Sécurité de l'information
Tunnels et VPN

Laurent Archimède
Thomas Chevalier
Julien Herbin
Sylvestre Ledru
Nicolas Pellegrin

DESS ISYDIS

4 mai 2004

Table des matières

1 Introduction

La quasi totalité des flux d'informations sur Internet utilisent le protocole TCP/IP (couches 3 et 4). Une grande partie des protocoles (POP/SMTP/FTP) utilisent d'autres protocoles de communication au mieux binaires, au pire en clair. Les protocoles encryptant leurs données sont relativement spécialisés et donc assez peu utilisés (ssh, https, pop3-ssl...) pourtant, il est extrêmement facile sur Internet pour une personne ayant accès à un périphérique de routage (serveur classique ou routeur) de voir tous les paquets circulants sur le réseau.

Par exemple, on peut facilement analyser les trames relatives à la consultation des emails (protocole POP3). Par défaut dans le protocole POP3, les mots de passe circulent en clair sur le réseau. Ainsi, il est facile d'obtenir le mot de passe du compte pop d'un utilisateur sans pour autant avoir le moindre accès au poste client ou au serveur.

Avec la commande ngrep, on voit très rapidement les problèmes de sécurité posés par ce protocole.

```
T 217.167.120.134:110 -> 81.249.31.222:1817 [AP]
  +OK <8182.1078671690@mail-hitomi.ecranbleu.org>..
#
T 81.249.31.222:1817 -> 217.167.120.134:110 [AP]
  CAPA..
##
T 217.167.120.134:110 -> 81.249.31.222:1817 [AP]
  -ERR authorization first..
#
T 81.249.31.222:1817 -> 217.167.120.134:110 [AP]
  USER berangere@jovialyteam.com..
#
T 217.167.120.134:110 -> 81.249.31.222:1817 [AP]
  +OK ..
#
T 81.249.31.222:1817 -> 217.167.120.134:110 [AP]
  PASS kangourou..
##
T 217.167.120.134:110 -> 81.249.31.222:1817 [AP]
  +OK ..
#
T 81.249.31.222:1817 -> 217.167.120.134:110 [AP]
  STAT..
##
T 217.167.120.134:110 -> 81.249.31.222:1817 [AP]
  +OK 0 0..
```

```
#
T 81.249.31.222:1817 -> 217.167.120.134:110 [AP]
  QUIT..
#
T 217.167.120.134:110 -> 81.249.31.222:1817 [AP]
  +OK ..
```

De plus, il est même possible de consulter le contenu des emails en sniffant le réseau. Cette opération est totalement transparente et indétectable pour le client ou le serveur.

```
[...]
T 217.167.120.134:110 -> 62.39.154.188:64269 [AP]
  1 2131..2 5312..3 2365..4 2615..5 1385..6 4169.....
#
T 62.39.154.188:64269 -> 217.167.120.134:110 [AP]
  RETR 1..
#
T 217.167.120.134:110 -> 62.39.154.188:64269 [AP]
  +OK ..
#
T 217.167.120.134:110 -> 62.39.154.188:64269 [A]
  Return-Path: <plop-return-1604-jovialyteam.com-chrispy=jovialyteam.com@jovialyteam
  .com>..Delivered-To: jovialyteam.com-chrispy@jovialyteam.com..Received: (qmail 289
  46 invoked by uid 98); 7 Mar 2004 03:41:26 -0000..Mailing-List: contact plop-help@
  jovialyteam.com; run by ezmlm..Precedence: bulk..X-No-Archive: yes..List-Post: <ma
  ilto:plop@jovialyteam.com>..List-Help: <mailto:plop-help@jovialyteam.com>..List-Un
  subscribe: <mailto:plop-unsubscribe@jovialyteam.com>..List-Subscribe: <mailto:plop
  -subscribe@jovialyteam.com>..Reply-to: plop@jovialyteam.com..Delivered-To: mailing
   list plop@jovialyteam.com..Received: (qmail 28938 invoked from network); 7 Mar 20
  04 03:41:24 -0000..From: Sylvestre Ledru <sylvestre@ledru.info>..To: Plop Mailing
  Liste Jovialyteam <plop@jovialyteam.com>..Cc: Pierre Marot <pierre.marot@jovialyte
  am.com>..Content-Type: text/plain; charset=ISO-8859-1..Message-Id: <1078630913.143
  03.123.camel@localhost>..Mime-Version: 1.0..X-Mailer: Ximian Evolution 1.4.5 ..Dat
  e: Sun, 07 Mar 2004 04:41:54 +0100..Content-Transfer-Encoding: 8bit..Subject: [plo
  p] faites chauffer les photos....Bon..suite a cette tres sympas soir.es a l'alien
  ce soir, j'ai mis rapido en..place le systeme de gallerie dont j'ai parl. un peu a
[...]
```

Ceci est le fait que le canal de communication est dit non sécurisé, c'est-à-dire que l'on a aucune garantie quant à la confidentialité du flux d'informations circulant sur le réseau. Sur le LAN d'une entreprise, un utilisateur consultant ses emails sur un serveur situé localement est à priori peu problématique

mais lorsqu'il consulte ses emails à partir de chez lui sur le serveur mail de sa société, il n'a aucun moyen de controle sur les flux passant chez différents prestataires (Fournisseurs d'accès internet, backbone...). Des informations sensibles circulant par email peuvent donc être interceptées.
Le protocole POP3 est loin d'être le seul protocole pouvant être écouté de manière très simple. En effet, il est possible d'écouter une bonne partie des protocoles de cette manière : HTTP, SMTP, IMAP, DNS, Messageries instantanées (MSN, ICQ...)... Pour peu que le "voyeur" espionne l'utilisateur à partir de la passerelle que ce dernier utilise, toute l'activité Internet de l'utilisateur est connue.

Pour sécuriser ces flux, il existe plusieurs manières de le faire :

- utiliser des connexions privées pour relier les différents points
- utiliser des protocoles de niveau 7 gérant de manière native le cryptage : HTTPS, POP3-SSL...
- sécuriser le canal de communication tout en restant sur un canal non sécurisé par l'utilisation d'un VPN

Connexions privées :
C'est la solution idéale, on reste maître de tout ce qui circule sur le canal. Les personnes ayant accès à des routeurs sur ce canal étant identifiées. Le principal (unique ?) problème de cette solution est son coût. En effet, il coute beaucoup plus cher de faire tirer une ligne spécialisée qu'utiliser des canaux existants.

Protocoles nativement cryptés :
C'est une solution efficace en général mais qui a le désavantage de laisser l'implémentation du système de cryptage/décryptage aux applications (serveurs ou clientes). En effet, si un client ou un serveur ne gère pas l'implémentation en crypté, il n'est pas possible d'utiliser le protocole crypté. Par exemple, certaines versions de Outlook ne supportent pas le POP3-SSL et donc obligent l'utilisation du POP3 classique.
Un autre point problématique est que les flux entre les deux machines restent identifiables. Les protocoles cryptés utilisent des ports identifiés. Il reste donc possible à l'attaquant de savoir à quel service le client fait appel.

Le VPN (Virtual Private Network / Réseau privé virtuel) :
Le VPN permet de relier deux réseaux distants à travers Internet. Il est ainsi possible de faire communiquer ces deux réseaux comme si ils étaient connectés directement ensemble. Dans la quasi totalité des implémentations d'un VPN, un cryptage est rajouté entre les deux connectiques qui vont initier la VPN. Ainsi, par exemple, on peut avoir deux réseaux distants d'une deux offices d'une entreprise reliées à travers un VPN.

2 Le Tunneling IP

2.1 Qu'est ce que c'est ?

Le tunneling IP est le procédé qui consiste à encapsuler un flux réseau dans les paquets d'un autre flux réseau du type TCP/IP. Il est possible, par exemple, d'encapsuler un flux IPX (réseau netware) dans une connexion TCP/IP. Un tunnel IP est un moyen d'assurer l'interconnexion entre deux (ou plus) réseaux dans un réseau plus grand.

Un tunnel IP s'effectue entre 2 machines, qui jouent le rôle de passerelles pour les autres machines de leur réseau respectif.

Le tunneling peut rendre des services de différents ordres :
- chiffrement et déchiffrement des données transmises.
- compression et décompression des données envoyées dans le tunnel.
- offrir l'impression à l'utilisateur de travailler en réseau local (voire sur la même machine)

2.2 Tunnels SSH

2.2.1 SSH

SSH est un protocole permettant d'établir une session interactive chiffrée entre un client et un serveur. Ainsi, les flux d'informations entre ces deux entités sont cryptés ce qui garantit la confidentialité. De plus, il permet l'identitification de la machine distante. L'algorithme utilisé pour la négociation des clés est RSA (dont le brevet a expiré aux USA ce qui permet une utilisation publique légale).

Une fois l'échange des clés effectué, la communication entre les deux machines se fait en utilisant un chiffrage symétrique (un chiffrage symétrique est environ 1000 fois plus rapide qu'un chiffrage asymétrique). Les principaux algorithmes utilisés dans SSH sont triple DES (3DES) ainsi que Blowfish. La plupart des fonctionnalités cryptographiques étant implémentés dans la bibliothèque OpenSSL.

La version du protocole ssh utilisée est la version 2, la première version de ce protocole souffrait d'une grosse faille de sécurité.

2.2.2 Exemple : Export X11 via SSH.

Sous Unix, la possibilité est offerte d'afficher une application graphique sur un autre serveur X que le serveur local. Ainsi, il est possible d'obtenir le rendu

graphique d'une application installée et lancée depuis une machine différente de celle que l'on utilise pour visionner l'application.

Pour pouvoir effectuer un export X11 classique, il faut lancer la commande "xhost +" sur la machine client (celà permet d'autoriser les connexions d'autres utilisateurs à la session X en cours). Il faut aussi que le serveur X acceptant la connexion (toujours sur la machine sur laquelle on désire afficher l'application) ne soit pas lancée avec la clause "-nolisten tcp".
Le fichier en question est le suivant : /etc/X11/xinit/x sous GNU Linux Debian

```
#!/bin/sh
exec /usr/bin/X11/X -dpi 100 #-nolisten tcp
```

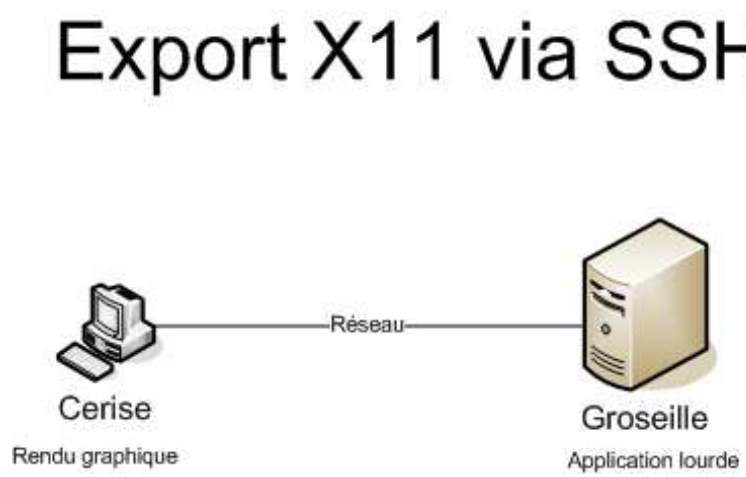
Les deux conditions précédentes réunies permettent d'arriver à nos fins. Néanmoins, cette procédure est un peu lourde et les données circulent en clair sur le réseau (touches frappées par exemple).

SSH propose de façons native la gestion de l'export X11, c'est à dire que lors d'une connexion à une machine distante, il est possible de faire en sorte que les informations concernant une application fenêtrée soient intégrées au flux SSH. Dans ce cas, l'application est lancée sur le serveur X local par l'utilisateur ayant ouvert la session, donc, nul besoin d'autoriser les connexions distantes au serveur X . Autrement dit, "-nolisten tcp" peut être spécifié au serveur X pour son lancement et "xhost" n'a pas besoin d'être lancé par l'utilisateur connecté (plus sûr).

Cependant, il faut que dans la configuration du serveur SSH la clause suivante soit spécifiée (/etc/ssh/sshd_config sous GNU Linux Debian) :

```
X11Forwarding yes
```

Sur le schéma suivant, imaginons par exemple que la machine "cerise" ne soit pas très performante d'un point de vue traitement en images, et que la machine "groseille" au contraire soit tout à fait en mesure d'exploiter ce genre d'application. Cependant, si "groseille" est utilisée par une autre personne, on peut vouloir utiliser "cerise" simplement pour obtenir le rendu grpahique.



Si “groseille” héberge un serveur SSH initialisé avec l’option “X11Forwarding yes”, il suffit de s’y connecter de la manière suivante depuis cerise :

```
ssh -X groseille
```

Puis de lancer l’application désirée dans le shell obtenu suite au login, et le rendu graphique apparaitra sur “cerise”.

