Interconnexion de deux LAN via VPN, avec une passerelle CheckPoint et une passerelle Linux-noyau2.6+IPsec-tools

Document version 1.1a du 13 décembre 2004, Emmanuel DESVIGNE (emmanuel@desvigne.org)

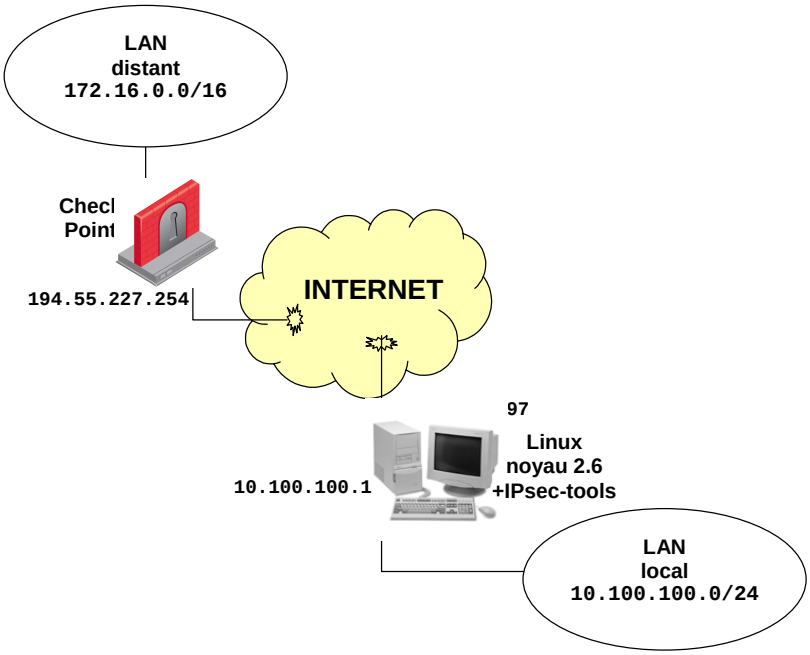
Table des matières :

1 Principe

Le présent document décrit comment interconnecter de façon sécurisée deux LANs en utilisant Internet, et en créant un VPN. Un des deux LANs, que nous appellerons par la suite « *réseau distant* », est connecté à Internet via une solution « *CheckPoint* » (http://www.checkpoint.com/). L'autre LAN, appelé par la suite « *réseau local* », se connectera en utilisant une passerelle sous Linux (impérativement avec un noyau 2.6.x minimum) et « *IPsec-tools* » (http://sourceforge.net/projects/ipsec-tools), une solution d'outils de gestion de VPN, développée initialement sous *BSD.

Par la suite, nous utiliserons les conventions suivantes (les adresses IP utilisées ici ne sont évidemment que des exemples ; il ne s'agit pas d'un cas concret) :

- LAN distant : `172.16.0.0/16`
- Adresse IP publique de la passerelle CheckPoint de ce LAN : `194.55.227.254`
- Adresse IP publique de la passerelle Linux-noyau2.6+IPsec : `82.66.254.97`
- LAN qui utilisera cette passerelle Linux comme routeur : `10.100.100.0/24`



2 Logiciels nécessaires sur la passerelle Linux

Le présent chapitre décrit comment installer par vous même « *IPsec-tools* » sous Linux. Si vous n'êtes pas curieux, et si vous utilisez une distribution comme la « *RedHat FedoraCore 3* » qui contient déjà les modules du noyau et les packages qui vont bien, je vous invite à passer directement au chapitre suivant.

2.1 Généralités

Plusieurs solutions existent pour créer des VPNs sous Linux. La plupart du temps, il s'agit de solutions relativement anciennes, qui risquent de ne plus être maintenues (si elles ne le sont pas déjà), et développées pour pallier au fait que le noyau Linux ne supportait pas nativement IPsec (exemple : FreeS/WAN). Or, depuis le noyau version 2.5.49, la pile IP du noyau Linux supporte de façon native IPsec (AH et ESP).

