Mise en place d'une passerelle Internet sous Linux

Article publié dans GNU/Linux Magazine France, numéro 65 (Octobre 2004).

Il est devenu assez courant de disposer d'un accès permanent haut-débit à Internet. Pourquoi, dans ces conditions, ne pas commencer à envisager tout naturellement le partage de cette même connexion entre plusieurs machines ?

Il y a pourtant plusieurs bonnes raisons qui devraient vous y mener :

- Il est raisonnable de penser que, vu l'évolution du parc informatique et le profil du magazine que vous tenez entre vos mains, vous disposez chez vous de plusieurs ordinateurs plus ou moins récents qui pourraient parfaitement servir de bornes d'accès Internet,
- Le prix de l'abonnement mensuel n'est pas négligeable et partager l'accès entre plusieurs individus est une bonne façon de rationaliser les coûts,
- Les offres proposant des bandes passantes de 1024k voire 2048k se généralisent, il est est donc de plus en plus envisageable de partager une même connexion tout en conservant un débit acceptable,
- La convivialité : Vous aurez noté comme il est particulièrement énervant de devoir attendre que le petit dernier, grand-maman ou votre petit(e) ami(e) ai fini de lire ses e-mails et de surfer sur Internet alors que vous avez tellement de choses plus importantes à faire ... Plus sérieusement, il est toujours plus malin de réunir sa petite famille autour de la toile plutôt que de vous isoler autour de votre petite boite, en solitaire.

Dans ce cas de figure, la première des choses à faire est de transformer une de vos machines, évidemment sous Linux, en passerelle Internet et de connecter tout votre jolie monde autour d'un petit réseau Intranet.

Il est bien évident que toutes les techniques présentées pourront bien être appliquées dans le contexte d'une petite PME. Un PC réformé peut toujours avantageusement être transformé en passerelle Internet et satisfaire le besoin de plusieurs dizaines d'utilisateurs avec un rapport prix-fiabilité imbattable !

Configurer Linux en passerelle Internet

Linux permet assez facilement de construire une telle architecture en utilisant des briques disponibles en standard sur la majorité des distributions, en particulier la couche NetFilter (intégrée au noyau Linux depuis la série 2.4) qui vous permettra de filtrer et de protéger l'ensemble de vos postes clients des intrusions en provenance du grand méchant Internet.

Vous avez bien évidemment besoin d'un accès à internet et d'une machine disposant d'une carte réseau Ethernet standard qui vous permettra de vous connecter aux autres postes par l'intermédiaire d'un Switch ou d'un Hub (si vous ne disposez que de deux machines, vous pourrez vous satisfaire d'un simple câble croisé).

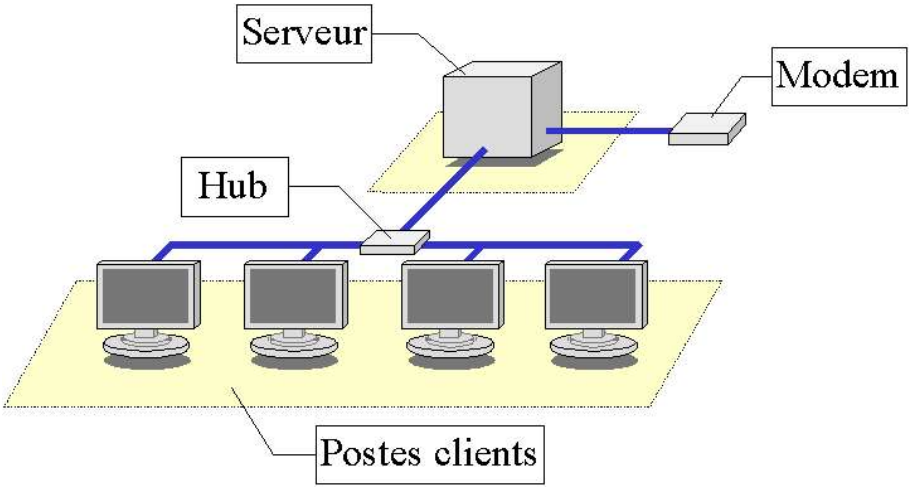


Figure 1 : Schéma de connexion classique d'un réseau Intranet

La plupart des distributions récentes proposent des interfaces graphiques qui mettent en place ce type de configuration en quelques clics. Cependant, nous allons vous proposer de mettre en place une solution équivalente par l'intermédiaire de votre propre script appelé dans la phase de démarrage du système : C'est tout de même plus formateur.

Vous pouvez utiliser comme base de travail le script shell que nous mettons à votre disposition sur [1] : Quoique modeste, celui-ci contient l'essentiel de ce que vous aurez besoin pour transformer votre serveur en passerelle Internet.

Ce script nécessite l'installation du programme **iptables** (lequel vous permet de configurer les règles de routage) et supporte les connexions par modem RTC aussi bien que les connexions par ADSL (testé sur un modem ADSL Ethernet).

Pour le mettre en place, suivez la procédure suivante sur une distribution à base de RPM (fonctionne sur SuSE, Redhat et Mandrake) :

```
# cp parefeu_passerelle /etc/init.d
# chmod 744 /etc/init.d/parefeu_passerelle
# chkconfig --add parefeu_passerelle
# chkconfig --list parefeu_passerelle
 parefeu_passerelle 0:off 1:off 2:off 3:on 4:off 5:on 6:off
 # /etc/init.d/parefeu_passerelle start
```

Sous la distribution Debian, vous pouvez utiliser la procédure suivante :

```
# cp parefeu_passerelle /etc/init.d
# chmod 744 /etc/init.d/parefeu_passerelle
# update-rc.d parefeu_passerelle start 20 3 5 . stop 80 0 1 2 4 6 .
# /etc/init.d/parefeu_passerelle start
```

Vous pouvez éditer ce script si vous souhaitez modifier la configuration par défaut : Les variables OUT pour l'interface connectée au réseau Internet (par défaut ppp0) et IN pour l'interface connectée au réseau privé (par défaut eth0).