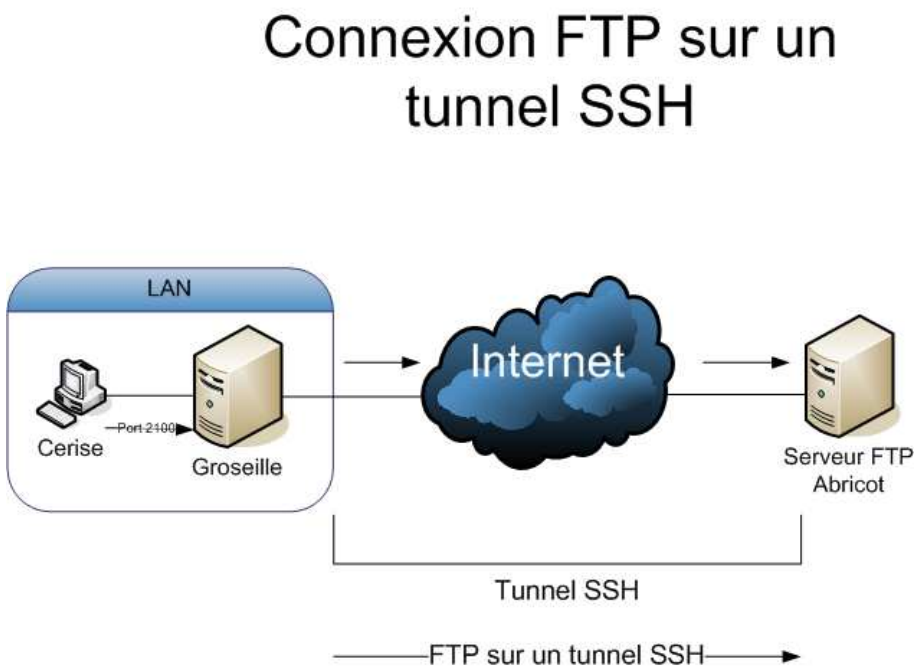
Les données sont protégées puisqu’elles circulent dans le flux SSH au lieu de circuler en clair sur le réseau. Il est donc possible de disposer d’un service non disponible sur la machine locale l’esprit tranquille. En fait, l’export X11 via SSH est un tunnel, puisque les données circulent grâce à une connexion réseau TCP/IP établie entre 2 machines, en entrant par un port TCP d’une machine et en sortant par un autre port d’une autre machine.

2.2.3 Exemple : sécurisation du traffic grâce à un tunnel SSH

Une solution pour palier à ce problème, est de mettre en place un tunnel SSH entre le serveur FTP, et le client. L’option -L de ssh permet d’encapsuler les connexions effectuées sur les sockets sur un port donné dans le flux SSH, en direction d’un autre port sur la machine distante.

Le schéma suivant montre comment s’établie une connexion FTP sécurisée avec SSH entre une machine du LAN et un serveur FTP distant. Pour cet exemple pratique, nous disposons de deux machines sur le LAN, “cerise”, sur laquelle on désire lancer le client FTP et “groseille” sur laquelle nous allons lancer un client ssh configuré pour rediriger les connexions arrivants sur le port 2100 vers “abricot” (encapsulées dans le flux SSH). Cette dernière héberge un serveur FTP séparé de notre LAN par Internet. Notons que nous devons

disposer d'un compte utilisateur sur chacune des machines. Néanmoins, il n'est pas nécessaire d'avoir un accès super utilisateur sur les machines, sauf dans le cas ou l'on désire travailler avec des ports TCP supérieurs à 1000.



La commande qui permet d'établir la connexion SSH entre "groseille" et "abricot" est la suivante :

```
[18:37:13][jam@groseille] ~\$ ssh -g -L 2100:abricot:21 abricot -N
Password:
```

L'option -N permet de pas donner de prompt après la connexion à SSH.
L'option -g permet d'autoriser les connexion d'autres machines, différentes de celle qui établie la connexion, à utiliser le canal sécurisé.
Enfin, l'option -L indique que le données arrivant sur le port 2100 de la machine locale doivent être envoyées sur le port 21 d'abricot en passant par la connexion SSH.
Dès lors, les données reçues sur le port 2100 de la machine sur laquelle est lancée la commande SSH (client) seront intégrées dans le flux sécurisé, puis débalées et déchiffrées à l'autre bout sur la machine qui héberge le service SSH.

Désormais, le fait de se connecter à groseille sur le port 2100 va devenir équivalent pour l'utilisateur à se connecter au serveur FTP d'abricot, mis à part le fait que les données sont cryptées.

```
[18:37:49][jam@cerise] ~\$ ftp groseille 2100
Connected to 127.0.0.1.
220 ProFTPD 1.2.9 Server (Debian) [abricot]
Name (127.0.0.1:jam):
```

voici le flux qu'on l'on peut maintenant sniffer sur le réseau :

```
interface: eth1 (192.168.0.0/255.255.255.0)
#
T 192.168.0.17:33387 -> 217.167.120.138:22 [AP]
  .&.j;...‘.zR.*.)}....?.6~.....kRmL..&.!;AL....LK
#
T 217.167.120.138:22 -> 192.168.0.17:33387 [AP]
  .<T.ye..-.v.Q.Q.!...:Z...r........1.<+3...j.=...6>P....G....j~.P_a....5.F.1
  ....<
##
T 192.168.0.17:33387 -> 217.167.120.138:22 [AP]
  .3..L........n.pI6SvGx........:...].......J..iT...X.=v.>n....-.]
#
T 217.167.120.138:22 -> 192.168.0.17:33387 [AP]
  X..r..B..1Y....P4..q"......]..GK\.....ldS...D...3.6...!..bu....}
##
T 192.168.0.17:33387 -> 217.167.120.138:22 [AP]
  ...d!.J..K.W..S.F..$"T...}P..b.H.....mD!..<..?..
#
T 217.167.120.138:22 -> 192.168.0.17:33387 [AP]
  vq...|...,.....g....$.J\..9dLr..!..v..#.t...Ui.}......m=..[....0
#
```

Donc les trames n'apparaissent plus en clair sur le réseau contrairement à ce que nous avions vu lors d'une connexion FTP non sécurisée.

2.3 Tunnels avec Stunnel

2.3.1 Stunnel

Stunnel est un programme qui permet de crypter les connexions TCP grâce aux librairies SSL (Secure Sockets Layer). Il peut être utilisé pour crypter des échanges d'informations pour des services qui ne sont pas sécurisés nativement, comme ceux que nous avons cité précédement. Pour utiliser stunnel, il est **obligatoire** d'avoir installé une librairie SSL du type OpenSSL ou SSLeay, puisqu'elles assurent le cryptage / décryptage pour stunnel.

Stunnel nécessite la création de certificats SSL (X509) pour fonctionner, ce qui permet au client et au serveur de s'identifier mutuellement de manière sûre

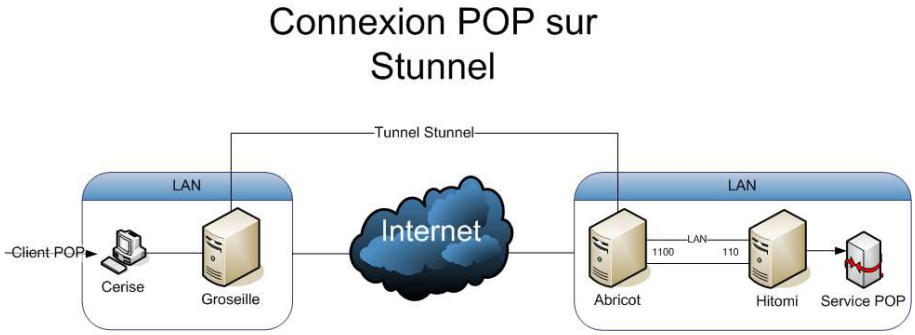
(dans la mesure où elles font confiance à une autorité tierce). Il est possible de se passer de certificats sur la machine cliente, mais pour lancer un service stunnel, il faut obligatoirement disposer d'un certificat valide. L'option "noauth" permet de lancer stunnel en autorisant les connexions sans authenfication sur le serveur, et l'option -d passera le programme en démon. Ici, nous nous interesserons bien sûr plus à la mise en place d'un tunnel qu'à la génération de certificats, c'est pourquoi les commandes de génération des certificats ne seront pas explicitées.

- création d'une autorité de certification
- génération du certificat pour le service
- génération de certificats pour les clients (sauf si on ne fait pas d'authentification au niveau du serveur)

Note : A partir de la version 4.0 de Stunnel, la configuration des tunnels doit se faire à partir des fichiers de configuration associés au programme stunnel. Le principe reste exactement le même bien sûr, mais les commandes ne sont plus tout à fait les mêmes.

2.3.2 Exemple : accès POP via Stunnel

L'idée de cet exemple est d'utiliser un canal sécurisé par stunnel pour récupérer des emails sur un serveur POP. Considérons 2 réseaux LAN reliés par Internet. Le serveur POP est "hitomi" et le service est accessible par le port TCP 110. "abricot" est la machine distante sur laquelle nous allons installer le service stunnel qui va tourner sur le port 1100. "groseille" va ouvrir une connexion Stunnel avec "abricot" et permettre de se connecter à son port 1100 pour utiliser le tunnel. Enfin, "cerise" va récupérer ses e-mails en POP en spécifiant comme serveur POP Abricot (port 1100).



Installation sous Debian :

```
abricot:~# apt-get install stunnel
Reading Package Lists... Done
Building Dependency Tree... Done
The following extra packages will be installed:
   openssl (0.9.7c-5)
```

```
The following NEW packages will be installed:
   openssl (0.9.7c-5)
   stunnel (3.26-1)
```

Création des certificats :

```
abricot:~# /usr/lib/ssl/misc/CA.pl -newca
abricot:~# openssl req -new -nodes -keyout pop.pem -out pop_req.pem
abricot:~# openssl ca -notext -infiles pop_req.pem >> pop.pem
abricot:~# openssl req -new -keyout groseille.pem -out groseille.pem
abricot:~# cp pop.pem  /etc/ssl/certs/
abricot:~# cp demoCA/cacert.pem /etc/ssl/certs/pop.pem
abricot:~# ln -s pop.pem /etc/ssl/certs/stunnel.pem
abricot:~# c_rehash
```

Il faut maintenant transférer le certificat groseille.pem sur la machine du même nom pour pouvoir lancer stunnel.

Lancement du démon sur le serveur (Abricot) :

```
abricot:~# stunnel -P/tmp/ -p pop.pem -d 1100 -r hitomi:110
```

(préciser "noauth pour" une connexion sans autentification)

Lancement du démon sur le client (Groseille) :

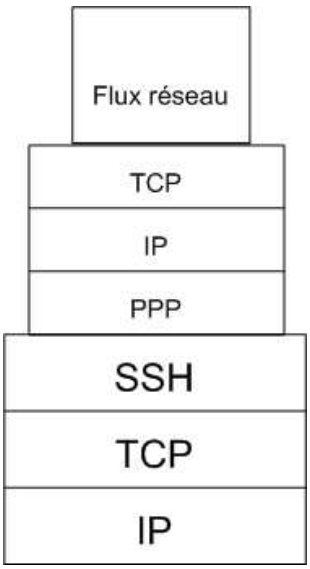
```
[16:37:10][jam@groseille] ~\$ /usr/sbin/stunnel -P/tmp/ -p groseille.pem
   -c -d 1100 -r abricot:1100
```

Le tunnel est maintenant en place ; il ne reste plus qu'à se connecter depuis "cerise" à "abricot" pour récupérer les mails en POP sur le port 1100.

3 PPP over SSH.

Cette technique consiste à établir une connexion PPP via un flux SSH.

Le protocole PPP (point à point) permet d'établir une connexion entre deux machines. Il est surtout utilisé pour les connexions modem. Il y a comme intérêt d'être facile d'utilisation mais aussi de pouvoir facilement fournir des services à la machine initiant la connexion (obtention d'une adresse IP fournie par le serveur...). C'est un protocole très standard utilisé par d'autres protocoles encore utilisé dans les connexions ADSL (PPPOE (PPP over Ethernet), PPPOA (PPP over ATM ...). Il permet aussi d'être interconnecté avec des systèmes RADIUS.



PPP over SSH permet de combiner la puissance de ces deux systèmes :

- De SSH, on tire la sécurité des transactions.
- De PPP, on tire l'établissement d'une connexion avec l'obtention d'une adresse ip.

3.1 Installation client

Pour interconnecter un client à notre VPN PPP over SSH, il faut qu'il ai dans son noyau les options nécessaires au PPP.
Dans la partie Network Device Support des kernels linux, les options suivantes doivent être activées :
PPP (point-to-point protocol) support
PPP filtering
PPP support for async serial ports
PPP support for sync tty ports
PPP Deflate compression
PPP BSD-Compress compression

Un script sh a été développé par le projet Insidenetworks pour faciliter l'interconnexion d'un client avec un serveur PPP over SSH. Ce script est disponible à l'adresse : http ://www.insidenetworks.net/connect/connect-linux
Il va tout d'abord vérifier que l'arborescence nécessaire à PPP existe bien. Ensuite, une clé ssh DSA va être générée (grâce au programmessh-keygen).
Cette clé devra être transmise à l'administrateur du serveur PPP over SSH (la clé se trouve dans le fichier root/.ssh/in.pub). C'est la clé qui ira dans le fichier authorized_keys.

Le script va ensuite générer les fichiers nécessaires à la mise en place du réseau (montage automatique et adéquat des routes : tous le trafic vers 10.0.0.0/8 est routé vers le serveur PPP over SSH). Pour initier une connexion, il suffit te taper la commande pon in qui va connecter la machine sur le serveur. (poff in pour déconnecter la machine) L'interface obtenue est :

```
23: ppp0: <POINTOPOINT,MULTICAST,NOARP,UP> mtu 1500 qdisc pfifo_fast qlen 3
    link/ppp
    inet 10.1.12.15 peer 10.1.12.2/32 scope global ppp0
```

Et tout le trafic vers le VPN passe par cette interface :

```
10.0.0.0/8 dev ppp0  scope link
```

Le client va donc pouvoir accéder aux services proposés par son serveur PPP over SSH mais aussi à toutes les machines qui sont interconnectés au réseau virtuel.

3.2 Installation serveur

Pour installer un serveur PPP over SSH pouvant accepter différentes connexions, il faut s'assurer d'avoir les options adéquates dans le kernel. Les options sont les mêmes que celles pour le client.