Aussi, il est devenu possible de compiler directement des produits développés à l'origine pour d'autres systèmes d'exploitations. Actuellement, un produit semble le plus utilisé : il s'agit de « *IPsec-tools* », développé initialement sous *BSD (avant d'être porté sous Linux, la version BSD s'appelait « *KAME's* »).

Le produit se compose d'une bibliothèque de fonctions « *libipsec* », d'un programme de configuration de IPsec dans le noyau (« *setkey* »), et d'un démon (« *racoon* ») qui gère le protocole d'échange de clés « *IKE* » (Internet Key Exchange Protocol, Cf. RFC 2409).

Attention : toutes les distributions, même récentes et utilisant un noyau 2.6.x, ne livrent pas nécessairement un noyau incluant IPsec. Par exemple, pour rédiger le présent document, deux distributions ont été testées :

- RedHat FedoraCore 3 : contient « *IPsec-tools* » et le noyau est compilé avec « *IPsec* » ;
- Mandrake 10.1 version finale : noyau 2.6 sans « *IPsec* », et « *IPsec-tools* » non fourni.

2.2 Options du noyau

Si les modules nécessaires pour faire du VPN sont compilés dans FedoraCore 3, ça n'est pas le cas pour la Mandrake.

Dans ce cas, il faut recompiler le noyau (pour ma part, j'ai téléchargé le 2.6.9, disponible ici : http://www.kernel.org/pub/linux/kernel/v2.6/), et activer les options suivantes (de façon native, ou sous forme de module) :

Device driver / Networking support / Networking options :

```
<*> PF_KEY sockets
...
<*> IP: AH transformation
<*> IP: ESP transformation
<*> IP: IPComp transformation
...
<*>   IPv6: AH transformation
<*>   IPv6: ESP transformation
<*>   IPv6: IPComp transformation
...
<*> IPsec user configuration interface
```

Evidemment, les lignes Ipv6... ne sont à activer que si vous compilez IP version 6 dans votre noyau. Dans les exemples du présent document, nous utiliserons l'algorithme de cryptage « *3des* », et l'algorithme « *md5* » comme algorithme de signature. Ces deux algorithmes sont automatiquement compilés dans le noyau. Toutefois, si vous en utilisez d'autres, n'oubliez pas de les compiler aussi dans le noyau (tous ne le sont pas par défaut dans « Cryptographic options »).

L'objectif ici n'est pas de faire un cours sur la compilation du noyau Linux. Pour les distributions citées ci-dessus, il faut faire un « `make oldconfig` » ou « `make menuconfig` » ou « `make xconfig` », selon votre préférence, afin d'activer les options indiquées précédemment. Ensuite, ne pas oublier le « `make bzlilo` » ou le « `make bzImage` », le « `make modules` » et le « `make modules_install` », le « `mkinitrd fichier_init.img version_noyau` », et enfin, la réinstallation de « *lilo* » ou de « *grub* » (ou tout autre logiciel de multiboot).

2.3 Outils nécessaires à la compilation de IPsec-tools

Pour compiler soit même « *IPsec-tools* », sachez qu'il est nécessaire de posséder les packages suivants, qui ne sont pas toujours livrés en standard avec toutes les distributions Linux :

- « *lex* » ou « *flex* »,
- « *yacc* » ou « *bison* »,
- « *open-ssl* ».

Et si vous souhaitez compiler vous-même « *flex* » et « *bison* », il vous faudra aussi le package « *m4* ». Voici les liens pour télécharger les trois programmes GNU :

- ftp://ftp.gnu.org/gnu/m4/
- ftp://ftp.gnu.org/gnu/non-gnu/flex/
- ftp://ftp.gnu.org/gnu/bison/

Il suffit de télécharger la dernière archive des sources (en .tar.gz ou .tar.bz2), de la décompresser, de se mettre sous shell logué root dans le répertoire des sources, et de faire un :

```
# ./configure --prefix=/usr --sysconfdir=/etc --localstatedir=/var
# make && make install
```

Pour « *open-ssl* », les sources sont disponibles ici : http://www.openssl.org/. Actuellement, il n'existe pas de fichier de configuration automatique « `./configure` ». Toutefois, la compilation et l'installation de ce package sous Linux est simple :

```
# ./config
```

et éditez le fichier « *Makefile* » afin de remplacer la ligne : « `INSTALLTOP=/usr/local/ssl` » par « `INSTALLTOP=/usr` ». Il suffit de faire ensuite un classique :

```
# make && make install
```

2.4 Compilation de "*IPsec-tools*"

Ce produit est disponible à l'adresse : http://sourceforge.net/projects/ipsec-tools. La version 0.3 (qui n'est pas la dernière version disponible) est livrée dans la Redhat FedoraCore 3.