Notez tout de même que d'autres exemples de configuration sont disponibles sur le site même de NetFilter [2] ainsi que toute la documentation nécessaire pour comprendre le contenu du script et l'améliorer (nous n'aborderons pas dans le détail la syntaxe des règles iptables, de nombreux articles ont parus sur le sujet qui s'acquittent parfaitement de cette tâche).

Les principes mis en oeuvre par ce type de scripts sont finalement assez simples : On distingue généralement une interface réseau publique connectée à Internet et une interface connectée au réseau privé (voir la figure 1). On bloque ensuite tous les accès de toutes les interfaces du système avant de distiller les droits d'accès : Cette approche permet d'expliciter clairement ce qui est autorisé de ce qui ne l'est pas.

Dans le cas de notre script, voici un petit résumé des principes utilisés avec les commandes mises en oeuvre (leurs simplicité vous donnera peut-être envie d'en savoir plus et d'aller plus loin ; elles ne sont données que pour vous permettre de comprendre ce que fait le script et pour évaluer la confiance que vous pouvez placer en elles) :

- On vide toutes les règles existantes avant de bloquer les accès sur toutes les interfaces (on ignore les paquets reçus) :

```
/sbin/iptables -F
/sbin/iptables -X
/sbin/iptables -Z
/sbin/iptables -P INPUT DROP
/sbin/iptables -P OUTPUT DROP
/sbin/iptables -P FORWARD DROP
```

- On autorise tout type de communication sur l'interface de loopback (lo0) pour autoriser les communications réseau en local sur la passerelle :

```
/sbin/iptables -A INPUT  -i lo -j ACCEPT
/sbin/iptables -A OUTPUT -o lo -j ACCEPT
```

- On autorise tout type de communication sur l'interface du réseau privé (eth0) ; C'est une approche dite « de confiance » où l'on considère que le risque ne peut pas venir de l'intérieur du réseau :

```
/sbin/iptables -A INPUT  -i eth0 -j ACCEPT
/sbin/iptables -A OUTPUT -o eth0 -j ACCEPT
```

- On autorise le flux IP à sortir de l'interface publique connectée à Internet ; Ne nous y trompons pas, à ce stade les paquets IP peuvent effectivement sortir mais ne peuvent pas revenir :

```
/sbin/iptables -A OUTPUT -o ppp0 -j ACCEPT
```

- On autorise les flux IP entrants que l'on connaît déjà ; Uniquement les paquets IP appartenant à des flux que la passerelle aura elle-même initiée seront autorisés à passer le pare-feu (ESTABLISHED fait référence à une connexion établie et RELATED aux flux assimilables à une connexion établie, comme par exemple FTP) :

```
/sbin/iptables -A INPUT -i ppp0 -m state --state ESTABLISHED,RELATED -j
ACCEPT
```

- On autorise le routage des paquets IP vers le réseau public (active la fonction passerelle) :

```
/bin/echo "1" > /proc/sys/net/ipv4/ip_forward
/sbin/iptables -A FORWARD -i eth0 -j ACCEPT
/sbin/iptables -A FORWARD -o eth0 -j ACCEPT
```

- On active une option pour changer l'origine des paquets IP : En effet, les adresses IP des machines de votre sous-réseau ne sont pas connues sur Internet et doivent donc être ré-écrites :

```
/sbin/iptables -t nat -A POSTROUTING -o ppp0 -j MASQUERADE
```

Comme nous venons de le préciser, toutes vos machines auront accès à Internet car elles seront « masquées » par la machine passerelle : Vu de l'intérieur du réseau Internet (réseau public), toutes les connexions en provenance de votre Intranet (réseau privé) sembleront venir de votre machine passerelle ; Ce qui est en soit naturel car l'adresse IP allouée par votre fournisseur d'accès est très logiquement la seule à être connue sur le réseau mondial. Cette technique est habituellement résumée sous le terme de masquerading.

Il vous reste à configurer votre machine serveur comme passerelle sur chacun de vos postes clients. Consultez la documentation sur la configuration du réseau sur les systèmes d'exploitations des machines de votre parc.

Par contre, dans cette configuration, il vous faudra référencer un serveur de nom sur chacune de vos machines clientes, car sinon vous ne serez pas en mesure de résoudre les adresses Internet : Consultez le contenu du fichier **/etc/resolv.conf** sur la machine passerelle lorsque cette dernière sera effectivement connectée à Internet pour obtenir une liste de serveurs DNS.

Notez que par l'intermédiaire du script proposé, tous les accès refusés seront tracés dans le fichier **/var/log/messages**. Consultez le régulièrement, vous aurez de quoi être particulièrement surpris. Internet est un réseau à ne pas mettre une adresse IP dehors ...

Ce script considère que vous ne mettez aucun service à disposition sur Internet, comme par exemple un serveur Apache ou un serveur FTP : Vous ne vous connectez strictement que comme client.

Si d'aventure cela vous intéresse, voici quelques recettes simples à appliquer si vous savez exactement ce que vous faites (ouvrir un port sur le pare-feu, c'est exposer fatalement sa machine à des attaques) :

Pour autoriser la connexion entrante sur certains ports de la passerelle (par exemple, HTTP et FTP), il vous faudra utiliser la règle suivante :

```
/sbin/iptables -A INPUT -i ppp0 -p tcp -m state --state NEW -m multiport --
destination-port 80,20,21 -j ACCEPT
```

Pour donner un accès transparent à un service présent sur une machine de votre sous-réseau, vous pouvez utiliser le principe du port forwarding (le service semblera venir de votre passerelle alors qu'il sera physiquement présent sur une autre machine) :

```
/sbin/iptables -t nat -A PREROUTING -p tcp -i ppp0 --dport 80 -j DNAT --to-
destination [adresse_ip]:80
```

Dans le cadre de la mise en oeuvre d'une passerelle Internet, ce type de solution est facile à mettre en place mais offre tout de même quelques désavantages notables :

- Lors de la consultation d'un site Internet par plusieurs personnes, la probabilité de transférer la même information plusieurs fois est assez importante, ce qui n'est guère optimal,
- Lorsque vous n'êtes pas là, comment pouvez-vous être certain que l'un de vos proches (un enfant ou un adolescent, par exemple) n'est pas en train de surfer sur un site présentant un contenu dangereux : En d'autres termes, comment tracer les accès et éviter les dérapages,
- Cela n'empêchera pas les logiciels espions potentiellement présents sur vos postes clients (de type Windows ou Mac) de se connecter à Internet pour y transférer des informations (à votre insu) puisque l'accès est libre. Sur ce thème, les utilisateurs de machines clientes Windows peuvent utiliser les logiciels Ad-aware [3] et Spybot [4] pour vacciner les machines clientes Windows.