Ensuite, une fois le noyau et les modules recompilés et installés correctement, les modifications à apporter au niveau du système sont :

- Créer un compte utilisateur 'vpn'
- Vérouiller le mot de passe (passwd -l vpn)
- Vérifier que l'utilisateur a bien les droits nécessaires pour l'éxécution de pppd (sous Debian, addgroup vpn dip)
- Dans le répertoire home de l'utilisateur vpn (ex : /home/vpn), Mettre le script vpnsh disponible sur http ://www.insidenetworks.net/serv/vpnsh, script qui deviendra le "shell" de l'utilisateur
- Rendre ce script éxécutable (chmod +x vpn/vpnsh)
- Modifier le shell de l'utilisateur vpn (chsh -s /home/vpn/vpnsh vpn)
- Créer le fichier /etc/ppp/peers/vpnserv qui contient uniquement la ligne "noauth" (echo "noauth" > /etc/ppp/peers/vpnserv)

Ensuite, pour pouvoir créer des connexions utilisateur, il faut ajouter un fichier authorized_keys (touch vpn/.ssh/authorized_keys) puis restreindre les droits d'accès au propriétaire, ceci pour d'évidentes raisons de sécurité (chown vpn vpn/.ssh/authorized_keys ; chmod 0600 vpn/.ssh/authorized_keys)

Pour ajouter une connexion, il faut entrer une ligne du type :

```
command="hostname ip_locale:ip_distante (options supplementaire de pppd)" clefs_ssh
```

Ce sont les clés RSA publics des clients.

Par exemple, pour connecter la machine yuko, on a rajoute la ligne suivante :

```
command="yuko 10.12.15.2:10.12.15.5" ssh-dss AAAAB3NzaC1kc3MAAACBAKmNOEHszYNVLNZU1wBR6wA
```

La clé ssh est de type ssh2. Elle nous a été communiquée par l'administrateur de la machine yuko. Dans cet exemple, lorsque yuko se connectera à au serveur PPP over SSH, si la vérification de la clé réussie, la machine yuko obtiendra l'ip 10.12.15.5 (ceci grâce à PPP) et aura comme passerelle 10.12.15.2 (ip locale du serveur PPP over SSH)

Si la connexion PPP réussi, une nouvelle interface ppp est montée :

```
806: ppp0: <POINTOPOINT,MULTICAST,NOARP,UP> mtu 1500 qdisc pfifo_fast qlen 3
    link/ppp
    inet 10.12.15.2 peer 10.1.12.15/32 scope global ppp0
```

et une nouvelle route est ajoutée dans la table de routage :

```
10.12.15.15 dev ppp0  proto kernel  scope link  src 10.12.15.2
```

Ainsi, pour un sniffer, les flux PPP over SSH donneront l'impression à l'attaquant qu'il n'y a qu'un flux SSH. Il n'aura aucun moyen de connaître la nature des données circulants à travers le flux SSH. Ainsi, il ne verra qu'apparaître ce genre de paquets circulant sur le réseau.

```
T 81.56.245.43:1027 -> 217.167.120.134:22 [AP]
  ....]..9.......  ...P......|...c*Z...B...~...y.Qb-...V5..<...6.n5
T 81.56.245.43:1027 -> 217.167.120.134:22 [AP]
  |D~P...u-.Lh......Gmy%..$..D..).....=G....=..C\.JO..%.U.....I...
```

Il est possible ensuite d'interconnecter plusieurs serveurs de VPN qui se partagent la classe 10.0.0.0/8 en différents sous-réseaux. Chacun ayant la délégation d'une plage d'ip et peuvent ainsi créer un grand réseau sécurisé basé sur Internet (ce qui implique l'utilisation de services de routages tel que zebra/quagga) le tout sans avoir aucune connaissance des ips réels des différentes

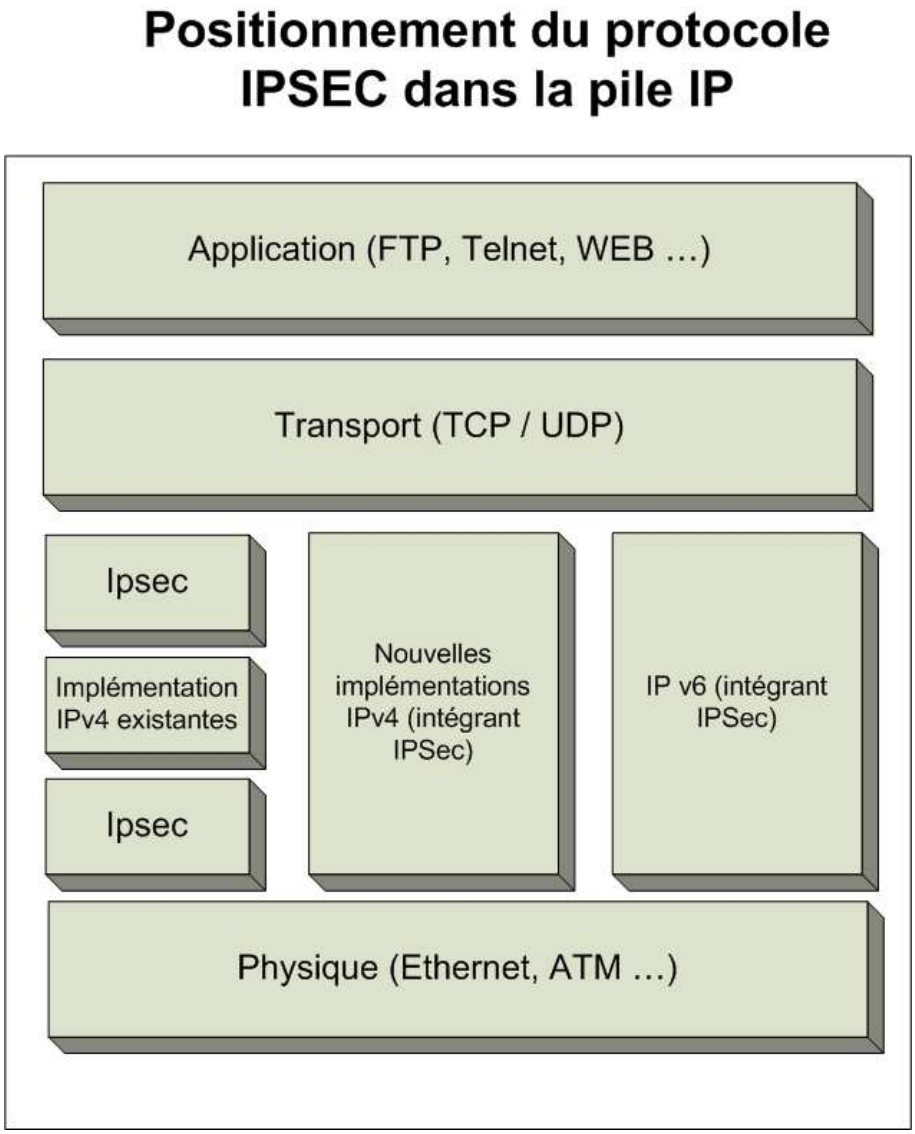
passerelles. Pour faciliter l'utilisation de ce "réseau Internet" dans Internet, il est aussi possible de créer un nouveau TLD (.com, .net, .info ... sont des TLD) pour faciliter l'accès aux services disponibles sur le VPN. C'est ces services que propose le projet InsideNetworks (http ://www.insidenetworks.net).
Ce genre d'infrastructure légère en terme de déploiement est intéressante pour interconnecter toutes les agences d'une société par exemple sur un même réseau virtuel.

4 IPSec : la théorie

4.1 Introduction

Pour sécuriser les échanges sur un réseau Internet, il existe plusieurs niveaux d'intervention. IPSec permet de sécuriser les échanges au niveau de la couche réseau.
Sur-couche du protocole IP, de nombreuses entreprises l'utilisent désormais pour les réseaux privés virtuels (VPN) ou bien la sécurisation des accès distants à un intranet.

- La couche IPSec donne donc les moyens d'assurer :
- la confidentialité des données échangées (lutter contre l'analyse du trafic) via le chiffrement,
- l'authentification des intervenants et des données dans la communication (signatures),
- l'intégrité des données (hashage),
- la protection contre le rejeu (remise de paquets déjà envoyés),
- le contrôle d'accès.



L'utilisation des implémentations d'IPSec permet un choix de configuration varié : la configuration des éléments d'interconnexion (Routeurs ou firewall) et des équipements terminaux (PC et serveur) offre la possibilité d'utiliser tous les types de chiffrement possibles.

4.2 Bases de données de politiques de sécurité (SAD/SPD)

IPsec fonctionne autour de bases de données de politiques de sécurité dont l'utilité vous est décrite ci-après.

4.2.1 SA (Security Association)

Une association de sécurité IPSec est une structure de données servant à stocker l'ensemble des paramètres de sécurité associés à une communication. Une SA étant unidirectionnelle, il faut deux SA pour protéger les deux sens d'une communication. Les services de sécurité définis par la SA sont fournis par l'utilisation des protocoles AH ou ESP que nous expliquons plus tard dans ce document.

Plus précisément, Le rôle d'une SA est de spécifier, pour chaque adresse IP avec laquelle IPsec peut communiquer, les informations suivantes :

- le Security Parameter Index (SPI) : l'identifiant de la SA choisi par le récepteur
- le numéro de séquence, (éviter le rejeu)
- une fenêtre d'anti-rejeu
- le dépassement de séquence
- les paramètres d'authentification (algorithmes et clés)
- les paramètres de chiffrement (algorithmes et clés)
- la durée de vie de l'association
- le mode du protocole IPSec (tunnel ou transport)

Chaque association est identifiée de manière unique à l'aide d'un triplet composé de :

- l'adresse de destination des paquets
- l'identifiant du protocole de sécurité (AH ou ESP)
- le SPI

Lorsqu'un destinataire reçoit un paquet IPSec, il vérifie à quelle SA correspond le paquet reçu. Si cette association de sécurité n'est pas trouvée le paquet est rejeté, sinon il utilise les informations de sécurité pour interpréter le paquet IPSec.

4.2.2 SAD (Security Association Database)

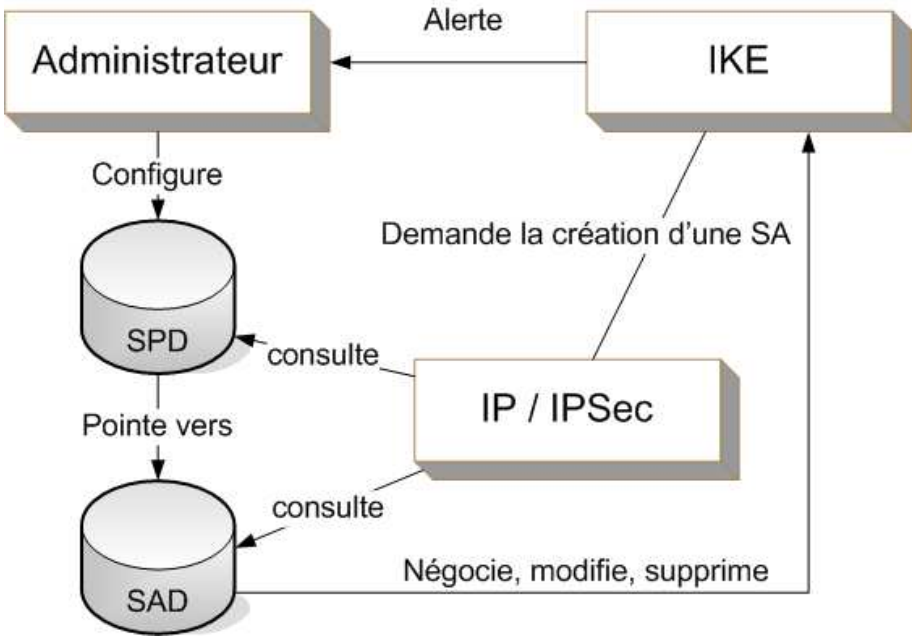
Les associations de sécurité sont stockées dans une base de données (Security Association Database, SAD). Cette base de donnée est consultée par l'hôte afin d'identifier la manière dont doit être traité chaque paquet reçu ou à émettre.

4.2.3 SPD (Security Policy Database)

IPSec offre des protections basées sur des choix définis dans une base de données de politique de sécurité (Security Policy Databas, SPD) établie par l'administrateur de la connexion. Elle permet de décider, pour chaque paquet, s'il doit être sécurisé ou autorisé à passer outre ou rejeté.

Les services de sécurité sont basés sur des mécanismes cryptographiques. IPSec met en jeu deux protocoles en complément du protocole IP classique : AH et ESP. Ce sont deux types d'encapsulation différents même si ESP reprend la plupart des principes d'AH et ajoute notamment des services de confidentialité. Par ailleurs, IPSec offre un service supplémentaire de cryptographie (chiffrement en mode Fast Forward) permettant de conserver des performances optimales en conservant des paquets de même taille. Néanmoins, ce mode garantie uniquement la confidentialité : L'en-tête IP et la longueur du datagramme restent les mêmes (le champ d'options IP peut être chiffré).
Les associations de sécurité contiennent toutes les données de sécurité nécessaires à un échange avec IPSec, notamment les algorithmes et les clés. Une SA peut être configurée manuellement mais la plupart des configuration utilisent un protocole de négociation dynamique des SA et d'échange des clés de session.
C'est le protocole IKE (Internet Key Exchange) association du protocole de gestion des clés et des associations de sécurité pour Internet (ISAKMP cadre générique permettant l'utilisation de plusieurs protocoles d'échange de clé) et d'une partie des protocoles SKEME et Oakley.

4.2.4 Illustration



Trafic sortant :

Dès que la couche IPSec doit envoyer des données, elle se réfère à la SPD pour connaître la manière dont elles doivent être envoyées. Dans le cas où elle doivent être sécurisées, elle consulte la SAD pour retrouver la SA et par conséquent, les paramètres de sécurité requis pour cette échange. Si cette SA n'existe pas, IPSec utilise le protocole IKE pour en définir une.