Remarque : dans la version stable à l'heure où ce document est écrit (version 0.4), il existe une petite erreur (corrigée depuis dans les pré-releases de la version 0.5) qui empêche la compilation du fichier « `src/racoon/sockmisc.c` ». Dans la fonction « `saddrwop2str()` », voici quelques modifications à effectuer (texte en rouge) :

```
char *
saddrwop2str(saddr)
        const struct sockaddr *saddr;
{
        static char buf[NI_MAXHOST + NI_MAXSERV + 10];
        char addr[NI_MAXHOST];
        // Déclarer une variable "port" :
        char port[NI_MAXSERV];

        if (saddr == NULL)
                return NULL;

        // et remplacez "GETNAMEINFO(saddr, addr, NULL);" par :
        GETNAMEINFO(saddr, addr, port);
        snprintf(buf, sizeof(buf), "%s", addr);

        return buf;
}
```

Une fois cette correction effectuée, compilation et installation se font avec un simple :

```
# CFLAGS="-O3" ./configure --prefix=/usr \
      --bindir=/usr/bin \
      --sbindir=/usr/sbin \
      --sysconfdir=/etc \
      --localstatedir=/var \
```

```
        --libdir=/usr/lib \
        --enable-static
    # make && make install
```

Remarque : avec la « *Mandrake* », j'ai eu un souci lors de l'édition de lien. J'ai du rajouter moi même dans les différents fichiers « *Makefile* » l'option « `-ldl` » (ajouter « `-ldl` » à la fin de la ligne « `LIBS=...` »).

3 Configuration du VPN

3.1 Généralités sur les VPN

Pour créer un VPN inter-LAN, il existe 3 solutions :

- Clés secrètes partagées (*Shared Secret Keys*) : sur les deux passerelles, les clés des algorithmes de chiffrement et de signatures de IPsec sont positionnées manuellement une fois pour toute. Simple à mettre en œuvre, cette solution n'est pas des plus sûres ;
- Clé pré-partagée (*Pre-Shared Key*) : ici, une clé est convenue entre les deux sites. Cette clé est utilisée par un protocole (*IKE*), pour sécuriser la négociation automatique des algorithmes de chiffrement et de signature, les clés de cryptage et de signature, le temps au bout duquel ces clés seront changées (renégociation automatique), etc. Dans la solution « *IPsec-tools* », ce protocole « *IKE* » est implémenté au travers du démon « *racoon* ». Coté sécurité, cette technique, qui change régulièrement de clés IPsec, est plus sûre que la première méthode ;
- Certificat X.509 : il s'agit de la méthode la plus sûre, mais elle nécessite la possession d'un certificat (délivré par une autorité de certification). Ici aussi, c'est le démon « *racoon* » qui gère l'utilisation de ce certificat pour la négociation des clés IPsec.

Dans les exemples suivants, nous utiliserons la technique « *Pre-Shared Key* ».

Le protocole « *IKE* » se décompose en une négociation en deux phases :

- phase 1 : négociation de « *ISAKMP SA* » (selon deux modes : soit « *main mode* », soit « *aggressive mode* ») : il s'agit de la négociation des algorithmes de chiffrement et de signature, des clés de ces algorithmes, des durées de validité de ces clés (en temps ou en nombre d'octets échangés), qui seront utilisés lors de la phase 2 ;
- phase 2 : quick mode (avec ou sans PSF) : utilisation des algorithmes et clés négociés en phase 1 pour négocier les clés et algorithmes qui seront cette fois-ci utilisés par IPsec (AH et ESP).