C'est là qu'il faut faire intervenir la notion de serveur proxy pour nous sortir de ce mauvais pas.

Serveur proxy ?

Une petite consultation rapide de mon petit Harrap's de poche nous précise très justement que le terme *proxy* signifie en anglais *procuration*, *pouvoir* ou encore *mandat*. Car, tout l'intérêt réside dans cette définition : Vous mandatez un bout de code logiciel afin qu'il réalise, à votre place, des requêtes réseaux protocolaires comme par exemple HTTP ou encore FTP.

Au lieu de réaliser directement une requête sur le serveur final, vous demandez au serveur proxy de réaliser cette opération à votre place : On perçoit immédiatement la finalité car, si seule la machine qui héberge le serveur proxy est connectée à Internet, toutes les personnes qui feront appel à ce serveur pourront aussi, à leur tour, avoir accès à Internet. De plus, les requêtes, vu du Net, sembleront venir d'une seule et même machine : Celle hébergeant le serveur proxy. Les connexions en provenance de votre petit Intranet seront « cachées » aux machines physiquement présentes sur le réseau Internet, vous n'êtes donc pas obligé de disposer d'autant d'adresses IP Internet que de machines.

En parlant de cacher, il nous faut aborder une fonctionnalité indispensable qui vient souvent de paire avec les serveurs proxy : La possibilité de conserver sur disque les éléments (images, pages html, sons, ...) récupérés par le serveur pour éviter d'avoir à les re-télécharger une seconde fois. C'est en somme une façon centralisée de mutualiser l'information comme le fait naturellement votre navigateur en conservant dans un cache disque l'historique de tous les éléments web récemment téléchargés. Les accès au net en sont artificiellement accélérés (réduction des temps d'accès et de la bande passante consommée), surtout lorsque plusieurs personnes partagent un tel cache.

D'ailleurs, on appelle assez logiquement ces systèmes des « proxy-cache ».

Dans le cadre de nos expérimentations, nous allons vous proposer d'utiliser le serveur **Squid** car ce dernier est le plus souvent livré en standard dans la plupart des distributions.

L'utilisation conjointe avec un pare-feu logiciel est plus que jamais conseillé car un proxy-cache ne vous protégera jamais des diverses tentatives d'intrusions : Il reste, par rapport au script précédemment proposé, de désactiver le routage des paquets IP ainsi que le masquerading en commentant la variable IN dont nous avons discutée précédemment (il devient alors impossible d'aller sur le Net dans le proxy). Les commentaires présents dans le script pourront vous aider sur ce sujet.

Attention, il faut bien avoir conscience qu'un serveur proxy-cache ne peut transporter généralement qu'un nombre limité de protocole : Dans le cas du serveur **Squid**, vous l'utiliserez généralement pour les protocoles HTTP, HTTPS et FTP. Cela est suffisant dans la plupart des cas mais devient un réel handicap lorsque vous souhaitez utiliser des protocoles comme irc ou icq sur vos postes clients.

Deux solutions s'offrent alors à vous :

- Modifier le script pare-feu pour autoriser uniquement le routage des protocoles irc et icq,
- Si vos clients irc ou icq supportent le protocole SOCKS, vous pouvez utiliser en parallèle au serveur **squid** un serveur SOCKS.

La première solution peut être mise en oeuvre simplement en modifiant les lignes concernant le routage des paquets IP pour accepter uniquement de router des paquets de type TCP sur une liste de ports :

```
/sbin/iptables -A FORWARD -i eth0 -p tcp -m multiport --destination-port
6667,5190 -j ACCEPT
```

La seconde solution demande d'abord une petite explication de texte : SOCKS est un protocole qui permet de router toute communication réseau de type TCP ou UDP. Plus précisemment la version 4 supporte les protocoles basés sur TCP alors que la version 5 supporte à la fois TCP et UDP.

On va alors parler de proxy SOCKS dont le travail va simplement consister à faire transiter des données entre un client et un serveur sans interférer avec leurs contenus et donc offrir un support à

des protocoles comme icq ou irc à l'intérieur d'un Intranet dont les accès Internet sont limités par un proxy cache classique, ce qui est en l'occurence notre cas (il permet aussi de rendre anonyme les connexions, ce qui est un autre débat).

Les personnes concernées par la mise en oeuvre d'un tel serveur peuvent se renseigner sur le serveur proxy réseau **Dante** [5] qui offre justement le support du protocole SOCKS.

Configuration du serveur squid

Grâce à ce logiciel, vous aurez en plus la possibilité de limiter les accès à des plages horaires précises ou encore autoriser les connexions sur certains sites ou groupes de sites. Les deux éléments sur lesquels vous allez vous appuyer pour contrôler ces accès sont les éléments ACL (Access Control List) et les listes d'accès : Le but de cet article n'étant pas de vous former à la configuration de ce type de serveur, nous allons vous présenter une configuration minimale type livrée avec quelques recettes standards, libre à vous ensuite de consulter les documentations et manuels si tripoter la chose plus précisément vous intéresse.

Le comportement d'un serveur **Squid** se contrôle par l'édition du fichier **squid.conf** habituellement placé sous le répertoire */etc*. Il est à noter que ce fichier héberge de nombreux commentaires qui vous seront très utiles afin d'en apprendre un peu plus sur le sujet, en particulier sur les différentes options disponibles.