Trafic entrant :
A la réception d'un paquet distant, IPSec vérifie dans l'entête si des services de sécurité ont été utilisé sur ce paquet. Au quel cas, il en extrait les identifiants de la SA pour la retrouver dans la SAD. Ceci afin de connaître les paramètres de sécurité permettant de traduire le paquet. Aussi, la SPD est interrogée pour savoir si la SA utilisée pour transmettre le paquet est bien celle requise par les politiques de sécurité. Dans le cas d'un paquet IP classique, la SPD est consultée pour savoir si le paquet a le droit ou non de transiter.

4.3 Modes de fonctionnement

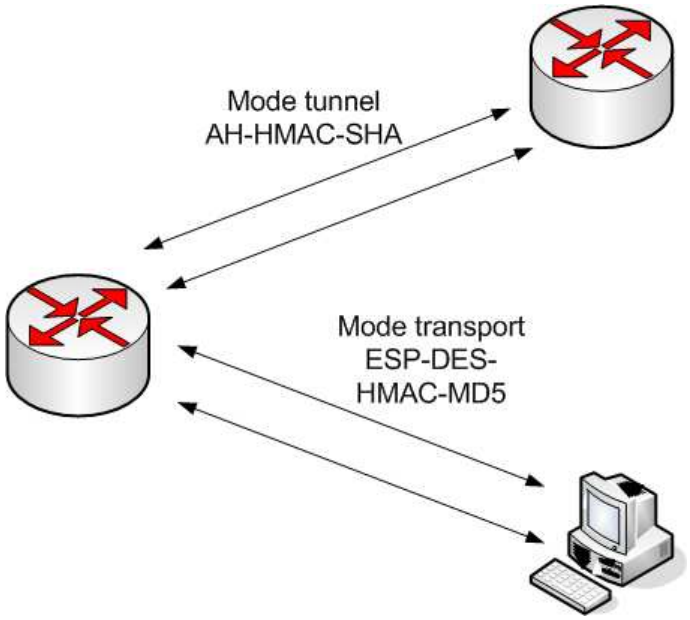
4.3.1 Mode Transport ou Transparent :

Dans le mode transport, IPSec intervient entre le niveau transport (TCP) et le niveau réseau (IP) du modèle OSI : le PDU de la couche transport se

voit appliqué les mécanismes de signature et de chiffrement puis le résultat est passé à la couche réseau (encapsulation IP).
Ce mode ne résout pas une problème majeur en matière de sécurité : l'en-tête du paquet est inchangé puisque produit par la couche IP. Il n'y a donc ni de masquage d'adresse ni de protection des options IP. Cependant ce mode est relativement aisé à mettre en oeuvre.

4.3.2 Mode Tunnel

Dans le mode tunnel, IPSec agit directement après l'encapsulation IP. La totalité du paquet IP est encapsulé dans un paquet IPSec sécurisé. Dans ce cas, l'en-tête IP d'origine est protégée et les adresses sont masqués. Ce mode est très utilisé pour la mise en place de VPNs.



Le mode tunnel est utilisé entre deux passerelles de sécurité (routeur, firewall, ?) alors que le mode transport se situe entre deux hôtes.

4.3.3 Mode Nesting

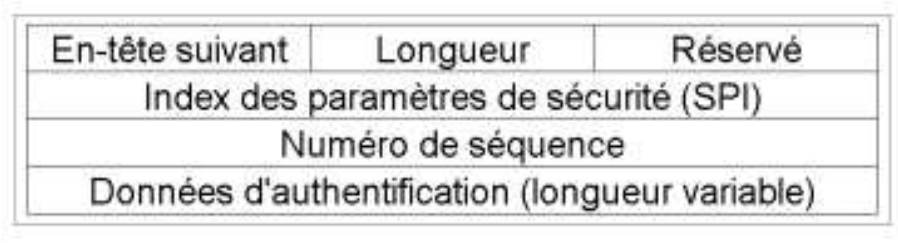
Le mode de Nesting utilise à la fois le mode transport et le mode tunnel : Un paquet IPSec et encapsulé dans un paquet IPSec.

4.4 Protocoles de sécurité (modes d'encapsulation)

4.4.1 Authentification Header (AH)

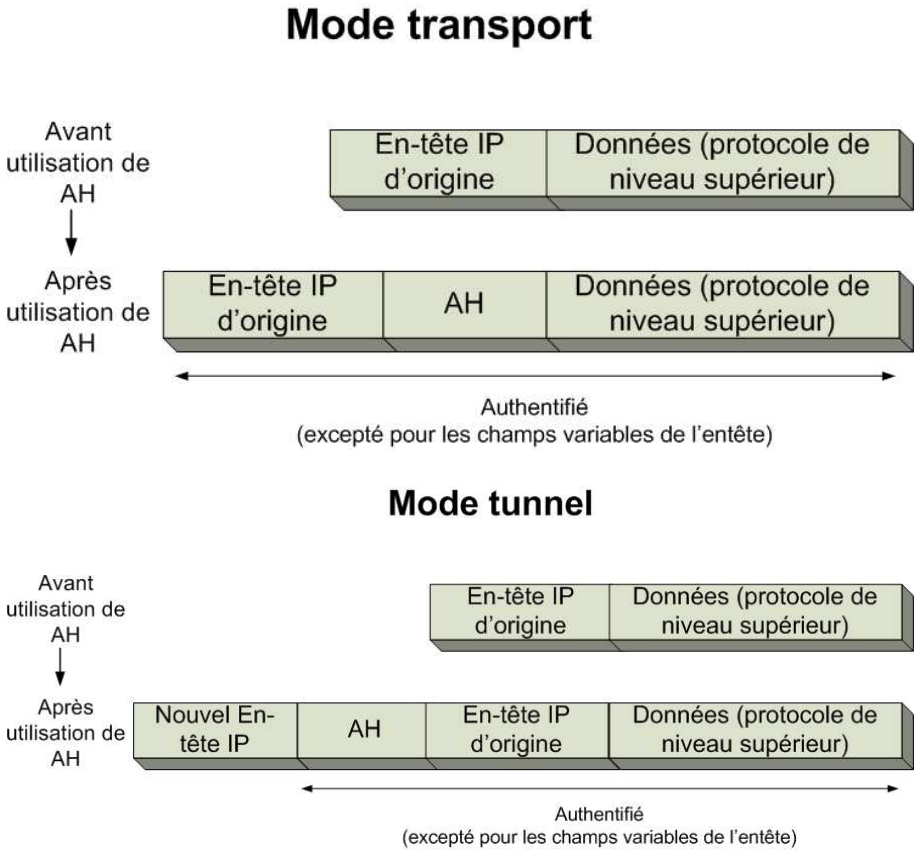
AH permet d'assurer l'intégrité et l'authentification de l'origine des paquets IP mais pas la confidentialité des données.

Le paquet IP se voit affecté d'un nouveau champ permettant de vérifier l'authenticité des données. Ce champ contient un hashé (digest MD5 ou SHA-1) appelé « Integrity Check Value ». Aussi, la protection contre le rejeu (réinjection de paquet) se fait grâce à un numéro de séquence et permet d'éviter les attaques par inondation. Les paramètres de sécurité liés à la communication sont identifiés par un identifiant unique (Security Parameters Index) caractérisant la Security Association (SA). C'est une combinaison de l'adresse du destinataire et du protocole utilisé.



Ce protocole donne les moyens de garantir :
- l'authentification du paquet et de son émetteur (si l'adresse source du paquet est celle de l'émetteur).
- L'unicité du paquet (pas de rejeu).
- L'intégrité des données (aucune altération volontaire ou non du paquet durant le transport).

Seuls certains champs sont certifiés dans le paquet : La version d'IP, longueur de l'entête/des données/du paquet, les données (en mode tunnel ou transport), l'identificateur de flux, Protocole ou entête suivant, Adresse IP de la source et du destinataire.
Les autres ne le sont pas car il changent de valeur au cours de la durée de vie du paquet (ex : TTL, Classe traffic ?).
Pour une protection totale du paquet IP, il faut utiliser le mode Tunnel.



Les différentes méthodes d'intégrité utilisées dans ce mode sont :

MD5 (Message Digest 5) : l'algorithme de hachage MD5 à été conçu par Rivest (un des concepteur de RSA). Il opère sur des blocs de 512 bits. Le résultat d'un hashage MD5 est un digest de 128 bits (4 * 32 bits).

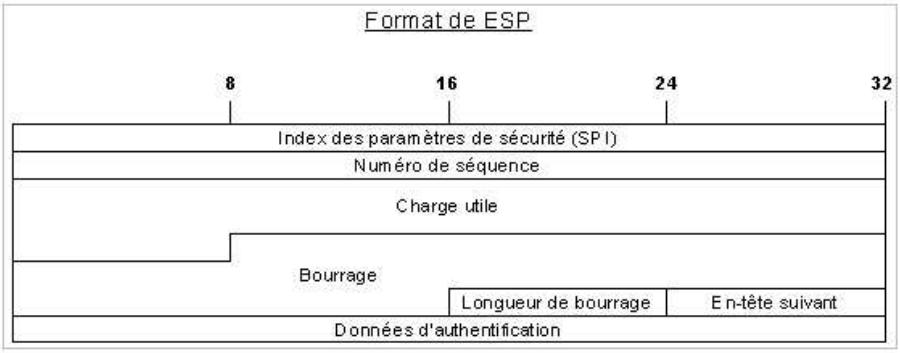
SHA-1 (Secure Hash Algorithme 1) : La NSA et le NIST s'ont à l'origine de cette méthode basée sur MD4. Il fonctionne avec des blocs de 512 bits, Clé de 160 bits.

L'application de ces méthodes sur un message produit un digest (hashé) permettant de certifier l'intégrité et l'authenticité du message : Il est impossible de retrouver le message à partir du digest ; Si un bit du message change, le digest résultant est très différent à cause de l'effet d'avalanche.
D'autres algorithmes peuvent être utilisés dan l'implémentation d'IPSec : exemple MAC (basé sur DES).

4.4.2 Encapsulating Security Payload (ESP) :

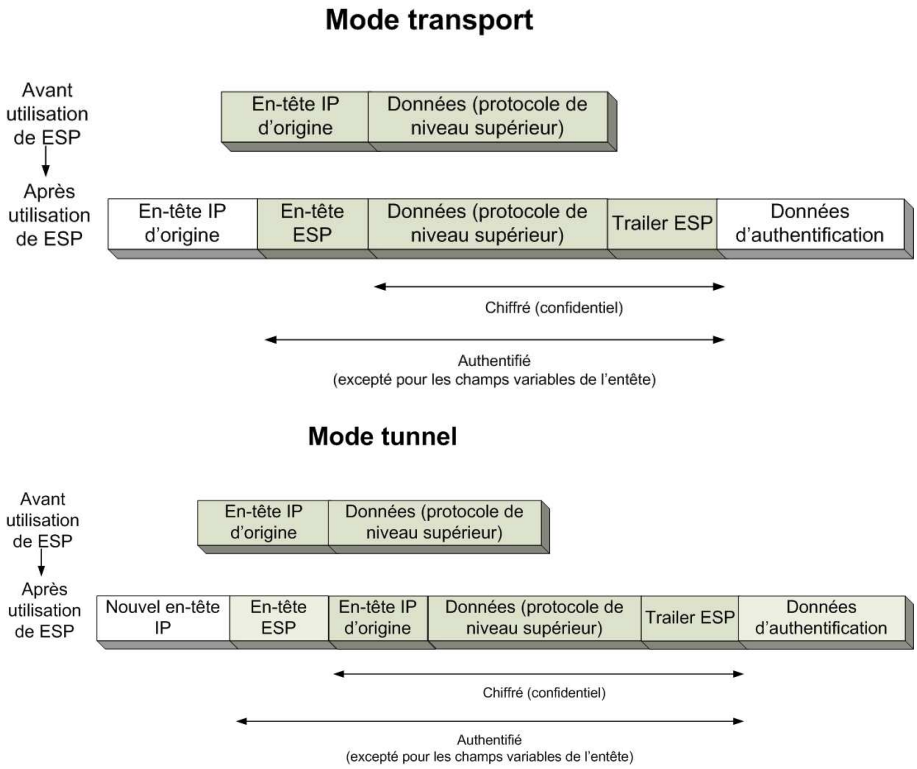
ESP permet de combiner, à volonté, plusieurs services de sécurité. A savoir, la confidentialité des données par l'utilisation d'un système de chiffrement ; l'authentification du paquet et de son émetteur (l'adresse source du paquet est celle de l'émetteur) ; l'intégrité des données (aucune altération volontaire ou non du paquet durant le transport) et l'unicité du paquet (pas de rejeu).

If faut noter que service d'authentification n'est pas obligatoire sauf si le service de confidentialité n'est pas utilisé.
Par opposition à AH, qui ajoute seulement une en-tête supplémentaire au paquet IP, ESP chiffre les données puis les encapsule.



ESP propose de l'authentification de la même manière que AH gràce à l'utilisation de données d'en-tête :
Le SPI (Security Parameters Index) permet de caractériser l'association de sécurité utilisée pour la communication (SA).
Les données d'authentification contiennent la valeur de vérification d'intégrité (ICV) permettant de vérifier l'authenticité des données du paquet.
Un numéro de séquence pour éviter le rejeu.

Les données chiffrées sont contenues dans la partie « champ libre » (ou PayLoad Data) du paquet. Ce champ contient éventuellement aussi des données de synchronisation. Du bourrage (Padding), peut être ajouté si nécessaire. Sa longueur est spécifiée dans le champ prévu à cet effet.
Enfin, le champs En-tête suivant (Next Header) indique la nature des informations contenues dans le Payload Data (champ libre).



