
::      |− port = '1080'
::      |− type = '4'
::      |− user = ''
::   sortrev = '0' [ sort files literally reverse on multifile xfers ]
::   sortup [ file extensions to transfer 1st in a listing ]
::      |− 'nfo'
::      |− 'sfv'
::   timeout [ various timeouts (in seconds) ]
::      |− data = '15'
::      |− connect = '15'
::      |− command = '10'
::   verbose = '2' [ set verbosity level (1 min, 3 max) ]
::   viewer = 'less' [ text file viewer program ]
::   xmode = '0' [ FXP mode (0−standard 1−alternative) ]
:: −−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
:: for simple toggles, '0' means OFF and '1' means ON
:: −/−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−/−
```

On voit que des alias existent déjà, c'est le cas pour `chmod` ou `del` par exemple. On voit aussi qu'il est possible de spécifier un serveur proxy par la configuration de la partie `socks` et qu'il y a bien d'autres options. Par exemple, si jamais vous souhaitez que les fichiers de 0 octets ne soient pas transférés, il suffit de mettre `skipzero` à la valeur '1' et non '0'.

xi

La commande que tout le monde attend :−), c'est elle qui permet de transférer un fichier (ou groupe de fichiers) depuis un serveur vers un autre. Pour cela il faut se connecter sur les 2 serveurs et faire en sorte que le serveur qui va recevoir le fichier ne soit pas celui sur lequel vous passez la commande (ie: vous devez taper cette commande sur le shell correspondant au serveur d'envoi). En gros si vous souhaitez transférer un fichier vous devez obtenir quelquechose comme l'exemple suivant :
[serveur1|/rep/reception]
[serveur2|/rep/envoi]
On a donc le serveur2 qui a le fichier toto.txt que l'on souhaite envoyer au serveur1, il suffit donc de faire :

```
xi toto.txt
```
Et voilà c'est parti, si vous souhaitez transférez tous les fichiers toto il suffit de faire :
```
xi toto*
```

Informations complémentaires

Pour le moment je n'ai pas trouvé le moyen de gérer des queues de fichiers, si jamais quelqu'un trouve, qu'il n'hésite pas à en faire part.
Sinon sachez que vous pouvez passer tout type de commandes compréhensibles par un serveur FTP, comme des commandes "site", mais pour cela vous devez les taper en majuscules ou tout du moins la première lettre en majuscule, pour que le logiciel n'essaie pas d'interpréter lui−même la commande et qu'il l'envoi tel quel au serveur, exemple :
```
Site user anonymous
```
Autre petite information pratique, la touche `[tab]` peut être utilisée à tout moment, pour la liste de vos sites, les noms de fichiers et répertoires sur les FTPs où vous êtes, cela évite si jamais vous avez donné un nom trop long à un site ou qu'un fichier a un nom trop long, de le taper en entier, il suffit de taper les premières lettres et un appui sur la touche `[tab]` et le tour est joué....

Conclusion

Voilà pour ce qui est de cette présentation de profxp. Ce type de logiciel est réservé à une utilisation bien spécifique, bien peu d'entre vous en auront besoin je pense, mais on sait jamais, moi je trouve ça très utile comme logiciel et permet de faire des transferts de fichiers depuis des connexions bas débits très facilement.
Il vous est aussi possible de télécharger les sources pour voir comment cela marche et pourquoi pas vous en servir comme exemple de programmation Perl :−)
Et n'oubliez pas, les mots de passes sont stockés en clair dans les fichiers .site du répertoire ~/.profxpv3/ et lorsque vous vous connectez sur un serveur le mot de passe est aussi affiché en clair à l'écran, alors soyez prudent.

Pour finir un lien vers la page de l'auteur : http://duncanthrax.net/profxp/

DNS BIND 1ère partie : serveur "cache DNS"

Par Serge

Pour cette rubrique, des connaissances sur TCP/IP sont nécessaires, ainsi qu'un minimum de savoir−faire sur la configuration des paramètres TCP/IP d'une machine Linux/Unix (bien que j'essaye d'expliquer au maximum toutes les étapes).

Introduction

Qu'est−ce donc qu'un serveur DNS?

C'est tout simplement ce qui sert à convertir des adresses noms en adresses IP. Par exemple, quand vous tapez dans votre navigateur préféré l'adresse :
http://lea−linux.org
celui−ci va tout d'abord faire une requête à un serveur DNS (généralement le serveur DNS que vous avez configuré pour votre connexion à l'Internet, donc les serveurs DNS de votre fournisseur d'accès) en lui demandant :
C'est quoi l'adresse IP de lea−linux.org ?
Et le serveur DNS lui donne l'adresse IP et le navigateur va alors se connecter à cette adresse IP et afficher le site.
Ceci est valable pour toute autre application qui manipule des noms DNS (ftp, telnet, mail, ...).

Dans quel cas installer un serveur DNS qui fait cache?

Tout d'abord, il faut savoir que l'on peut remplir à la main un fichier, qui s'appelle **/etc/host**, avec les correspondances "adresse IP et nom DNS des machines". Donc si l'on a un petit réseau local de quelques machines, on peut remplir ce fichier à la main. Mais dans le cas d'un réseau beaucoup plus conséquent, il vaut mieux installer un serveur DNS.
Là où un serveur DNS peut être utile aussi, c'est si vous partagez une connexion internet sur votre réseau local. Vous installez sur la machine qui partage la connexion internet un serveur DNS qui fait cache. De ce fait, toutes les machines qui vont se connecter via ce serveur à l'internet vont demander la résolution d'adresse à votre serveur en interne plutôt qu'aux serveurs DNS externes, ce qui va permettre de réduire le traffic réseau sortant. Bon, en fait pour les premières connexions cela ne va servir à rien, car comme votre serveur DNS sera presque vide, celui−ci va demander aux serveurs externes les réponses afin de remplir sa base DNS. Mais comme votre serveur va stocker les réponses, au bout d'un moment, cela va accélérer les réponses DNS, car c'est votre serveur local et non pas les serveurs externes qui vont répondre à ces requêtes DNS.
Même dans le cas où vous n'avez qu'une machine "poste de travail" sous linux et que vous vous connectez très souvent à l'internet, le fait d'installer là aussi un serveur DNS sur votre poste va accélérer votre connexion, ce qui est assez agréable, surtout si vous êtes en RTC (connexion via le téléphone). De plus, mon FAI avait fréquemment des problèmes de serveurs DNS, ce qui m'empêchait régulièrement d'atteindre certains sites, alors je me suis décidé à installer un serveur DNS qui fait cache des serveurs DNS de référence de l'internet. Depuis je n'ai plus aucun probléme.

Pré−requis

Bon, avant de configurer le tout, il faut vérifier que vous avez sur votre machine ce qu'il faut pour faire votre serveur DNS, c'est−à−dire le package *bind*. Vérifiez donc que ce package est installé sur la machine qui va faire serveur DNS, dans le cas d'une Mandrake ou Red Hat, vérifiez avec un utilitaire pour les RPM, dans le cas d'une Slackware, vérifiez avec *pkgtool*, ou dans tous les cas rechercher un fichier nommer "*named*" qui se trouve dans la plupart des cas dans */usr/sbin/named*. Si vous ne trouvez pas de package "Bind...." ou si vous ne trouvez aucun fichier "named", installez alors le package "Bind....", qui doit sûrement se trouver sur le CD d'install de votre distribution.

Théorie : fonctionnement du service DNS.

Pour comprendre la configuration d'un serveur DNS, un minimum de théorie sur le fonctionnement de celui−ci est nécessaire. Tout d'abord, il faut savoir que les noms DNS sont organisés de façon hiérarchique. Le haut de la hiérarchie est le root (la racine), désigné par ".", puis viennent les "Top Level Domains" (TLD) que vous connaissez : .com, .net, .fr, .org, ... En fait, quand un nom est recherché, par exemple lea−linux.org, la "question" est posé dans l'ordre hiérarchique, tout d'abord on demande à un serveur racine (un serveur connu sur internet et déclaré comme étant un serveur racine) la liste des serveurs pouvant répondre au TLD ".org", puis on descend d'une hiérarchie et on recommence. Donc à partir de la liste de serveurs obtenue, on demande ceux qui répondent à "lea−linux.org", puis de même pour lea−linux.org. C'est en gros le fonctionnement du service DNS.

Un serveur DNS qui fait cache

Le fichier /etc/named

On va voir ici une première configuration possible, un serveur DNS qui fait cache. Vous avez besoin de ce type de serveur si votre serveur DNS ne vous sert que pour accélérer votre connexion internet (très utile si vous étes connecté via un modem RTC), ou si vous partagez une connexion internet via votre machine.

Tout d'abord, éditons le fichier /etc/named.conf (créez ce fichier s'il n'existe pas) :

```
// Fichier /etc/named.conf

// pour serveur DNS cache
options {
                // Répertoire des fichiers de configuration

                directory "/var/named";

                // Si vous traversez un firewall
                // et que vous avez des problèmes DNS
                // décommentez la ligne ci−dessous
```

```
                // query-source port 53;

                version "SECRET";
                //permet de masquer la version de Bind - voir précisions ci-dessous
        };

controls {
          inet 127.0.0.1 allow { localhost; } keys { rndc_key; };
          };


key "rndc_key" {
                   algorithm hmac-md5;
                   secret "c3Ryb25nIGVub3VnaCBmb3IgYSBtYW4gYnV0IG1hZGUgZm9yIGEgd29tYW4K";
                };



//Zone racine (root)

zone "."
                {
                type hint;
                file "root";
                };
//Zone locale

zone "0.0.127.in-adr.arpa"
                {
                type master;
                file "zone/127.0.0";
                };
```

Explication de ce fichier :

Toutes les lignes qui débutent par "**//**" sont des lignes de commentaires, elles ne sont là que pour documenter le fichier.
La ligne "directory" indique à *named* où il doit trouver tous ses fichiers de configurations. Tous les autres fichiers auront donc un chemin relatif à celui-ci. On a ensuite la déclaration de la zone ".", la zone racine (root en anglais), avec son fichier de description que l'on va voir plus bas. Ensuite, on a la zone qui peut vous paraître bizarre, la zone "0.0.127.in-adr.arpa". En fait la zone "in.adr.arpa" est une zone "spéciale" qui nous permet de faire des recherches inverses, c'est-à-dire trouver le nom d'une machine à partir de son adresse IP cette fois-ci. Ce fonctionnement est identique à celui qui permet de trouver les noms. Pour trouver une machine d'adresse IP 192.168.1.1 par exemple, la requête va être de trouver en premier *in-adr.arpa* (en quelque sorte le root) , puis *192.in-adr.arpa*, puis *168.192.in-adr.arpa*, etc... En fait, dans notre fichier, on déclare ici la zone de recherche inverse sur le domaine 127.0.0 qui est la zone de notre domaine local (rappelez-vous que les adresses IP 127.0.0.x sont des adresses IP spéciales, 127.0.0.1 est l'adresse de loopback d'une machine), ce fichier est détaillé aussi plus bas.

version "SECRET" : permet de masquer la version de Bind utilisée et de limiter ainsi l'exploitation de failles e sécurité.
Exemple :

```
[root admin]# dig @digital-connexion.info version.bind txt chaos

; <> DiG X.x <> @digital-connexion.info version.bind txt chaos
; (1 server found)
 ;; res options: init recurs defnam dnsrch
 ;; got answer:
 ;; ->>HEADER<
```

Les fichiers de zones

Comme nous avons vu au-dessus, il nous faut d'autres fichiers, qui sont **`/var/named/root`** et **`/var/named/zone/127.0.0`**

`/var/named/root` :

```
; Fichier de zone des serveurs root
; A vérifier et à maintenir réguliérement



.               6D IN NS        A.ROOT-SERVERS.NET.
.               6D IN NS        B.ROOT-SERVERS.NET.
.               6D IN NS        C.ROOT.SERVERS.NET.
.               6D IN NS        D.ROOT-SERVERS.NET.
.               6D IN NS        E.ROOT-SERVERS.NET.
.               6D IN NS        F.ROOT.SERVERS.NET.
.               6D IN NS        G.ROOT-SERVERS.NET.
.               6D IN NS        H.ROOT-SERVERS.NET.
.               6D IN NS        I.ROOT.SERVERS.NET.
.               6D IN NS        J.ROOT-SERVERS.NET.
.               6D IN NS        K.ROOT-SERVERS.NET.
.               6D IN NS        L.ROOT.SERVERS.NET.
.               6D IN NS        M.ROOT.SERVERS.NET.

A.ROOT-SERVERS.NET.     5w6d16h IN      A       198.41.0.4
```

```
B.ROOT-SERVERS.NET.      5w6d16h IN      A       128.9.0.107
C.ROOT-SERVERS.NET.      5w6d16h IN      A       192.33.4.12
D.ROOT-SERVERS.NET.      5w6d16h IN      A       128.8.10.90
E.ROOT-SERVERS.NET.      5w6d16h IN      A       192.203.230.10
F.ROOT-SERVERS.NET.      5w6d16h IN      A       192.5.5.241
G.ROOT-SERVERS.NET.      5w6d16h IN      A       192.112.36.4
H.ROOT-SERVERS.NET.      5w6d16h IN      A       128.63.2.53
I.ROOT-SERVERS.NET.      5w6d16h IN      A       192.36.148.17
J.ROOT-SERVERS.NET.      5w6d16h IN      A       198.41.0.10
K.ROOT-SERVERS.NET.      5w6d16h IN      A       193.0.14.129
L.ROOT-SERVERS.NET.      5w6d16h IN      A       198.32.64.12
M.ROOT-SERVERS.NET.      5w6d16h IN      A       202.12.27.33
```

Comme décrit précédemment, ce fichier contient la liste des serveurs DNS root (racine).

`/var/named/zone/127.0.0`:

```
$TTL 3D
@       IN      SOA        nom_machine.votre_domaine.votre_TLD. (
                                1       ; Serial
                                8H      ; Refresh
                                2H      ; Retry
                                1W  ; Expire
                                1D)     ; Minimum TTL
             nom_machine.votre_domaine.votre_TLD.
1               PTR             localhost.
```

Remplacez bien sur "*nom_machine.votre_domaine.votre_TLD"* par les valeurs réélles de votre machine (par exemple azizlinux.icebox.org). Les "**.**" après les noms de domaines **ne sont pas une erreur et doivent être présents**. J'insiste bien sur ce fait, car l'erreur typique du débutant et de ne pas mettre le "**.**" (j'avoue que ca m'arrive encore parfois).

Configuration de l'utilitaire de controle rndc

Si vous regardez a nouveau le fichier *named.conf* que l'on a fait précédement, vous avez vu deux sections appelées `"controls"` et `"key"`. Ces deux directives permettent le controlle de votre serveur de nom via le programme "***rndc***". La ligne `"controls"` définit les adresses IP des machines autorisées à contrôler le serveur de nom (dans notre exemple il s'agit de la machine 127.0.0.1), et la ligne "*keys*" définit un couple algorythme/clef qui est en fait une sorte de mot de passe pour autorisé là aussi le contrôle à distance. Pour que la machine puisse alors s'identifier, il faut qu'elle envoie ce mot de passe. Pour cela, il faut créer un fichier `/etc/rndc.conf` contenant la clef :

`/etc/rndc.conf`:

```
key rndc_key {
      algorithm "hmac-md5";
      secret "c3Ryb25nIGVub3VnaCBmb3IgYSBtYW4gYnV0IG1hZGUgZm9yIGEgd29tYW4K";
  };

  options {
      default-server localhost;
      default-key    rndc_key;
  };
```

Faite bien attention à ce que les informations entres les fichiers `named.conf` et `rncd.conf` soient cohérents, c'est à dire que `"algorithm"`, `"secret"`, `"default-key"` aient exactement les même valeurs, autrement l'identification entre le serveur et rndc echouera.
Pour la valeur de "*default-server*" dans le fichier *rndc.conf*, vous mettez là l'adresse ou le nom de la machine que vous voulez controler à distance (dans notre exemple c'est la même machine).
Changez au moins la valeur de la key que je donne ici, elle est issue du fichier exemple de BIND, c'est à dire que tout le monde possède cette clef par défaut, donc si vous ne voulez pas que quelqu'un contrôle votre serveur à votre insu, modifiez la!

rndc permet de stopper, démarrer, recharger la configuration du serveur BIND à distance. Voir la documentation de rndc, ou les autres parties de ce document.

Configuration des fichiers systèmes

Bon maintenant que votre DNS est configuré en tant que serveur cache, il reste encore deux fichiers systèmes à configurer pour indiquer à notre machine que nous avons un serveur DNS et que nous pouvons nous en servir. Commençons par **/etc/resolv.conf** :

```
search  votre_domaine.votre_TLD
nameserver      127.0.0.1
```

Remplacez *votre_domaine.votre_TLD* par les valeurs de votre machine bien sûr (par exemple icebox.org). **Ne pas mettre que votre TLD sans spécifier le domaine**.
En fait, à quoi sert ce fichier ? Lorsqu'une application va lancer une requête DNS, elle ne fait pas appel directement à votre serveur DNS mais au "*résolveur*". Celui-ci lit ce fichier et regarde la ligne ***search*** pour savoir dans quel domaine chercher en premier. Cela peut paraître stupide, mais quand vous lancez une requête sur "www.netscape.com", via le résolveur vous allez chercher en premier "www.netscape.com.votre_domaine.votre_TLD". Pourquoi ? me demanderez-vous. Tout simplement parce que les personnes qui ont inventé cela pensaient que la plupart des requêtes de noms seraient en interne sur un réseau et rarement vers l'extérieur, donc par défaut, il cherche en premier dans votre nom de domaine.
La ligne ***nameserver*** indique l'adresse à laquelle on peut contacter un serveur de nom.

Aussi bien pour *search* que pour *nameserver*, on peut spécifier plusieurs entrées de cette façon :

```
search  domaine1.tld1   domaine2.tld2 ....
nameserver adresse1
nameserver adresse 2

nameserver adresse 3

...
```

Attention tout de même à ne pas surcharger en nombre d'entrées dans la ligne search car ca va prendre du temps à chercher dans tous les domaines spécifiés. Différentes entrées sur cette ligne ne servent que si vous êtes sur un réseau local avec plusieurs noms de domaines et que vous avez l'habitude d'appeler les machines juste par leurs noms d'hôte. Dans le cas d'une machine reliée au net, mettre juste le nom de domaine de votre machine et rien de plus.

Il reste encore un fichier à configurer, le fichier ***/etc/host.conf***, éditez ce fichier et vérifiez que la **première** ligne de ce fichier comporte :

```
order hosts,bind
```

Cette ligne oblige le résolveur à regarder en premier les entrées du fichier ***/etc/hosts*** puis de faire appel au serveur de nom qui est spécifié dans */etc/resolv.conf*.

Test de la configuration

Nous allons tester maintenant que la configuration fonctionne. Lancez votre connexion à l'internet. Vérifiez que le serveur DNS n'est pas déjà démarré par un :

```
ps −aux | grep named
```

Si la seule ligne que vous renvoie cette commande est "grep named", c'est que named n'est pas lancé. Sinon relevez le numéro de PID de named et killez−le (kill *numéro_du_PID*). Relancez ou lancez named par la commande :

```
/usr/sbin/ndc start
```

(remplacez /usr/sbin par le répertoire ou ndc est installé, si vous ne le trouvez pas, lancez alors named la commande "named&").

Exécutez alors la commande "*nslookup*" et vous devriez voir apparaître:

```
olio@azizlinux:~$ nslookup
Default Server:  localhost
Address:  127.0.0.1
```

Ca veut dire que tout marche bien, du moins que le serveur named est lancé et est prêt à répondre aux requêtes DNS (allez voir plus bas si ce que vous obtenez n'est pas la même chose que ce qui est affiché ci−dessus).

On va tester que le cache marche vraiment en tapant maintenant:

```
> lea−linux.org
```

et vous devez voir apparaîitre quelque chose comme:

```
Server: localhost
Address: 127.0.0.1
Name: lea−linux.org
Address: 212.73.209.252
```

Puis retapez exactement la même commande et cette fois ci, le résultat devrait être :

```
> lea−linux.org
Server:  localhost
Address:  127.0.0.1

Non−authoritative answer:
Name:    lea−linux.org
Address:  212.73.209.252

>
```

La ligne "*Non−authoritative answer*" signale justement que l'information a été récupérée via le cache du serveur DNS, et nous avertit donc que cette adresse peut être fausse si l'adresse a changé depuis, ce qui est très improbable. Quittez alors l'utilitaire *nslookup* par "*exit*".
Voilà, votre serveur "DNS cache" est opérationnel. Il ne vous reste qu'à ajouter le "*/usr/sbin/ndc start*" dans un script de démarrage (par exemple /etc/rc.local).

Ca marche pas!

Bon, soit c'est votre serveur DNS qui n'est pas lancé, soit vos fichiers systèmes sont mal configurés. Pour le savoir, lancez la commande suivant pour voir si *named* tourne en tâche de fond :