La première chose à définir est la variable http_port avec laquelle vous irez configurez vos navigateurs ou autres programmes Internet (par défaut, on utilise le port 3128 pour la connexion au serveur **Squid**) :

```
http_port 3128
```

On indique ensuite sous quel utilisateur le serveur **Squid** va effectuer ses requêtes :

```
cache_effective_user nobody
cache_effective_group nogroup
```

Il nous faut maintenant configurer l'emplacement sur le disque dur où le serveur ira stocker ses données et tracer les connexions :

```
cache_mem 8 MB
maximum_object_size 4096KB
cache_dir ufs /var/cache/squid 250 16 256
cache_access_log /var/log/squid/access.log
cache_store_log none
```

On aura, au préalable, pris soin de créer les répertoires */var/cache/squid* et */var/log/squid* si ces derniers n'existent pas avec les droits d'écriture pour l'utilisateur nobody (chmod 744).

Suit la configuration du contrôle d'accès de vos machines clientes. Si l'on part du principe que votre sous-réseau Intranet est architecturé autour d'un masque de classe B de type 192.168, alors il vous faudra insérer les lignes :

```
acl all src 0.0.0.0/0.0.0.0
acl allowed_hosts src 192.168.0.0/255.255.0.0
acl localhost src 127.0.0.1/255.255.255.255
acl SSL_ports port 443 563
acl Safe_ports port 80          # http
acl Safe_ports port 21          # ftp
acl Safe_ports port 70          # gopher
acl Safe_ports port 443 563     # https nttps
acl CONNECT method CONNECT
http_access deny !Safe_ports
http_access deny CONNECT !SSL_ports
http_access allow localhost
http_access allow allowed_hosts
http_access deny all
```

La figure 2 présente la façon dont vous devrez configurer votre navigateur pour vous connecter à votre serveur Squid en considérant que celui-ci se situe sur une machine configurée avec l'adresse IP 192.168.1.1

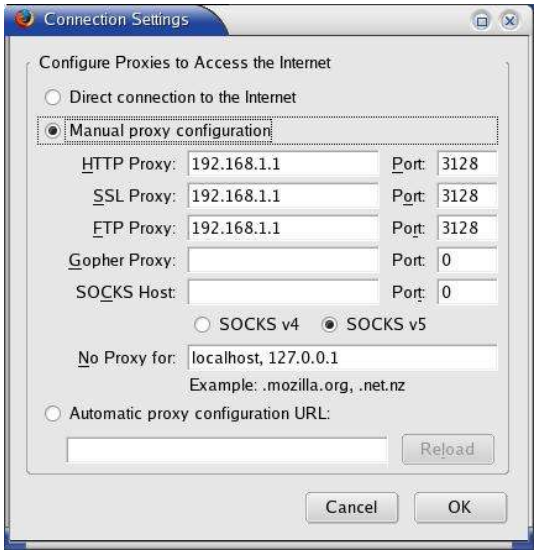


Figure 2 : Configuration du navigateur pour établir la connexion avec le serveur proxy

Si vous souhaitez restreindre l'accès strictement sur une plage horaire précise et sur un groupe de machines, il aurait fallu saisir les lignes suivantes :

```
acl all src 0.0.0.0/0.0.0.0
acl allowed_hosts src 192.168.2.200 192.168.2.211
acl allowed_hosts src 192.168.2.1
acl localhost src 127.0.0.1/255.255.255.255
acl calendrier time MTWHF 17:00-19:45
http_access allow localhost
http_access allow allowed_hosts calendrier
http_access deny all
```

Dans cet exemple, on autorise les accès uniquement pour trois machines entre 17 heures et 19 heures 45 du lundi au vendredi. Au passage, vous aurez compris que l'on peut répéter les règles ACL, ce qui est très utile pour la lisibilité de votre fichier de configuration.

On utilise le format suivant pour filtrer sur les jours et les heures :

```
acl [mot_clef] time [liste_jours] [heure:minute-heure:minute]
```

Sachant que l'on distingue les jours avec la table d'équivalence suivante (qui vous permettra, par ailleurs, de réviser un peu votre anglais!) :

- M : Lundi (Monday),
- T : Mardi (Tuesday),
- W : Mercredi (Wednesday),
- H : Jeudi (tHursday),
- F : Vendredi (Friday),
- A : Samedi (sAturday),
- S : Dimanche (Sunday).

Et puisque qu'une approche paranoïaque reste encore de nos jours la meilleur façon de prévenir les dérives, une option intéressante permet de mettre en oeuvre un filtrage par liste noire des sites ou à l'aide d'expressions régulières :

```
acl allowed_hosts src 192.168.0.0/255.255.0.0
acl localhost src 127.0.0.1/255.255.255.255
acl liste_sites url_regex "/etc/blacklist.txt"
http_access allow localhost
http_access allow allowed_hosts
http_access deny liste_sites
```

On utilise l'ACL **url_regex** pour interdire directement les sites web présents dans une liste noire et le mot clef **urlpath_rgex** pour interdire les sites qui contiennent certains mots-clefs.

Exécutez la ligne suivante pour relancer ou démarrer le serveur (sous root) et tester une nouvelle configuration :

```
# /etc/init.d/squid restart
```

Si le serveur ne veut pas se lancer, il vous faudra consulter le fichier de trace **/var/log/messages** pour connaître l'origine du problème. Sachez que vous devez disposer d'un fichier **/etc/resolv.conf** (contenant l'adresse IP du serveur de nom) correctement rempli pour pouvoir lancer le serveur Squid.

Pour ceux qui souhaitent aller plus loin dans cette démarche de filtrage des URL, vous vous apercevrez rapidement qu'il vous sera impossible de constituer seul une liste exhaustive de sites à interdire.

Dans ce dernier cas, le logiciel **squidGuard** [6] est votre ami. Ce programme a été conçu pour être utilisé en conjonction avec un serveur **Squid** afin de filtrer et interdire l'accès à des sites aux contenus litigieux. Les listes d'interdictions couvrent de nombreux domaines et sont mises à jour régulièrement.