Pour l'authentification et l'intégrité des données, ESP utilise HMAC, une évolution des algorithmes de hashage classiques. En effet, il s'agit d'un algorithme de hashing MD5 ou SHA-1 avec en plus l'utilisation d'une clef secrète partagée entre les hôtes. Le hashé chiffré représente une signature du message.

Les méthodes de cryptage utilisé par ESP :
DES (Data Encryption Standard) : C'est une système de cryptage symétrique mis au point par IBM : il met en jeu une clé unique pour chiffrer et déchiffrer les messages. La confidentialité du message repose donc sur le secret de cette clé. La transmission de la clé doit donc être sécurisée. Il opère sur des blocs de 64 bits (dont huit derniers bits de parité).
3DES (Triple Data Encryption Standard) : 3DES consiste en l'utilisation à trois reprises de DES. Ce qui le rend plus fiable car il met en jeu trois clefs différentes, mais donc plus lent.

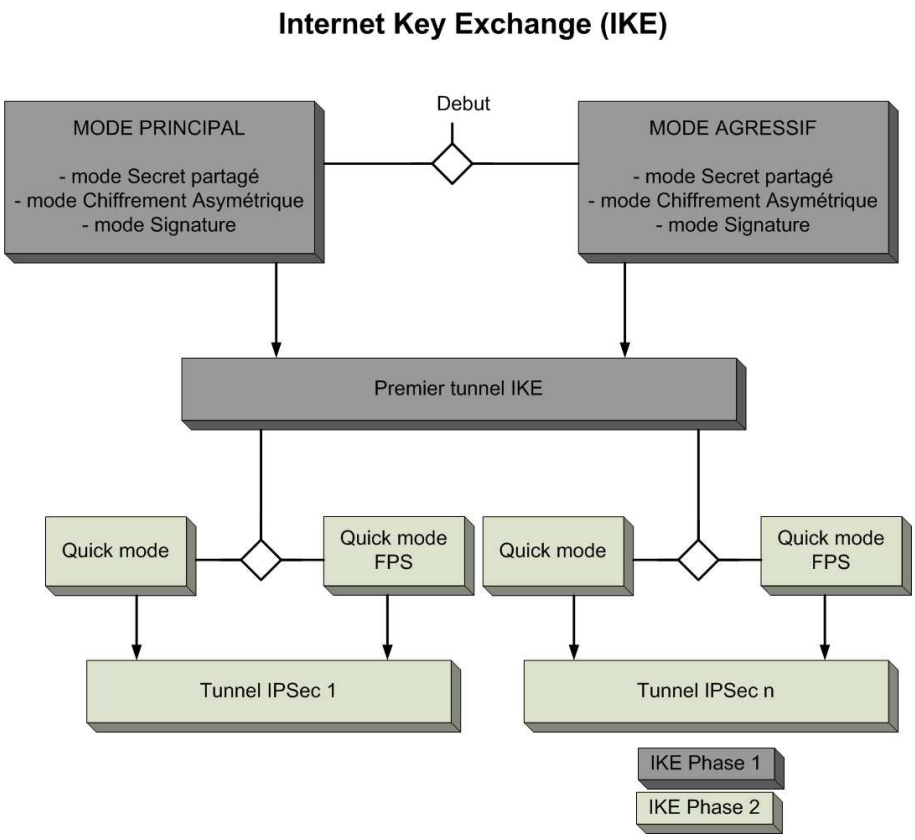
Par ailleurs, il existe deux codes pour les protocoles sans effet :
Algorithme de chiffrage NULL
Algorithme d'authentification NULL

4.4.3 Internet Key Exchange

Ce protocole a pour but dans sa première phase de construction d'un premier tunnel sécurisé entre les 2 hôtes (le tunnel IKE). Il est utilisé pour gérer les tunnels IPSec (la négociation des SAs et leur mise à jour) constituant la deuxième phase du protocole IKE. Ce sont ces tunnels qui serviront aux échanges de données entre les hôtes. Cependant, IPSec offre la possibilité d'opérer une authentification manuelle, sans utiliser IKE.

Il est la combinaison de plusieurs autres protocoles :
ISAKMP, protocole de gestion des clés et des associations de sécurité pour Internet : cadre générique permettant l'utilisation de plusieurs protocoles d'échange de clé
SKEME et Oakley, systèmes d'échange de clés.

Les différentes étapes du protocole IKE :



IKE Phase 1

Cette phase va servir à la création d'une première clé qui va permettre par la suite la génération de 3 autres clés dérivant de celle-ci. Cette clé peut être générée selon 3 modes offerts par IKE. Le mode «secret partagé» implique que les hôtes partagent déjà un secret qui permettra la mise au point de cette clé. Le mode « Chiffrement asymétrique » se base sur les crypto système à clé publique pour échanger les données sensibles et donc établir le secret partagé. Le mode «Signature», quant à lui, se sert du chiffrement asymétrique pour signer et authentifier les hôtes alors que le secret partagé est établi grâce à Diffie-Hellman.

Une fois la première clé générée, elle est dérivée en 3 autres clés qui serviront à la création du tunnel IKE sécurisé entre les hôtes (en faîte une association de sécurité ISAKMP). L'une des clés sera utilisée pour l'authentification, l'autre pour le chiffrement et la dernière sera utilisé lors de la phase 2 du protocole.
Ce canal, sécurisé, est ensuite utilisé pour la deuxième phase IKE.

Plus précisément, lors de cette phase, les échanges permettent de définir l'association de sécurité puis d'établir le secret partagé et enfin d'authentifier les hôtes.

Il faut noter que le mode aggressif permet de limiter les communications en utilisant certains paramètres d'office. D'autre part, les SA ISAKMP utilisent un chiffrement (DES ou 3DES) lors de l'échange des clefs de session.

IKE Phase 2

L'objectif de la deuxième phase à pour objectif de créer les tunnels IPSec (SA) pour les échanges effectifs entre les hôtes : Deux SA par hôtes, un pour chaque sens de communication, conservées dans la SAD (Security Association Database).
C'est lors de cette phase que chaque hôte donne ses préférences en matière d'algorithme et établissent le matériel cryptographique. Les clés de session sont générées à partir de l'une des clés dérivées, générée durant la phase 1 de IKE.
Cependant, lorsque le mode «Perfect Secrecy» est utilisé, les hôtes doivent échanger de nouveaux secrets, ceci afin de couper la relation systématique entre les nouvelles clés générés et la clé de la phase 1 IKE. Cet échange s'effectue via le protocole d'échange Diffie-Hellman.
Cette phase sert aussi à spécifier les échanges devant bénéficier des services IPSec (utilisation de la Security Policy Database),

Rappels sur certains systèmes utilisés :
RSA (Rivest, Shamir, Adleman) :
RSA est un crypto système à clé publique faisant intervenir une paire de clé par intervenant, dans un échange sécurisé :

Une clé privée, utilisée par l'émetteur du message pour chiffrer. Cette clé n'est jamais divulguée.
Une clé publique, utilisée par le destinataire pour déchiffrer le message. Cette clé est divulgué à tous les destinataire via, par exemple, un certificat numérique certifié par un CA.
Cet algorithme s'appuie sur la difficulté de factorisation de deux nombres entiers.

Diffie-Hellman :
Il s'agit d'un algorithme d'échange de clefs pour les algorithmes à clefs publiques. Cet algorithme permet un établissement d'un secret partagé entre 2 hôtes, et ce via un réseau non sécurisé.
DH est basé sur les nombres premiers, le modulo et le logarithme discret :
Une fois les paires de clés générées à l'aide des données partagées, les clés publiques sont échangées et vont servir à l'aide des clés privées à générer le secret (une clef de session qui servira au criffrement des messages). Pour cela, chaque hôte utilise sa clef privée et la clé publique de l'hôte distant pour créer la clef de session commune.

5 Partie technique Ipsec

La principale implémentation libre sous Linux est Freeswan mais pour des raisons politiques et internes, une partie des patchs sont refusés (les patchs de contributeurs américains sont automatiquement refusés). Ainsi, Freeswan n'intègre pas le NAT traversal, les certificats (x509) ... De plus, le projet freeswan a été arrêté le 1er mars 2004.
C'est pour ces raisons que nous utilisons un fork (reprises du projet par un autre groupe) appelé openswan (http ://www.openswan.org) qui intègre tous ces patchs qui apportent d'intéressantes fonctionnalités. Il existe aussi le projet strongswan (http ://www.strongswan.org/) qui proposent aussi ipsec avec d'autres patchs.

Les protocoles IP étant modifiés, il faut donc modifier la pile IP. Sous linux, ceci se fait en patchant le noyau du système et en le recompilant.

5.1 Procédure de compilation et installation de ipsec/openswan

Il est supposé que le kernel est correctement configuré et les sources de celui ci disponible dans /usr/src/linux.

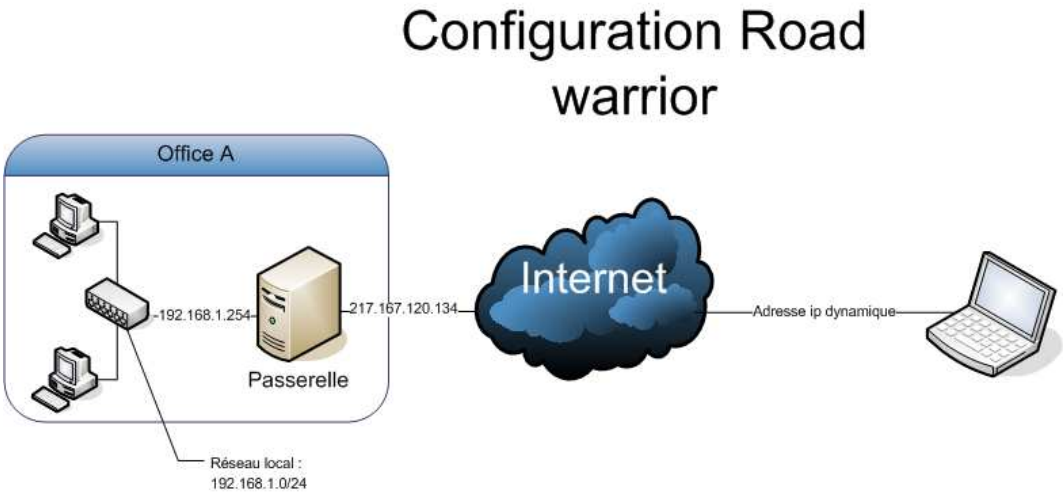
Téléchargez la dernière version sur le site officiel (actuellement, la dernière est la version 2.1.1).

```
# cd /usr/src/
# wget http://www.openswan.org/code/openswan-2.1.1.tar.gz
# tar -zxvf openswan-2.1.1.tar.gz
# cd openswan-2.1.1
# make kpatch # cette option va patcher le kernel pour ajouter les fonctionnalités ipsec
# make nattpatch > nat_patch
# cd ../linux
# patch -p1 < ../openswan-2.1.1/natpatch # pour obtenir  la fonctionnalité de
  nat traversal
# make menuconfig # pour verifier que les options IPSEC sont bien activées
  (dans la section Networking options)
Lancez la compilation du nouveau kernel (la procédure est variable selon
les distribs linux)

# cd ../openswan-2.1.1/
# make programs install # pour installer les daemons et services nécessaires au
 fonctionnement de openswan. Attention, les sources de la libraire de calcul
mathématique gmp doivent être installé pour pouvoir compiler openswan.
```

5.2 Utilisation de Ipsec en mode Road Warrior

Le mode road warrior est le mode qui permet à un utilisateur distant de se connecter au réseau privé de l'entreprise comme si il était physiquement présent. C'est typiquement l'exemple de l'un employé qui veut se connecter au réseau de l'entreprise à partir de chez lui.
La particularité réside dans le fait que l'utilisateur n'a pas une ip fixe. En effet, avec la plupart des fournisseurs d'accès internet, les ips sont alloués dynamiquement.



Pour effectuer cette connexion, il faut chaque machine ai la clé public RSA de l'autre machine pour valider le processus d'authentification.
La partie de configuration ipsec pour une connexion Road Warrior se divise en deux parties
La partie Left (Local) et Right (Remote, distante).
Ainsi, pour la machine cliente, on va avoir :

```
conn road
    left=%defaultroute              # Pour récuperer l'ip dynamique
    leftnexthop=%defaultroute       #
    leftid=@client_road_warrior.example.com       # Information locale
    leftrsasigkey=0sAQPIPN9uI[...]   # Clé RSA de la machine road warrior
    right=192.0.2.10                # Adresse ip de la passerelle ipsec
    rightsubnet=10.0.0.0/24         # Réseau privé géré par la passerelle
    rightid=@reseau.example.com        #  Information distante
    rightrsasigkey=0sAQOnwiBPt[...]  # Clé RSA de la passerelle ipsec
    auto=add                        # autorise mais ne lance pas la connexion au
                                      démarrage d'ipsec

Pour le serveur ipsec, le fichier ipsec.conf aura l'apparence suivante :
conn road
    left=192.0.2.10                # Adresse ip de la passerelle ipsec
    leftsubnet=10.0.0.0/24         # Réseau privé géré par la passerelle
    leftid=@reseau.example.com        #  Information distante
    leftrsasigkey=0sAQOnwiBPt[...]  # Clé RSA de la passerelle ipsec
    right=%defaultroute             # Pour récuperer l'ip dynamique
    rightnexthop=%defaultroute      #
```

```
rightid=@client_road_warrior.example.com       # Information locale
rightrsasigkey=0sAQPIPN9uI[...]   # Clé RSA de la machine road warrior
auto=add                        # autorise mais ne lance pas la connexion
                                  au démarrage d'ipsec
```

Pour initier la connexion, il suffit de taper la commande : ipsec auto –up road

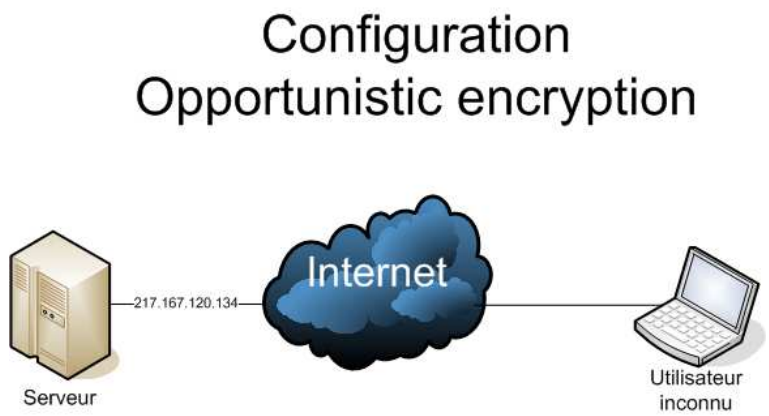
Pour obtenir sur une machine la clé RSA public, il suffit de taper la commande suivante : ipsec showhostkey –left pour la machine "située" à gauche ou ipsec showhostkey –right pour l'autre coté. (les clés générées à "gauche" et à "droite" sont similaires dans le cas de RSA). Si la clé n'a pas été générée au premier démarrage (ce que openswan fait normalement par défaut), il est possible de la générer avec la commande : ipsec newhostkey –output /etc/ipsec.secrets –hostname machine.example.com

Si le client se trouve derrière un NAT (ou appelé aussi masquerading), il faut absolument que le serveur ainsi que le client gèrent la RFC 3715 (IPsec-Network Address Translation (NAT)). Dans le cas contraire, le NAT réécrivant les paquets la vérification AH (Authentication Header) d'ipsec va rejeter le paquet et rendre impossible toute connexion.