```
ps −aux | grep named
```

Si named ne tourne pas en tâche de fond, éditez alors le fichier */var/log/messages* et repérez les lignes qui comporte ***named[xxxx]: .....*** , elles devraient vous indiquez les erreurs de configuration que vous avez faites (fichier de zone non trouvé, etc...).

Si named tourne, vérifiez alors que le résolveur se connecte bien à votre serveur DNS et pas sur un autre. Pour cela, quand vous lancez nslookup, celui−ci doit vous répondre que le serveur par défaut est bien localhost, avec comme adresse "127.0.0.1". Si ce n'est pas le cas, revérifiez les fichiers /etc/resolv.conf et /etc/host.conf. Vérifiez aussi le fichier de zone 127.0.0.

Remarques sur un serveur DNS cache

- Un serveur DNS cache n'écrit pas dans un fichier les différentes adresses qu'il récupère, il les stocke en mémoire, ce qui veut dire qu'à partir du moment où vous éteignez votre machine (ou que vous stoppez le service *named*), vous perdez toutes les adresses en mémoire. Donc si vous utilisez votre serveur DNS juste pour accélérer votre connexion internet, ne stoppez pas votre serveur DNS et ne rebootez pas votre machine, où vous n'utiliserez en fait jamais votre cache. Le seul bénéfice dans ce cas est que vous utilisez votre serveur DNS qui fait cache avec les serveurs "root" d'internet, ce qui vous garantit des réponses fiables.
- Dans le cas où vous partagez votre connexion internet (modem cable, adsl, ou même simple modem) il est très utile d'utiliser un serveur DNS cache. Par contre, pour que vos stations qui utilisent cette connexion partagée se servent de ce serveur cache DNS, n'oubliez surtout pas de configurer toutes les stations pour qu'elles utilisent comme serveur DNS votre serveur et pas un autre. Pour cela, donnez comme adresse de serveur DNS l'adresse interne (côté LAN donc) de votre serveur.

DNS BIND 2ème partie : serveur de Zone

Par Serge

Pour cet article, je considère que vous avez lu la 1ère partie, sur les serveurs DNS cache, qui permet déjà de comprendre les fichiers de zones et donne la configuration minimale d'un serveur DNS. Sans cette configuration minimale votre serveur DNS ne fonctionnera pas.

J'explique dans cet article comment configurer une zone personnelle pour un serveur primaire et secondaire. Je n'explique pas les zones inverses ni la sécurité que l'on peut ajouter sur les zones que notre serveur va contenir. Ces parties seront vu dans un 3ème article configuration avancée.

Un serveur DNS pour mon domaine.

Nous prenons ici un exemple de configuration pour le domaine .org. Remplacez bien sûr avec votre propre domaine.

Attention:

Votre serveur DNS doit, pour pouvoir fonctionner, avoir une adresse IP fixe sur l'Internet. De plus votre domaine doit être enregistré chez un registar (Gandi, Namebay, ou tout autre registar) et lors de l'enregistrement du domaine vous devez fournir l'IP du serveur de nom en serveur primaire et l'IP du serveur secondaire. Nous détaillons en 1er la configuration du serveur primaire, celle du secondaire étant triviale (voir plus bas dans l'article).

Le fichier /etc/named

On reprend le fichier déjà créé dans l'article dns cache.

Pour rappel, voici ce qu'il doit donc déjà contenir:

```
// Fichier /etc/named.conf
// pour serveur DNS cache
options {
    // Répertoire des fichiers de configuration

    directory "/var/named";

    // Si vous traversez un firewall
    // et que vous avez des problèmes DNS
    // décommentez la ligne ci−dessous
    // query−source port 53;

    version "SECRET";
    //permet de masquer la version de Bind − voir précisions ci−dessous
    };

controls {
    inet 127.0.0.1 allow { localhost; } keys { rndc_key; };
    };


key "rndc_key" {
    algorithm hmac−md5;
    secret "c3Ryb25nIGVub3VnaCBmb3IgYSBtYW4gYnV0IG1hZGUgZm9yIGEgd29tYW4K";
    };

//Zone racine (root)

zone "."
    {
    type hint;
    file "root";
    };
//Zone locale

zone "0.0.127.in−adr.arpa"
    {
    type master;
    file "zone/127.0.0";
    };
```

On ajoute alors dans ce fichier l'entrée pour notre fichier de zone lea−linux.org:

```
zone lea−linux.org
   {
   type master
   file zone/lea−linux.org
   }
```

L'ajout de zone est très simple comme vous pouvez le voir. Il suffit de dire quelle zone ce serveur va servir, mot clef zone , d'indiquer si nous si sommes le serveur d'autorité pour cette zone, mot clef type master et d'indiquer le fichier contenant les informations de la zone, mot clef file.

Voyons maintenant ce que doit contenir ce fichier de zone.

Le fichier de zone de mon domaine.

Nous devons donc créer le fichier de zone de notre domaine lea−linux.org.

Voyons le fichier, puis expliquons le:

```
$TTL 3D
@       IN    SOA     ns1.kilio.com. serge.kilio.net. (
                      2003091801; serial
                      86400 ; refresh
                      3600 ; retry
                      3600000 ; expire
                      604800 ; default_ttl
                      )

@       IN     NS         ns1.kilio.com.
@       IN     NS         ns2.kilio.com.
@       IN     MX   50    ns1.kilio.net.
@       IN     MX   10    nice.kilio.net.

        IN     A          80.245.32.131
irc     IN     A          80.245.32.129
www     IN     CNAME      linuxmutuel.kilio.fr.
wiki    IN     A          80.67.179.10
vizu    IN     A          195.202.0.21
devel   IN     A          80.67.179.10
sniper  IN     A          62.212.100.216
ftp     IN     A          80.245.32.131
beta    IN     CNAME      linuxmutuel.kilio.fr
```

Cela peut vous paraître très barbare mais en fait rien n'est très compliqué. On peut découper ce fichier en deux parties:

- L'en−tête: qui commence au début du fichier $TTL et se finit à la)
- Les enregistrements de la zone: le reste du fichier.

Détail de l'en−tête.

Nous l'avons déjà vu dans l'article précédent mais voici ce qu'elle doit contenir obligatoirement:

$TTL : l'indication du TTL (Time To Live) ou la durée de vie de la zone, exprimée en secondes par défaut, ou dans une autre unité si on la spécifie comme dans l'exemple, ici le D spécifiant que l'unité est en jours, donc 3 jours pour notre exemple.

Le bloc suivant défini tous les autres paramètres techniques et administratifs de la zone de la forme:

```
@       IN    SOA     dns_primaire. adresse_mail. (
                      xxxxxxxx; serial
                      xxxxx; refresh
                      xxxxx; retry
                      xxxxx; expire
                      xxxxx; default_ttl
                      )
```

dns_primaire: mettre ici le nom du serveur faisant autorité pour la zone

adresse email: email du responsable technique de la zone en remplaçant le @ de l'email par un .

Le reste, encadré par des parenthèses renseigne les données techniques de la zone avec:

serial: le numéro de version de la zone.

Ce chiffre est particulièrement important. A chaque fois que vous modifiez quoi que ce soit dans un fichier de zone, vous devez impérativement incrémenter ce numéro, autrement les changements ne seront pas pris en compte par le reste du monde et particulièrement par le serveur secondaire. C'est ce numéro, s'il est incrémenté, qui indique au reste du monde que votre zone a subit un changement et que donc les autres serveurs DNS doivent redemander la zone à votre serveur pour prendre en compte ces changements.

On a l'habitude de suivre une règle simple pour être sûr d'incrémenter ce numéro de version, on le compose par la suite des chiffres de l'année en cours, mois, jour et le nombre de changements effectués ce jour là:

AAAAMMJJNN

Donc par exemple si je modifie la zone le 10 juillet 2003 , je vais mettre 2003071001, puis je m'aperçois que je me suis trompé donc je la modifie à nouveau le même jour, donc ce serial devient 2003071002, et si je la modifie ensuite le 15 août de la même année le serial devient alors 2003081501.

Cette règle n'est pas du tout obligatoire, vous pouvez utiliser votre propre règle du moment que le nombre soit incrémenté et que ce nombre ne comporte pas plus de 10 chiffres.

Refresh: Temps en secondes d'attente du serveur secondaire avant de contrôler si le serveur primaire a subi une modification au niveau de sa zone. Sur 8 chiffres max.

Retry: Temps d'attente du serveur secondaire avant de faire à nouveau une demande si le serveur primaire n'a pas répondu à une requête. Sur 8 chiffres max.

Expire: Temps pendant lequel le serveur secondaire va conserver les données en cache. Si ce délai est dépasse et que le serveur secondaire n'a pas pu contacter le serveur primaire, il va alors considérer que les données qu'il a en cache ne sont plus fiable et ne pourra plus servir de serveur secondaire pour la zone tant qu'il n'aura pas réussi à contacter le serveur primaire. Sur 8 chiffres max.

Ttl: Valeur par défaut de ttl des enregistrements. On peut spécifier les ttls au niveau de chaque enregistrement, mais d'une manière générale on défini ici un ttl qui vaut pour tous les enregistrements.

Enregistrement de la zone.

Tout les enregistrements d'une zone suivent cette syntaxe:

hôte_ou_wildcard (ttl) classe type (priorité_si_besoin) valeur

Voici les explications de ces champs, puis en dessous quelques exemples pour mieux comprendre. Par défaut nous n'utilisons pas de ttl pour les enregistrements (voir la remarque au dessus pour le ttl dans l'en−tête de zone).

Hôte ou wildcard: indique si on défini une machine ou un ensemble de machine.

Classe: type de classe, a comme valeur IN pour l'Internet. Nous ne détaillerons pas ce champ, vous utiliserez toujours la classe IN pour vos besoins.

Type: Indique quel type d'enregistrement nous sommes en train de définir. Les types les plus utilisés sont A pour une adresse, CNAME pour un alias de nom, NS pour un serveur de nom, MX pour un serveur de Mail, TXT pour des commentaires.

Priorité: Si le type à besoin d'une priorité, nous l'indiquons ici.

Valeur: la valeur ou la donnée de l'enregistrement que nous définissons.

Détaillons alors les exemples de notre zone pour mieux comprendre:

```
@    IN    NS    ns1.kilio.com.
@    IN    NS    ns2.kilio.com.
```

Nous indiquons ici que les deux serveurs DNS (type NS) de la zone (wildcard @) lea−linux.org sont ns1.kilio.com et ns2.kilio.com

```
@   IN    MX    50    ns1.kilio.net.
@   IN    MX    10    nice.kilio.net.
```

Nous indiquons ici que les deux serveurs de Mail (type MX) de la zone (wildcard @) lea−linux.org sont : nice.kilio.net avec la priorité 10 ns1.kilio.net avec la priorité 50. Cela veut dire que tout mail du type xxxxxx@lea−linux.org doit être envoyé en priorité sur nice.kilio.net et que si ce serveur ne répond pas on l'envoie alors sur le serveur ns1.kilio.net. Faites attention, c'est la priorité la plus basse qui indique le premier serveur à contacter.

```
irc    IN    A    80.245.32.129
```

Nous indiquons ici que l'hôte irc.lea−linux.org correspond à l'adresse 80.245.32.129

```
www    IN    CNAME    linuxmutuel.kilio.fr.
```

Nous indiquons ici que l'hôte www.lea−linux.org correspond à l'hôte linuxmutuel.kilio.fr .
On aurait pu aussi le définir en type A et mettre l'adresse ip correspond à linuxmutuel.kilio.fr comme valeur. Mais dans ce cas, si la machine linuxmutuel.kilio.fr change d'adresse IP on doit modifier la valeur de www pour la zone lea−linux.org. De plus, comme cette machine héberge environ 200 sites web différents, il y aurait 200 fichiers de zones à modifier avec la nouvelle IP. Pour éviter cela, on définit alors www.lea−linux.org comme CNAME de linuxmutuel.kilio.fr.

```
    IN     A     80.245.32.131
```

Vous remarquerez que le 1er champ est vide. Cela indique tout simplement que si l'on demande lea−linux.org directement (sans spécifier d'hôte) on aura l'IP 80.245.32.131. Ce qui permet de taper dans votre navigateur http://lea−linux.org sans le "www"..

Voici un autre exemple qui n'est pas dans notre zone mais qui définit le type TXT:

```
info     IN     TXT     Zone du fabuleux site linux entre amis
```

Indique que info à comme valeur Zone du fabuleux site Linux entre amis. Aucune application n'a besoin de ce type d'enregistrement, mais il peut être utile pour donner des infos sur une zone, etc..

Il existe d'autres types d'enregistrements que ceux vu ci−dessus, mais ils sont très rarement utilisés sur l'Internet. Consultez les RFC sur les DNS si vous souhaitez les connaître.

Serveur secondaire.

Tout ce que nous avons vu au dessus concerne la configuration du serveur primaire de votre zone. Il faut généralement déclarer lors de l'enregistrement de votre domaine chez un registar un serveur secondaire (et parfois des tertiaires etc.. mais la configuration est exactement identique à celle d'un serveur secondaire, et seul le secondaire et obligatoire). Il faut donc, vous vous en doutez, le configurer lui aussi.

Le serveur secondaire sert tout simplement à répondre aux requêtes à la place du primaire pour ne pas surcharger le serveur primaire ou si le serveur primaire est hors service temporairement.

Il doit contenir tout d'abord la configuration de base vue dans l'article serveur cache comme n'importe quel serveur DNS qu'il soit primaire ou secondaire. Et il doit aussi contenir dans son fichier named.conf les zones pour lesquelles il est serveur secondaire sous la forme:

```
zone lea−linux.org
    {
    type slave
    file zone/lea−linux.org
    masters {ip_serveur_primaire;}
    }
```

et rien d'autre. Pas de fichier de zone ni rien. Les fichiers de zones seront créés automatiquement en interrogeant le serveur primaire directement via la directive masters.

Et si ça marche pas?

Vous pouvez alors faire exactement les mêmes vérifications que celles données dans l'article serveur cache. La seule différence est que si vous faites les tests nslookup sur le serveur primaire vous ne devez pas avoir de réponse Non−authoritative.

Qmail: installation d'un serveur de mail multi−domaine et sécurisé

par serge

Un serveur de mail gérant plusieurs domaines, sécurisé et protégé contre les virus.

Introduction

Nous allons voir dans cet article comment installer un serveur de mail pouvant gérer plusieurs domaines (des centaines, voir plus), sécurisé contre l'open relay mais permettant aux utilisateurs authentifiés d'envoyer des mails. De plus, les mails entrant seront scannés à la recherche de virus. Car même si sous linux la plupart des virus mail n'ont aucun effet, il se peut que certains des utilisateurs récupèrent leurs mails sous windows.

Ce serveur de mail comprend aussi un gestionnaire de mailing list et une interface web d'admin des domaines mails.

Pour la compréhension de cet article, vous devez avoir des notions assez avancées sur Linux et son utilisation (commandes shell en ligne, compilation, etc...), et des notions sur les réseaux TCP/IP et de l'internet.

Cet article est une "compilation" d'articles trouvés sur l'internet (en partie l'article qmail sur toolinux), d'HOWTO, et de mon expérience professionelle de Qmail.

Récupération des sources et compilation

Télécharger les sources

Récupérez tout d'abord les sources de ces différents programmes (attention, vérifiez les liens et versions, je vous donne les versions à jour, à l'heure où j'écris cet article, c'est−à−dire en avril 2002):

− qmail−1.03
− autorespond−2.0.2
− daemontools−0.76
− ucspi−tcp−0.88
− ezmlm−0.53
− ezmlm−idx
− maildrop−1.3.8
− qmail−scanner−1.10
− vpopmail−5.2
− F−PROT (version linux gratuite pour un usage privé, prenez aussi les mises à jours des définitions de virus sur la page download du site).
− Qmailadmin

Oui je sais, ca fait beaucoup de chose à récupérer, j'espére que vous avez un bon modem :)

Compiler Qmail et l'installer

Bon on va commencer par installer Qmail lui même. Détarrez / dégzippez les sources de qmail (`tar zxvf qmail−1.0.3.tar.gz`). Nous allons devoir patcher légèrement les sources de Qmail pour notre besoin.
Entrez dans le répertoire des sources de Qmail, et éditez le fichier Makefile, et remplacez :

− La ligne **1486** `auto_split.o` par `auto_split.o env.a`
− La ligne **1491** `substdio.a error.a str.a fs.a auto_qmail.o auto_split.o` par `substdio.a error.a str.a fs.a auto_qmail.o auto_split.o env.a`

De même, éditez le fichier qmail.c et :

−ajoutez en dessous de la ligne **8** `#include "env.h"` en dessous de la ligne `#include "auto_qmail.h"`
−remlacez la ligne **10** `static char *binqqargs[2] = { "bin/qmail−queue", 0 } ;` par `static char *binqqargs[2] = { 0, 0 } ;`

−Ajoutez juste au−dessous de la ligne que vous venez de modifier les lignes suivantes:

```
static void setup_qqargs()
{
if(!binqqargs[0])
binqqargs[0] = env_get("QMAILQUEUE");
if(!binqqargs[0])
binqqargs[0] = "bin/qmail−queue";
}
```

et enfin au−dessous des lignes (vers la ligne **23**)

```
int qmail_open(qq)
struct qmail *qq;
{
int pim[2];
int pie[2];
```

vous ajoutez:

```
setup_qqargs();
```

Remarque: Ce patch est uniquement nécessaire pour pouvoir utiliser qmail−scanner, qui va nous permettre de scanner tout mail entrant à la recherche de virus. En fait, ce patch ajoute une variable d'environnement QMAILQUEUE. Si cette variable est vide, Qmail utilise son propre programme de "queue" (file d'attente des mails entrant et sortant), autrement cette variable contient le nom du programme de queue de remplacement à utiliser. Nous utiliserons alors la "queue" de qmail−scanner qui appellera notre programme antivirus (F−PROT) pour scanner tous les mails.

Maitenant que Qmail est patché, nous allons pouvoir le compiler. Mais avant cela, nous devons créer des comptes utilisateurs, qui de plus vont servir pour le chemin d'installation de Qmail. Dans cet article, le chemin d'installation est celui par défaut, c'est à dire `/var/qmail`
Nous créons ces comptes utilisateurs par les commandes (en **root** bien sur):

```
groupadd nofiles
useradd −g nofiles −d /var/qmail/alias alias
useradd −g nofiles −d /var/qmail qmaild
useradd −g nofiles −d /var/qmail qmaill
useradd −g nofiles −d /var/qmail qmailp
groupadd qmail
useradd −g qmail −d /var/qmail qmailq
useradd −g qmail −d /var/qmail qmailr
useradd −g qmail −d /var/qmail qmails
```

Remarque: Si vous souhaitez installer Qmail dans un autre répertoire que `/var/qmail`, modifiez les commandes ci−dessus avec le chemin souhaité.

Il nous reste plus qu'à compiler Qmail:

```
make setup check
```

Configurons maintenant Qmail, via la commande :

`./config−fast `hostname −−fqdn``.

Vous devez voir apparaître quelque chose du style :

```
Your fully qualified host name is mailhub.lea−linux.org
Putting tarsier.lea−linux.org into control/me...
Putting lea−linux.org into control/defaultdomain...
Putting lea−linux.org into control/plusdomain...
.....
```

Si avez une erreur ou autre insulte de la sorte, essayez en remplaçant "**`hostname −−fqdn`**" par votre nom d'hôte complet (par exemple **./config−fast mailhub.lea−linux.org**)

Qmail utilise deux styles de mailbox (boîte aux lettres) différents : le mbox traditionel comme sendmail (un fichier nommé comme le login de l'utilisateur contenant tout les mails de l'utilisateur dans /var/mail) ou un répertoire Maildir à structure spéciale contenant les mails dans des fichiers séparés. Pour notre cas nous utiliserons Maildir, car vpopmail se base dessus.

Pour cela, il suffit de copier un fichier:

```
cp /var/qmail/boot/home /var/qmail/rc
```

Qmail n'est pas tout à fait complètement installé, mais pour la suite nous allons devoir installer les daemontools et ucspi de façon à avoir des performances optimales pour notre serveur de mails.

Installer les utilitaires supplémentaires

De façon à avoir de meilleures performances pour notre serveur de mails et pour une plus grande souplesse, pour "logger" les évements mails, nous devons installer les outils développés par Dan Bernstein : **daemontools** et **ucspi−tcp**.

Pour les installer, rien de plus simple, vous dégzippez / détarrez les sources comme d'habitude.

Pour les daemontools, un répertoire admin a été créé lors du "détarrage":

```
cd admin/daemontools/
./package/install
```

Il ne reste plus qu'à compiler / installer uscpi:

```
cd ucspi−tcp−0.88
make setup check
```

Compiler Vpopmail et l'installer

Comme d'habitude, on dégzippe / détarre les source de Vpopmail. Avant de lancer la compilation, il faut, comme pour Qmail, créer les comptes utilisateurs de vpopmail, qui vont aussi déterminer le chemin d'installation de vpopmail.
Pour ma part, pour des questions de "cohérence" avec Qmail, je l'installe dans /var/vpopmail. De plus je trouve logique de l'installer dans /var, car c'est le repertoire où l'on trouve normalement tout les fichiers qui varient en permanence (mail, logs, pid, etc...). Donc, nous faisons (en **root** bien sur) :

```
groupadd -g 89 vchkpw
useradd -g vchkpw -u 89 -d /var/vpopmail vpopmail
```

Remarque: changez le chemin `/var/vpopmail` par le chemin d'installation que vous désirez bien sur

Vous devez créer le répertoire de Vpopmail avant de continuer, pour notre exemple:

```
mkdir /var/vpopmail
```

Vpopmail possède un script de configuration avant la compilation (le bien connu **./configure**), auquel nous allons passer les options nécessaires à notre cas.
Voici les options que nous allons utiliser, avec une brève explication pour chaque option:

`--prefix=/var/vpopmail`
Répertoire de base de vpopmail

`--enable-clear-passwd=y`
Ce qui permet de stocker une copie des mots de passe POP des utilisateurs en clair. C'est peut-être moins sécurisé, mais c'est toujours pratique pour retrouver le mot de passe de l'un de vos utilisateurs qui l'aurait perdu. Si vous êtes parano, n'activez pas cette option.

`--enable-valias=y`
Permet d'utiliser les alias mails (plusieurs noms possibles pour un même compte POP)

`--enable-default-domain=`***votre domaine***
Indique le domaine primaire de votre machine. A mettre absolument, autrement vous allez devoir créer le compte de chaque utilisateur pour le domaine principal du serveur de mail.

`--enable-roaming-users=y`
Permet une authentification POP before SMTP pour activer le relaying.

Explication sur cette option:

Vous avez sûrement entendu parler du "relaying" pour les serveurs mails, l'open relay, etc... Pour résumer ce probléme, normalement un serveur de mail (comme la plupart des serveurs de mails des FAI) interdisent aux utilisateurs d'envoyer des mails si l'IP de l'utilisateur n'est pas une IP du même réseau que le serveur de mail.
Par exemple, les serveurs de mails wanadoo refusent que vous envoyiez des mails si vous n'avez pas une IP wanadoo (c.a.d si vous ne vous etes pas connecté à l'internet via leur propre service), cela pour des raisons de sécurité, de lutte anti-spam, de charge serveur et aussi économique.
La plupart des sociétés ayant leur propre serveur de mail font de même. Si vous ne faites pas ça, votre serveur est considéré comme un "open-relay", ce qui n'est pas bien du tout (je vais pas expliquer pourquoi ici, ca prendrait trop de temps).
Mais se pose alors un probléme: vous êtes une grosse société, avec 5 000 collaborateurs. Vos collaborateurs voyagent beaucoup et ont besoin d'utiliser le serveur de mail de la société. Comme ils sont en déplacement à l'extérieur de la société, ils n'ont pas d'IP de la société (ils sont connectés via un FAI quelquonque), et le serveur de mail n'autorise pas le relaying (c'est à dire envoyer un mail d'un utilisateur dont l'IP n'est pas une IP de la société). Comment faire alors ?
On a alors inventé le POP before SMTP pour permettre le relaying. Comment ca marche? Très simple. L'utilisateur doit tout d'abord "popper" sa boîte aux lettres (c'est à dire relever ses mails, c'est la chose que vous faites chaque fois que vous regardez avec votre client mail si vous avez reçus un nouvel e-mail). Lorsque l'utilisateur "poppe" sa boîte au lettre, il s'identifie sur le serveur en fait (le POP demande obligatoirement un nom d'utilisateur et un mot de passe). Si l'identification est OK, le serveur distribue les mails .
Comme l'utilisateur est identifié, on va alors se dire "ok l'utilisateur est connu, il est de chez nous", on va alors enregistrer son adresse IP actuelle et permettre a cette adresse IP d'envoyer des mails pendant un certain temps que l'on peut définir (voir plus bas dans l'article). Donc en résumé (si vous n'aviez rien compris), si vous voulez que votre serveur de mail soit utilisable pour envoyer des mails à partir de n'importe où dans le monde, à partir du moment où l'on aune boîte aux lettres sur votre serveur, activez cette option.

Donc, je vous redonne les options que nous utilisons ici :

```
./configure --enable-clear-passwd=y --enable-valias=y --enable-default-domain=lea-linux.org
--enable-roaming-users=y
```

Puis on le compile / installe:

```
make
make install-strip
```

Comme les "headers" de vpopmail sont nécessaires pour la compilation d'autres programmes (qmailadmin par exemple), nous devons les copier dans un répertoire d'include. Pour cela:

```
cp /var/vpopmail/include/* /usr/include
```

Pour que vpopail trouve bien les fichiers de tcpserver, il faut refaire le repertoire `etc/` de vpopmail:

```
mv /var/vpopmail/etc/* /etc
rmdir /var/vpopmail/etc
ln −s /etc /var/vpopmail/etc
```

Qmail−scanner et F−PROT

Avant d'installer Qmail−scanner, vous devez vous assurez d'avoir Perl 5.005_03 (ou supérieur) sur votre machine (c'est le cas avec toutes les distrib récentes) et les modules Perl:

Time::HiRes
DB_File
Sys::Syslog

Pour vous en assurer, installez−les via CPAN. Pour se faire, vous devez être connecté à internet, et lancez alors la commande:

```
perl −MCPAN −e shell;
```

Remarque: Si c'est la premiére fois que vous lancez CPAN, vous allez devoir le configurer. Je ne vous explique pas ici comment configurer CPAN. Toutefois, la plupart du temps il suffit de valider les options proposées par défaut

Une fois CPAN lancé, demandez l'installation des modules via:

```
install Time::HiRes
install DB_File
install Sys::Syslog
```
`exit` (pour sortir du CPAN)

Nous allons aussi dès à présent installer F−PROT. Dgzippez / Détarrez f−prot dans le répertoire `/usr/local` et installez le via:

```
cd /usr/local
tar zxvf /ou/se/trouve/votre/fp−linux_sb.tar.gz
ln −fs fp−linux_312/ f−prot
ln −fs /usr/local/f−prot/f−prot.sh bin/f−prot
ln −fs /usr/local/f−prot/f−prot.8 man/man8/
chmod +x /usr/local/f−prot/f−prot*
```

Mettez aussi à jour les signatures antivirales de F−PROT (**Faites−le le plus souvent possible si vous voulez un bonne protection antivirale de vos mails!**)

Téléchargez les deux zip de mise à jour de signature de f−prot et dézipé−les dans `/usr/local/f−prot`.

Avant d'installer Qmail−scanner, nous devons installer Maildrop, indispensable pour Qmail−scanner.

```
tar zxvf maildrop−1.3.8.tar.gz
cd maildrop−1.3.8
./configure
make
make install
```

Passons maitenant à Qmail−scanner.

Dégzippé / Détarré Qmail−scanner, puis son répertoire de source lancez `./configure` avec les options suivantes:

`−−admin user`
(voir en−dessous pour la valeur de user)

`−−domain votre_domaine`
Pour comprendre les valeurs à mettre dans ces options, prenons le cas où nous voulons que tous les mails d'alerte de détection virale soient envoyés à admin@lea−linux.org, vous devez alors mettre admin pour le user et lea−linux.org pour le domaine.

`−−notify all`
Pour prévenir l'administrateur, l'envoyeur et la personne qui auraient dû recevoir le mail, qu'un mail contaminé a été intercepté

`−−redundant yes`
Permettre le scan des fichiers zips, etc...

Pour notre exemple, nous lançons donc:

```
./configure −−admin admin −−domain lea−linux.org −−notify all −−redundant yes −−install
```

Attention: Vous risquez d'avoir une erreur du type:

```
YOU HAVEN'T DISABLED SET-ID SCRIPTS IN THE KERNEL YET!
FIX YOUR KERNEL, PUT A C WRAPPER AROUND THIS SCRIPT, OR USE -u AND UNDUMP!


******* FATAL ERROR *******
```

Cette erreur est "normale" si votre distribution comporte une version de Perl qui interdit que des script Perl soient lancés en SET-ID pour des raisons de sécurité (cas de la slackware par exemple).
En effet, lancer les scripts en SET-ID signifie que le script est lancé avec les droits d'un utilisateur spécifique. Certaines distributions interdisent cela (pour empêcher de lancer des script Perl qui prendraient les droits root par exemple et pourraient faire des choses pas très gentilles).
Dans ce cas (et **que** dans ce cas-là, c'est-à-dire que vous avez l'erreur énoncée ci-dessus) vous devez:

```
cd contrib
make
make install
```

Vous devez voir apparaître alors:

```
install -o qmailq -g qmail -m4755 qmail-scanner-queue /var/qmail/bin/qmail-scanner-queue
```

Revenez dans le répertoire des sources de Qmail-scanner (`cd ..`) et vous éditez le fichier `qmail-scanner-queue.pl` et vous remplacez:

```
#!/usr/bin/suidperl -T
```

par:

```
#!/usr/bin/perl
```

Copiez enfin ce fichier dans `/var/qmail/bin` en placant les bons droits :

```
cp qmail-scanner-queue.pl /var/qmail/bin/
chown qmailq.qmail /var/qmail/bin/qmail-scanner-queue*
```

Dans tout les cas, il faut initialiser qmail-scanner maitenant.

Si vous n'avez pas eu l'erreur ci-dessus, tapez ces commandes:

```
qmail-scanner-queue.pl -g
qmail-scanner-queue.pl -z
```

Si vous avez eu l'erreur lors de l'installation, les commandes alors sont celles-ci:

```
qmail-scanner-queue -g
qmail-scanner-queue -z
```

Installation du gestionnaire de mailing list

Ezmlm est le gestionnaire de mailing de Qmail. Pour l'installer, détarrez / dgzippez ezmlm et ezmlm-idx.
Puis copiez le contenu de ezmlm-idx dans le répertoire de ezmlm:

```
tar zxvf ezmlm-0.53.tar.gz
tar zxvf ezmlm-idx-0.40.tar.gz
mv ezmlm-idx-0.40/* ezmlm-0.53/
```

Patchez alors ezmlm:

```
cd ezmlm-0.53
patch < idx.patch
```

et compilez, installez le tout:

```
make clean
make
make man
```

Configuration des service mails

Maitenant que le plus "gros" est installé, nous allons créer le script de démarrage de ces services.
Créez un fichier `rc.startmail` dans `/etc/rc.d` comprenant:

```
#!/bin/bash
```

```
export PATH="/usr/local/bin:/var/qmail/bin:/var/vpopmail/bin:/usr/local/bin/ezmlm:$PATH"
echo "Starting Qmail and Vpopmail daemons ..."
export QMAILQUEUE="/var/qmail/bin/qmail-scanner-queue.pl"
/var/qmail/rc &
/usr/local/bin/tcpserver -v -H -R -x /etc/tcp.smtp.cdb -c20 -u1033 -g103 0 smtp
\ /usr/local/bin/recordio /var/qmail/bin/qmail-smtpd 2>&1 >/dev/null &
/usr/local/bin/tcpserver -v -H -R 0 pop3 /usr/local/bin/recordio /var/qmail/bin/qmail-popup
\ mailhub.lea-linux.org /var/vpopmail/bin/vchkpw /var/qmail/bin/qmail-pop3d Maildir &
```

Remarque: Ne coupez pas les lignes de ce script. Regardez bien ce qui est marqué au dessus, le caractére "\" dans le texte au dessus signale qu'il s'agit de la même ligne que celle qui a commencé au dessus, le retour de ligne est du a la mise en page HTML. C'est à dire que si vous lisez:

```
tuc machin
\ bidule
```

vous devez lire une seulle ligne:

```
truc machin bidule
```

Il ne faut pas, bien sur copier, le "\".

Attention: Si vous avez eu l'erreur dans Qmail-scanner à propos des script SET-ID, modifiez la ligne comprenant `export QMAILQUEUE` par:
`export QMAILQUEUE="/var/qmail/bin/qmail-scanner-queue"`

Ajoutez alors dans votre `/etc/rc.d/rc.local` une ligne du type:

```
/etc/rc.d/rc.startmail
```

Ou si votre distribution est basé sur sysVinit,créez le scripts correspondants dans les niveaux d'init que vous souhaitez.

Reste à configurer quelques fichiers. Tout d'abord, il faut régler le temps d'autentification des utilisateurs par le pop (POP before SMTP). Pour cela, nous editons la contrab et donnons l'intervale de temps pu il faut nettoyer les identifications:

```
crontab -e
```

et placez une ligne du type:

```
40 * * * * /var/vpopmail/bin/clearopensmtp 2>&1 > /dev/null
```

Pour nettoyer toutes les 40 minutes.

Il faut définir aussi quel est votre adresse réseau, pour authoriser ce réseau àutiliser le serveur de mail. Prenons l'exemple où les utilisateurs de votre serveur de mails se trouvent sur les réseaux 192.168.0.0 (local) et 213.30.139.0 (public), il faut alors éditer le fichier /etc/tcp.smtp et y mettre:

```
127.:allow,RELAYCLIENT=""
192.168.0.:allow,RELAYCLIENT=""
213.30.139:allow,RELAYCLIENT=""
```

l'adresse 127 doit **absolument** y être, autrement le serveur ne peut pas fonctionner!

Construisez alors la base de données de ces adresses:

```
tcprules /etc/tcp.smtp.cdb /etc/tcp.smtp.tmp < /etc/tcp.smtp
```

Installation d'autoresponder

Ce programme permet de répondre automatiquement aux mails reçus (pour signaler une absence par exemple).
Pour l'installer, c'est très simple:

```
tar zxvf autorespond-2.0.2.tar.gz
cd autorespond-2.0.2
make
make install
```

Installation de Qmailadmin

Il nous reste plus qu'à installer l'interface cgi Web de la gestion des comptes mails.
Pour cela:

```
tar zxvf qmailadmin-1.0.2
cd qmailadmin-1.0.2
./configure --enable-htmldir=/var/www
```

```
make
make install
```

Attention: Remplacez `--enable-htmldir=/var/www` par le chemin de votre racine web.
Je suppose que vous savez configurer apache pour que le répertoire de cgi soit valide et fonctionnel.

Utilisation, question courantes, etc ...

Je vous donne ici brièvement les commande pour créer un nouveau domaine mails, les comptes, etc...
Je suppose bien sûr que vous savez gérez vos DNS pour que le champ MX du domaine pointe vers le serveur de mails.

- Ajouter un nouveau domaine que votre serveur de mails va héberger :

```
/var/vpopmail/vadddomain domaine.com
```

Vous allez devoir rentrer le mot de passe du "postmaster", c'est-à-dire de l'administrateur mails de ce domaine.

Pour créer les comptes pop, utiliser qmailadmin. Tapez l'url dans votre navigateur pour pointer sur le cgi de qmailadmin, dans user laissez "postmaster", dans domaine, mettez le domaine que vous voulez gérer et pour le mot de passe, mettez le mot de passe du postmaster.
Dans l'interface se trouvent tout les liens pour créer les mails etc...

- Effacer un domaine:

```
/var/vpopmail/vdeldomain domaine.com
```

Quand vous configurez votre client mail pour lire les mails reçus, faites attention dans la partie "nom utilisateurs" de mettre de la forme:

```
utilisateur@domaine.com
```

car les logiciels de mail mettent par défaut "utilisateur" tout court.

Si lors du démarrage des services vous avez des erreurs du style:

```
unable to bind: adresses in use
```

ou quelque chose de similaire, c'est que vous avez déjà un serveur de mail et/ou pop3 de lancé. Désinstallez tout autre serveur de mail/pop3 (sendmail, gnu-pop3d,...), vérifiez aussi dans .etc/inetd.conf ou dans /etc/xinet.d/ que les services smtp, pop3 ne sont pas utilisés par d'autres programmes.

Lisez les documentations d'utilisation de ezmlm, ezmlm-idx pour la gestion des mailing lists.

Qmailadmin vous permet de "fabriquer" vos propres pages d'administration mail, etc... avec de nombreux exemples installés dans /usr/local/share/qmailadmin

Pour que vous puissiez gérer les mails d'un domaine, vous devez configurer un MX dans le fichier de zone de ce domaine qui pointe vers votre serveur. Lisez les documentions des serveurs DNS.

Interface graphique de messagerie

par Laurent DUBETTIER−GRENIER

Installer une interface graphique de messagerie IMP

Introduction

Ce guide est basé sur l'excellent travail original en anglais de Olivier Schulze L, site web http://www.geocities.com/oliversl/imp/, pour distribution RedHat 9.

Dans ce document, nous allons décrire étape par étape les instructions nécessaires pour configurer une interface graphique, pour un serveur local de messagerie postfix, sur une distribution Mandrake Linux 9.1.

Vous devez auparavant récupérer les paquetages nécessaires à l'installation de l'interface web sur le site de Horde. Il vous faut HORDE 2.2.4 (Architecture PHP), IMP 3.2.2 (Client IMAP), TURBA 1.2 (Gestionnaire de contacts), KRONOLITH 1.1 (Calendrier/Organiseur journalier), MNEMO 1.1 (Gestionnaire de note), NAG 1.1 (Gestionnaire de tâche) et PEAR 1.1, ou leurs versions plus récentes. Le tout est diffusé sous licence libre GNU/GPL, sauf Horde qui est en licence Lesser GNU/GPL.

Si vous souhaitez autoriser le changement en ligne de mot de passe pour la boîte à lettres, ce guide décrit aussi l'installation de POPPASSD−CETI, et de PASSWD 2.2 (Gestionnaire de mot de passe).

En plus de Passwd, Il est aussi possible d'installer facilement les modules FORWARDS 2.2 (Redirection) et VACATION 2.2 (Répondeur automatique), le tout groupé dans le module ACCOUNTS 2.1 (Gestionnaire de Passwd, Forwards et Vacation). Cela n'est pas expliqué dans ce guide mais ne présente aucune difficulté particulière.

Ce guide vous aidera à configurer une installation standard, uniquement testée pour un réseau local, non relié à Internet. Cela doit vous permettre de réaliser un petit serveur de messagerie interne, pour une petite structure, avec une interface graphique ergonomique accessible depuis un navigateur Internet.

Il est ensuite possible de "tuner" votre configuration pour qu'elle corresponde à vos souhaits, ou/et pour réaliser un serveur de messagerie Internet (Le webmail de Free est basé sur horde/imp), mais cela sort du cadre de ce guide et de mes compétences : une attention particulière doit être notamment accordée à la sécurité.

Pré−requis

Pour suivre les instructions ci−dessous, vous aurez besoin d'un minimum de connaissances relatives à Linux, notamment sur l'usage de la console, si vous installez un serveur de messagerie minimum (sans interface graphique type KDE ou Gnome).
Nous utiliserons un serveur Mandrake Linux 9.1, d'adresse IP 192.168.10.1. Je suppose qu'Apache, Mysql, Php et Postfix sont installés et fonctionnent correctement. Sendmail est éventuellement utilisable à la place de Postfix.
Le nom de domaine du serveur est : `mail.example.com`
Vous pouvez vérifier le nom de domaine votre machine avec l'instruction : `hostname`
Les paquetages (et leurs dépendances) suivants doivent être présents :

- apache2−2.0.44−11mdk
- apache2−mod_php−2.0.44_4.3.1−2mdk
- php−pear−4.3.0−3mdk
- php−imap−4.3.0−3mdk
- php−ldap−4.3.0−3mdk
- php−mysql−4.3.0−2mdk
- php−xml−4.3.0−2mdk
- mysql−4.0.11a−5mdk
- mysql−client−4.0.11a−5mdk
- imap−2002a−2mdk
- postfix−2.0.6−1mdk
- tcp_wrappers−7.6−22mdk
- poppassd−ceti−1.8−3mdk
- xinetd−2.3.10−1mdk

Quelques rappels

- Pour visionner les paquetages installés sur votre système : `rpm −qa | less (Taper q pour quitter)`
  Si vous n'avez pas tout, il faut installer les paquetages manquants : `rpm −Fvh nom_du_paquetage.rpm pour mettre à jour un paquetage déjà installé`
  `rpm −ivh nom_du_paquetage.rpm pour installer un nouveau paquetage`
- Vous devez avoir les services httpd, mysql, postfix, xinetd activés. Pour visionner les services fonctionnant :
  `chkconfig −−list | less`
- Pour mettre en marche un service (exemple avec apache) :
  `service httpd stop (Arrêt d'Apache)`
  `service httpd start (Mise en marche d'Apache)`
  `service httpd restart (Redémarrage d'Apache)`
- Pour faire en sorte qu'à chaque démarrage de votre poste, le service Apache fonctionne :
  `chkconfig apache on`

- Pour avoir une documentation sur une instruction Linux :
  `man instruction (exemple : man chkconfig)`.
  A propos de la configuration des services, consulter également l'article sur la gestion des démons.
- Pour savoir où vous vous situez dans l'arborescence de linux : `pwd`
- Enfin, n'oubliez pas que l'appui sur la touche de tabulation "tab" permet de compléter un nom :
  `cd /v "tab" donne automatiquement cd /var`

Configurer Apache

Le répertoire d'installation sera : /var/www/html/mail/.

Lancer le service imap, dépendant de xinetd :

chkconfig imap on
service xinetd restart

Modifier php.ini

Modifier le fichier /etc/php.ini :

file_uploads = On
;short_open_tag = On # ligne à commenter
extension_dir = "/usr/lib/php/extensions"

Fichiers de configuration

Modifier le fichier /etc/httpd/conf/httpd.conf :

DocumentRoot /var/www/html/mail
ServerName mail.example.com

Modifier le fichier /etc/httpd/conf/commonhttpd.conf : `<directory /var/www/html/mail>`

Ajouter la ligne suivante dans le fichier /etc/hosts : `92.168.10.1 mail.example.com`

Remarque : une autre méthode consiste à configurer un **VirtualHost**.

Création répertoire

" mkdir −p /var/www/html/mail/

Tester Apache

Tester la configuration puis redémarrer Apache pour que les changements prennent effet :

service httpd configtest
service httpd restart

Créer un fichier temporaire : `touch /var/www/html/mail/essai.html`

Puis à l'aide d'un navigateur, aller à l'adresse Internet : `http://mail.example.com/essai.html`

Remarque : On peut utiliser un navigateur texte comme Lynx pour tester la configuration, si l'on n'a pas installé d'interface graphique sur le serveur ou si aucun poste distant n'est connecté au serveur :
`lynx mail.example.com/essai.html`

Configurer Horde

Une fois Apache configuré, vous avez besoin d'installer et de configurer Horde.

Installer Horde

A partir du répertoire où vous avez stocké les paquetages Horde :

tar zxvf horde−2.2.4.tar.gz −C /var/www/html/mail
cd /var/www/html/mail
mv horde−2.2.4 horde

Installer Pear

Si vous avez installé le paquetage rpm php−pear, il faut rajouter les modules Pear Date, HTML_Common et HTML_Select. Pour cela, télécharger ces paquetages et les décompresser dans un répertoire quelconque (tar zxvf nom_du_paquetage). Il faut ensuite copier le contenu de ces paquetages dans le répertoires /usr/share/pear en respectant l'arborescence suivante :

Paquetages Date :
mkdir /usr/share/pear/Date
/usr/share/pear/Date.php
/usr/share/pear/Date/Calc.php
/usr/share/pear/Date/Human.php
/usr/share/pear/Date/TimeZone.php
Paquetage HTML_Common :
mkdir /usr/share/pear/HTML
/usr/share/pear/HTML/Common.php
Paquetage HTML_Select :
/usr/share/pear/HTML/Select.php

Si vous n'avez pas installé php−pear, voici comment installer Pear à partir du paquetage original PEAR 1.1 :

tar zxvf pear−1.1.tar.gz −C /usr/share
cd /usr/share
mv pear−1.1 pear
chown root.root −R pear

Cette procédure sera alors à effectuer à chaque mise à jour de php.

Configurer MySQL

Tester si MySQL est lancé :

service mysql restart

Choisissez un mot de passe pour la base Horde :

vi mysql_create.sql # Définir le mot de passe pour l'accès à la base de données horde

Créer la base Horde :

cd /var/www/html/mail/horde/scripts/db
mysql −−user=root −−password < mysql_create.sql

Le mot de passe demandé est celui d'accès root à MySQL
Redémarrer MySQL : `# service mysql restart`

Tester si vous pouvez vous connecter à la base :

mysql −h localhost −D horde −u horde −p

Un mot de passe vous est demandé : rentrer celui choisi ci−dessus.
Quitter mysql avec la commande 'exit'

Modifier les fichiers de config Horde

cd /var/www/html/mail/horde/config

Renommer tous les fichiers :

for fichier in *.dist; do cp −v $fichier $(basename $fichier .dist); done

Modifier le fichier /var/www/html/mail/horde/config/horde.php :