Remarque concernant PSF (« *Perfect Forward Secrecy* »). C'est un mode de fonctionnement paranoïaque où les clés négociées en phase 1 sont de nouveau signées en phase 2. Il est alors aussi possible de spécifier la longueur de la clé utilisée pour cette signature (groupes Diffie-Hellmann, comme en phase 1).

3.2 Exemple de configuration

Suivant le schéma décrit au début du présent document, voici un exemple de configuration :

- Méthode utilisée en phase 1 : « *main mode* » (pas d'utilisation du « *mode agressif* »),
- Utilisation de la méthode « *Pre-Shared Key* » (pas de certificat X.509),
- Algorithme de cryptage utilisé en phase 1 : 3des,
- Algorithme de signature utilise en phase 1 : md5,
- Durée de validité des clés négociées en phase 1 : 1440 minutes,
- Groupe de Diffie-Hellmann (Cf. RFC 2409) en phase 1 : groupe 2 (1024 bits),
- Utilisation du « Perfect Forward Secrecy » en phase 2 : non,
- Algorithme de cryptage utilisé en phase 2 : 3des,
- Algorithme de signature utilise en phase 2 : md5,
- Durée de validité des clés IPsec négociées en phase 2 : 3660 secondes,
- Clé secrete : bla-bla-bla

Avec ce schéma, les différentes actions à effectuer sous Linux sont :

- Définir la police IPsec (prévenir le noyau que les flux entre les LANs `172.16.0.0/16` et `10.100.100.0/24` se feront avec un VPN entre les passerelles ayant comme adresse `194.55.227.254` et `82.66.254.97`. Cette opération s'effectue avec la commande « *`setkey`* » :

```
# setkey -c
flush;
spdflush;

spdadd 10.100.100.0/24 172.16.0.0/16 any -P out ipsec
esp/tunnel/82.66.254.97-194.55.227.254/require;

spdadd 172.16.0.0/16 10.100.100.0/24 any -P in  ipsec
esp/tunnel/194.55.227.254-82.66.254.97/require;

^D    (CTRL + D)
```

- Configurer le démon « *racoon* » et le lancer afin qu'il effectue les négociations « *IKE* » selon les paramètres définis ci-dessus. Sans certificat X.509, le démon « *racoon* » n'utilise alors que deux fichiers :
  - le fichier « */etc/racoon/psk.txt* », qui contient la clé secrète convenue entre les deux sites. Attention : ce fichier doit absolument appartenir à l'utilisateur « *root* », et doit avoir comme droits Unix « *600* ». Il est constitué de lignes ayant la forme :
    `adresse_ip_distante  clé_partagée`
  - le fichier « */etc/racoon/racoon.conf* ». Il contient deux principales sections : la section « *remote* », qui définit les paramètres négociés en phase 1, et la section « *sainfo* », qui contient les paramètres utilisés en phase 2. Faire un « `man racoon.conf` » pour plus de détails sur la syntaxe de ce fichier de configuration ;
- Lancer le démon « *racoon* » ;
- Activer le routage dans le noyau. En effet, dans la plupart des distributions, la fonction routage n'est pas activée. De façon simple, il suffit d'écrire la valeur « 1 » dans le fichier « `/proc/sys/net/ipv4/ip_forward` » ;
- A ce stade, toute machine du réseau « `10.100.100.0/24` » qui aurait comme passerelle par défaut « `10.100.100.1` » arrivera à dialoguer avec le réseau « `172.16.0.0/16` ». Par contre, la passerelle Linux elle même ne peut dialoguer avec ce réseau « `172.16.0.0/16` ». En effet, si, sur la passerelle Linux, on faisait un « `ping 172.16.x.y` », le paquet serait directement envoyé à sa « *default gateway* » depuis l'interface publique. Nous pouvons forcer la passerelle Linux à utiliser le VPN pour causer au LAN distant en ajoutant la route suivante :

```
# route add -net 172.16.0.0/16 gw 10.100.100.1
```

L'ensemble de ces opérations peut être automatisée en utilisant les deux scripts suivants :