Cette configuration va permettre à la machine dite cliente d'accéder au réseau privé situé derrière la passerelle et ainsi accéder aux ressources du réseau local. Le tout sécurisé par le cryptage appliqué par ipsec.

5.3 Opportunistic encryption

L'opportunistic encryption (ou appelé OE) est une fonctionnalité non RFC ajoutée par les développeurs de freeswan sur le protocole IPSEC. Il a pour intérêt de pouvoir négocier avec n'importe quel serveur (connu ou non) acceptant l'OE la création d'un tunnel ipsec temporaire entre une machine et l'autre.



Pour cela, il utilise les propriétés des DNS pour pouvoir authentifier la machine distante.
Sans rentrer dans les arcanes des DNS, il va faire une double vérification en se basant sur les champs de commentaires de DNS (champs TXT).
Lorsqu'une machine se connecte à une serveur ipsec gérant l'OE, il va faire deux résolutions DNS :

- une sur le champs de commentaire du reverse de l'ip
- une sur le champs de commentaire du nom annoncé par ipsec du serveur

Ainsi, lors du processus de connexion, l'ipsec va vérifier la présence de ces deux champs. Et ceux, des deux cotés de la connexion.

Il est possible de tester l'existence de ces deux champs à la main (ici avec le programme host).

```
# host -t TXT machine.ecranbleu.org
machine.ecranbleu.org  TXT     "X-IPsec-Server(10)=81.255.82.6"
 " AQNnycIot0WaNFL6LHInCoWnLz72qHb6Cvpw7Za3oWkWO5uksyKn2EN8Ckw
6N/bX2Opbz53h+PjFBUcyjO65UzamUqbOdiJemEORZq2Jv1fmcAOTw+8oKoq6N
a0pMD3Nl9MP0bhrW5KnExTZ8IBAWIGF7yqlFaSXW6rsCYmavkU+kHFo7NsEa0W
oHUwdzEscyapV0WR0Apvqu4t8AB3pgkTg5/Xpo2qD64QQybjkUFx7QKYLg7QX1
WcnwAgHW2" "EyEz+Z9tAwoamhAG4Ckunbwm3ot8mhu4EFdukrG+mXDOVPXiZn
w3zmvWYDiJFdqcfl6oy8cbTm0ao2UDoOGkjgzk+SE7L1EAod    ipsec auto
 --up net-to-netlhBU/z4bidc+y3xxcVs5NaCKOTXUVBw/qirOHnmCC+FYYg
bwS7qobaL1XOb+w7N2ipgba/f6NFmxC+WHuTbepYv7+4iafajeizDQEeEYSZVD
7qFqe9H0HeY5jF+x+gKFKM/cgrMYEoBJQb3s2f4qWOnf6s1Kv" "p1fCER+qD0U
Af4JZEbBAvNMWcdK6ll3HkvwtfVlSaEW8Bs="

# host -t TXT 6.82.255.81.in-addr.arpa.
6.82.255.81.in-addr.arpa        CNAME   6.0-16.82.255.81.in-addr.arpa
```

```
6.0-16.82.255.81.in-addr.arpa   TXT     "X-IPsec-Server(10)=81.255.82.6"
 " AQNnycIot0WaNFL6LHInCoWnLz72qHb6Cvpw7Za3oWkWO5uksyKn2EN8Ckw6N/bX2Opb
z53h+PjFBUcyjO65UzamUqbOdiJemE0RZq2Jv1fmcAOTw+8oKoq6Na0pMD3Nl9MP0bhrW5K
nExTZ8IBAWIGF7yqlFaSXW6rsCYmavkU+kHFo7NsEa0WoHUwdzEscyapV0WR0Apvqu4t8AB
3pgkTg5/Xpo2qD64QQybjkUFx7QKYLg7QX1WcnwAgHW2" "EyEz+Z9tAwoamhAG4Ckunbwm
3ot8mhu4EFdukrG+mXDOVPXiZnw3zmvWYDiJFdqcfl6oy8cbTmOao2UDoOGkjgzk+SE7L1E
AodlhBU/z4bidc+y3xxcVs5NaCKOTXUVBw/qirOHnmCC+FYYgbwS7qobaL1XOb+w7N2ipgb
a/f6NFmxC+WHuTbepYv7+4iafajeizDQEeEYSZVD7qFqe9H0HeY5jF+x+gKFKM/cgrMYEoB
JQb3s2f4qWOnf6s1Kv" "p1fCER+qD0UAf4JZEbBAvNMWcdK6ll3HkvwtfVlSaEW8Bs="
```

Pour rajouter ces champs, il faut générer la clé public. Il existe un programme inclus dans openswan qui permet de générer l'enregistrement DNS au format utilisé par BIND (principal serveur DNS sur le net) :

```
# ipsec showhostkey --txt @‘hostname --fqdn‘ # la partie entre entre backquote permet de
```

Une fois le processus de vérification et d'échange des clés a été effectué, un tunnel est effectué de manière transparente entre les deux machines. Par exemple, si une machine gérant l'OE essaye de consulter un site internet héberge sur une machine supportant cette technologie, la sécurisation se ferra automatiquement.

Le projet freeswan met à disposition publique une machine de test pour vérifier que cette fonctionnalité marche de manière correcte. Ainsi, si on essaye de se connecter (quelque soit le protocole) sur la machine oetest.freeswan.org, le tunnel va être mis en place pour une certaine durée.

Le tunnel ainsi obtenu est similaire à celui que l'on a pu obtenir précédemment en dehors du fait qu'il a une durée de vie.

```
# ipsec eroute
0 193.17.15.19/32 -> 81.255.82.7/32 => tun0x1002@81.255.82.7
```

Si on sniffe le réseau, on ne va voir que des informations chiffrées circuler, sans même d'informations quant au port de destination ou source. Voila ce que tcpdump nous retourne quand on sniffe une connexion cryptée.

```
Entête : 14:16:52.493098 81.255.82.7 > 193.17.15.19:
ESP(spi=0x629ad466,seq=0x19)
Contenu du packet : E..h^U..42..Q.R.....b..f.....B..+.....-.o/|....
.]q.g[...Y."".....y..4r..P."..Kx..
Entête : 14:16:52.493199 193.17.15.19 > 81.255.82.7:
ESP(spi=0xf9e66fad,seq=0x1b)
Contenu du packet : E..‘i...@2......Q.R...o........<..^...z9Rt....
```

```
.Q..~6..Y..8..:n.G...Q,.....I...wx..
```

Alors que si la connexion n'avait pas été chiffré, on aurait obtenu bien plus d'informations, tel que le port source, de destination, les numéros de séquence et surtout le contenu du paquet :

```
Entête : 14:19:29.709798 81.255.82.7.21 > 193.17.15.19.36111:
P 62:94(32) ack 11 win 32416 <nop,nop,timestamp 90859454 188978569>
 (DF) [tos 0x10]
Contenu du packet : 220 ProFTPD 1.2.9 Server (Debian) [brannigan.ecranbleu.org]..
Entête : 14:20:42.840999 193.17.15.19.36113 > 81.255.82.7.21:
P 11:26(15) ack 94 win 5840 <nop,nop,timestamp 188985899 90866560>
 (DF)
Contenu du packet : PASS mon_pass..
```

Intêret de l'OE L'opportunistic encryption est très intéressante pour sécuriser les flux sur internet. Si toutes les machines utilisaient ce système, les problèmes de sécurité a l'échelle d'internet seraient sensiblement diminués étant donné que tous les flux sont automatiquement cryptés entre deux périphériques. Mais malheureusement, cette technologie est difficile à déployer pour principalement trois raisons :

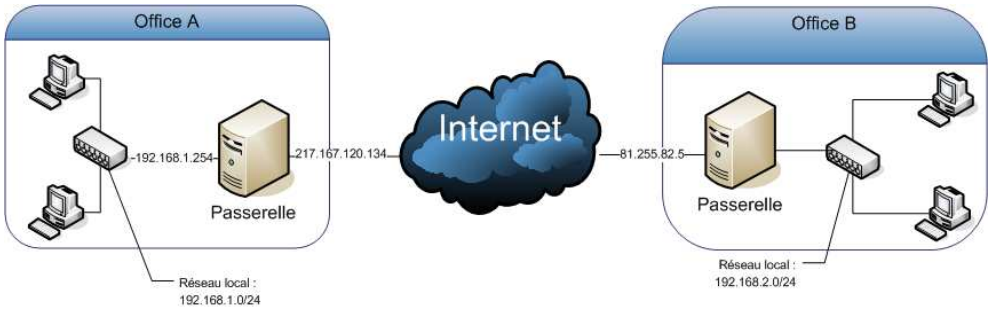
- elle est spécifique à Free/Swan et à ses forks (openswan/strongswan) aucune autre implémentation d'ipsec ne l'implémentant jusqu'à maintenant.
- elle nécessite une délégation de la résolution inverse de son adresse ip pour pouvoir rajouter les champs commentaires nécessaires.
- le procédé d'authenfication reste relativement long selon les critères habituellement toléré en réseau : il faut en effet entre 2 et 4 secondes pour vérifier les clés (en grande partie dû à la "lenteur" de la résolution DNS).

D'autres méthodes de mise en place de la connexion OE sont actuellement à l'étude (en particulier un draft d'une RFC pour utiliser IKE (Internet Key Exchange) au lieu des serveurs DNS).

5.4 Réseau à réseau

Le mode de réseau à réseau permet d'interconnecter deux réseaux d'entreprise distante. Par exemple, dans le cas d'une entreprise ayant deux locaux distants ayant besoin d'utiliser des ressources réseau communes (serveur de fichier, accès au serveur de messagerie ...). Il va être nécessaire d'interconnecter les deux réseaux. La solution d'une connexion physique entre les deux locaux est la plupart du temps inenvisageable car très coûteuse. La solution d'un VPN sur Internet interconnectant les deux réseaux est donc la plus économique et réaliste d'un point de vue technique.

Pour interconnecter les deux réseaux, il faut interconnecter les deux passerelles et autoriser l'accès aux réseaux qu'elles gèrent.



La configuration nécessaire pour interconnecter les deux réseaux présentés dans le schéma est la suivante :

```
conn net-to-net
    left=217.167.120.134     # ip publique de la passerelle du réseau A
    leftsubnet=192.168.1.0/24      # Réseau privé de l'office A
    leftid=@officea.masociete.com  # identifiant de la passerelle
    leftrsasigkey=0s1LgR7/oUM[...] # Clé RSA de la passerelle
    leftnexthop=%defaultroute      # Récupère la route de la passerelle
    right=81.255.82.5              # ip publique de la passerelle du réseau B
    rightsubnet=192.168.2.0/24     # Réseau privé de l'office B
    rightid=@officeb.masociete.com # Identifiant de la passerelle
    rightrsasigkey=0sAQOqH55O[...] # Clé RSA de la passerelle
    rightnexthop=%defaultroute     # Récupère la route de la paserelle
    auto=add
```

Dans ce cas, il n'y a pas à intervertir les deux définitions (right/left).

Pour initier la connexion, la commande est :

```
# ipsec auto --up net-to-net
```

Si le processus se déroule parfaitement, ipsec va produire l'affichage suivant :

```
104 "net-to-net" #223: STATE_MAIN_I1: initiate
106 "net-to-net" #223: STATE_MAIN_I2: sent MI2, expecting MR2
108 "net-to-net" #223: STATE_MAIN_I3: sent MI3, expecting MR3
004 "net-to-net" #223: STATE_MAIN_I4: ISAKMP SA established
```

```
    112 "net-to-net" #224: STATE_QUICK_I1: initiate
    004 "net-to-net" #224: STATE_QUICK_I2: sent QI2, IPsec SA established

On peut vérifier que le tunnel est bien établi grâce à la commande :

# ipsec eroute
0          192.168.1.0/24    -> 192.168.2.0/24     => tun0x1002@81.255.82.5
```

Ainsi, toutes les machines reliés au réseau de l'office A seront accessibles à celle du réseau de l'office B et réciproquement. Ce qui potentiellement entraîne des problèmes de sécurité car si un des deux réseaux venait à être compromis, le pirate aurait un accès facile et rapide à l'autre réseau.