```
// utiliser la compression. Mandrake PHP supporte zlib
$conf['compress_pages'] = true;
// utiliser IMAP pour authentifier les utilisateurs
$conf['auth']['driver'] = 'imap';
$conf['auth']['params']['dsn'] = '{localhost:143/imap}INBOX';
// utiliser MySQL pour stocker les données de Horde
$conf['prefs']['driver'] = 'sql';
$conf['prefs']['params']['phptype'] = 'mysql';
$conf['prefs']['params']['hostspec'] = 'localhost';
$conf['prefs']['params']['username'] = 'horde';
$conf['prefs']['params']['password'] = 'mon_mot_de_passe';
$conf['prefs']['params']['database'] = 'horde';
$conf['prefs']['params']['table'] = 'horde_prefs';
// Le SMTP postfix, pour envoyer les emails
$conf['mailer']['type'] = 'smtp';
```

Remarque 1 : mon_mot_de_passe est à remplacer par le mot de passe de la base horde mysql choisi au paragraphe Configurer mysql.
Remarque 2 : Si vous souhaitez utiliser sendmail à la place de postfix, il faut remplacer smtp par sendmail ci−dessus. Il faudra alors aussi ajouter dans

le fichier /etc/mail/trusted−users le mot apache, pour l'autoriser à envoyer des mails par l'intermédiaire de sendmail.

Modifier le fichier /var/www/html/mail/horde/config/lang.php :/p>

$nls['defaults']['language'] = 'fr_FR';

Tester Horde

Vous pouvez tester la configuration initiale de Horde en allant avec un

Puis essayer :

http://mail.example.com/horde/

VERIFICATION DE SECURITE
Vous ne devez pas pouvoir accéder avec un navigateur à : http://mail.example.com/horde/config/

Configurer IMP

Installer IMP

A partir du répertoire où vous avez stocké le paquetage IMP :

tar zxvf imp−3.2.2.tar.gz −C /var/www/html/mail/horde
cd /var/www/html/mail/horde
mv imp−3.2.2 imp

Configurer Horde pour IMP

Déclarer IMP dans Horde :
Modifier le fichier /var/www/html/mail/horde/config/registry.php :

```
// Décommentez les lignes suivantes
$this−>registry['auth']['login'] = 'imp';
$this−>registry['auth']['logout'] = 'imp';
// Montrer l'icône imp dans horde
$this−>applications['imp'] = array(
...
'status' => 'active'
);
```

Configurer les fichiers de config d'IMP

cd /var/www/html/mail/horde/imp/config

Renommer tous les fichiers :

for fichier in *.dist; do cp −v $fichier $(basename $fichier .dist); done

Modifier le fichier /var/www/html/mail/horde/imp/config/servers.php :

```
// Modifier ces lignes
$servers['imap'] = array(
'name' => 'IMAP Server',
'server' => 'localhost',
'protocol' => 'imap/notls',
'port' => 143,
'folders' => 'mail/',
'namespace' => '',
'maildomain' => 'example.com',
'smtphost' => 'mail.example.com',
'realm' => '',
'preferred' => ''
);
```

Remarque : Il doit être possible à ce niveau de configurer le service sécurisé d'imap (protocol imap/ssl, port 993).

Modifier le fichier /var/www/html/mail/horde/imp/config/prefs.php :

```
// langage de l'utilisateur
// regarder dans /horde/config/lang.php pour les alias de langue
$_prefs['language'] = array(
'value' => 'fr_FR',
'locked' => false,
```

```
'shared' => true,
'type' => 'select',
'desc' => _("Select your preferred language:")
);
// boite postale de l'utilisateur par défaut
// la valeur par défaut INBOX ne peut être changée
$_prefs['mailbox'] = array(
'value' => 'INBOX',
'locked' => true,
'shared' => false,
'type' => 'implicit'
);
// Utiliser IMAP
// Montre seulement les répertoires souscrits par IMAP
$_prefs['subscribe'] = array(
'value' => 1,
'locked' => true,
'shared' => false,
'type' => 'checkbox',
'desc' => _("Use IMAP folder subscriptions")
);
// Répertoire des courriers envoyés
// Utilise le même nom que Mozilla, Outlook Express, etc
$_prefs['sent_mail_folder'] = array(
'value' => 'Sent',
'locked' => false,
'shared' => true,
'type' => 'implicit'
);
// Répertoire poubelle
// Utilise le même nom que Mozilla, Outlook Express, etc
$_prefs['trash_folder'] = array(
'value' => 'Trash',
'locked' => false,
'shared' => false,
'type' => 'implicit'
);
```

Tester IMP

Vous pouvez tester la configuration initiale de IMP en allant avec un navigateur web sur : http://mail.example.com/horde/imp/test.php.

Se logger avec un utilisateur/passe valide (en l'occurrence un compte pop valide).
Essayer d'envoyer des mails à un autre compte et à vous même.
Utiliser une adresse mail du type utilisateur@mail.example.com Nous verrons plus loin (paragraphe 8) comment configurer postfix pour utiliser une adresse mail du type utilisateur@example.com

Pour créer un compte sans possibilité de connection sur votre serveur, uniquement destiné à la messagerie :

useradd −c "nom_utilisateur ou commentaire" −s /bin/false −g popusers nom_utilisateur

Puis définir le mot de passe du compte nouvellement créé avec passwd :

passwd nom_utilisateur

Remarque 1 : Avec un poste Windows distant, il faut configurer le fichier hosts. Dans le répertoire : C:\WINNT\SYSTEM32\DRIVERS\ETC), rajouter la ligne : 192.168.10.1 mail.example.com Ceci est probablement inutile si votre serveur de messagerie est aussi serveur de nom (DNS).

Remarque 2 : Vous pouvez aussi modifier le fichier 'trailer.txt' situé dans /horde/imp/config/. Ce fichier vous permet d'ajouter un texte à la fin de tous les mails qui seront envoyés par IMP. Par défaut, le texte suivant est ajouté à tous les messages : This mail sent through IMP: http://horde.org/imp/. Si vous ne voulez rien ajouter, effacer tout ce qu'il y a dans ce fichier.

VERIFICATION DE SECURITE
Vous ne devez pas pouvoir accéder avec un navigateur à :
`http://mail.example.com/horde/config/`

Configurer Turba

Maintenant que vous pouvez lancer Horde et IMP, vous avez besoin d'un carnet d'adresses afin de gérer vos contacts.

Installer Turba

A partir du répertoire où vous avez stocké le paquetage TURBA :

```
tar xzf turba−1.2.tar.gz −C /var/www/html/mail/horde
cd /var/www/html/mail/horde
mv turba−1.2 turba
cd /var/www/html/mail/horde/turba/config
```

Renommer tous les fichiers :

```
for fichier in *.dist; do cp −v $fichier $(basename $fichier .dist); done
```

Configurer Turba

Déclarer Turba dans Horde :
Modifier le fichier /var/www/html/mail/horde/config/registry.php :

```
// Montrer l'icône Turba dans horde
$this−>applications['turba'] = array(
...
'status' => 'active'
);
```

Configurer dans IMP l'icône pour Turba :
Modifier le fichier /var/www/html/mail/horde/imp/config/conf.php :

```
$conf['menu']['apps'] = array('turba');
```

Configurer dans Turba l'icône pour IMP :
Modifier le fichier /var/www/html/mail/horde/turba/config/conf.php :

```
$conf['menu']['apps'] = array('imp');
```

Modifier le fichier /var/www/html/mail/horde/turba/config/prefs.php :

```
// Language de l'utilisateur
// Règler le même langage par défaut que Horde et IMP
$_prefs['language'] = array(
'value' => 'fr_FR',
'locked' => false,
'shared' => true,
'type' => 'select',
'desc' => _("Select your preferred language:")
);
```

Configurer Turba pour utiliser MySQL pour enregistrer les données des contacts :
Modifier le fichier /var/www/html/mail/horde/turba/config/sources.php :

```
// Compléter cette partie de la configuration avec les données
// de la base de données comme dans /horde/config/horde.php
// Configurer aussi le titre dans votre langue
$cfgSources['localsql'] = array(
'title' => 'My Address Book',
'type' => 'sql',
'params' => array(
'phptype' => 'mysql',
'hostspec' => 'localhost',
'username' => 'horde',
'password' => 'mon_mot_de_passe',
'database' => 'horde',
'table' => 'turba_objects'
),
);
```

Remarque : mon_mot_de_passe est à remplacer par le mot de passe de la base horde mysql choisi au paragraphe Configurer mysql

Configurer MySQL pour Turba

Créer la base dans MySQL que Turba utilisera :

```
cd /var/www/html/mail/horde/turba/scripts/drivers/
mysql −−user=root −−password −D horde < turba.sql
```

Tester Turba

Vous pouvez tester la configuration initiale de Turba en allant avec un navigateur web sur :

http://mail.example.com/horde/

Créer une nouvelle entrée dans le carnet d'adresse et aller dans IMP, puis Options et choisissez d'utiliser le carnet d'adresses avec le nom "My Address Book".

Configurer poppassd (optionnel : non sécurisé)

Maintenant que vous avez configuré Horde avec IMP et Turba, vous avez besoin de fournir à vos utilisateurs une méthode pour changer leur mot de passe au travers de Horde.

Installer poppassd−ceti

rpm −ivh poppassd−ceti−1.8−3mdk.i586.rpm

Activer le service poppassd :

chkconfig poppassd on
service xinetd restart

Installer passwd

A partir du répertoire où vous avez stocké le paquetage PASSWD :

tar zxvf passwd.2.2.tar.gz −C /var/www/html/mail/horde/
cd /var/www/html/mail/horde/
mv passwd−2.2 passwd
cd /var/www/html/mail/horde/passwd/config

Renommer tous les fichiers :

for fichier in *.dist; do cp −v $fichier $(basename $fichier .dist); done

Configurer Horde pour Passwd

Modifier le fichier /var/www/html/mail/horde/config/registry.php :

$this−>applications['passwd'] = array(
'fileroot' => dirname(__FILE__) . '/../passwd',
'webroot' => $this−>applications['horde']['webroot'] . '/passwd',
'icon' => $this−>applications['horde']['webroot'] . '/passwd/graphics/lock.gif',
'name' => _("Password"),
'allow_guests' => false,
'status' => 'active'
);

Configurer IMP pour Passwd

Configurer l'icône Passwd dans IMP :
Modifier le fichier /var/www/html/mail/horde/imp/config/conf.php :

$conf['menu']['apps'] = array('turba', 'passwd');

Configurer le fichier de config de Passwd

Configurer l'icône pour IMP dans Passwd :
Modifier le fichier /var/www/html/mail/horde/passwd/config/conf.php :

$conf['menu']['apps'] = array('imp');

Editer le fichier /var/www/html/mail/horde/passwd/config/backends.php :

$backends['poppassd'] = array(
'name' => 'nom de votre organisation',
'preferred' => '',
'password policy' => array(),
'driver' => 'poppassd',
'params' => array(
'host' => 'localhost',
'port' => '106'
)
);

Tester Passwd

Vous pouvez tester la configuration initiale de Passwd en allant avec un navigateur web sur : http://mail.example.com/horde/passwd/

Regardez dans /var/log/messages pour les messages de débuggage.
Changez votre mot de passe. Attention : n'importe quel mot de passe de n'importe quelle longueur sera accepté car poppassd est lancé sous l'identité root.

Kronolith, Mnemo et Nag

Décompresser les paquetages et renommer chaque répertoire :

```
tar zxvf kronolith−1.1.tar.gz −C /var/www/html/mail/horde/
tar zxvf mnemo−1.1.tar.gz −C /var/www/html/mail/horde/
tar zxvf nag−1.1.tar.gz −C /var/www/html/mail/horde/
cd /var/www/html/mail/horde/
mv kronolith−1.1 kronolith
mv mnemo−1.1 mnemo
mv nag−1.1 nag
```

Installer Kronolith

Créer la base dans MySQL que Kronolith utilisera :

```
# cd /var/www/html/mail/horde/kronolith/scripts/drivers/
# mysql −−user=root −−password −D horde < kronolith.sql
```

Modifier le fichier /var/www/html/mail/horde/config/registry.php :

```
$this−>applications['kronolith'] = array(
'fileroot' => dirname(__FILE__) . '/../kronolith',
'webroot' => $this−>applications['horde']['webroot'] . '/kronolith',
'icon' => $this−>applications['horde']['webroot'] . '/kronolith/graphics/kronolith.gif',
'name' => _("Calendar"),
'allow_guests' => false,
'status' => 'active'
);
```

Configurer l'icône Kronolith dans IMP :
Modifier le fichier /var/www/html/mail/horde/imp/config/conf.php :

```
$conf['menu']['apps'] = array('turba','passwd','kronolith');
cd /var/www/html/mail/horde/kronolith/config
```

Renommer tous les fichiers :

```
for fichier in *.dist; do cp −v $fichier $(basename $fichier .dist); done
```

Configurer l'icône IMP dans Kronolith et les paramètres d'accès à la base MySQL :
Modifier le fichier /var/www/html/mail/horde/kronolith/config/conf.php :

```
// utiliser MySQL pour stocker les données de Kronolith
$conf['calendar']['driver'] = 'sql';
$conf['calendar']['params']['phptype'] = 'mysql';
$conf['calendar']['params']['hostspec'] = 'localhost';
$conf['calendar']['params']['username'] = 'horde';
$conf['calendar']['params']['password'] = 'mon_mot_de_passe';
$conf['calendar']['params']['database'] = 'horde';
$conf['calendar']['params']['table'] = 'kronolith_events';
$conf['menu']['apps'] = array('imp');
```

Remarque : mon_mot_de_passe est à remplacer par le mot de passe de la base horde mysql choisi au paragraphe Configurer mysql

Installer Mnemo

Créer la base dans MySQL que Mnemo utilisera :

```
cd /var/www/html/mail/horde/mnemo/scripts/drivers/
mysql −−user=root −−password −D horde < mnemo_memos.sql
```

Modifier le fichier /var/www/html/mail/horde/config/registry.php :

```
$this−>applications['mnemo'] = array(
'fileroot' => dirname(__FILE__) . '/../mnemo',
```

```
'webroot' => $this−>applications['horde']['webroot'] . '/mnemo',
'icon' => $this−>applications['horde']['webroot'] . '/mnemo/graphics/mnemo.gif',
'name' => _("Memos"),
'allow_guests' => false,
'status' => 'active'
);
```

Configurer l'icône Mnemo dans IMP :
Modifier le fichier /var/www/html/mail/horde/imp/config/conf.php :

```
$conf['menu']['apps'] = array('turba','passwd','kronolith','mnemo');
```
cd /var/www/html/mail/horde/mnemo/config

Renommer tous les fichiers :

```
for fichier in *.dist; do cp −v $fichier $(basename $fichier .dist); done
```

Configurer l'icône IMP dans Mnemo et les paramètres d'accès à la base MySQL :
Modifier le fichier /var/www/html/mail/horde/mnemo/config/conf.php :

```
// utiliser MySQL pour stocker les données de Mnemo
$conf['storage']['driver'] = 'sql';
$conf['storage']['params']['phptype'] = 'mysql';
$conf['storage']['params']['hostspec'] = 'localhost';
$conf['storage']['params']['username'] = 'horde';
$conf['storage']['params']['password'] = 'mon_mot_de_passe';
$conf['storage']['params']['database'] = 'horde';
$conf['storage']['params']['table'] = 'mnemo_memos';
$conf['menu']['apps'] = array('imp');
```

Remarque : mon_mot_de_passe est à remplacer par le mot de passe de la base horde mysql choisi au paragraphe Configurer mysql

Installer Nag

Créer la base dans MySQL que Nag utilisera :

```
cd /var/www/html/mail/horde/nag/scripts/drivers/
mysql −−user=root −−password −D horde < nag_tasks.sql
```

Modifier le fichier /var/www/html/mail/horde/config/registry.php :

```
$this−>applications['nag'] = array(
'fileroot' => dirname(__FILE__) . '/../nag',
'webroot' => $this−>applications['horde']['webroot'] . '/nag',
'icon' => $this−>applications['horde']['webroot'] . '/nag/graphics/nag.gif',
'name' => _("Tasks"),
'allow_guests' => false,
'status' => 'active'
);
```

Configurer l'icône Nag dans IMP :
Modifier le fichier /var/www/html/mail/horde/imp/config/conf.php :

```
$conf['menu']['apps'] = array('turba','passwd','kronolith','mnemo','nag');
```
cd /var/www/html/mail/horde/nag/config

Renommer tous les fichiers :

```
for fichier in *.dist; do cp −v $fichier $(basename $fichier .dist); done
```

Configurer l'icône IMP dans Nag et les paramètres d'accès à la base MySQL :
Modifier le fichier /var/www/html/mail/horde/nag/config/conf.php :

```
// utiliser MySQL pour stocker les données de Mnemo
$conf['storage']['driver'] = 'sql';
$conf['storage']['params']['phptype'] = 'mysql';
$conf['storage']['params']['hostspec'] = 'localhost';
$conf['storage']['params']['username'] = 'horde';
$conf['storage']['params']['password'] = 'mon_mot_de_passe';
$conf['storage']['params']['database'] = 'horde';
$conf['storage']['params']['table'] = 'nag_tasks';
$conf['menu']['apps'] = array('imp');
```

Remarque : mon_mot_de_passe est à remplacer par le mot de passe de la base horde mysql choisi au paragraphe Configurer mysql

Sécurité

Vérifier que les répertoires config de chaque module Horde sont inaccessibles à un navigateur web. Si ce n'est pas le cas, revoir la configuration de Apache.
Une fois l'installation terminée, lancer le script :

```
cd /var/www/html/mail/horde
sh scripts/set_perms.sh
```

Répondre 'apache' pour le groupe, et toutes les autres questions par oui (y).
Ce script va changer le propriétaire, le groupe et les autorisations de chaque fichier à l'intérieur du répertoire horde.

Configurer Postfix

En réseau local, lorsque machin@mail.example.com envoie un mail à truc@mail.example.com, l'adresse de réponse est machin@example.com. Postfix ne peut envoyer la réponse.

Pour que cela fonctionne correctement, il faut rajouter (ou décommenter) les lignes suivantes du fichiers /etc/postfix/main.cf :

```
myorigin = $mydomain
mydestination = $myhostname, localhost.$mydomain, $mydomain
```

Conclusion

Si vous avez suivi ce document jusqu'à ce point, vous disposez d'une interface web complète de messagerie, avec la possibilité de changer de mot de passe en ligne, et avec un gestionnaire d'adresse, de calendrier, mémos et tâches.

Toutes les réponses à vos questions en tant qu'utilisateur de cette interface sont probablement dans la FAQ utilisateur de horde.

En tant qu'administrateur, il vous est maintenant possible de configurer plus finement votre interface. Beaucoup d'informations intéressantes sont disponibles dans la FAQ administrateur de Horde. Et en cas de problème insoluble, après avoir consulté la documentation présente avec chaque paquetage (dans le répertoire /docs associé), le site Horde et les FAQ, il y a encore la possibilité de poser vos questions sur la mailing−list de Horde...

Client/Serveur VNC

par Jonesy

Installer, configurer et utiliser VNC.

Qu'est que c'est ?

Une petite définition, **VNC** signifie **Virtual Network Computing**.

Cela permet de se connecter à une machine distante de sa propre machine, comme telnet ou openSSH, mais à la différence que vous êtes sous l'interface graphique du système hôte, et que vous avez la main sur le clavier et la souris.

> Note de Jicé : On dit que VNC permet de "déporter" l'affichage d'une machine sur une autre.
> Sachez qu'entre 2 machines dotées de serveurs X, VNC ne sert à rien, car c'est une fonctionnalité native des serveurs X. Il suffit de lancer votre application sur la machine jouant le rôle de serveur en lui spécifiant sur quel écran elle doit s'afficher.
> Par exemple, je lance `xeyes` depuis la machine *taz* avec affichage sur l'écran 0 (par défaut) de la machine *papoune* :
> `[jice@taz jice] xeyes −display papoune:0`

VNC rend ceci possible quelque soit le système d'exploitation de la machine distante et/ou le système de votre machine.
C'est à dire, par exemple, vous êtes sous Windows, vous avez un serveur Linux sur lequel vous avez un compte, vous vous connectez au serveur Linux avec un client et, c'est comme si vous vous retrouviez sur le serveur avec votre environnement graphique préféré !

Pour information, il existe d'autres solutions pour utiliser l'interface graphique d'une machine distante.
Note de Jicé : allez Jonesy, dis−le que c'est similaire à ce que fait PCAnywhere® qui, lui, ne fonctionne qu'avec des serveurs et des clients Windows © Microsoft ;)

Le principe de fonctionnement

En fait, c'est un système client/serveur. D'un coté, il vous faut installer un serveur VNC et de l'autre un client VNC. Le client va interroger le serveur, qui va lui renvoyer une *image* de l'environnement graphique du serveur.

Bien sûr, il existe des clients/serveurs VNC pour différents systèmes d'exploitation : Windows (9x/NT/2000/CE), Linux, Solaris, Macintosh et DEC.
Dans notre exemple, le serveur sera à installer sur Linux et le client sur Windows. Bien entendu, vous installez la version Linux du serveur sur Linux et la version Windows du client sur Windows. Comme toutes les versions utilisent le même protocole de communication, cela ne pose aucun problème.

Installation

Les clients et les serveurs VNC sont sous licence GPL, mais vos coordonnées sont demandées pour vous laisser les télécharger. Sachez que certaines distributions fournissent un (ou plusieurs) package VNC.

Vous pouvez trouver le client et le serveur, fournis ensemble, ici :
http://www.uk.research.att.com/vnc/download.html

Ou allez voir sur vos CDs d'installation ou ici pour le RPM :
http://rpmfind.net/linux/rpm2html/search.php?query=vnc&submit=Search+...
et là pour le DEB :
http://packages.debian.org/stable/x11/vncserver.html

Installez votre RPM, DEB, tarball ou l'éxécutable. C'est tout bête, il n'y a rien de particulier à savoir, je vous le jure... ;−)

Important : Pour le serveur VNC sous Linux, surtout ne l'ajoutez pas dans les services à démarrer automatiquement ! Cela bloquerait la machine au cours du démarrage et vous seriez obligé d'aller le désactiver d'une facon ou d'une autre.
Lorsque que cela m'est arrivé, j'ai "mounté" la partition Linux concernée avec mon deuxième système Linux.
Note de Jicé : j'ai aussi eu ce problème, surtout quand j'utilisais Aurora (knterface graphique de boot). Il suffit en fait de démarrer sans Aurora (depuis je l'ai supprimé) en mettant `textboot` (heu... ou `boottext` ?) sur la ligne de commande du noyau (ex : `linux textboot` à l'invite LILO). Ensuite, on voit qu'où vient le problème (en général, c'est VNC qui demande un mot de passe la première fois, ce qui bloque Aurora).

Utilisation du serveur sous Linux

Il faut que chaque utilisateur lance le serveur VNC. Ou que l'administrateur le fasse pour eux. Un utilisateur peut très bien lancer VNC via une connexion telnet.

Lorsqu'on lance le serveur VNC, celui−ci vous retourne un numéro sous cette forme :