Parmi les principales caractéristiques de ce programme particulièrement efficace et rapide :

- Limiter l'accès au net à partir d'une liste de sites web autorisés,
- Bloquer l'accès à des sites web présents dans une liste noire,

- Bloquer les accès aux sites présentant dans leurs URL certains mots clefs jugés ambigus,
- Interdire les accès aux sites accessibles uniquement sous forme d'adresses IP,
- Rediriger les utilisateurs vers une page d'information lorsqu'un site est interdit,
- Rediriger les bannières de pub vers une image vide locale,
- Gestion des règles d'accès selon les jours et les heures.

Proxy transparent

L'utilisation d'un proxy-cache nécessite la configuration de chaque application du client (généralement, le navigateur) pour préciser l'adresse IP du serveur proxy et le port de connexion (dans notre cas 3128) : Selon le nombre de machines clients à configurer, cela peut rapidement devenir rédhibitoire.

Si cette opération n'est pas réalisée, il vous sera impossible de surfer sur Internet alors que le serveur proxy est pourtant bien en place.

Une seule solution pour résoudre partiellement vos problèmes : Mettre en place un **proxy transparent**.

Ajoutez les lignes suivantes à la fin de votre fichier *squid.conf* :

```
httpd_accel_host virtual
httpd_accel_port 80
httpd_accel_with_proxy on
httpd_accel_uses_host_header on
```

Et activer la ligne suivante dans le script (elle est présente en commentaire) :

```
/sbin/iptables -t nat -A PREROUTING -p tcp -i eth0 --dport 80 -j REDIRECT --
to-port 3128
```

Pour simplifier, toutes les requêtes HTTP à destination du port 80 seront redirigées vers le port 3128, celui du proxy-cache. Par contre, cela ne fonctionne pas dans le cas des protocoles FTP et SSL : Le serveur proxy utilise en effet les en-tête du protocole HTTP pour déterminer ou il est sensé effectuer la connexion.

Ne pas oublier aussi de mettre à disposition un serveur de nom sur la machine passerelle (par exemple **bind**) : En effet, puisque vous ne passez plus en direct par le proxy-cache, ce dernier ne résoudra pas pour vous le nom des machines du réseau Internet et votre navigateur cherchera donc à le faire en local sur votre machine. Cette opération échouera, à moins que vous n'ayez accès à un serveur de nom dans votre Intranet ou que vous décidiez de modifier le parefeu pour laisser passer les requêtes DNS vers Internet.

Et maintenant ?

Votre passerelle est à ce stade parfaitement configurée, quelque soit la solution que vous avez appliquée. Cependant, il reste encore un dernier point à aborder : Que pouvons nous faire pour obtenir la liste nominative des accès à Internet ?

Il est indispensable, dans ce cas, de forcer les utilisateurs à s'authentifier lors du premier accès car sinon vous ne pourrez obtenir que l'adresse IP de la machine appelante dans les traces du serveur **Squid** (ce qui peut amplement suffire, c'est à vous de voir).

Authentification, principes et mise en oeuvre

Squid n'intègre pas directement le support de l'authentification mais s'appuie intelligemment sur des modules externes afin de sous-traiter cette phase. Une authentification demandant fort

logiquement deux paramètres : Un nom d'utilisateur et un mot de passe.

On peut distinguer les plugins suivants :

- LDAP : L'authentification se base sur un annuaire de type Lightweight Directory Access présent sur le réseau. A privilégier lorsque le nombre d'utilisateurs est plutôt élevé : Cela n'a pas d'intérêt particulier dans notre contexte familiale ou de petite PME,
- SMB/MSNT : On utilise une connexion sur un serveur Windows ou le logiciel d'émulation SAMBA pour gérer l'authentification. A moins que votre parc ne soit composé que de machines Windows, cette solution est aussi à écarter,
- GETPWAM : Cette solution est élégante car on utilise directement le fichier de mot de passe système présent sur la machine Linux hébergeant **Squid**. Par contre, cela demande de créer un compte pour chaque utilisateur ce qui peut être rédhibitoire,
- NCSA : On utilise un fichier comportant des noms d'utilisateurs et des mots de passe, le tout au format NCSA. C'est cette solution que nous allons mettre en oeuvre car elle est la plus adaptée à un parc de machines hétérogènes et dans un contexte où le nombre d'utilisateurs est plutôt réduit.

Pour les puristes, l'authentification NCSA est utilisée dans la restriction d'accès des données d'un serveur apache par l'intermédiaire du fichier **.htaccess**. Nous aurons l'occasion de le mettre en oeuvre, par ailleurs, un peu plus loin dans cet article.

Il faut tout d'abord constituer notre fichier de mots de passe dans le répertoire */usr/etc* que nous auront préalablement créé (sous root) :

```
# mkdir -p /usr/etc
# cd /usr/etc
# htpasswd -c passwd [votre nom d'utilisateur]
```

La commande précédente (disponible en général avec le serveur web Apache) vous permet de créer un nouveau fichier de mots de passe (avec l'option -c) et d'ajouter un utilisateur. Il vous faudra alors taper deux fois le mot de passe pour valider la saisie.

Pour ajouter un nouvel utilisateur ou changer son mot de passe, il vous faudra utiliser la même commande mais sans l'option **-c**.

Il faut ensuite ajouter les lignes suivantes au début de votre fichier **squid.conf** et redémarrer le serveur :

```
acl authentification proxy_auth REQUIRED
http_access allow authentification
authenticate_program /usr/sbin/ncsa_auth /usr/etc/passwd
authenticate_children 5
authenticate_ttl 1 hour
authenticate_ip_ttl 60 seconds
```

Le chemin du module **ncsa_auth** peut être différent selon les distributions : Pensez donc à adapter la configuration en conséquence.

Les options **authenticate_ttl** ou **authenticate_ip_ttl** vous permettent aussi de modifier le comportement du processus d'authentification comme le fait de revalider le nom d'un utilisateur et son mot de passe. Celui-ci vous sera demandé à la première connexion Internet ou lorsque vous aurez relancé votre navigateur.