6 Connexion Ipsec entre machines Windows et Linux

6.1 Pré-requis

Comme nous l'avons vu précédemment, IKE assure la négociation des paramètres de la connexion IPSec. Il négocie les algorithmes utilisés, les clés, les durées de vie, etc.
Différentes méthodes pour gérer les clés sont possibles :

- Secrets partagés : nécessite un secret connu et partagé entre les entités. Il est appelé le "pre-shared secret" et va servir de base à l'élaboration de la clé mère qui va elle-même servir lors de l'authentification finale. il faut installer la clé sur chacune des machines, il y a une clé par couple de machines.
- Kerberos : fonctionne en environnement Microsoft
- DNS : méthode de choix pour FreeS/WAN, encore nouveau, ne fonctionne qu'avec certains serveurs DNS, n'est pas disponible pour Windows 2000, repose sur la confiance que l'on peut faire au DNS.
- Certificats x509 ou mode Signature : Se base sur l'utilisation de certificats.

Le but est d'établir un VPN entre une machine Windows cliente et un serveur Linux dans l'optique de démontrer l'interopérabilité des différentes implémentations d'Ipsec sur des plateformes hétérogènes. Windows XP et 2000 incluent le mode "Certificat" en standard (il n'y aura rien à installer de particulier sur les poste Windows), nous nous baseront donc dessus pour la mise en pratique. Nous expliqueront dans un premier temps ce mode Certificats. L'implémentation d'Ipsec devra comporter du côté serveur, le patch X.509 permettant la gestion des certificats d'authentification.

6.2 Généralités

La certification est le procédé qui permet de relier une clé publique à un individu, une organisation ou autre entité. La validation est l'action de vérifier qu'une certification est toujours valide.
Le certificat est un document électronique attestant qu'une clef publique appartient réellement à un individu ou à une organisation. Il est possible de transmettre une clef publique en authentifiant son propriétaire, mais tous les destinataires doivent faire confiance à l'autorité de certification qui aura émis le certificat.

6.3 Définitions

Un certificat permet d'assurer l'identité du propriétaire de la clef publique qu'il contient.
La première méthode consiste à établir une relation de confiance directe avec le détenteur de la clef publique (principe du protocole PGP)
La seconde méthode consiste à ce que tous les intervenants d'un échange fassent confiance en un tiers appelé Autorité de Certification (CA pour Certification Authority), qui se chargera de vérifier l'identité du propriétaire de la clef publique, de délivrer et de signer le certificat assurant la relation. L'autorité de certification aura en charge de préciser les méthodologies mises en place pour vérifier les identités des propriétaires des certificats, pour assurer la pérennité des informations dans le temps, etc.
Le format d'un certificat est défini par le standard X.509

Le but de la gestion par certificats est de rendre automatique la distribution des clés entre toutes les entités participant à une transaction donnée.
Lors de la phase d'authentification, IKE utilise les "RSA signatures" (certificat numérique au format X.509 authentifié par une signature RSA). Dans cette architecture tri-parties avec les 2 peers et l'autorité de certification (ayant délivré et signé les certificats), chaque peer reçoit un certificat émanant d'une autorité de certification. les deux machines signent leurs certificats avec leurs clefs privées et les cryptent avec leurs clefs publiques (elles mêmes ayant au préalable été signées par l'autorité de certification), puis l'envoie à l'autre peer. L'identification de l'équipement distant va donc se faire au moyen du certificat reçu du serveur de certificat par l'équipement voulant réaliser l'identification.

6.3.1 Format des certificats X509

- Première version : 1988
- Deuxième version : 1993
- Troisième version : version actuellement utilisée (X509v3). Cette version contient des extensions qui augmentent la flexibilité des certificats.

Un certificat contient les données suivantes :
Certificate format version (Ce champ donne la version du certificat : 1, 2

ou 3)
Certificate serial number (Numéro de série unique pour l'autorité de confiance qui a établi le certificat qui l'identifie de façon unique. C'est ce numéro de série qui sera posté dans la liste de révocation en cas de révocation.)
Signature algorithm identifier for CA (Désigne les algorithmes utilisés pour signer le certificat : (norme ISO). Il s'agit d'un algorithme asymétrique et d'une fonction de hachage. Exemple : RSA with SHA .)
Issuer X.500 name (Nom de l'émetteur du certificat. Spécifie le DN (Distinguished Name) dans la norme X.500 du CA qui a généré le certificat. o = organization name c = country)
Validity period (Période de validité du certificat. Donne les dates de début et de fin de validité.)
Subject X.500 name (Nom de propriétaire du certificat (celui qui possède la clef privée correspondant à la clef publique contenue dans le certificat. Spécifie le DN dans la norme X.500. O = organization C = country CN = name)
Subject public key information (Ce champ contient la valeur de la clé publique du détenteur du certificat et les algorithmes avec lesquels elle doit être utilisée. Exemple : RSA with MD5)
CA signature (C'est la signature de l'autorité de certification (CA). Cette signature est effectuée en passant l'ensemble du certificat au travers d'une fonction de hachage puis en chiffrant le résultat à l'aide de la clé privée de l'autorité de certification.)

Ces informations sont certifiées être justes par une autorité de certification (Certification Authority ou CA ; par exemple Verisign) qui est censé vérifier les informations avant de valider le certificat, notamment le " Distinguished Name ". Pour cela, le CA hache et signe le certificat à l'aide de sa clé privée. Il suffit donc de connaître sa clé publique largement distribuée pour vérifier la validité d'un certificat distribué par elle.

6.4 Mise en application

6.4.1 Création de l'autorité de certification sur le serveur Linux

Cette autorité permettra par la suite de valider l'ensemble des certificats pour le serveur et les clients.

```
./CA.sh -newca
Spécification des informations :

C=FR,S=France,L=Paris,O=Ledru,CN=zoidberg.ecranbleu.org,Email=sylvestre@ledru.info

Obtention du certificat : /etc/ipsec.d/cacerts/cacert.pem
```

6.4.2 Création du certificat de la passerelle

```
./CA.sh -newreq
./CA.sh -sign
```

zoidberg.ecranbleu.org est le nom de la passerelle.
zoidberg.ecranbleu.org.pem : certificat de la passerelle (contient la clé publique)
zoidberg.ecranbleu.org.key : clé privée

```
mv zoidberg.ecranbleu.org.pem /etc/ipsec.d/certs
mv zoidberg.ecranbleu.org.key /etc/ipsec.d/private
```

Mise à jour de /etc/ipsec.secrets :

```
RSA zoidberg.ecranbleu.org.key "PassGateway"
```

6.4.3 Création d'un certificat pour un client Windows

On génère en premier lieu un certificat standard.
Windows nécessite de convertir les deux fichiers .pem et .key en un fichier .p12

```
openssl pkcs12 -export -in winHost.zoidberg.ecranbleu.org.pem
    -inkey winHost.zoidberg.ecranbleu.org.key
    -certfile /etc/ipsec.d/cacerts/cacert.pem
    -out winHost.zoidberg.ecranbleu.org.com.p12
```

La machine Windows doit posséder ce fichier .p12 et le certificat de l'autorité de certification du serveur.

Le certificat utilisé pour IPSec est associé à l'ordinateur et non pas à l'utilisateur comme on aurait pu l'imaginer. Windows possède 3 catégories de magasins de certificats qui correspondent aux 3 comptes suivants :

- Le compte de l'ordinateur (global)
- Le compte de l'utilisateur (un par utilisateur)
- Le compte du service (un par service)

Par ailleurs le magasin peut correspondre à un magasin physique (carte à puce ou token USB) ou logique (rangé dans le registre). Cependant seul un utilisateur peut avoir un magasin physique. En outre il n'est pas possible de protéger par un mot de passe la clé privée associée au certificat de l'ordinateur car le processus doit pouvoir s'exécuter sans intervention humaine (il n'a pas de fenêtre pour demander un code).

La clé privée et les certificats doivent être importés à l'aide de la MMC (Microsoft Management Console).

On utilisera, pour gérer les politiques de sécurité, un outil recommandé

sur le site Freeswan (ipsec.exe). Celui-ci permet, associé à un fichier de configuration de spécifier les connexions possibles.

```
conn road
  left=%any
  leftsubnet=81.51.226.209
  right=81.255.82.6
  rightsubnet=192.168.10.0/24
  rightca="C=FR,S=France,L=Creteil,O=Miage, CN=zoidberg.ecranbleu.org,email=sylvestre"
  network=auto
  auto=start
  pfs=yes
```

La syntaxe est très proche du langage de description du fichier de configuration /etc/ipsec.conf du serveur Linux.

6.5 Conclusion sur la certification

Les certificats utilisés par IPSec sont associés à la machine et non à une personne physique. Il est quasiment impossible de protéger par un mot de passe la clé privée. IPSec apporte une bonne garantie que l'on dialogue avec la bonne machine mais n'offre rien en matière d'authentification de l'individu. Le risque le plus banal est le vol de l'ordinateur portable. Il faut être suffisamment réactif dans la gestion des listes de révocation. La protection de la clé privée repose entièrement sur le système d'exploitation.

7 Failles des VPN

7.1 Introduction

Cette partie ne constituera pas une banale récapitulation des failles relatées dans des listes de diffusion comme “ bugtrack ” sans intérêt pédagogique mais bel et bien les questions qui sont posées lors de la mise en ?uvre de réseaux virtuels privés.
Il est possible de définir plusieurs axes de réflexion pour plusieurs types de failles :

- Les problématiques relatives à l'essence même, le principe fondamental du VPN.
- Les problèmes soulevés par les protocoles utilisés suivant le type de VPN à différents niveaux OSI ou différentes étapes de phases critiques préalables ou lors du déroulement de l'étape du chiffrement.
- Les architectures réseaux inhérentes à ces solutions en fonction du besoin de l'utilisateur.
- Les failles d'implémentation de cette solution c'est-à-dire au sein même des programmes utilisés (clients ou serveurs).

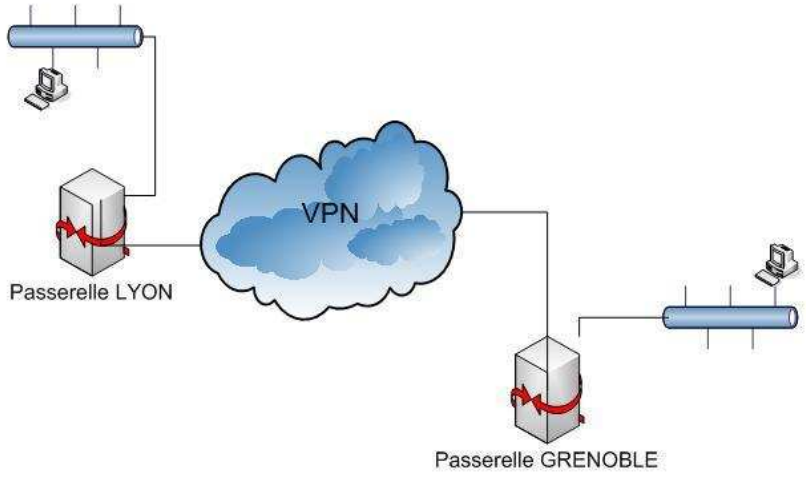
Dans le cas présent même si cette expression est très populaire dans les salons de sécurité, elle illustre une partie des questions à se poser concernant les VPN.

“ Le niveau d'une chaîne de sécurité est égal à celui de son maillon le plus faible ”.

Autrement dit, sécuriser la plupart des aspects d'une solution en en occultant certains prétextant qu'ils appartiennent à d'autres domaines ne permet pas de sécuriser une architecture. Par cette notion de “ sécuriser ”, il n'est pas question de prétendre que la solution est inviolable mais bien de garantir un niveau minimum de sécurité dans le cadre d'une classification des risques opérationnels selon des axes aujourd'hui consensuels (Auditability, Availability, Integrity, Confidentiality).

Nous allons maintenant voir plus en détails les problématiques qui méritent d'être soulevées.

7.2 VPN : “ Cesam ouvre toi ”



Voici le principe du VPN réduit à sa plus simple expression mais elle suffit à comprendre l'idée qui consiste à dire que comme dans le cadre de ce genre de solution si une isolation (chiffrement, contrôle de l'intégrité etc. ?) est assurée entre par exemple un client VPN à Lyon et le serveur qui se trouve à Grenoble, quiconque réussit par un quelconque moyen à compromettre la sécurité du client s'ouvre les portes du réseau distant. Cette idée si elle parait triviale est lourde de conséquence car elle peut se traduire concrètement par l'accès à des

partages NFS ou netBios, la modification d'entrées dans les annuaires Ldap, l'accès à la comptabilité de l'agence de Grenoble, etc.

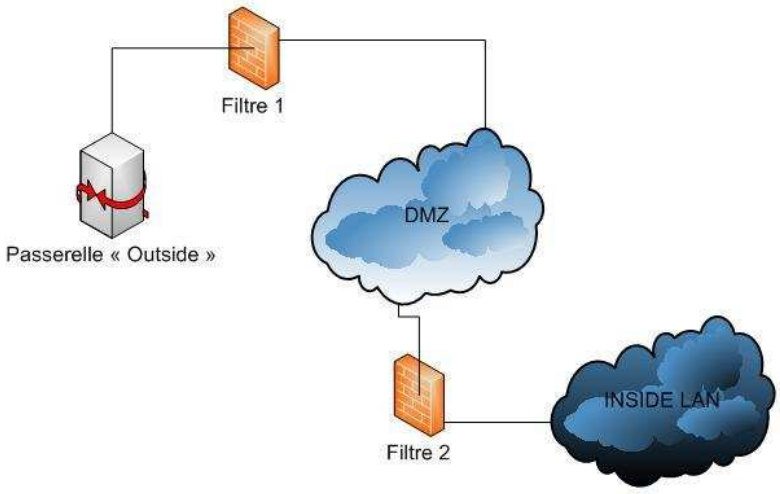
En réalité cette situation fait immédiatement penser à une architecture pare-feu de type Bastion.