```
New 'X' desktop is pingouin.pingouin.fr:1

Starting applications specified in /home/jonesy/.vnc/xstartup
Log file is /home/jonesy/.vnc/pingouin.pingouin.fr:1.log
```

Ici, le numéro est 1. Gardez le bien, car vous en aurez besoin.

Pour lancer le serveur VNC
`vncserver`

Pour arrêter le serveur VNC
`vncserver -kill :X`
X étant le numéro renvoyé lorsque vous avez lancé le serveur.

La **première fois** que vous lancerez le serveur pour un utilisateur, il vous demandera un mot de passe, pour des raisons de sécurité évidentes, n'utilisez pas le mot de passe du compte Linux ! Et il créera sous le $HOME de l'utilisateur le répertoire `.vnc` avec différents fichiers. Dont **`xstartup`**, qu'il est intéressant de modifier afin de choisir votre environnement graphique et d'autres petites choses.

Chez moi, **`$HOME/.vnc/xstartup`** ressemble a ceci :

```
#!/bin/sh

xrdb $HOME/.Xresources
xsetroot -solid grey
rxvt -ls -title "$VNCDESKTOP Desktop" &
starticewm &
```

`rxvt -ls -title "$VNCDESKTOP Desktop" &` : Lance automatiquement un terminal rxvt à la connexion.
`starticewm &` : Je choisis **iceWM** comme environnement graphique.

Pour Gnome ou KDE, il suffit de modifier la dernière ligne comme ceci:

```
startgnome &
ou
startkde &
```

Pour que les modifications de **`xstartup`** soient prises en compte, il faut arrêter et relancer le serveur VNC.

Note : Je ne sais pas pour quelle raison, mais chez moi, VNC ne marche pas avec **KDE** !

Et bien entendu, il faut que l'interface graphique soit installée sous Linux, VNC ne va pas la simuler. ;-)

Utilisation du serveur sous Windows

Sous Windows, le serveur peut être lancé automatiquement au démarrage, mais il ne peut y avoir qu'un seul et unique utilisateur.
Donc si vous êtes plusieurs à vouloir vous connecter au serveur VNC, il faudra le faire sous le même nom d'utilisateur et le dernier à se connecter chassera sans préavis celui qui est déjà connecté !
VNCServer a l'air assez limité sous le système de Microsoft... ;-p C'est certainement dû au fait que l'interface graphique de Windows n'est pas un serveur graphique comme XFree sous Linux. Et aussi que Windows n'est pas réellement multi-utilisateurs.

Utilisation du client

Le client est très simple d'utilisation, lorsque vous le lancez, il vous demande le nom de la machine où est installé le serveur VNC. Supposons que notre serveur s'appelle "pingouin".
Il vous faut donc taper :
`pingouin:X`
X étant le numéro renvoyé lorsque vous avez lancé le serveur. Puis il vous demande le mot de passe.
A la place du nom de la machine, "pingouin", vous pouvez mettre son adresse IP, "`10.0.0.2:X`" par exemple.
Et voilà, vous vous retrouvez sous votre compte sur le serveur avec votre environnement graphique préféré !

Pour quitter le client, c'est comme d'habitude. Sachez tout de même, que si vous fermez votre session X avant de quitter le client, il vous faudra relancer le serveur VNC pour la prochaine fois.
Si vous ne fermez pas votre session X et que vous quittez le client VNC avec Netscape ouvert, la prochaine fois que vous vous connecterez à VNC, vous retrouverez votre Netscape toujours ouvert. Bien sûr, si la machine hôte avec le serveur n'a pas été arrêtée.

Note : N'oubliez pas, vous êtes sur le serveur ! C'est comme un telnet ou openSSH. Donc vous n'avez pas accès à votre disque dur local, CDROM ou lecteur de disquette. Sauf si le serveur y a accès via le réseau...

Remarques

Voici les problèmes que j'ai pu rencontrer à l'utilisation du client VNC sous Windows vers un serveur VNC Linux :

- La définition de l'écran est celle du serveur. Donc la taille de la fenêtre du client dépend de la configuration vidéo du serveur. Il est possible d'agrandir la fenêtre du client mais la perte de qualité est importante. La meilleure solution est donc d'avoir la même configuration vidéo sur le serveur et sur le client (votre machine).
- Pour je ne sais quelle raison, il est impossible de faire exécuter certains scripts Shell de facon automatique. Par exemple le `~/.bashrc`.
- La configuration du clavier n'est pas forcément correcte. Impossible de taper "ê", par exemple.

Voilà, c'est enfin terminé... ;-)

Le proxy Junkbusters

BRARD Emmanuel

Installer le proxy Junkbusters

Introduction

Junkbusters est un proxy http (web) qui bloque les bannières de publicité des sites web.

Il est facile à mettre en oeuvre, et est très pratique puisque vous ne perdez plus de temps à télécharger ces bannières.

Pré−requis

Vous pouvez trouver Junkbusters à cette adresse : http://www.junkbusters.com

Mise au Point

Un proxy est une "passerelle" entre vous et le web (pour le http).

Il gère les requêtes http et, pour Junkbuster, bloque les requêtes indésirables.

Installation

Téléchargez−le, puis décompressez−le où vous le voulez.

Compilez les sources (si vous n'avez pas l'exécutable) par

```
./configure &make &make install
```

Vous pouvez, si vous le souhaitez, effacer le fichier `junkbusters.exe`, puisque c'est la version pour Windows 9x.

Bien maintenant, on a 3 fichiers de configuration à écrire :

- `blockfile`
- `cookiefile`
- `config`

On va commencer par `cookiefile` puisqu'on veut recevoir TOUS les cookies.

Ouvrez votre éditeur et mettez−y :
*
Enregistrez votre fichier en `cookiefile`.

Maintenant `config` :

```
listen−address localhost:8080          # machine locale port 8080
blockfile /etc/junkbusters/blockfile   # où est le fichier blockfile
cookiefile /etc/junkbusters/cookiefile # où est le fichier cookiefile
```

Comme vous le voyez les fichiers sont tous dans `/etc/junkbusters` ; créez ce repertoire et déposez y les fichiers que vous avez créés.

Le dernier c'est `blockfile`, c' est une liste de sites (là où sont les bannières) à bloquer. Ecrire ce fichier serait ridicule puisqu'on le trouve tout prêt sur internet.

Evidemment vous devrez voir comment il marche pour éventuellement ajouter des sites. Je vous en laisse un en téléchargement : cliquer ici.

enregistez le aussi dans `/etc/junkbusters`.

Voilà, maintenant il faut qu'il se lance automatiquement à chaque boot de Linux.

Pour cela éditez `/etc/rc.d/rc.local`, et ajoutez ces lignes :

```
echo junkbusters loadling...
/usr/sbin/junkbusters /etc/junkbusters/config
```

Pour que ça marche copiez l'executable `junkbusters` dans `/usr/sbin`.

Bien, c'est presque fini, il faut juste dire à votre navigateur d'utiliser le proxy.

Pour Netscape:
Ouvrez les péférences, et choisissez "proxy", Puis configuration manuelle du proxy.
Entrez `localhost` pour la machine, et 8080 pour le port.

Si vous utilisez un autre navigateur, réglez le de façon à utiliser le proxy de la même façon.

Voilà c'est tout, pour vérifier que Junkbusters fonctionne, rebootez votre machine, et ouvrez Netscape sans être connecté ; tapez une adresse, et regardez si vous voyez le message de Junkbusters, tout marche !

c' est bon, les pubs seront filtrées !!!

Mur pare feu pas à pas

par Fred

Ce document explique comment constuire un mur pare feu, pas à pas, sans théorie, sans blabla.

Introduction

Il n'est pas question d'expliquer ici le fonctionnement d'`iptables`, il existe pour cela de très bons articles dont celui de Léa. Le mur pare feu que je vous propose de construire sera adapté à vos besoins. Il ne comportera aucun gadget, aucune optimisation du réseau. Ce sera un mur pare feu rien qu'un mur pare feu.

On commence

Vous devrez mettre dans le fichier `/usr/bin/startfirewall` tout ce qui va suivre.

Comme tout script, le script de notre mur pare feu doit commencer par :

```
#!/bin/sh
```

Mais, pour être facilement modifiable, nous allons définir en plus quelques variables :

```
# (suite du script...)
# mettez ici l'emplacement d'iptables :
IPTABLES=/sbin/iptables
# mettez ici le nom de l'interface réseau vers internet :
EXTERNAL_IF="ppp0"
# mettez ici le nom de l'interface réseau vers votre lan :
INTERNAL_IF="eth0"
```

Ensuite, il nous faut charger les modules dont nous aurons besoin :

```
# (suite du script...)
# si vous voulez pouvoir autoriser les connexions ftp :
modprobe ip_conntrack_ftp
# si vous voulez pouvoir autoriser le DCC sous IRC :
modprobe ip_conntrack_irc
```

Suivant votre configuration, il peut être nécessaire de charger un autre module, pour en avoir la liste :

```
ls /lib/modules/`uname −r`/kernel/net/ipv4/netfilter/
```

Politique par défaut

Un mur pare feu correctement configuré se doit de rejeter tout ce qui n'a pas été explicitement autorisé. C'est le rôle de la politique par défaut (`default policy`). Pour fixer celle−ci, nous utilisons les 3 règles suivantes :

```
# (suite du script...)
$IPTABLES −P INPUT DROP
$IPTABLES −P OUTPUT DROP
$IPTABLES −P FORWARD DROP
```

La dernière règle peut être omise si vous souhaitez tout transmettre [aux]/[depuis les] machines de votre réseau local.

Les règles locales

Pour ne pas être ennuyé, il faut tout autoriser pour ce qui est du traffic réseau local (sur '`lo`' et '`$INTERNAL_IF`' aka: '`eth0`') :

```
# (suite du script...)
# "On accepte le traffic sur 'lo'"
$IPTABLES −A INPUT −i lo −j ACCEPT
$IPTABLES −A OUTPUT −o lo −j ACCEPT
$IPTABLES −A FORWARD −i lo −j ACCEPT
$IPTABLES −A FORWARD −o lo −j ACCEPT

# "On accepte le traffic sur le réseau local"
$IPTABLES −A INPUT −i $INTERNAL_IF −j ACCEPT
$IPTABLES −A OUTPUT −o $INTERNAL_IF −j ACCEPT
$IPTABLES −A FORWARD −i $INTERNAL_IF −j ACCEPT
$IPTABLES −A FORWARD −o $INTERNAL_IF −j ACCEPT
```

Suivre son mur pare feu

Ceci n'est pas vraiment obligatoire.

Si vous ne regardez pas régulièrement les logs de votre mur pare feu, celui−ci a toutes les chances de devenir inefficace. Nous allons donc 'logguer' une partie des connexions que nous refuserons :

```
# (suite du script...)
# On loggue les packets DROPés
$IPTABLES −A LOG_DROP −j LOG −−log−prefix "[IPT] "
$IPTABLES −A LOG_DROP −j DROP
```

Le texte `[IPT]` peut être remplacé par n'importe quel texte de votre choix. Vous pouvez, de la même façon, créer plusieurs cibles '`DROP`' pour les logguer différemment.

Partager la connexion

Le mur pare feu peut aussi servir à partager la connexion, pour cela il faut faire deux choses :

1. l'autoriser au niveau du noyau :

   ```
   # (suite du script...)
   echo 1 > /proc/sys/net/ipv4/ip_forward
   ```

2. cacher les autres machines du réseau local derriére le mur pare feu :

   ```
   # (suite du script...)
   $IPTABLES −A POSTROUTING −t nat −o $EXTERNAL_IF −j MASQUERADE
   ```

Autoriser des connexions

A partir de maintenant je vais vous donner des recettes de cuisine.

Tout d'abord, il faut savoir que vous pouvez au choix utiliser un numéro de port ou son nom dans le fichier `/etc/services`. J'utiliserais, dans la mesure du possible, cette dernière solution. Je ne détaillerais pas tous les ports, si vous souhaitez utiliser un port que j'aurais omis, consultez `/etc/services`. Dans ce fichier, vous constaterez que certains ports sont indiqués comme utilisant le protocol `tcp`, `udp` ou autre. Pour autoriser ces protocols vous devrez changer le "`−p tcp`" par "`−p udp`" ou autre dans les exemples qui vont suivre.

Cas général : réseau local vers internet

Pour la plupart des connexions, tout ce que je vais dire ici s'applique (exceptions notables : l'irc/dcc, le ftp/actif).

Pour autoriser les machines de votre réseau local (si le masquerading est activé) ainsi que votre serveur mur pare feu à se connecter à un port sur internet, il faut procéder de la maniére suivante (dans l'exemple j'autorise la connexion au WEB : `www`):

```
# (suite du script...)
$IPTABLES −A INPUT −i $EXTERNAL_IF −p tcp −−sport www −m state −−state ESTABLISHED,RELATED −j ACCEPT
$IPTABLES −A OUTPUT −o $EXTERNAL_IF −p tcp −−dport www −m state −−state NEW,ESTABLISHED,RELATED −j ACCEPT
```

Chaque commande `$IPTABLES` doit être tapée sur une seule ligne depuis le `$IPTABLES` jusqu'au `−j ACCEPT` ou `−j DENY` (dans la suite...).

Une petite explication, la règle `$IPTABLES −A INPUT` s'applique au traffic entrant (tous les paquets IP entrant seront soumis à cette règle). Comme on a précisé `−p tcp`, la règle s'applique aux paquets utilisants le protocol `tcp`. On a aussi préciser `−−sport www`, donc la règle s'applique au paquet qui proviennent (`sport` est mis pour `source port`, le port d'entrée) du port `www` (le web quoi !). Puis on a mis `−m state −−state ESTABLISHED,RELATED` pour préciser que cette règle ne s'applique qu'aux paquets IP qui proviennent soit d'une liaison "établie" (`ESTABLISHED`), soit d'une liaison en relation (`RELATED`) avec une liaison déjà établie. Enfin on précise `−j ACCEPT` pour dire qu'on accepte tous les paquets qui vérifient toutes ces conditions : les paquets venant d'une liaison WEB déja établie.

La seconde règle est symétrique de la première : elle autorise la paquets IP à sortir (`−A OUTPUT`), sauf qu'elle précise en plus que les paquets sortants ont le droit d'initier une nouvelle connexion (`−−state NEW`). Sinon la premiére règle aurait trés peu de chance de fonctionner : aucune liaison ne serait jamais 'établie'.

Pour autoriser plusieurs ports en même temps, on peut soit taper plusieurs règles comme la précédente (une par port) ou alors utiliser la syntaxe (exemple pour le http et le https):

```
# (suite du script...)
$IPTABLES −A INPUT −i $EXTERNAL_IF −p tcp −m multiport −−sports www,https −m state −−state ESTABLISHED,RELATED −j ACCEPT
$IPTABLES −A OUTPUT −o $EXTERNAL_IF −p tcp −m multiport −−dports www,https −m state −−state NEW,ESTABLISHED,RELATED −j ACCEPT
```

Cas général : internet vers réseau local

Pour autoriser les machines d'internet à se connecter à votre serveur local (dans l'exemple j'autorise la connexion à votre serveur WEB : `www`):

```
# (suite du script...)
$IPTABLES −A OUTPUT −o $EXTERNAL_IF −p tcp −−sport www −m state −−state ESTABLISHED,RELATED −j ACCEPT
$IPTABLES −A INPUT −i $EXTERNAL_IF −p tcp −−dport www −m state −−state NEW,ESTABLISHED,RELATED −j ACCEPT
```

Si vous voulez autoriser une connexion d'internet vers l'un des serveurs de votre réseau local à la place du serveur qui joue le rôle de mur pare feu, alors il faut faire du `port forwarding` en utilisant les cibles `SNAT` et `DNAT` en plus d'`ACCEPT`, mais cela dépasse le cadre de cet article.

Cas particuliers

Il existe des cas particuliers, ce sont les protocoles qui ouvrent plusieurs liaisons en même temps : le DCC(pour irc), le ftp passif, les protocole vidéo (comme le h323). Ils fonctionnent souvent (mais pas tous) sur le principe suivant : le client (vous) réclame quelque chose (un fichier) au serveur (le serveur ftp) ; celui−ci répond à la demande en ouvrant une nouvelle connexion (aléatoire en général : c'est ce qui nous pose problème, on ne sait pas quel port ouvrir) réseau sur le client (vous). Pour ne pas ouvrir n'importe quel port (plus précisément, pour ne pas ouvrir TOUS les ports) pour être en mesure de répondre à la demande du serveur, il faut que nous soyons capable de "tracer" (to track en anglais) la provenance de la demande de connexion pour être sur que la demande de connexion provient d'une liaison que nous avons nous méme initié. C'est le rôle du module "`ip_conntrack`" et de ses acolytes : "`ip_conntrack_ftp`" (pour le ftp) et "`ip_conntrack_irc`" pour l'irc. Pour que cela (ce qui va suivre) fonctionne, il faut donc que ces modules soient chargés. C'est ce que nous faisons au début du script, donc plus besoin de nous en occuper, si ce n'est pour autoriser les connexions qui sont en relation avec celles déja établies sur les ports qui sont utilisés par irc et ftp :

```
# (suite du script...)
$IPTABLES −A INPUT −i $EXTERNAL_IF −p tcp −−sport 1024:65535 −−dport 1024:65535 −m state −−state ESTABLISHED,RELATED −j ACCEPT
$IPTABLES −A OUTPUT −o $EXTERNAL_IF −p tcp −−sport 1024:65535 −−dport 1024:65535 −m state −−state ESTABLISHED,RELATED −j ACCEPT
```

Voilà, c'est tout, le ftp passif doit maintenant pouvoir bien passer votre mur pare feu.

Par contre pour le DCC, il faut encore ajouter une règle, autoriser les liaisons sortantes à initier une connexion. Je ne vois pas pourquoi, mais chez moi c'est nécessaire ;−) :

```
# (suite du script...)
$IPTABLES −A OUTPUT −o $EXTERNAL_IF −p tcp −−sport 1024:65535 −−dport 1024:65535 −m state −−state NEW −j ACCEPT
```

Pour le DCC, j'ai comme l'impression que le module 'ip_conntrack_irc' a du mal à suivre les connexions et c'est pour ça qu'il faut ajouter la troisième régle. Ce n'est pas une trop grande faille de sécurité puisque c'est nous qui initions la connexion, mais tout de même ce serait mieux si ce n'était pas obligé. Si quelqu'un trouve mieux, je serais content qu'il m'en fasse part.

Le ping !

Pour l'instant, si vous avez ouvert quelques port, vous avez du remarquer que vous n'avez plus accès au ping. Nous allons remédier à cela :

```
# (suite du script...)
$IPTABLES −A OUTPUT −p icmp −m state −−state NEW,RELATED,ESTABLISHED −j ACCEPT
$IPTABLES −A INPUT −p icmp −m state −−state NEW,RELATED,ESTABLISHED −j ACCEPT
```

ou alors, vous pouvez vouloir limiter les ping entrant (pour éviter d'être floodé) et vous remplacez la seconde règle par les deux suivantes :

```
# (suite du script...)
$IPTABLES −A INPUT −p icmp −m state −−state RELATED,ESTABLISHED −j ACCEPT
$IPTABLES −A INPUT −p icmp −m state −−state NEW −m limit −−limit 10/min −j ACCEPT
```

Vous pouvez remplacer `10/min` par `1/s` etc...

Envoyer une requête entrante vers un autre PC

Cette pratique s'appelle le 'port forwarding'.

Vous aurez besoin du port forwarding si vous avez un serveur qui ne fonctionne pas sur le poste qui partage la connexion. Par exemple, si le poste qui a un serveur http a pour IP 192.168.0.3, on utilisera cette commande pour que les requetes reçues lui soient automatiquement transmises :