Figure 3 : Exemple d'authentification sous Konqueror

Vous trouvez que l'ajout d'un utilisateur ainsi que le changement du mot de passe n'est pas très pratique ? Nous sommes d'accord avec vous. Nous allons vous présenter deux petits utilitaires qui vont vous simplifier la vie et vaincre vos dernières réticences, en particulier dans le contexte d'une petite PME où le nombre d'utilisateurs peut être assez important.

Gestion des utilisateurs

Nous allons donner la possibilité aux utilisateurs de changer eux même leur mot de passe par le biais d'une interface web. Vous-même ne serez pas oublié car vous pourrez aussi gérer directement la liste des utilisateurs par l'intermédiaire de votre navigateur.

Nous allons pour cela nous appuyer sur deux utilitaires indispensables que nous allons tout d'abord devoir compiler (ils sont uniquement livrés sous forme de source) et qui utilisent les interfaces CGI (Common Gateway Interface) et HTML.

Nous allons donc considérer que vous disposez, sur la machine passerelle, d'un serveur apache configuré et fonctionnel. Vous placerez les binaires ainsi générés directement dans un répertoire de type **cgi-bin** pour les rendre disponibles et accessibles.

A ce stade, il vous reste à déterminer où se situe le répertoire **cgi-bin/** de votre serveur. Un simple grep sur le fichier **httpd.conf** devrait pouvoir faire l'affaire :

```
# grep « ScriptAlias » httpd.conf
```

Pour la suite de cet article, nous allons considérer que le chemin est : */usr/local/httpd/cgi-bin*

Le logiciel qui va vous permettre d'ajouter, supprimer ou encore modifier vos utilisateurs s'appelle **admuser** : Vous pouvez le trouver sur [7].

Il vous reste donc à compiler le programme et à installer le logiciel :

```
# tar xzvf admuser-2.3.tar.gz
# cd admuser-2.3
# ./configure --prefix=/usr/etc --enable-language=French \
--enable-cgidir=/usr/local/httpd/cgi-bin
# make
# make install
```

Ce programme peut supporter plusieurs listes (fichiers) de mots de passe, il vous faut donc préciser l'emplacement d'un premier fichier qui contiendra le nom et le libellé de chaque liste. Dans le cas

qui nous intéresse, voilà ce que cela peut donner :

```
# cat /usr/etc/admuser.conf
password_file /usr/etc/pwd_files
# cat /usr/etc/pwd_files
/usr/etc/passwd;Squid Password File
```

Pour administrer vos utilisateurs, il vous faudra simplement pointer sur l'url *http://localhost/cgi-bin/admuser.cgi* pour obtenir la copie d'écran qui suit. Le reste est assez intuitif.

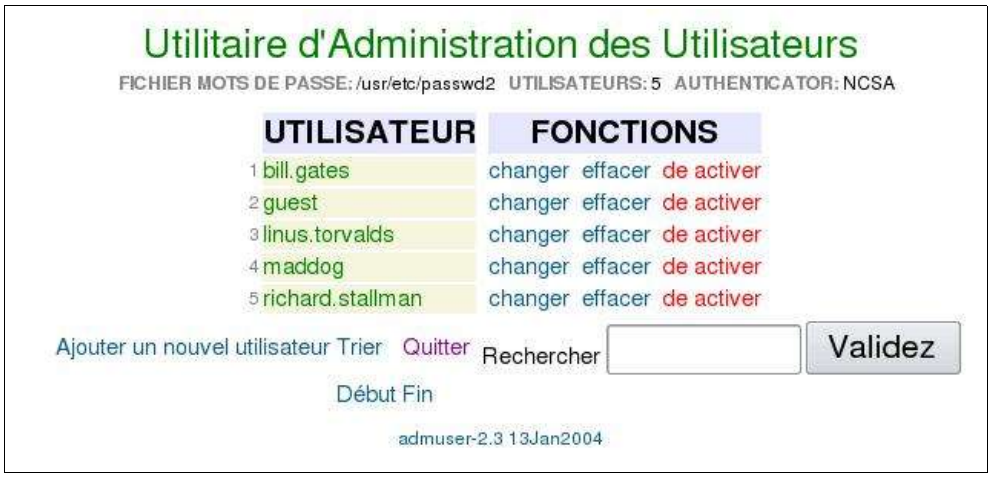


Figure 4 : Administration graphique des utilisateurs sous admuser

Si cela ne fonctionne pas, il vous reste à autoriser apache à exécuter les scripts de type CGI. Vérifiez donc que les éléments suivants sont correctement configurés dans le fichier **httpd.conf** :

```
ScriptAlias /cgi-bin/ "/usr/local/httpd/cgi-bin/

<Directory "/usr/local/httpd/cgi-bin">
    AllowOverride None
    Options None
    Order allow,deny
    Allow from all
</Directory>

<Location /cgi-bin>
AllowOverride None
Options +ExecCGI -Includes
SetHandler cgi-script
</Location>
```

Puis lancez ou relancez votre serveur apache :

```
# /etc/init.d/apache restart
```

La méthode sera identique pour le logiciel **chpasswd** que vous pourrez trouver sur [8] (chpasswd-2.2.1.tar.gz). Configurez alors le fichier **/usr/etc/chpasswd.conf** pour qu'il contienne au moins les lignes suivantes :

```
password_file /usr/etc/passwd
enable_log /usr/etc/chpasswd.log
alert_mail_user root
alert_mail_subject "CHPASSWD EVENT"
```

Tout comme dans l'exemple précédent, vos utilisateurs pourront avoir accès à cette interface à travers l'url *http://localhost/cgi-bin/chpasswd.cgi* qui se révélera, à l'utilisation, suffisamment intuitive pour ne pas nécessiter de plus amples explications.