Il est important d'expliquer, et ceci rejoint l'idée que se fait W.R Bellovin(*) des laboratoires Bell, que le terme de firewall est en réalité dans la plupart des cerveaux des administrateurs réduit à sa plus simple expression concrète : le filtre de PDU(**) (Protocol Data Unit) suivant le niveau OSI où l'on se trouve.

(*) Steven M Bellovin est chercheur pour les laboratoires Bell AT&T et est un pionnier des scientifiques qui se sont penchés sur la question de la sécurité et du chiffrement, il est membre de la " National Academy of Engineering ".

(**) l'unité de donnée de chaque couche OSI comme les N-PDU (N comme Network) peuvent être des "datagrammes" IP ou les T-PDU (T comme Transport) peuvent être des "segments" dans le cas de TCP , les LPDU (L comme Link) des " trames " dans le cas de la couche de liaison. On devrait en réalité parler d' " architectures pare feu ", les deux plus courantes sont les suivantes :

Une seule DMZ (partie du réseau dans restriction d'accès) :



Et l'architecture en bastion citée plus haut ou une seule et même machine dédiée constitue le firewall (filtre de paquets), ce peut être un routeur (Cisco

propose des filtres dans ses routeurs) ou une machine avec un système réduit à sa plus simple expression.
Cette solution n'est pas moins sécurisée que l'autre plus complexe cela dépend entièrement du contexte.

Revenons à notre situation du client VPN dans son réseau privé, la première solution qui vient à l'esprit est de sécuriser le client en faisant des mises à jour régulières du noyau, du système (démons, softs) et de filtrer correctement via un " pare feu " comme iptables etc.

En réalité, la porte d'entrée du réseau distant a été déporté dans ce réseau local et nécessite une architecture un minimum sécurisée et en l'occurrence centraliser la sécurisation sur le client revient à en faire un bastion du point de vue du réseau distant.

7.2.1 Questions sur les protocoles utilisés

Les protocoles utilisés dépendent bien sûr au moins du type de VPN que l'on met en oeuvre Ici nous prendrons comme cas les VPN légers du type PPP over SSH.

Dans ce paragraphe il est question des protocoles qui font l'objet de la sécurisation (couche OSI >2) et non de ceux sous-jacents à un réseaux local.

Sur le papier le fait d'utiliser par exemple RSA en algorithme de chiffrement symétrique pour rendre la clé de session secrète pourrait laisser imaginer que cette phase est sûre or en passant par l'attaque frontale par factorisation de la clé publique RSA (n=pq et PGCD(p,q)=1) à celles basées sur le mauvais choix de l'exposant publique (WIENER, DE WEGER) entre autres ou la simple faille d'implémentation ou de protocole il n'est pas censé de faire confiance en du chiffrement sans se poser de questions.
Par ailleurs GnuPG propose par exemple en plus de leur outil un petit utilitaire dit générateur d'entropie (à décortiquer !) ce qui montre bien que leur seul outil ne peut être en soi une solution à tous les problèmes.

Si gérer les trousseaux de clé de façon non automatique réduit les chances en apparence de voir l'intégrité de ces données compromises il n'en n'est rien. L'administrateur qui envoie ces données par mail par exemple doit s'assurer de la sécurité (confidentialité, intégrité) des connexions générées. En l'occurrence, le fait que le message en clair (données qui suivent la commande DATA du protocole SMTP terminées par un " . " final) circule sur parfois plusieurs relais SMTP fait dors et déjà trembler . Quelqu'un pourrait rétorquer qu'il est possible de chiffrer le message avec GnuPG par exemple. Oui mais comment se

communiquer les clés publiques ? Le poisson se mort la queue.

L'utilisation de certificats signés (digest) par une autorité de certification comme Verisign oblige d'un part à dépenser de l'argent et d'autre part à faire confiance à un tiers.

En réalité le protocole d'échange des clés dans le cas d'une grosse infrastructure se doit d'être automatisé car personne n'a le temps de gérer cette phase. Des protocoles comme IKE proposent une solution.

7.3 Les architectures Réseaux : Le périmètre des VPN

7.3.1 Passerelle mais pourquoi ?

Avant toute chose, considérons un cas très présent aujourd'hui d'un réseau local Ethernet avec par exemple des Vlan taggués, des routeurs pour interconnecter les différents réseaux et une passerelle dévouée pour permettre aux stations du réseau interne d'accéder au Net.

Une des caractéristiques de l'inter connecteur nommé " passerelle " est qu'il est capable d'assurer la traduction d'un protocole de niveau OSI Réseau (couche 3). Concrètement un routeur pourra interconnecter deux réseaux HDLC et CSMA/CD (Ethernet) alors que la passerelle elle pourra interconnecter deux réseaux IP et X25 respectivement.

La raison pour laquelle on appelle dans notre cas l'accès au Net une passerelle est qu'elle assure la traduction entre le monde OSI (CSMA/CD(IP(TCP par exemple) avec un monde non OSI (le net) basé sur TCP/IP.

Autrement dit les données de niveau 2 transmises par la station initialement (adresse MAC sur 48 bits, données etc.) ne seront pas communiquées au monde extérieur par la passerelle.

Cette explication est nécessaire car c'est justement ce monde là que les VPN proposent de sécuriser. Le périmètre des VPN est donc censé se limiter à sécuriser l'interconnexion via Internet de réseaux locaux pour donner l'illusion d'un réseau privé (adresse IP de type privé A, B, C) alors que tout passe par Internet.

7.3.2 Attaque ARP (couche de liaison OSI)

Si les couches plus basses comme CSMA/CD sont utilisées alors il est possible d'exploiter des failles de ces couches qui, plus elles sont de bas niveau (les failles et les couches), plus il est difficile de les contrer.

Parlons maintenant un peu de l'attaque d' " ARP cache poisonning " dans notre cas.

Sans expliquer en détail le fonctionnement d'ARP/RARP il est important de savoir que ce protocole est du type " cry for help " c'est-à-dire basé sur un requête de type 'who-is' en broadcast (@mac de destination FF-FF-FF-FF-FF-FF) pour demander à qui de droit à quelle adresse MAC correspond telle adresse IP. Le propriétaire habituel ou occasionnel (DHCP) de cette adresse IP va recevoir cette trame comme tout le monde et répondre cette fois en unicast qu'il possède telle adresse MAC.

Cependant ce genre de conversations génère du trafic réseau et un système de cache ARP est utilisé pour ne pas demander à chaque fois qui est qui. C'est la raison pour laquelle le temps de réponse au premier " ping " (message ICMP de type echo, type 8 code 0) sur une machine du réseau local est toujours plus grand que les suivants.

Exécuter une requête ARP en unicast (autorisé par la RFC) pour demander une conversion en forgeant sa propre trame (arp-sk de frederic Raynal ou Nemesis pour win32) en usurpant l'identité IP d'un hôte mais avec sa propre adresse MAC occasionnera une mise à jour du cache ARP du destinataire avec de fausses données. Plus tard le destinataire enverra à l'IP de l'attaquant toutes les trames destinées à l'hôte dont l'identité réseau aura été usurpée.

Une solution qui n'en est pas une est le cache statique (lourd à administrer) mais Sun propose sous Solaris d'augmenter la fréquence de rafraîchissement du cache ARP.

Dans notre cas précis du client VPN qui tente de joindre son serveur en chiffrant ses données il émettra comme tout le monde une trame CSMA/CD vers la passerelle mais avec l'adresse MAC de l'attaquant qui pourra par exemple faire un " drop " de toutes les trames (option -mac de iptables) venant du client et créer ainsi un dénis de service.

La conclusion de cette simulation est que même un VPN est soumis aux failles de couches OSI dont il n'a pas à s'occuper.

7.3.3 Les failles d'implémentation

Les failles d'implémentation des VPN sont facilement accessibles pour qui accède le réseau des réseaux via des sites comme " securityfocus ", " securiteam " ou les listes de diffusion spécialisées (bugtrack). Certains articles de " Phrack " pourraient être amenés à en parler un jour donc il faut les lire (en anglais)

Nous citerons 3 failles à titre d'exemple à savoir la vulnérabilité IPSEC CISCO du " group password " qui peut occasionner des détournements de sessions ou des attaques du type " Man In the Middle "
Ref : http ://www.cisco.com/warp/public/707/cisco-sn-20040415-grppass.shtml
Ou encore la faille SSL trouvé par l'école polytechnique de Lausanne en Février 2003 Ref : http ://lasecwww.epfl.ch/memo_ssl.shtml

Enfin la plus récente de Avril 2004 :
Cible : gentoo Soft : le package Racoon http ://www.linuxsecurity.com/advisories/gentoo_advisory-4207.html

Il en existe d'autres comme des buffers overflows des clients VPN Cisco par exemple dans différentes zones (heap stack bss etc.) de la mémoire pour exécuter du code malveillant via un shellcode.

7.4 Avantages et inconvénients de la solution PPP

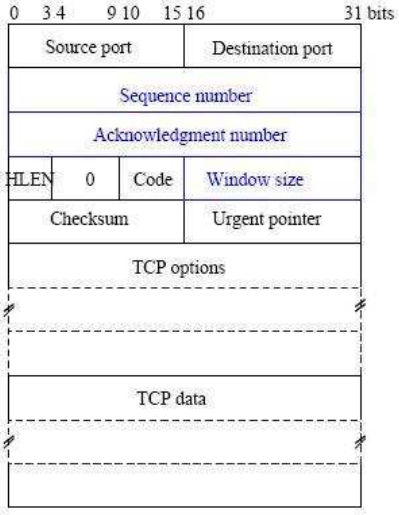
7.4.1 TCP over TCP

Encapsulé au sein du champ DATA d'un segment TCP rien de moins qu'un autre segment TCP peut être lourd de conséquence car il s'agit ici d'un mode de connexion connecté synchrone. Sans entrer trop dans le détail et les notions de " piggybacking " et d'acquittement de trame, il faut savoir qu'après la phase d'initiation de connexion TCP appelée TCP three way handshake qui a pour but la synchronisation de part et d'autre du réseau les segments TCP sont séquencés par un numéro de séquence initialement choisi aléatoirement ISN (*). Ce numéro incrémenté de la taille des données transmises (explicit_data_segment_length) va donner le numéro d'acquittement de ce segment.
(*) La commande Hping2 permet de visualiser les Numéros de séquence TCP.

```
Source : man hping2
#hping2 win98 --seqnum -p 139 -S -i u1 -I eth0
HPING uaz (eth0 192.168.4.41): S set, 40 headers + 0 data bytes
2361294848 +2361294848
2411626496 +50331648
2545844224 +134217728
2713616384 +167772160
2881388544 +167772160
3049160704 +167772160
3216932864 +167772160
3384705024 +167772160
3552477184 +167772160
3720249344 +167772160
3888021504 +167772160
4055793664 +167772160
4223565824 +167772160
```

Dans le cas d'un problème d'ordonnancement de segment, l'émetteur réémet le segment dit 'perdu'. Mais pour laisser une chance à son pair il est possible de spécifier un taille de fenêtre de transmission (deuxième demi mot de 16 bits du 4ème octet de l'entête fixe du segment TCP). Dans le cas précédent, la fenêtre sera décrémentée de la taille de la donnée transmise.

Ce mécanisme est en plus appuyé sur le biggybacking c'est-à-dire que un pair acquitte le segment précédemment reçu en même temps qu'il en envoie un nouveau.



Sequence number :
. Numéro de premier octet du champ de données dans le flux d'octets transmis (modulo 2^{32}).
. Initialement la valeur du champ est quelconque!

Acknowledgment number :
. Numéro du prochain octet à recevoir.
. Acquitte tous les octets de numéro inférieur.

Window size :
. Nombre d'octets pouvant être envoyés par anticipation.
. Capacité de stockage du récepteur.
. W= 0 $\Rightarrow$ contrôle de flux

Il est simple d'imaginer que ce 'time-out' s'il est différent dans le segment TCP père et dans son fils encapsulé peut générer jusqu'à la terminaison de la connexion.

Citons comme solution à ce problème, l'exemple de CIPE (Olaf Titz), qui lui fonctionne en UDP c'est-à-dire en mode non connecté ce qui ôte les soucis de synchronisation en général.

Cependant la solution de PPPD comme il est dit dans la revue MISC n°10 ne nécessite pas d'installation de logiciels supplémentaires et même si SSH peut souffrir de certaines faiblesses, il reste un standard pour sécuriser la connexion entre un client et un serveur.

Références

[1] MISC numéro 10, http ://www.miscmag.com

[2] TooLinux, http ://www.toolinux.com/linutile/reseau/tunnel/index2.htm

[3] Stunnel, http ://www.stunnel.org

[4] Natecarlson.com, http ://www.natecarlson.com/linux/ipsec-x509.php

[5] Jussieu.fr, https ://www-ext.lmcp.jussieu.fr/informatique/IPSec/IPSec.htm

[6] Laboratoire-microsoft.org, http ://www.laboratoire-microsoft.org/articles/network/ipsec/#installation

[7] Securiteinfo.com, http ://www.securiteinfo.com

[8] Guill.net, http ://www.guill.net

[9] 194.51.152.252 http ://194.51.152.252/documentation.htm.

[10] "FireWall et Sécurité Internet" S.M Bellovin & W.R Cheswick

[11] Cours sur TCP de Berbard Cousin, Université de Rennes I

[12] Projet Inside Networks http ://www.insidenetworks.net

[13] Projet freeswan http ://www.freeswan.org

[14] Projet openswan http ://www.openswan.org

[15] Linux IPSec Overview http ://www.linux-tech.com/fswan.html