```
iptables −t nat −A PREROUTING −p tcp −−dport 80 −j DNAT −−to−destination 192.168.0.3:80
```

Cela vous sera aussi utile si vous voulez héberger un serveur de jeu (crack−attack, freeciv, etc) sur une autre machine que le mur de feu qui peut, de la sorte, rester une machine de faibles capacités.

Si ça ne vous suffit pas réalisation d'une zone démilitarisée (DMZ) sera sans doute nécessaire (mais cela sort du cadre de cet article).

Quels ports autoriser ?

Il est un port que vous êtes obligé d'autoriser dans le sens 'réseau local' vers 'internet' : le port "`domain`" (aka: 53) en `udp` ET `tcp`. Sinon vous serez dans l'impossibilité d'utiliser la résolution de nom (aka: l'identification d'un nom de domaine avec son IP).

Sinon, moi j'autorise, dans le sens 'reseau local' vers 'internet' en `tcp` :

- domain (obligatoire),
- ftp,

- ftp−data,
- www,
- https,
- pop−3,
- imap2,
- imap3,
- smtp,
- ircd,
- cvspserver,
- rsync,
- 7070 (realaudio),
- 11371 (keyserver),
- ssh,
- 1441 (flux ogg de radio france)

Et en `udp` :

- domain (obligatoire),
- 6970 et 7170 (realaudio)

Et dans le sens : 'internet' vers 'reseau local' en `tcp` :

- auth (accélère l'établissement de la connexion à plein de serveurs irc)

Rappel : tous ces noms de 'ports' se trouvent dans le fichier `/etc/services`

Fin de script

Pour finir, on loggue tout via `syslogd` :

```
$IPTABLES −A FORWARD −j LOG_DROP
$IPTABLES −A INPUT −j LOG_DROP
$IPTABLES −A OUTPUT −j LOG_DROP
```

Arrêter le mur pare feu

Il n'est en général pas nécessaire d'arrêter le mur pare feu, mais, sait−on jamais ? Pour faire les tests, il est nécessaire de l'arrêter et de le redémarrer (parce que l'ordre des règles est important). Voici un petit script pour arrêter le mur pare feu (celui qui est proposé ici, il ne fonctionnera pas avec tous les murs de feu) :

```
#!/bin/sh
# Script d'arrêt du mur pare feu
# mettez ici l'emplacement d'iptables :
IPTABLES=/sbin/iptables

echo "On vide toutes les régles."
$IPTABLES −F INPUT
$IPTABLES −F OUTPUT
$IPTABLES −F FORWARD
$IPTABLES −t nat −F POSTROUTING
$IPTABLES −F LOG_DROP
$IPTABLES −X
$IPTABLES −P INPUT ACCEPT
$IPTABLES −P OUTPUT ACCEPT
$IPTABLES −P FORWARD ACCEPT

echo "On décharge les modules."
rmmod `lsmod | grep −E "^ip" | cut −d" " −f 1`
```

Conclusion

Ce mur pare feu n'est certainement pas parfait, mais j'espere qu'il vous permettra de mieux comprendre comment sont fabriqués les murs de feu que vous trouvez dans des scripts tous faits.

Installation de l'IDS SNORT

par julien Lecubin

Introduction

Ce document va tenter d'expliquer les différentes étapes pour mettre en place le détecteur d'intrusions SNORT à partir des sources. Un détecteur d'intrusions s'appelle aussi "IDS" pour Intrusion Detection System. SNORT est un système de détection d'intrusions réseau en OpenSource, capable d'effectuer l'analyse du trafic en temps réel. On l'utilise en général pour détecter une variété d'attaques et de scans tels que des débordements de tampons, des scans de ports furtifs, des attaques CGI, des scans SMB, des tentatives d'identification d'OS, et bien plus.

Avant de commencer l'installation de SNORT, vous devez avoir installé :

PACKAGES	REMARQUES
`MySQL`	La base de données MySQL
`MySQL-client`	La partie cliente de mysql (connexion BD)
`php-mysql`	le module php de mysql
`Apache`	Le serveur web Apache
`mod_php`	Le module php pour Apache
`libpcap/libpcap0-devel`	Librairie utilisée par SNORT pour capturer les paquets (rpm téléchargeable sur rpmfind.net)
`gcc`	indispensable pour compiler les sources de SNORT

Si vous n'avez pas encore installé le trio Apache/PHP/MySQL, il y a un article sur Lea vous expliquant comment le faire. C'est ici.

Les étapes pour l'installation de SNORT sont les suivantes :

- Installation de l'outil SNORT
- Installation des règles SNORT
- Liaison Mysql et SNORT
- Mise en place de ACID (Interface php pour visualiser les logs SNORT)

Installation de SNORT

Téléchargez la dernière release de SNORT à l'adresse suivante : http://www.SNORT.org/dl . La compilation de ce programme reste traditionnelle :

COMMANDES	REMARQUES
`cd /usr/local/snort`	...
`tar -xvzf SNORT-1.9.*.tar.gz`	Décompacte l'application
`./configure --with-mysql=/usr/lib/mysql`	Retirez l'argument `--with-mysql` si vous ne souhaitez pas rediriger les logs SNORT vers une base de données mysql *
`make`	Compilation
`make install`	Installation

Pour l'argument `--with-mysql`, vous pouvez l'adapter si vous utilisez une base de données autre que MySQL :

- `--with-odbc=$PATH_ODBC` : pour une base de données Microsoft SQL server
- `--with-postgresql=$PATH_POSTGRE` : pour une base PostegreSQL
- `--with-oracle=$ORACLE_HOME` : pour une base de données Oracle.

Installation des règles SNORT

Maitenant, il faut télécharger les règles de SNORT. En effet, SNORT utilise des règles pour détecter les intrusions. Il existe aujourd'hui environ 1200 règles différentes. Ces règles se caractérisent par un ensemble de fichiers (ftp.rules, p2p.rules,telnet.rules etc...). Vous devez téléchargez les sources de ces règles à l'adresse suivante :
http://www.SNORT.org/dl/signatures

Créez le répertoire de configuration SNORT, et installez-y les règles :

COMMANDES	REMARQUES
`mkdir /etc/snort`	Création du répertoire contenant la configuration SNORT
`cp /usr/local/snort*/etc/snort.conf /etc/snort`	Copie du fichier de config snort dans /etc/snort
`cp snortrules.tar.gz /etc/snort`	Mise en place des règles dans le répertoire de configuration SNORT
`cd /etc/snort`	On se place dans le répertoire de configuration SNORT
`tar -xvzf snortrules.tar.gz`	Décompactage des règles

Les règles SNORT sont alors placées dans le répertoire `/etc/snort/rules`.

Maintenant, Il faut éditer le fichier de configuration snort (`/etc/snort/snort.conf`) et spécifier le réseau sur lequel l'IDS travaille. Il faut pour cela modifier la variable `HOME_NET` :

```
var HOME_NET [10.1.1.0/24] # SNORT travaille sur le réseau 10.1.1.0

var HOME_NET (10.1.1.0/24,192.168.1.0/24] # Si votre carte réseau possède 2 alias
```

Dans le fichier de configuration de SNORT (`/etc/snort/snort.conf`), vous avez toute une série de include. Il s'agit des règles utilisées par SNORT pour détecter d'éventuelles intrusions. Il y a des règles de telnet, ICMP, FTP, ... Bref, commentez celles que vous ne voulez pas et décommentez celles qui vous parait utile. Conseil : Décommentez les règles ICMP, car elles ne cessent pas de vous remonter des alarmes très souvent inutiles.

Pour des explications plus détaillées concernant les règles SNORT, allez voir ici.

Lancement de SNORT

Deux possibilités s'offrent à nous. Soit vous lancez SNORT tout seul, et dans ce cas, il génerera ces logs dans un fichier plat. Soit vous décidez de l'interfacer avec une base de données. Suivant le cas, SNORT ne se lancera pas de la même façon.

Sans Mysql :
```
/usr/local/snort*/src/snort -c /etc/snort/snort.conf -i eth0 -D
```

Avec Mysql :
```
/usr/local/snort*/src/snort -c /etc/snort/snort.conf
```

Remarque : Si vous souhaitez interfacer SNORT avec une base de données, ne lancez pas SNORT avec l'argument `-L` qui spécifie l'emplacement des logs.

Lier les logs SNORT avec MySQL

Maintenant, nous allons éditer le fichier de configuration de SNORT afin de lui indiquer qu'il faut rediriger les logs dans une base de données (ici MySQL). Avec vos yeux de lynx, retrouvez la ligne suivante dans le fichier de configuration SNORT `/etc/snort/snort.conf` :

#output database:log,mysql,user=root password=test dbname=SNORT host=localhost

Décommentez et modifiez cette ligne par :

output database:log,mysql,user=user_snort password=snort_pwd dbname=snort host=localhost

Ici, l'utilisateur MySQL accédant à la base de données s'appelle "user_snort", son password associé est "snort_pwd", le nom de la base MySQL utilisée par snort est "snort" et la machine qui fait tourner la base Mysql est la même que celle où SNORT tourne.

Création de la base de données SNORT

Au préalable, assurez-vous d'avoir installé :

PACKAGES	**REMARQUES**
`MySQL-client-*`	partie cliente de MySQL
`MySQL-devel-*`	

Astuce : La commande "rpm -qa | grep client" vous permet de vérifier que votre station Linux possède bien ces packages installés.

Suivez alors les instructions suivantes :

COMMANDES	**REMARQUES**
`cd /usr/local/snort*/contrib`	on se place à l'endroit du fichier contenant les tables SQL de SNORT
`mysql -u root -p`	Connexion à la base de données en tant qu'administrateur (au passage, si ce n'est pas encore fait, définissez un password pour l'administrateur de la base par la commande 'set password for root@localhost=PASSWORD('totomdp');
`create database SNORT;`	Création de la base de données SNORT
`use mysql;`	On se place ici pour créer l'utilisateur MySQL qui gérera la base de données snort
`insert into user values('localhost', 'user_snort', password('snort_pwd'), 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', '', '', '', '', 'Y', 'Y', 'Y');`	Création utilisateur MySQL "user_snort". Attention le nombre de 'Y' dépend de votre version de MySQL. (faites un `select * from user;` pour voir combien il faut en mettre)
`grant ALL PRIVILEGES ON SNORT.* TO user_snort@localhost IDENTIFIED BY 'snort_pwd' WITH GRANT OPTION;`	Attribution des droits de la base "snort" à l'utilisateur "user_snort"
`flush privileges;`	Recharge les tables de droits pour prendre en compte les nouvelles modifications
`use snort;`	on se place dans la base où l'on veut créer les tables pour SNORT

`Source create_mysql`	Création des tables pour SNORT

Vérifiez que les tables sont bien créées. Allez voir dans `/var/lib/mysql/snort` et vous y verrez tout un tas de fichiers correspondant au nom des tables de la base de données SNORT (il doit y avoir 3 fichiers par tables).

Lancez SNORT. Désormais, SNORT envoie les informations dans la base de données (astuce : installez PhpMyAdmin, et vérifiez la taille de la base de données SNORT. Si tout fonctionne, vous la voyez augmenter si bien évidemment il y a du trafic !).

Installation/Configuration ACID

ACID est une interface PHP qui permet de visualiser les remontées d'alarmes générées par SNORT. Cette partie sous−entend que vous avez une base de données qui récupère les informations envoyées par SNORT. Avant de suivre l'installation de cette application, assurez−vous d'avoir téléchargé :

- Adodb : Contient des scripts PHP génériques de gestion de bases de données. L'installer dans la racine d'apache (`/var/www/html/adodb par exemple`)
- PHPlot : librairie de scripts PHP utilisée par ACID pour présenter graphiquement certaines données statistiques (optionnel)

Le téléchargement de ACID se fait ici. Imaginons que la racine de votre serveur web est /var/www/html. Installez ACID dans la racine d'apache :

COMMANDES	REMARQUES
`cd /var/www/html`	Placez−vous dans la racine du serveur web
`tar −xvzf acid*`	Décompactage de ACID
`tar −xvzf adodb*`	Décompactage de AdoDB
`tar −xvzf phplot*`	Décompactage de PHPlot
`vi /var/www/html/acid/acid_conf.php`	Renseignez les champs suivants : ♦ `$DBlib_path="../adodb";` ♦ `$Chartlin_path="../phplot";` ♦ `alert_dbname="snort"` ♦ `alert_host="localhost"` ♦ `alert_user="user_snort"` ♦ `alert_password="snort_pwd"`

Voilà, maintenant vous pouvez vérifier que ACID est bien configuré (allez voir sur http://localhost/acid). Si vous le souhaitez, L'accès peut se faire via certificat SSL de manière à crypter l'échange entre vous et le détecteur d'intrusions.

Sachez que ce document a pour but de vous apporter quelques éléments de réponse concernant l'installation et la configuration de l'IDS SNORT. Il est loin d'être parfait. Vos remarques sont les bienvenues. Je prévois de modifier le présent document suivant les remarques que vous y apporterez.

Pour me contacter : guitarparts chez fr point st

SmokePing

par Dimitri Clatot

Supervision de réseaux avec SmokePing

SmokePing

Présentation

SmokePing permet de mesurer et d'afficher sous forme graphique les temps de réponse aux ping, http, https, smtp d'une machine ou d'un groupe de machines, et de déclencher des alertes si une machine ne répond pas aux tests.
Cet article se veut seulement être un moyen de mise en place rapide de cet outil et non un substitut à la documentation officielle.

Installation des prérequis de SmokePing

Voici la liste de ces programmes :

- RRDTool 1.0.x
- FPing 2.4b2
- echoping
- Apache
- Perl 5.6.1
- SpeedyCGI

Je vais détailler l'installation de tous ces programmes à l'exception d'Apache et de Perl qui sont fournis généralement avec votre distribution. Il est à noter que les programmes FPing IPV6 et le module perl Socket6 seront à installer si vous souhaitez gérer l'IPV6 (non développé ici).

Installation de RRDtool

RRDtool permet de stocker les résultats des mesures faites et de les afficher sous forme graphique :

```
$ wget http://people.ee.ethz.ch/~oetiker/webtools/rrdtool/pub/rrdtool−1.0.42.tar.gz
$ tar zxvf rrdtool−1.0.42.tar.gz
$ cd rrdtool−1.0.42
$ ./configure
$ make
$ su
Password:
# make install
# ln −s /usr/local/rrdtool−1.0.42 /usr/local/rrdtool
```

Installation de FPing

FPing est une version améliorée de la commande ping :

```
$ wget http://people.ee.ethz.ch/~oetiker/webtools/smokeping/pub/fping−2.4b2_to.tar.gz
$ tar zvxf fping−2.4b2_to.tar.gz
$ cd fping−2.4b2_to
$ ./configure
$ make
$ su
Password:
# make install
```

Installation de echoping

echoping permet de tester les services echo, http, https, smtp... d'une machine :

```
$ wget ftp://ftp.internatif.org/pub/unix/echoping/echoping−5.0.1.tar.gz
$ tar zxvf echoping−5.0.1.tar.gz
$ cd echoping−5.0.1
$ ./configure −−with−ssl=/usr/include (pour Openssl)
$ make
$ su
Password:
# make install
# ln −s /usr/local/bin/echoping /usr/bin/echoping
```

Installation de SpeedyCGI

SpeedyCGI dope la vitesse d'exécution de SmokePing :

```
$ wget http://daemoninc.com/SpeedyCGI/CGI−SpeedyCGI−2.21.tar.gz
$ tar zxvf CGI−SpeedyCGI−2.21.tar.gz
$ cd CGI−SpeedyCGI−2.21
$ perl Makefile.PL

Optional mod_speedycgi support.

Mod_speedycgi increases performance under Apache by avoiding the fork/exec
overhead associated with each request under normal SpeedyCGI. However, it
requires a working copy of "apxs" in your path, Apache with mod_so
support, and additional Apache configuration.

Compile mod_speedycgi (default no)? no
Checking if your kit is complete...
Looks good
Writing Makefile for src
Writing Makefile for speedy_backend
Writing Makefile for speedy
Writing Makefile for CGI::SpeedyCGI

$ make
$ su
Password:
# make install
```

Installation de SmokePing

Une fois toutes les dépendances installées, il ne reste plus qu'à faire celle de SmokePing :

```
$ wget http://people.ee.ethz.ch/~oetiker/webtools/smokeping/pub/smokeping−1.20.tar.gz
$ tar zxvf smokeping−1.20.tar.gz
$ cd smokeping−1.20
$ su
Password:
# mkdir /var/www/html/smokeping/data/.simg −p
# ln −s /var/www/html/smokeping/data/.simg /var/www/html/.simg
# mkdir /usr/lib/smokeping
# cp lib/* /usr/lib/smokeping −R
# mkdir /etc/smokeping
```

SmokePing est composé de plusieurs fichiers qu'il faut adapter à la configuration de votre installation et copier dans les répertoires les plus adaptés à votre système.

Le premier d'entre eux est smokeping.dist, c'est l'éxecutable :

```
# cp bin/smokeping.dist /usr/bin/smokeping
# vi /usr/bin/smokeping
#!/usr/bin/perl −w
# −*−perl−*−

use lib qw(/usr/local/rrdtool/lib/perl);
use lib qw(/usr/lib/smokeping);

use Smokeping 1.20;

Smokeping::main("/etc/smokeping/config");

−−−−−−−−−−Suite du fichier−−−−−−−−−−
```

Le suivant est le script CGI qui permet d'accéder à l'interface HTML de SmokePing :

```
# cp htdocs/smokeping.cgi.dist /var/www/cgi−bin/smokeping.cgi
# vi /var/www/cgi−bin/smokeping.cgi

#!/usr/bin/speedy −w
# −*−perl−*−

use lib qw(/usr/local/rrdtool/lib/perl);
use lib qw(/usr/lib/smokeping);

use Smokeping 1.20;

Smokeping::cgi("/etc/smokeping/config");
```

−−−−−−−−−−*Suite du fichier*−−−−−−−−−−

Le suivant, basepage.html, est le fichier template qui permet de personnaliser l'interface HTML :

cp etc/basepage.html.dist /etc/smokeping/basepage.html

Le fichier config est comme son nom l'indique le fichier de configuration principal de SmokePing. Dans celui−ci on déclare les machines que l'on souhaite pinger (via fping) et/ou tester les services echo, http, https, smtp... (via EchoPing)

cp etc/config.dist /etc/smokeping/config

Voici un exemple de configuration de fichier dans lequel on teste le ping ainsi que les services echo, http, https et smtp :

```
# cat /etc/smokeping/config

# Note that all IP addresses in this file are false, to prevent some
# machine falling uder a deadly DOS storm because all users keep
# the same addresses in their config.

*** General ***

owner    = Joe Random
contact  = joe@some.place.xyz
mailhost = smtp.mailhost.abc
sendmail = /usr/lib/sendmail
imgcache = /var/www/html/smokeping/data/.simg
imgurl   = ../.simg
datadir  = /var/www/html/smokeping/data
piddir  = /var/run
cgiurl   = http://127.0.0.1/cgi−bin/smokeping.cgi
smokemail = /etc/smokeping/smokemail
# specify this to get syslog logging
syslogfacility = local0


*** Database ***

step     = 300        # Le temps en secondes entre chaque test
pings    = 20        # Le nombre de tests lancé à chaque fois

# consfn mrhb steps total

AVERAGE  0.5   1  1008
AVERAGE  0.5  12  4320
   MIN  0.5  12  4320
   MAX  0.5  12  4320
AVERAGE  0.5 144   720
   MAX  0.5 144   720
   MIN  0.5 144   720

*** Presentation ***

template = /etc/smokeping/basepage.html

+ overview

width = 600
height = 50
range = 10h

+ detail

width = 600
height = 200
unison_tolerance = 2

"Last 3 Hours"    3h
"Last 30 Hours"   30h
"Last 10 Days"    10d
"Last 400 Days"   400d

*** Probes ***

+ FPing
```

```
binary = /usr/local/sbin/fping

+ EchoPing  # uses TCP or UDP echo (port 7)
+ EchoPingHttp # HTTP (80/tcp) for web servers and caches
+ EchoPingHttps # HTTPS (443/tcp) for web servers
+ EchoPingSmtp # SMTP (25/tcp) for mail servers

*** Alerts ***

to = joe@some.place.xyz
from = joe@some.place.xyz

+lossdetect
type = loss
# in percent
pattern = ==0%,==0%,==0%,==0%,>20%,>20%,>20%
comment = suddenly there is packet loss

+rttdetect
type = rtt
# in milli seconds
pattern = <10,<10,<10,<10,<10,<100,>100,>100,>100
comment = routing mesed up again ?

*** Targets ***

probe = FPing

menu = Top
title = Network Latency Grapher
remark = Welcome to the SmokePing website of Linux Company. \
         Here you will learn all about the latency of our network.