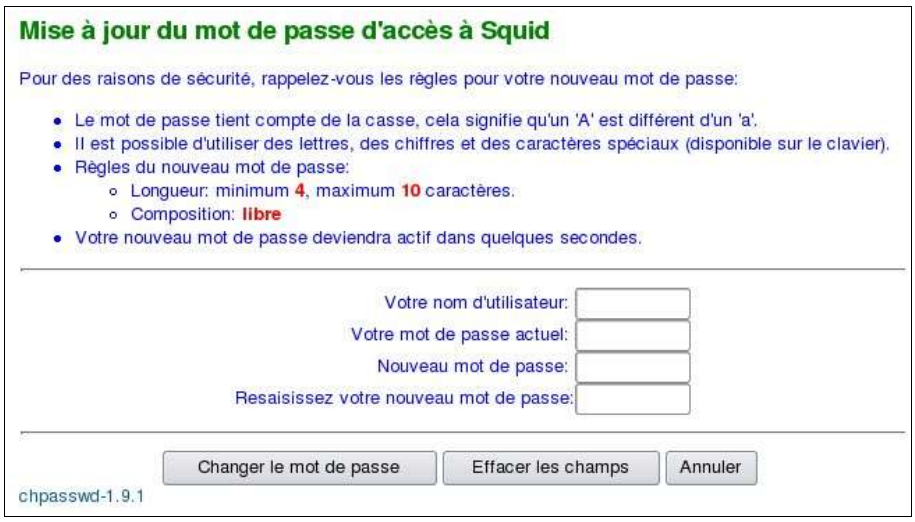


Figure 5 : Changement du mot de passe à travers une interface web sous chpasswd

La dernière action à faire sera de créer une page d'accueil sur votre machine passerelle pointant au minimum sur l'interface de mise à jour des mots de passe et de configurer cette page comme URL par défaut sur tous les navigateurs clients.

Vous noterez que, en bon administrateur, il vous est donné un moyen de tracer tous les appels à ce programme pour traquer les tentatives de fraudes (changements de mot de passe), en l'occurrence le fichier de log **/usr/etc/chpasswd.log**.

Les plus malins d'entre vous auront noté une faille dans notre raisonnement : Comment éviter qu'un utilisateur indélicat ajoute un nouvel utilisateur fantôme puisque, en pratique, les deux programmes peuvent être appelés par n'importe qui ?

Pour éviter ce cas de figure, il vous faudra disposer de deux répertoire de type cgi-bin : L'un hébergeant le programme chpasswd, le second le programme admuser. Dans le second répertoire, par exemple **cgi-bin2/**, vous placerez un fichier **.htaccess** qui sera utilisé par le serveur Apache pour restreindre l'accès au répertoire en demandant, avant tout préalable, un nom d'utilisateur et un mot de passe.

Le fichier devra présenter le contenu suivant :

```
AuthUserFile /usr/etc/.htpasswd
AuthName admin
AuthType basic
<Limit GET>
require valid-user
</Limit>
```

Il restera à votre charge de créer un fichier **.htpasswd** (possédant les droits 644) comportant l'utilisateur **admin**. Vous utiliserez pour ce faire le programme htpasswd cité plus tôt dans ce document dont nous allons détailler la mise en oeuvre maintenant.

Pour activer le support des fichiers .htpasswd, utilisez la directive « AllowOverride All » dans le serveur apache :

```
<Directory "/usr/local/httpd/cgi-bin2">
    AllowOverride All
    Options None
    Order allow,deny
    Allow from all
</Directory>
```

Une superbe fenêtre d'authentification vous sera demandée pour tout appel au programme d'administration des utilisateurs. Pas mal non ?

Traçabilité des accès

La phase logique qui suit l'authentification des utilisateurs reste celle de l'exploitation des logs générés par le serveur **Squid**.

Vous pouvez, à tout instant, consulter manuellement ces derniers dans le fichier **access.log** qui se trouve habituellement dans le répertoire */var/log/squid* (paramétrable dans le fichier squid.conf) :

```
[...]
1083327810.189   1301 192.168.2.51 TCP_MISS/200 8620 GET http://login.yahoo.com/config/login?
prenom.nom DIRECT/216.109.127.60 text/html
1083327810.409   515 192.168.2.51 TCP_MISS/200 359 GET http://66.102.9.99/search? prenom.nom
DIRECT/66.102.9.99 text/html
1083327810.459   562 192.168.2.51 TCP_MISS/200 934 GET http://uk.adserver.yahoo.com/a?
prenom.nom DIRECT/217.12.4.96 application/x-javascript
1083327810.949   367 192.168.2.51 TCP_REFRESH_HIT/304 212 GET
http://eur.a1.yimg.com/eur.yimg.com/a/eu/any/0304ukbuttonloisirs425x600.gif prenom.nom
DIRECT/212.73.231.112 image/gif
1083327834.707   882 192.168.2.57 TCP_MISS/200 1691 GET http://images.google.fr/favicon.ico
prenom.nom DIRECT/66.102.9.99 image/x-icon
[...]
```

Vous noterez que, selon votre distribution Linux, vous utilisez peut être sans le savoir le programme **logrotate** afin de purger vos fichiers de logs. Dans ce cas, vérifiez que le champ *size* du fichier **/etc/logrotate.d/squid** est suffisamment important pour ne pas perdre d'informations (adaptez-le à la fréquence d'analyse de vos logs).

Par contre, il semble plutôt impossible d'exploiter directement ce fichier tant le volume de données à traiter peut rapidement excéder celui de la capacité d'analyse normale d'un être humain. Heureusement, plusieurs logiciels existent qui excellent dans ce travail de synthèse : Calamaris, Webalizer, Squid-Log-Analzer, SARG, ... (une liste est disponible sur [9]).

Nous allons, pour notre part, vous proposer d'utiliser le logiciel SARG [10] pour les raisons suivantes :

- Ce dernier est proposé par les mêmes auteurs que ceux qui ont commis admuser et chpasswd,
- Sa mise en oeuvre est plutôt agréable et accessible,
- Ce logiciel va digérer les logs du serveur Squid avant de recracher une petite arborescence web statique que vous pourrez consulter à travers un simple navigateur.