- le script « `start-vpn.sh` », qui active le VPN :

```
#!/bin/sh
# fichier "start-vpn.sh"

MON_ADRESSE_PUBLIQUE="82.66.254.97"
MON_ADRESSE_PRIVEE="10.100.100.1"
MON_RESEAU_PRIVE="10.100.100.0/24"
ADRESSE_PUBLIQUE_DISTANTE="194.55.227.254"
RESEAU_PRIVE_DISTANT="172.16.0.0/16"
ALGO_ENC_PHASE1="3des"
ALGO_AUT_PHASE1="md5"
LIFETIME_TIME_PHASE1="1440 min"
DIFFIE_HELLMANN_GROUP="2"
ALGO_ENC_PHASE2="3des"
```

```
ALGO_AUT_PHASE2="hmac_md5"
LIFETIME_TIME_PHASE2="3660 sec"
CLE_SECRETE="bla-bla-bla"
REP_CONF_RACOON="/etc/racoon"
FIC_LOG_RACOON="/var/log/racoon.log"

# 1) Enregistre la police IPsec
cat > $REP_CONF_RACOON/setkey.conf <<EOF
flush;
spdflush;
# Linux-2.6.9+racoon -> CHU CheckPoint
spdadd $MON_RESEAU_PRIVE $RESEAU_PRIVE_DISTANT any -P out ipsec
esp/tunnel/$MON_ADRESSE_PUBLIQUE-$ADRESSE_PUBLIQUE_DISTANTE/require;
# CHU CheckPoint > Linux-2.6.9+racoon
spdadd $RESEAU_PRIVE_DISTANT $MON_RESEAU_PRIVE any -P in  ipsec
esp/tunnel/$ADRESSE_PUBLIQUE_DISTANTE-$MON_ADRESSE_PUBLIQUE/require;
EOF
setkey -f $REP_CONF_RACOON/setkey.conf

# 2) Crée le fichier de configuration de racoon
if [ ! -d $REP_CONF_RACOON ] ; then
      mkdir $REP_CONF_RACOON
fi
if [ ! -d $REP_CONF_RACOON/cert ] ; then
      mkdir $REP_CONF_RACOON/cert
fi
cat > $REP_CONF_RACOON/racoon.conf <<EOF
path include "$REP_CONF_RACOON";
path pre_shared_key "$REP_CONF_RACOON/psk.txt";
path certificate "$REP_CONF_RACOON/cert";

# "padding" defines some parameter of padding.
# You should not touch these.
padding
{
      maximum_length 20;      # maximum padding length.
      randomize off;          # enable randomize length.
      strict_check off; # enable strict check.
      exclusive_tail off;     # extract last one octet.
}

# Specification of default various timer.
timer
{
      # These value can be changed per remote node.
      counter 5;        # maximum trying count to send.
      interval 20 sec;  # maximum interval to resend.
      persend 1;        # the number of packets per a send.
      # timer for waiting to complete each phase.
      phase1 30 sec;
      phase2 15 sec;
}

# Niveau de log pour le deboguage (notify=minimum, debug=moyen,
# et debug2=maximum).
log notify;

# Parametres de la phase 1 :
remote $ADRESSE_PUBLIQUE_DISTANTE
{
      exchange_mode main;
      lifetime time $LIFETIME_TIME_PHASE1;
      proposal {
            authentication_method pre_shared_key;
            dh_group $DIFFIE_HELLMANN_GROUP;
              encryption_algorithm $ALGO_ENC_PHASE1;
```