+ Machine1

menu = Machine1
title = Titre Machine1
host = Machine1.com

+ Machine2

menu = Machine2
title = Titre Machine2

++ icmp
menu = icmp
title = Icmp Server
probe = EchoPing
alerts = lossdetect
host = Machine2

++ http
menu = http
title = Web Server (www-server) / HTTP
probe = EchoPingHttp
alerts = lossdetect
host = www.Machine2.com

++ https
menu = https
title = Https Server
probe = EchoPingHttps
alerts = lossdetect
host = www.Machine2.com

++ smtp
menu = smtp
title = Smtp Server
probe = EchoPingSmtp
alerts = lossdetect
host = smtp.Machine2.com
```

Démarrage de SmokePing

```
# cp etc/smokemail.dist /etc/smokeping/smokemail
# vi /etc/syslog.conf

------Rajouter la ligne en fin de fichier--------
# Save smokeping messages
local0.info /var/log/smokeping

# touch /var/log/smokeping
# chmod 600 /var/log/smokeping
# /etc/rc.d/init.d/syslog restart

# /usr/bin/smokeping
```

Une fois l'exécution terminée, on peut visualiser le résultat de votre travail :

```
$ mozilla http://127.0.0.1/cgi-bin/smokeping.cgi
```

Voici un exemple de ce l'on peut faire avec Smokeping : ici

Ajout d'un service smokepingd

Un script est disponible ici pour lancer SmokePing au démarrage de votre système

```
# cp smokeping-start-script-rh72 /etc/rc.d/init.d/smokepingd
# vi /etc/rc.d/init.d/smokepingd

#!/bin/sh
#
# smokeping This starts and stops the smokeping daemon
# chkconfig: 345 98 11
# description: Start/Stop the smokeping daemon
# processname: smokeping
# Source function library.
. /etc/rc.d/init.d/functions

SMOKEPING=/usr/bin/smokeping
LOCKF=/var/lock/subsys/smokeping
CONFIG=/etc/smokeping/config

----------Suite du fichier----------

# cd /etc/rc.d/init.d/
# chmod 755 smokepingd
# /sbin/chkconfig --add --level 345 smokepingd
```

Pour finir

J'espère vous avoir fait découvrir ou redécouvrir cet outil. Pour ma part je compte l'utiliser pour tester la qualité de liaisons à Relais de Trame ainsi que les services http de serveurs intranet. Je vous encourage très fortement à consulter la documentation officielle afin de découvrir les nombreuses possibilités offertes par cette application.

Configurer l'IPv6 Natif

par Franck Paillaret

Ou comment installer sa connexion IPv6

Introduction

Bienvenue dans un monde parallèle, un monde en 128bits. Vous allez maintenant affronter la configuration de votre machine. Cachez vos peurs, ignorez votre flemmardise et chaussez vos bottes, nous y allons. Si jamais vous avez des problèmes ou quoi que ce soit, n'oubliez pas que les newsgroups et irc sont les meilleurs remèdes.

Les tests de grossesse

Je suppose à la base que vous avez déjà configuré vos interfaces, (voir votre réseau) en ipv4 ou que vous savez le faire (http://lea-linux.org/reseau/lan.php3)
La première chose à faire est de configurer le noyau pour qu'il prenne en charge le v6. Je me baserai sur un noyau 2.4.x. Détectons déjà si l'ipv6 est configuré (ce qui doit être le cas pour la plupart des distributions récentes). Je ne parlerai pas également des kernel USAGI ni Vanilla, à tout casser allez voir http://www.linux-ipv6.org.
`test -f /proc/net/if_inet6 && echo "IPv6.. ca roule ma poule"`
S'il est compatible, passez à la section 2. Sinon continuez à lire :)

La config du noyau

Networking options ---->

<M> The IPv6 protocol (EXPERIMENTAL)
IPv6: Netfilter Configuration ---->

IPv6: Netfilter Configuration
< > Userspace queueing via NETLINK (EXPERIMENTAL)
<M> IP6 tables support (required for filtering/masq/NAT)
<M> limit match support
<M> MAC address match support
< > Routing header match support (EXPERIMENTAL) (NEW)
< > Hop-by-Hop and Dst opts header match (EXPERIMENTAL) (NEW)
< > Fragmentation header match support (EXPERIMENTAL) (NEW)
< > HL match support (NEW)
<M> Multiple port match support
<M> Owner match support (EXPERIMENTAL)
<M> netfilter MARK match support
< > IPv6 Extension Headers Match (EXPERIMENTAL) (NEW)
< > AH/ESP match support (EXPERIMENTAL) (NEW)
< > Packet Length match support
<M> EUI64 address check (EXPERIMENTAL)
<M> Packet filtering
<M> LOG target support
<M> Packet mangling
<M> MARK target support

Dans le menu *Networking options*, mettez en module ou en dur (dur=intégré au noyau) l'option The IPv6 protocol (EXPERIMENTAL) et ensuite mettez en module ou en dur toutes les options du menu IPv6 : *Netfilter Configuration* si vous n'etes pas sûr (ça ne risque rien). Dès que nous avons un noyau en état d'accepter l'ipv6 (c'est à dire après recompilation et reboot - http://lea-linux.org/kernel/kernel.php3), passons à la section 2.

La configuration du système

Les Net-tools sont nos amis...

Il faut réinstaller les net-tools (ifconfig, route...). Vous pouvez les trouver à http://www.tazenda.demon.co.uk/phil/net-tools/ (en n'oubliant pas de préciser bien-sûr le support ipv6).

Module es-tu là ?

Lancer le module ipv6 si jamais vous avez configuré le kernel avec le support ipv6 en module, avec: `modprobe ipv6` (pour vérifier: `lsmod |grep -w 'ipv6' && echo "IPv6.. ça roule ma poule"`)

PPPd

Il faut recompiler en décommentant HAVE_INET6 dans pppd/Makefile.linux (pour les flemmards: `perl -pi -e "s/#HAVE_INET6/HAVE_INET6/;" pppd/Makefile.linux`), ensuite un ./configure && make && make install.
Ensuite, pour une connexion pppd simple, il vous faut changer les chap-secrets/pap-secrets avec votre login et ajouter à votre /etc/ppp/options `ipv6 ,` (Attention la virgule est très importante !!!).

Pour une connexion pppoe, vous devez aussi éditer votre chap−secrets/pap−secrets avec votre login (login@net1.dual.nerim par exemple) puis éditez votre /etc/ppp/pppoe.conf et là−aussi remettez votre login et ajoutez l'option `PPPD_EXTRA="ipv6 ,"`.Nous voilà déjà bien avancés (poil au nez).

La Connexion

Lançons la connexion, comme vous avez l'habitude de faire. Si tout se passe bien, en faisant un `ifconfig ppp0 | grep inet6` vous devriez avoir un truc du genre: inet6 addr: fe80::fc75:ecb8:7e35:1c75/10 Scope:Link.

Puis, il faut attribuer l'adresse ipv6 à l'interface avec `/sbin/ifconfig ethX inet6 add votre:bloc::/48` (2001:7a8:6ee8::/48 par exemple). Enfin mettre la route par défaut: `/sbin/route add −A inet6 2000::/3 ppp0` .

Pour ajouter une adresse à votre interface: `/sbin/ifconfig eth0 inet6 add 2001:7a8:2569::2` par exemple.

Réinstaller...

Maintenant il vous reste à réinstaller les logiciels dont vous vous servez avec le support ipv6 (apache, XChat...) En général un `−−enable−ipv6` au script configure suffit, mais il faut toujours lire les READMEs !!!

Mise en réseau

Maintenant dans le cas où la machine qui vous sert à vous connecter est une passerelle, il faut la configurer en tant que tel...normal en fait :)

Sur le gateway, hop! un p'tit forwarding des paquets: `echo 1 > /proc/sys/net/ipv6/conf/all/forwarding`

Ensuite le routage des packets gateway<=>clients, on va se servir de ip (comme on aurait pu se servir de ifconfig, c'est kif−kif bourrico)

Sur le gateway:

```
ip −6 addr add 2001:mon:bloc::1 dev eth0
ip −6 ro add 2001:mon:bloc::/48 dev eth0
ip −6 ro add 2000::/3 dev ppp0
```

Sur le client:

```
ip −6 addr add 2001:mon:bloc::2 dev eth0
ip −6 ro add 2001:mon:bloc::/48 dev eth0
ip −6 ro add 2000::/3 via 2001:mon:bloc::1
```

Pour ceux qui voudraient utiliser radvd, voici un exemple de configuration:

```
interface eth0
{
	AdvSendAdvert on;
	MinRtrAdvInterval 3;
	MaxRtrAdvInterval 10;
	prefix 2001:mon:bloc:1::0/64
	{
		AdvOnLink on;
		AdvAutonomous on;
	};
};
```

De même, réinstallation pour les clients des softs avec le support ipv6.

Conclusion

Vous êtes maintenant une autre personne. Changez de travail, changez d'amis, changez de chaussettes aussi. Vous pouvez surfer l'esprit....et humm...aussi.....heu ouais ! Si vous avez des problèmes ou quoi que ce soit, n'oubliez pas que les newsgroups et irc sont les meilleurs remèdes (j'ai déjà entendu ça quelque part...)

Serveur de messagerie instantanée Jabber

par Laurent DUBETTIER−GRENIER

Installation d'un serveur de messagerie instantanée Jabber

Introduction :

Cet article est une variante de l'article original paru dans le numéro 25 (Août 2003), du magazine Planète−Linux.

Jabber est un protocole de communication XML développé sous l'égide de la Jabber Software Foundation. Un protocole, c'est un "langage" particulier permettant à des ordinateurs de communiquer entre eux… Jabber est plus particulièrement développé pour la messagerie instantanée, et c'est un protocole libre (licence GPL/JOSL). Comme ses principaux concurrents privés (AIM, ICQ, MSN, Yahoo), il permet de connaître et de faire connaître votre état de présence sur le réseau Internet et d'envoyer ou de recevoir des messages instantanément. Si vous n'êtes pas connecté, vos messages sont stockés et distribués dès que vous revenez en ligne.

L'intérêt de Jabber ? il y en a plusieurs. Le protocole Jabber est libre, public, ouvert et basé sur un langage connu (XML), gage de pérennité et de développement futur. Il est et restera non payant. Il permet donc de créer des serveurs extensibles (ajout d'une brique au serveur), décentralisés (création de son propre serveur) et sûr (sans espion logiciel, avec cryptage SSL). Il n'est par contre pas nativement multi−protocole, mais des passerelles vers les autres services de messagerie instantanée existent, à installer en complément du serveur de base.

Outre le serveur de messagerie instantanée Jabber, il existe de nombreux autres serveurs utilisant le protocole Jabber. WPJabber est une alternative libre à Jabber, tandis que le site http://www.jabber.com diffuse un des nombreux serveurs sous licence payante et propriétaire. Un bref comparatif des serveurs existant est disponible ici. Le choix est donc vaste et devrait permettre de couvrir vos besoins : de Windows à Solaris, en passant par OS X, intégrable avec une base de données MySql, Oracle ou LDAP, vous trouverez forcément une version publique ou commerciale du serveur Jabber qui vous conviendra.

Enfin, pour se connecter à votre serveur, une multitude de clients existent, en licence libre ou propriétaire : on peut citer Gabber, Gaim et Psi sous Linux, Exodus sous Windows.

L'article qui suit devrait vous permettre d'installer un serveur de messagerie instantanée Jabber (version 1.4.2), soit à partir des sources, soit à partir des paquetages rpm, sur un serveur Mandrake 9.1. Mais le principe est identique pour toute distribution. Le hostname du serveur sera jabber.masociete.com et l'adresse IP 192.168.0.1.

Matériel requis :

Le matériel requis dépend du nombre de personnes qui vont se connecter à votre serveur : un système Linux équipé d'un processeur de type Pentium avec 512 Mo de Ram peut supporter de 100 à 1000 utilisateurs, avec un taux de charge maximum de 50%. La bande passante nécessaire est de 15 bits par seconde par utilisateur connecté… Attention, sans recompilation, Linux n'accepte pas plus de 1024 connexions simultanées…

Installation à partir des sources :

Se logger comme root.
Récupérer la source du serveur Jabberd à l'adresse suivante : http://jabberd.jabberstudio.org/downloads/jabber−1.4.2.tar.gz
et la copier dans le répertoire `/usr/local`

Décompresser le fichier source :

```
# tar zxvf jabber−1.4.2.tar.gz
```

Renommer le répertoire jabber−1.4.2 :

```
# mv jabber−1.4.2 jabber
```

Récupérer le code source de openssl (openssl−0.9.7b.tar.gz) à l'adresse suivante : http://www.openssl.org. Et la copier dans le répertoire /usr/local.
Décompresser le fichier source :

```
# tar zxvf openssl−0.9.7b.tar.gz
```

Renommer le répertoire openssl−0.9.7b :

```
# mv openssl−0.9.7b ssl
```

Changer de répertoire :

```
cd ssl
```

Compiler openssl (Perl5 doit notamment être installé) :

```
# ./config
# make
# make test
```

make install

Changer de répertoire :

cd /usr/local/jabber

Installer le serveur jabber, avec prise en charge du cryptage de la connexion ssl :

$ ./configure −−enable−ssl 1>sortie.txt 2>

Vérifier que ssl a bien été pris en compte. Les premières lignes du fichier sortie.txt doivent contenir :

Running Jabber Configure
Searching for SSL... Found.

Une erreur existe dans le fichier résultant platform−settings, qu'il faut corriger. Editer le fichier avec vi par exemple :

$ vi platform−settings

Repérer la ligne :

CFLAGS= −I/usr/local/ssl/include/openssl −DHAVE_SSL

Et la transformer en :

CFLAGS=−I/usr/local/ssl/include/openssl −I/usr/local/ssl/include −DHAVE_SSL

Faire la même modification pour la ligne `CCFLAGS`. Taper alors :

$ make

Et c'est terminé, votre serveur de messagerie instantanée est installé (enfin presque...) !

Configuration :

L'essentiel de la configuration du serveur s'effectue en intervenant sur le fichier `/usr/local/jabber/jabber.xml` (ou `/etc/jabber/jabber.xml` si vous avez utilisé un paquet Mandrake). Avant toute modification, il est judicieux de faire une copie de secours du fichier original

NB : le fichier jabber.xml étant au format XML, la mise en commentaire est réalisée en utilisant des balises : `<!−−` (balise ouvrante) et `−−>` (balise fermante), comme en HTML

Ouvrir le fichier jabber.xml et modifier la ligne :

<host> <jabberd:cmdline flag="h">localhost</jabberd:cmdline> </host>

en remplaçant localhost par votre nom de domaine (jabber.masociete.com par exemple), ou par l'adresse IP de votre serveur Jabber (déconseillé). Si vous créer un serveur jabber interne, sans lien avec internet, vous pouvez commenter cette ligne :

<update> <jabberd:cmdline flag="h">localhost</jabberd:cmdline> </update>

C'est la commande permettant de contrôler automatiquement la présence de mise à jour sur le serveur jabber.org. Configurer alors le répertoire destiné à stocker les fichiers de profils des utilisateurs :

mkdir −p /usr/local/jabber/spool/jabber.masociete.com

en remplaçant jabber.masociete.com par le nom que vous avez indiqué ci−dessus à la place de localhost dans la balise `<host>` du fichier jabber.xml. Démarrez Jabber et tester son fonctionnement sans cryptage ssl (voir rubrique Démarrer/Arrêter le serveur). Si tout fonctionne normalement, on peut passer à la suite.

Il faut maintenant créer la clé de cryptage SSL. Rester dans le répertoire /usr/local/jabber et repérer l'emplacement de l'éxécutable openssl :

which openssl

Il réside normalement dans le répertoire `/usr/bin/openssl`. C'est ce chemin qui devra figurer dans la première ligne du fichier keygen.sh.
Créer un fichier keygen.sh :

$ vi keygen.sh

et taper le code suivant :

#!/bin/sh
régler le chemin ci desous sur le chemin de openssl
OPENSSL=/usr/bin/openssl

```
## Génération du certificat et de la clé
$OPENSSL req −new −x509 −newkey rsa:1024 −keyout privkey.pem −out key.pem
## effacement de la phrase servant de mot de passe
$OPENSSL rsa −in privkey.pem −out privkey.pem
## Assemblage
cat privkey.pem >> key.pem
## Effacement
rm privkey.pem
```

Enregistrer et rendre exécutable ce fichier :

```
chmod u+x keygen.sh
```

Exécuter ce fichier :

```
./keygen.sh
```

Répondre aux questions posées. A la fin de l'éxécution, vous devez trouver dans le répertoire `/usr/local/jabber` un fichier `key.pem`. Tester alors le mode ssl :

```
openssl s_client −connect 192.168.0.1:5223
```

en remplaçant 192.168.0.1 par l'adresse ip de votre serveur. Vous devez recevoir une réponse du type :

```
CONNECTED(00000003)
depth=0 /C=FR/ST=FRANCE/L=Departement/O=organization/OU=Jabber/CN=contact name/Email=email address
verify error:num=18:self signed certificate
verify return:1
depth=0 /C=FR/ST=FRANCE/L=Departement/O=organization/OU=Jabber/CN=contact name/Email=email address
verify return:1
−−−
Certificate chain
0 s:/C=FR/ST=FRANCE/L=Departement/O=organization/OU=Jabber/CN=contact name/Email=email address
i:/C=FR/ST=FRANCE/L=Departement/O=organization/OU=Jabber/CN=contact name/Email=email address
−−−
Server certificate
−−−−−BEGIN CERTIFICATE−−−−−
MIIDdDCCAt2gAwIBAgIBADANBgkqhkiG9w0BAQQFADCBiTELMAkGA1UEBhMCVVMx
EDAOBgNVBAgTB0FyaXpvbmExEDAOBgNVBAcTB1Bob2VuaXgxETAPBgNVBAoTCERJ
TExJR0FGMQ8wDQYDVQQLEwZKYWJiZXIxFzAVBgNVBAMTDkNocmlzIE1jRG9uYWxk
MRkwFwYJKoZIhvcNAQkBFgpwaHgtamFiYmVyMB4XDTAxMDYwNjAxNTMwNloXDTAx
MDcwNjAxNTMwNlowgYkxCzAJBgNVBAYTAlVTMRAwDgYDVQQIEwdBcml6b25hMRAw
DgYDVQQHEwdQaG9lbml4MREwDwYDVQQKEwhESUxMSUdBRjEPMA0GA1UECxMGSmFi
YmVyMRcwFQYDVQQDEw5DaHJpcyBNY0RvbmFsZDEZMBcGCSqGSIb3DQEJARYKcGh4
LWphYmJlcjCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEArbyosgcyf9VNpPZc
+nU6yKdfAsOSpBu/n/MkChis5POuLkXo62WEoiYuDYF6bmd6XYaVC7ZwltcCwTIv
OqdErh4u82E2qeArN0j9eq6EX+MMrYBSkv2nzwabNkkWPCS9VaOsVWx+kvRw598p
ACyANf52liFhfDGISIoTlBOn+ysCAwEAAaOB6TCB5jAdBgNVHQ4EFgQUv9mxa1Yj
o7Um9ZK0OSW0phiG23AwgbYGA1UdIwSBrjCBq4AUv9mxa1Yjo7Um9ZK0OSW0phiG
23ChgY+kgYwwgYkxCzAJBgNVBAYTAlVTMRAwDgYDVQQIEwdBcml6b25hMRAwDgYD
VQQHEwdQaG9lbml4MREwDwYDVQQKEwhESUxMSUdBRjEPMA0GA1UECxMGSmFiYmVy
MRcwFQYDVQQDEw5DaHJpcyBNY0RvbmFsZDEZMBcGCSqGSIb3DQEJARYKcGh4LWph
YmJlcoIBADAMBgNVHRMEBTADAQH/MA0GCSqGSIb3DQEBBAUAA4GBAGzAwYHlBY+P
ioqT8O4t4o30MkIDR7Q9Wqc0Uv1dUXViOKDzkzHXV/gB89yOmFVf6qNg7GB4rexs
0/fMmlSOkvXJPRCvmX05J9c5JpvcprkTjp7ECHPNZgaGsdNT+lilJ2f83uTvpJgM KoQY9OKbV4NnkxDM8lxCjvIvGvrbvnkR
−−−−−END CERTIFICATE−−−−−
subject=/C=FR/ST=FRANCE/L=Departement/O=organization/OU=Jabber/CN=contact name/Email=email address
issuer=/C=FR/ST=FRANCE/L=Departement/O=organization/OU=Jabber/CN=contact name/Email=email address
−−−
No client certificate CA names sent
−−−
SSL handshake has read 1042 bytes and written 320 bytes
−−−
New, TLSv1/SSLv3, Cipher is DES−CBC3−SHA
Server public key is 1024 bit
SSL−Session:
Protocol : TLSv1
Cipher : DES−CBC3−SHA
Session−ID: A23C1FB04F635EC09F92CBD722DAB8BB1503B54D4A0E9C61B3708CB33D6ED372
Session−ID−ctx:
Master−Key: C8C25C17D5B4312E1440DBC956FF5738829C50E16E8E704010B84B1A8D33C405995D8B7FB02E06988890C7ED400ACF32
Key−Arg : None
Start Time: 991792771
Timeout : 300 (sec)
```

Verify return code: 0 (ok)
−−−

Ouvrir le fichier `/usr/local/jabber/jabber.xml` et valider la prise en charge du mode SSL en repérant la ligne :

<ip port='5222'/>

et en ajoutant la ligne suivante à la suite :

<ssl port='5223'>192.168.0.1</ssl>

De même, repérer la ligne :

<ssl> <key ip='192.168.0.1'>/usr/local/jabber/key.pem</key> </ssl>

Redémarrez le serveur et tester avec cryptage ssl (voir rubrique Démarrer/Arrêter le serveur).

Installation simplifiée :

Si compiler les sources vous paraît trop compliqué, sous Mandrake 9.1 et KDE 3.1, il suffit d'utiliser l'interface graphique (Configuration / Paquetages / Installer des logiciels). Le paquetage actuel est jabber−1.4.2a−6mdk.i586.rpm. Le fichier principal de configuration est alors `/etc/jabber/jabber.xml`. Les seules modifications à effectuer sont celles indiquées ci−dessus concernant ce fichier. La clé SSL est générée automatiquement.

Démarrage/Arrêt du serveur :

Si vous avez installé Jabber depuis les sources (fichier tar.gz), pour démarrez le serveur, tapez :

./usr/local/jabberd/jabberd

Il y a un mode debug :

./usr/local/jabber/jabberd/jabberd −D

Si vous avez installé Jabber à partir du fichier rpm, pour démarrer/arrêter/redémarrer, votre serveur Jabber, tapez :

service jabber start/stop/restart

et le mode "debug" (option −D) :

service jabber stop
/usr/sbin/jabberd −h jabber.masociete.com −c /etc/jabber/jabber.xml −B −D

Pour vérifier que le serveur Jabber est effectivement en service, tapez :

ps ax | grep jabber

Vous devez obtenir un résultat ressemblant aux deux lignes suivantes :

3052 ? S 0:00 /usr/sbin/jabberd −h jabber.masociete.com −c /etc/jabber/jabber.xml −B
> 3053 ? S 0:00 /usr/sbin/jabberd −h jabber.masociete.com −c /etc/jabber/jabber.xml −B

où jabber.masociete.com est remplacé par le hostname de votre serveur.

Le serveur jabber utilise les ports 5222 (connexion normale avec le client), 5223 (connexion sécurisée avec le client) et 5269 (connexion entre serveurs). En tapant :

netstat −an | grep −E '5222'

vous devez obtenir la ligne suivante, qui vous indique que le serveur Jabber écoute le port 5222 : :

tcp 0 0 0.0.0.0:5222 0.0.0.0:* LISTEN

De même pour 5223 et 5269.

Pour gérer le démarrage, vous pouvez utiliser chkconfig :

chkconfig −−list | less # liste des services démarrés
chkconfig −−add jabber # ajouter jabber à la liste des services
chkconfig jabber off # inhibition du démarrage automatique
chkconfig jabber on # démarrage automatique au démarrage

Passerelles :

Il existe plusieurs briques additionnelles au serveur jabber de base : mu−conference (conférence multi−utilisateurs), jud (Jabber User Directory : répertoire des utilisateurs), et les passerelles pour les utilisateurs de AIM, ICQ, MSN et Yahoo.
Les paquetages sont distribués avec la Mandrake 9.1, et peuvent donc être installés facilement (Configuration / Paquetages / Installer des logiciels).
Il faut ensuite les déclarer dans le fichier jabber.xml : c'est généralement expliqué dans le fichier README du paquetage (pour savoir ou il est : `rpm −ql nom_du_paquetage`) et le fichier jabber.xml est abondamment commenté.
L'exemple suivant s'applique au paquetage jabber−jud. Une fois le paquetage installé, il existe un fichier `jud.so` dans le répertoire `/usr/lib/jabber/jud`
Pour activer ce service, il faut insérer les lignes suivantes dans le fichier jabber.xml, dans la zone où sont défini les services :