Téléchargez le fichier sarg-1.4.1.tar.gz (des paquets sont disponibles sur le site) et installez-le :

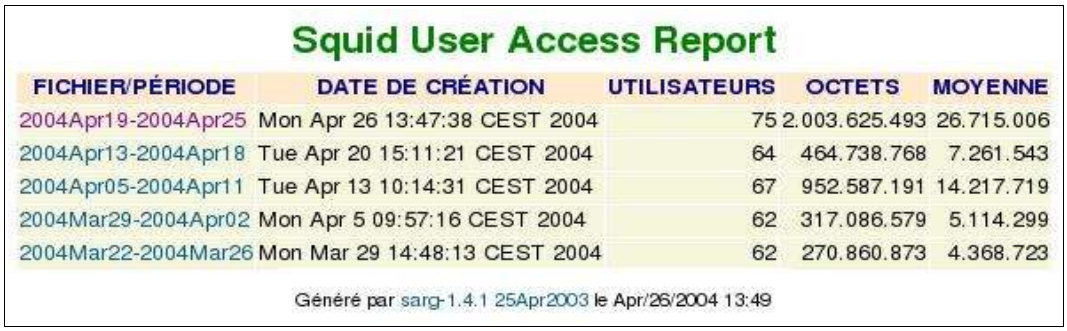
```
# tar xzvf sarg-1.4.1.tar.gz
# cd sarg-1.4.1
# ./configure --enable-sysconfdir=/usr/etc
# make
# make install
```

Vous devez ensuite configurer SARG pour qu'il aille lire le fichier de traces du serveur **Squid** :

```
# cat /usr/etc/sarg.conf
language French
access_log /var/log/squid/access.log
```

Il vous sera alors possible de trier les accès selon les critères suivants :

- Liste des cent sites les plus visités avec la bande passante utilisée et le nombre d'accès,
- Liste de tous les sites visités avec le nom des utilisateurs,
- Liste des utilisateurs triés sur le critère de la bande passante utilisée,
- Par utilisateur :
  - La liste des sites visités par ordre d'importance (nombre d'accès ainsi que la date et l'heure des visites),
  - Un planning des connexions pour balayer d'un seul coup d'oeil la charge par utilisateur,
  - Par site, la date et l'heure de tous les accès,
- La liste des connexions échouées (utile pour localiser les logiciels espions).



Squid User Access Report

FICHIER/PÉRIODE	DATE DE CRÉATION	UTILISATEURS	OCTETS	MOYENNE
2004Apr19-2004Apr25	Mon Apr 26 13:47:38 CEST 2004	75	2.003.625.493	26.715.006
2004Apr13-2004Apr18	Tue Apr 20 15:11:21 CEST 2004	64	464.738.768	7.261.543
2004Apr05-2004Apr11	Tue Apr 13 10:14:31 CEST 2004	67	952.587.191	14.217.719
2004Mar29-2004Apr02	Mon Apr 5 09:57:16 CEST 2004	62	317.086.579	5.114.299
2004Mar22-2004Mar26	Mon Mar 29 14:48:13 CEST 2004	62	270.860.873	4.368.723

Généré par sarg-1.4.1 25Apr2003 le Apr/26/2004 13:49

Figure 6 : Consultation de vos connexions Internet à travers le logiciel SARG

Sauf besoin ponctuel, il est préférable d'automatiser la génération de ces informations en plaçant l'appel à la commande **sarg** dans la CRONTAB du système.

Exemple d'utilisation en ligne de commande :

```
# /usr/bin/sarg -n -o /home/maison/public_html/logs/ -i
```

Dans le cadre d'une utilisation professionnelle de ces trois derniers logiciels (admuser, chpasswd et sarg), et si vous estimez que vous gagnez grâce à eux en efficacité, je vous invite à faire une petite donation sur la page [11] afin de motiver les développeurs qui font tout de même un excellent boulot. Cette règle vaut en générale pour tous les autres projets libres, cela n'est pas inutile de le rappeler.

Le mot de la fin

Linux constitue une plate-forme idéale pour partager une connexion Internet entre plusieurs machines. N'hésitez donc pas à placer ce type de service critique entre les mains du pingouin :

Correctement configuré, la sécurité et les performances seront largement au rendez-vous.

Par contre, quelque soit la solution technique que vous avez sélectionnée pour partager votre connexion haut-débit (utilisation d'un proxy-cache ou configuration de votre machine en passerelle IP), je ne saurais vous conseiller d'activer au minimum un pare-feu logiciel. Ne jamais oublier que le haut-débit offre comme effet pervers de constituer une cible de choix plus évidente que lorsque vous vous connectiez par modem RTC de part sa connexion permanente.

Avant de finir, un petit récapitulatif exhaustif des logiciels utilisés pour vous prouvez que le libre est un monde formidable : Linux pour le système d'exploitation couplé à Netfilter pour le routage des paquets IP, Apache pour le serveur web, Squid pour le proxy-cache, Squiguard pour le filtrage préventifs des sites, admuser/chpasswd/sarg pour l'authentification et le suivi des connexions.

N'hésitez pas à me contacter si vous trouvez une faille ou un point à améliorer dans le script pare-feu/passerelle : Toutes les contributions seront les bienvenues.

Lionel Tricon

Références :

[1] Script Netfilter : htttp://lionel.tricon.free.fr/Articles/proxy/LISEZMOI.TXT

[2] Netfilter : http://www.netfilter.org/

[3] Ad-aware : http://www.lavasoftusa.com/software/adaware/

[4] Spybot : http://www.safer-networking.org/

[5] Dante : http://www.inet.no/dante/

[6] Squidguard : http://www.squidguard.org/

[7] Logiciel admuser : http://sarg.sourceforge.net/admuser.php

[8] Logiciel chpasswd : http://sarg.sourceforge.net/chpasswd.php

[9] Liste de programmes analyseurs : http://www.squid-cache.org/Scripts/

[10] Logiciel SARG : http://sarg.sourceforge.net/

[11] Page de donation : http://sarg.sourceforge.net/donations.php