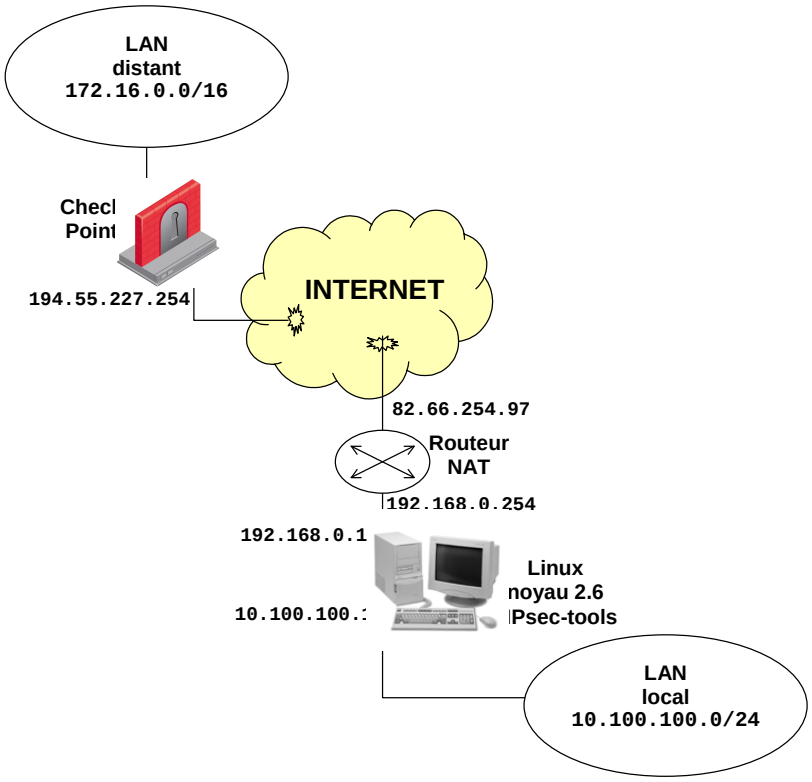
```
                hash_algorithm $ALGO_AUT_PHASE1;
        }
}

# Parametres de la phase 2 (net2net)
sainfo address $MON_RESEAU_PRIVE any address $RESEAU_PRIVE_DISTANT any
{
        compression_algorithm deflate;
        lifetime time $LIFETIME_TIME_PHASE2;
        encryption_algorithm $ALGO_ENC_PHASE2;
        authentication_algorithm $ALGO_AUT_PHASE2;
}
EOF

# 3) Création du fichier qui contient la clé secrète
echo $ADRESSE_PUBLIQUE_DISTANTE $CLE_SECRETE > $REP_CONF_RACOON/psk.txt
chmod 600 $REP_CONF_RACOON/psk.txt
chown -R root.root $REP_CONF_RACOON

# 4) Lance le demon racoon (uniquement en IP v4 pour l'instant)
if [ -f $FIC_LOG_RACOON ] ; then
      mv -f $FIC_LOG_RACOON $FIC_LOG_RACOON.old
      touch $FIC_LOG_RACOON
fi
racoon -4 -f $REP_CONF_RACOON/racoon.conf -l $FIC_LOG_RACOON

# 5) Active le routage
echo 1 > /proc/sys/net/ipv4/ip_forward

# 6) Ajoute la route pour forcer la passerelle Linux a utiliser le VPN
route add -net $RESEAU_PRIVE_DISTANT gw $MON_ADRESSE_PRIVEE

exit 0    # FIN du fichier "start-vpn.sh"
```

- le script « `stop-vpn.sh` », qui arrête le VPN :

```
#!/bin/sh
# fichier "stop-vpn.sh"

MON_ADRESSE_PRIVEE="10.100.100.1"
RESEAU_PRIVE_DISTANT="172.16.0.0/16"

# 1) Annulation du routage vers le site distant
route del -net $RESEAU_PRIVE_DISTANT gw $MON_ADRESSE_PRIVEE

# 2) Arrêt du démon racoon
PID_RACOON=`pidof racoon`
if [ "$PID_RACOON" != "" ] ; then
      kill -SIGTERM $PID_RACOON
fi

# 3) Remise a zéro de la police IPsec
setkey -c <<EOF
flush;
spdflush;
EOF

exit 0    # FIN du fichier "stop-vpn.sh"
```

3.3 Cas d'une passerelle « cachée » derrière un routeur NAT

Dans l'exemple précédent, la passerelle NAT a une interface publique directement connectée à Internet. Or, nous pouvons imaginer que le VPN se fasse au travers d'un routeur NAT, selon le schéma suivant :



Dans ce schéma, la passerelle Linux communique vers Internet en utilisant comme passerelle par défaut (« *default gateway* ») l'adresse IP du routeur NAT « `192.168.0.254` ».