```
<service id="jud">
<host>jud.monserveurjabber</host>
<load><jud>./jud/jud.so</jud></load>
<jud xmlns="jabber:config:jud">
<vcard>
<fn>Annuaire des utilisateurs locaux</fn>
<desc>Ce service est un annuaire des utilisateurs locaux</desc>
<url></url>
</vcard>
</jud>
</service>
```

Et à la place du jud existant, qui déclare l'annuaire du serveur jabber.org (users.jabber.org), dans la zone `<browse>` :

```
<service type="jud" jid="jud.monserveurjabber" name="Annuaire des utilisateurs locaux">
<ns>jabber:iq:search</ns>
<ns>jabber:iq:register</ns>
</service>
```

Redémarrez jabber : le service d'annuaire devrait maintenant exister.
Depuis Gabber, il faut choisir Actions/Consultations des agents IM pour s'inscrire dans l'annuaire des utilisateurs locaux de jabber.masociete.com

Liste des extensions serveurs :

- Conférence multi−utilisateurs : http://mu−conference.jabberstudio.org/
- JUD (Jabber User Directory) : http://download.jabber.org/dists/1.4/final/
- Passerelle AIM : http://aim−transport.jabberstudio.org/
- Passerelle ICQ : http://icqv7−t.sourceforge.net/
- Passerelle MSN : http://msn−transport.jabberstudio.org/
- Passerelle Yahoo : http://yahoo−transport.jabberstudio.org/

Intranet :

Pour une utilisation en Intranet, sans communication avec l'extérieur, il faut réaliser les opérations suivantes :

- Interdire la communication entre serveurs : pour cela, fermer le port 5269 du firewall de votre organisation.
- Interdire la communication avec les clients extérieurs : fermer les ports 5222 et 5223 du firewall de votre passerelle internet.
- Commenter les sections `<service id="dnsrv"> … </service>` et `<service id="s2s"> … </service>`
- Commenter la ligne `<update> … </update>` (mise à jour automatique : ce sera à vous de la faire manuellement)

NB : Pour bloquer des ports sur le firewall shorewall, à partir du panneau de contrôle (Configuration / Panneau de contrôle Mandrake), il faut choisir Sécurité/DrakFirewall, décocher la case Tout et cocher uniquement les services que vous souhaitez autoriser.

Votre serveur ne peut alors plus communiquer avec l'extérieur. Pour des raisons de sécurité, il est important de désactiver l'authentification par mot de passe en clair : commentez la ligne `<mod_auth_plain> <mod_plain>`. N'oubliez pas de redémarrer le serveur jabber chaque fois que vous effectuez une modification du fichier jabber.xml (service jabber restart)

Webmin

Pour configurer facilement votre serveur, éventuellement depuis un poste distant, il est judicieux d'utiliser webmin. Si ce n'est pas déjà fait, installer Webmin puis, depuis un navigateur Internet, rentrez l'adresse `https://198.168.0.1:10000` (toujours en remplaçant 192.168.0.1 par l'adresse de votre serveur), et se connecter en tant que root.

NB : Si vous accéder à votre serveur depuis un poste distant Windows, sans service DNS, il sera nécessaire de rajouter la ligne suivante dans le fichier `C:\WINNT\SYSTEM32\DRIVERS\ETC\HOSTS` :

```
192.168.0.1 jabber.masociete.com
```

Ceci est d'ailleurs valable aussi pour vous connecter au serveur Jabber depuis un client installé sur un poste Windows.

Il faut ensuite choisir l'onglet "serveurs" et trouver l'icône "Jabber IM Server". L'interface graphique permet la définition de nombreux paramètres :

- Général Options : emplacement des fichiers log.
- IP Access Control : gestion des autorisations/refus d'accès.
- Messages : définition des messages de bienvenue de votre serveur.
- User filters : options de filtrage de message.
- Jabber modules : modules chargés par le serveur.
- Administration users : définition des administrateurs et de leurs droits.
  Le droit lire permet de lire les messages destinés à l'administrateur du serveur et de voir la liste des utilisateurs connectés
  Le droit écrire permet d'envoyer un message à tous les utilisateurs.
- Karma traffic control : réglage de la bande passante allouée au serveur (taux de charge).
- Edit config file : édition du fichier jabber.xml.

Debugage

Il est possible de tester le bon foctionnement de Jabber via Telnet. Ouvrir une session telnet :

telnet jabber.masociete.com 5222

Le serveur doit vous répondre :

Trying 127.0.0.1...
Connected to jabber.masociete.com.
Escape character is '^]'.

Taper les instructions suivantes :

<stream:stream
to='jabber.masociete.com'
xmlns='jabber:client'
xmlns:stream='http://etherx.jabber.org/streams'>

Vous devez recevoir immédiatement une réponse :

<?xml version='1.0'?> <stream:stream
xmlns:stream='http://etherx.jabber.org/streams' id='nombre−aleatoire'
xmlns='jabber:client' from='jabber.masociete.com'>

Demander alors les informations nécessaires pour enregistrer un compte :

<iq id='reg1' type='get'>
<query xmlns='jabber:iq:register'/>
</iq>

Vous devriez recevoir quelque chose du type :

<iq id='reg1' type='result'>
<query xmlns='jabber:iq:register'>
<instructions>
Choose a username and password to register with this server.
</instructions>
<name/>
<email/>
<username/>
<password/>
</query>
</iq>

Envoyer alors vos informations d'enregistrement :

<iq id='reg2' type='set'>
<query xmlns='jabber:iq:register'>
<username>jabberuser</username>
<password>secret</password>
<name>Votre−Nom</name>
<email>votre−email@domaine.com</email>
</query>
</iq>

Si tout fonctionne correctement, vous devez recevoir :

<iq id='reg2' type='result'/>

Si le répertoire de spool (`/usr/local/jabber/spoll/jabber.masociete.com`) n'est pas configuré correctement :

<error code='500'>Password Storage Failed</error></iq>

Si votre nom d'utilisateur existe déjà :

<error code='409'>Username Not Available</error></iq>

Avant de vous logger sur votre compte nouvellement créé, demander au serveur quelles infos sont nécessaires :

<iq id='auth1' type='get'>
<query xmlns='jabber:iq:auth'>
<username>jabberuser</username>
</query>
</iq>

Le serveur doit vous renvoyer les informations suivantes :

<iq id='auth1' type='result'>
<query xmlns='jabber:iq:auth'>
<username>jabberuser</username>
<password/><digest/>
<resource/>
</query>
</iq>

Connectez−vous alors à votre compte :

<iq id='auth2' type='set'>
<query xmlns='jabber:iq:auth'>
<username>jabberuser</username>
<password>secret</password>
<resource>telnet</resource>
</query>
</iq>

Le serveur doit vous répondre :

<iq id='auth2' type='result'/>

Indiquer au serveur que vous êtes en ligne :

<presence/>

Le serveur doit vous envoyer immédiatement un message de bienvenue (paramètrable dans le fichier jabber.xml) :

<message from='jabber.masociete.com' to='jabberuser@jabber.masociete.com'>
<subject>Welcome!<subject>
<body>
Welcome to the Jabber server at localhost −− we
hope you enjoy this service! For information about
how to use Jabber, visit the Jabber User's
Guide at http://docs.jabber.org/
</body>
</message>

Refaire la même démarche depuis un autre poste distant, avec un autre nom d'utilisateur, puis essayer d'envoyer un message à votre premier compte :

<message to='jabberuser@jabber.masociete.com'>
<body>hi!</body>
</message>

Pour quitter correctement vos sessions telnet :

</stream:stream>

Si tout a fonctionné correctement, vous pouvez installer un client (Gabber, Gaim, Psi, ...).

D'autres fichiers contiennent des informations importantes :

- /var/run/jabber/jabber.pid contient le numéro de processus affecté à jabber, lorsqu'il a démarré.
- /var/log/jabber/error.log contient la liste des erreurs.
- /var/log/jabber/record.log contient la liste des personnes qui se sont connectées à votre serveur, leur adresse IP, le client qu'ils utilisent et la durée de leur connexion.
- /var/lib/jabber contient un répertoire correspondant au nom (hostname) de votre serveur, dans lequel seront stockés les profils utilisateurs. Ce sont des fichiers XML contenant notamment (en clair) le mot de passe de connexion et les contacts de l'utilisateurs.
- /usr/share/doc/jabber−1.4.2a est un répertoire contenant une documentation succincte.

En cas de problème, ou pour obtenir plus de renseignement :

- la communauté francophone des utilisateurs de Jabber : http://www.jabberfr.org
- La FAQ utilisateur
- La FAQ administrateur
- Enfin, après avoir consulté les archives pour vérifier que votre question (et sa réponse) ne figure pas déjà quelquepart, la mailing−list administrateur : http://mailman.jabber.org/listinfo/jadmin/

SquirrelMail

par Laurent DUBETTIER−GRENIER

Installer une interface graphique de messagerie SquirrelMail

Introduction

Vous souhaitez installer votre propre serveur de mail au sein de votre entreprise, équipé d'une interface web permettant à vos collaborateurs d'accéder et de gérer leur courrier électronique à distance ? C'est ce que rend possible SquirrelMail. Rapide et stable, facile à installer, extensible par plugins à volonté, il vous permettra de créer votre propre webmail.

Un webmail est une interface web permettant la lecture de votre courrier électronique. Vous avez certainement déjà utilisé ce type d'interface, si votre adresse électronique est hébergée par La Poste, par exemple. L'intérêt ? Pouvoir accéder de n'importe où à vos messages (chez vous, au travail, dans un cybercafé...). SquirrelMail n'est pas le seul webmail disponible : IMP, basé sur le projet Horde est une autre possibilité, très puissante, utilisée entre autres pour le webmail de Free. Mais SquirrelMail est beaucoup plus simple à installer, configurer et étendre que IMP. Il est de plus soutenu par une large communauté francophone.

L'objet de cet article et d'expliquer la procédure d'installation et de configuration d'une interface graphique de messagerie basée sur SquirrelMail, sur un serveur Linux Mandrake 9.2, au sein d'un réseau local d'entreprise. Le hostname du serveur est `mail.exemple.net`, son adresse IP `192.168.0.1`. Pour configurer correctement ce qui suit, il faut être connecté au serveur en tant qu'administrateur (root). Cet article ne gère pas tous les éventuels problèmes de sécurité liés à la mise en place de ce service.

Installation

Pour installer SquirrelMail, il faut un serveur Linux, sur lequel sont installés, entre autres : Apache2, Apache2−mod_php, php−ini, Apache2−mod_ssl, imap, postfix, perl, xinetd, ispell. L'installation de Perl permet de configurer ultra−rapidement squirrelMail grâce à une interface développé dans ce langage, et ispell permet le fonctionnement correct du correcteur orthographique squirrelspell intégré. MySQL n'est pas requis, sauf pour certains plugins. Enfin pour un fonctionnement correct, les navigateurs des postes qui utiliseront votre webmail doivent accepter les cookies.

Vous devez avoir les services httpd, postfix, xinetd, imap activés.
Pour visionner les services fonctionnant :

```
# chkconfig −−list | less
```

Pour mettre en marche un service (exemple avec apache) :

```
# service httpd start
```

Pour faire en sorte qu'à chaque démarrage de votre poste, le service Apache fonctionne :

```
chkconfig apache on
```

Il faut notamment veiller à bien lancer le service imap, dépendant de xinetd :

```
# chkconfig imap on
```

service xinetd restart.

Php

Premièrement, vérifier que PHP est installé et fonctionne correctement en créant un fichier de test dans le répertoire `/var/www/html` :
`vi phpinfo.php`
Insérer les instructions php suivantes dans ce fichier :

```
<?
phpinfo();
?>
```

Puis tester le fonctionnement de php :

```
# lynx http://127.0.0.1/phpinfo.php
```

Vous devez obtenir une page donnant les caractéristiques complète de votre installation de PHP et de ces modules additionnels. Si cela fonctionne correctement, vous pouvez effacer le fichier `phpinfo.php`.

Il faut ensuite configurer `/etc/php.ini`, notamment :

session.use_cookies = 1
file_uploads = On (pour autoriser les fichiers attachés)
upload_max_filesize = 2M (règle la taille maxi des fichiers attachés)
register_globals=off (il est placé par défaut sur On par la distribution Mandrake)
expose_php = Off (sécurité : n'expose pas le fait que votre serveur héberge PHP)

Le paramètre `session.save_path` doit pointer vers un répertoire lisible uniquement par le serveur web (`/var/squirrel/session` par exemple) :

```
mkdir −p /var/squirrel/session
chgrp apache /var/squirrel/session
chown apache /var/squirrel/session
chmod 730 /var/squirrel/session
```

Redémarrer le serveur Apache :

```
service httpd restart
```

SquirrelMail

Pour installer SquirrelMail, il faut télécharger la dernière version (`squirrelmail−1.4.2.tar.gz` : 2.74 Mo), la décompresser dans un répertoire lisible par le serveur web (`/var/www/html/` par exemple), changer le nom du répertoire (`mv squirrelmail−1.4.2 squirrelmail`), créer les deux répertoires de stockage des données et des fichiers joints :

```
mkdir −p /var/squirrel/data
chown apache /var/squirrel/data
chgrp apache /var/squirrel/data
mkdir −p /var/squirrel/attachement
chgrp apache /var/squirrel/attachement
chmod 730 /var/squirrel/attachement
```

Ceci cause un problème : si quelqu'un prépare un mail avec pièce jointe, mais interromps son action avant de l'expédier, les pièces jointes resteront éternellement dans le répertoire `/var/squirrel/attachement`. Il faut donc prévoir une tâche cron qui supprime l'ensemble des fichiers de ce répertoire à intervalle régulier, en espérant que personne n'est en train d'envoyer une pièce jointe au même moment... Voici un exemple permettant de supprimer tous les jours à 23H15 le contenu du répertoire `/var/squirrel/attachement` :
éditer le fichier crontab et ajouter la ligne :

```
# crontab −e
15 23 * * * /root/rm_attachement.sh
```

le fichier `rm_attachement.sh`, executable (`chmod +x rm_attachement.sh`), contient les lignes suivantes :

```
# cd /var/squirrel/attachement
rm −f *
```

Puis il faut alors configurer squirrelmail soit manuellement en renommant le fichier `config_default.php` en `config.php` (répertoire `/var/www/html/squirrelmail/config`) puis en l'éditant pour modifier ses paramètres, soit, beaucoup plus convivial, en utilisant le petit script perl `conf.pl` situé dans ce même répertoire (nécessite que le paquetage perl soit installé). Pour le lancer : `./conf.pl` (point slash conf.pl). Un menu comportant 9 options apparaît, qui permet de configurer complètement SquirrelMail. Quelques modifications à effectuer :

- Organization Preferences / Default Language : remplacer en_US par fr_FR
- Server Settings / Domain : exemple.net
- General Options / Data Directory : /var/squirrel/data
- General Options / Attachment Directory : /var/squirrel/attachement
- Themes : pour choisir celui qui vous plaît !
- Plugins : pour installer un plugins, parmi ceux disponibles (Available Plugins)
- S sauve ces modifications, Q quitte l'application de configuration.

Puis, à partir d'un navigateur sur un poste distant, se rendre à l'url `http://192.168.0.1/squirrelmail/` ou `http://mail.exemple.net/squirrelmail`
si vous avez un serveur DNS correctement configuré. L'interface de squirrelmail apparaît.

Remarque : Si vous n'avez pas de serveur DNS, avec un poste Windows distant, il faut configurer le fichier hosts situé dans le répertoire `C:\WINNT\SYSTEM32\DRIVERS\ETC` , et rajouter la ligne : `192.168.0.1 mail.exemple.net`. Sur un poste Linux distant, il faut rajouter la même ligne dans le fichier `/etc/hosts`.

Plugins

Une fois que tout est installé et en état de fonctionnement, vous aurez peut−être envie d'étendre les fonctionnalités de votre interface graphique de messagerie. Pour cela, il faut installer un ou plusieurs plugins, et le choix est vaste... Il en existe plus de 170 différents ! Par défaut, 16 plugins sont déjà installés (mais non activé !) :

- Abook_take : pour récupérer facilement l'adresse de l'expéditeur d'un mail dans votre carnet d'adresse.
- Administrator : pour administrer à distance la configuration de SquirrelMail.
- Bug_report : pour reporter facilement un bug à l'équipe de développement.
- Calendar : un calendrier simple.
- Delete_move_next : pour effacer et déplacer facilement vos mails.
- Filters : pour filtrer vos mails (antispam)
- Fortune : pour insérer une devise dans vos mails.
- Info : pour dépanner le serveur IMAP (ne pas activer en usage normal !)

- Listcommands : ajoute une ligne à tout mail en provenance d'une mailing−list, permettant notamment de se dés−inscrire ou de visionner les archives de la mailing−list.
- Mail_fetch : pour récupérer vos mails sur un serveur pop.
- Message_details : pour accéder à la source du mail reçu.
- Newmail : ajoute une musique vous signalant un nouveau message reçu.
- Sent_subfolders : pour gérer les sous répertoires.
- Spamcop : pour rapidement signaler à spamcop.net un spammeur.
- Squirrelspell : un dictionnaire basé sur ispell.
- Translate : Envoi votre message vers un serveur de traduction en une autre langue.

Pour installer un plugin c'est plus ou moins simple : certains requièrent l'installation de paquetages supplémentaires pour fonctionner (serveur de base de données MySQL, par exemple), d'autres s'installent en quelques instants. Par exemple, le plugin nommé "`Show Username and IP`" permet d'afficher l'adresse IP et le nom d'utilisateur de la personne connectée au webmail. Pour l'installer, il suffit de télécharger le paquetage (`show_user_and_ip−2.2.3.tar`), de le copier dans le répertoire plugins (`/var/www/html/squirrelmail/plugins`), de le décompresser (`tar xvf show_user_and_ip−2.2.3.tar`) et de l'activer par l'intermédiaire de l'interface `conf.pl` (rubrique plugins). On peut difficilement faire plus simple !

Utilisateurs

Seul les utilisateurs ayant un compte sur votre serveur de mail auront la possibilité d'utiliser SquirrelMail. Pour créer un compte uniquement destiné à la messagerie, sans possibilité de connections sur votre serveur, il faut d'abord créer un groupe d'utilisateur dédié à l'utilisation de SquirrelMail :

groupadd squirrelmail

Puis créer chaque utilisateur par l'instruction suivante :

useradd −c "nom_utilisateur ou commentaire" −s /bin/false −g squirrelmail nom_utilisateur

Enfin définir le mot de passe pour chaque compte nouvellement créé :

passwd nom_utilisateur

Pour tester le fonctionnement, essayer d'envoyer des mails à un autre compte et à vous même. Utiliser d'abord une adresse mail du type `utilisateur@mail.exemple.net`.

Postfix

Pour utiliser une adresse du type `utilisateur@exemple.net`, il faut configurer Postfix. Sinon, lorsque `machin@mail.exemple.net` envoie un mail à `truc@mail.exemple.net`, l'adresse de réponse est `machin@exemple.net`. Si `truc` essaie de répondre à `machin`, Postfix ne peut envoyer la réponse. Pour que cela fonctionne correctement, il faut rajouter (ou décommenter) les lignes suivantes du fichiers `/etc/postfix/main.cf` :

myorigin = $mydomain
mydestination = $myhostname, localhost.$mydomain, $mydomain

Conclusion

Et voilà, grâce à SquirrelMail, en quelques dizaines de minutes, vous avez maintenant un serveur de mail équipé d'une interface de type webmail complète ! Il n'y a plus maintenant qu'à convaincre votre hiérarchie à acheter le serveur nécessaire à son installation, et à expliquer à vos collaborateurs comment utiliser l'interface de SquirrelMail...