Ce cas ne change quasiment rien par rapport à l'exemple du chapitre précédent. En effet, du point de vue de la passerelle « *Check Point* », les paquets IP sont toujours reçus de l'adresse IP « `82.66.254.97` ».

Dans les scripts « `start-vpn.sh` » et « `stop-vpn.sh` », il suffit de mettre :

```
MON_ADRESSE_PUBLIQUE="192.168.0.1"
```

Pour que tout fonctionne, il faut que :

- la passerelle Linux ait pour passerelle par défaut « `192.168.0.254` »,
- le routeur NAT doit être configuré pour re-router le port 500/UDP vers l'adresse « `192.168.0.1` » (les négociations « *IKE* » se font via ce numéro de port),
- et que ce routeur doit savoir router les paquets IPsec.

Remarque : dans ce schéma, rien n'empêche en pratique que la passerelle ne possède en fait qu'une seule carte réseau. L'adresse IP « `192.168.0.1` » est située sur l'interface « `eth0` », et l'adresse IP « `10.100.100.1` » est affectée à l'interface « `eth0:0` » (utilisation de la notion d'alias sous Linux). Dans ce cas, les réseaux IP « `192.168.0.0/24` » et « `10.100.100.0/24` » sont alors physiquement situés sur le même brin Ethernet. Aussi, pour toute machine connectée physiquement à ce brin, il est possible de lui assigner deux adresses IP : une adresse « `192.168.0.x` » avec comme « *default gateway* » l'adresse « `192.168.0.254` », et un alias pour cette même carte réseau « `10.100.100.y` », et une route :

- si la machine tourne sous Linux : `route add -net 172.16.0.0/16 gw 10.100.100.1`
- si la machine tourne sous Windows 2K, XP, ou 2003 : `ROUTE ADD 172.16.0.0 MASK 255.255.0.0 10.100.100.1`

Attention : en terme de sécurité, cette machine devient sensible. En effet, si elle venait à être piratée (par exemple en se faisant infecter par un cheval de Troie), un pirate qui arriverait à la contrôler pourrait l'utiliser comme passerelle pour accéder au site distant.

3.4 Les logs de « *racoon* »

Lorsque tout fonctionne normalement, voici ce qui est lisible dans les fichiers logs de « *racoon* » (dans notre exemple, fichier « `/var/log/racoon.log` » pour un niveau de log de type « *notify* ») :

```
2004-12-12 15:22:12: INFO: @(#)ipsec-tools 0.4 (http://ipsec-
tools.sourceforge.net)
2004-12-12 15:22:12: INFO: @(#)This product linked OpenSSL 0.9.7e 25 Oct 2004
(http://www.openssl.org/)
2004-12-12 15:22:12: INFO: ::1[500] used as isakmp port (fd=6)
2004-12-12 15:22:12: INFO: 10.100.100.1[500] used as isakmp port (fd=7)
2004-12-12 15:22:12: INFO: 82.66.254.97[500] used as isakmp port (fd=8)
2004-12-12 15:22:12: INFO: 127.0.0.1[500] used as isakmp port (fd=9)
2004-12-12 15:22:23: INFO: IPsec-SA request for 194.55.227.254 queued due to no
phase1 found.
2004-12-12 15:22:23: INFO: initiate new phase 1 negotiation:
82.66.254.97[500]<=>194.55.227.254[500]
2004-12-12 15:22:23: INFO: begin Identity Protection mode.
2004-12-12 15:22:23: INFO: ISAKMP-SA established 82.66.254.97[500]-
194.55.227.254[500] spi:d8607a7def600b90:1901e4a23ee72d1d
2004-12-12 15:22:24: INFO: initiate new phase 2 negotiation:
82.66.254.97[0]<=>194.55.227.254[0]
2004-12-12 15:22:24: WARNING: ignore RESPONDER-LIFETIME notification.
2004-12-12 15:22:24: INFO: IPsec-SA established: ESP/Tunnel 194.55.227.254-
>82.66.254.97 spi=96943484(0x5c73d7c)
2004-12-12 15:22:24: INFO: IPsec-SA established: ESP/Tunnel 82.66.254.97-
>194.55.227.254 spi=2227466425(0x84c470b9)
```