e i g

Laboratoire de Transmissions de données

# Interopérabilité avec les protocoles IPSec & Personal Firewall

**Informations**


| | |
|---|---|
| Date | *04-12-2001* |
| Etudiant | *Sébastien Borsa* |
| Email | *sborsa@bluewin.ch* |

Professeur *Eric Jenny*

# Table de matière :

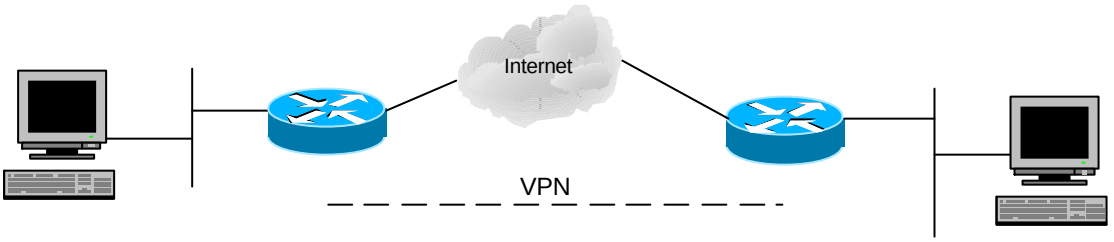
## Annexe :

# 1. Introduction

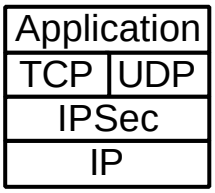
La sécurité au sein des réseaux informatiques devient de plus en plus un problème, c'est pourquoi il est nécessaire d'utiliser de nouveaux protocoles permettant de sécuriser le transite des données. Dans ce document, nous étudierons le protocole IPSec.
Les tests ont été effectués avec des routeurs Cisco 2621 (Version 12.0(7)T avec le module : c2600-ik2o3s-mz.120-7.T.bin), SSH Sentinel 1.1 bêta 3 avec pilot 1.2.0.15 du coté client et un firewall CheckPoint FW1 4.1 sp 5.

## 1.1 IPSec

L'utilisation de VPN devient de plus en plus fréquente. En effet, un VPN présente des avantages non négligeables au niveau du coût par rapport à une ligne louée. Cependant, le fait de passer dans Internet pour échanger les données ne permet pas d'assurer la même sécurité qu'une ligne louée.




Pour résoudre ces problèmes, le protocole IPSec a été implémenté. Il permet de créer des connexions sécurisées entre deux équipements (PC, Routeur, FireWall ,…).
Ce protocole permet de garder la structure IP des paquets (source, destination) et donc ne nécessite pas de changer le matériel déjà présent (router, switch, …).
Ce protocole est inséré juste au-dessus de la couche IP.




Les données se trouvant au-dessus de la couche IPSec sont chiffrées et il n'est donc plus possible d'utiliser certain système tel que NAT où les ports de connexion UDP/TCP sont utilisés.

## 1.2 Cisco

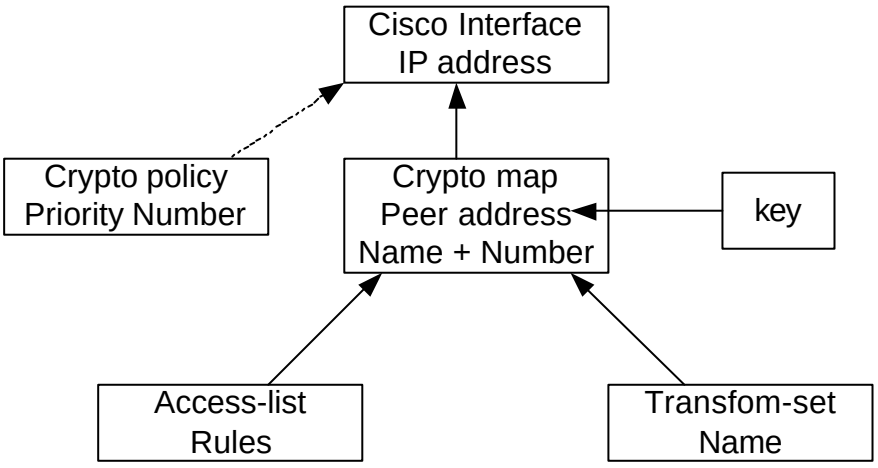
Un routeur est un équipement permettant, lorsque l'on a un réseau, de séparer ainsi que de diriger les paquets au bon endroit.

### 1.2.1 Philosophie et configuration

Les routeurs Cisco se configurent en mode terminal par une suite de commande. Pour obtenir plus d'information, se référer au manuel : *Software Configuration Guide* fournit avec l'équipement Cisco. Il ne faut pas négliger le temps d'adaptation au routeur car bien que la configuration en mode console présente certains avantages, l'apprentissage est un peu plus long.
Suite à des problèmes avec l'utilisation de Hyper Terminal (fournit avec Win2k) j'ai utilisé un autre émulateur de terminal ''Tera Term''. Celui-ci me permet d'obtenir un buffer plus grand, ce qui est appréciable en mode debug.

### 1.2.2 Commandes importantes

Voici un arbre permettant de comprendre comment s'imbrique les différents modules des routeurs Cisco.




*crypto isakmp policy 1*
Les policys sont des "modules" qui permettent de définir un schéma de connexion pour les SA de ISAKMP. Le numéro définit la priorité, plus il est bas plus il est prioritaire.

| | |
|---|---|
| authentication | : Type d'authentification (secret partagé, signature, chiffrement du nonce (voir § 2.4) |
| encryption | : Type de chiffrement |
| group | : Set the Diffie-Hellman group |
| hash | : Type d'algorithme de Hash |
| lifetime | : durée de vie de la SA ISAKMP |

*crypto isakmp key 1234567890 address 129.194.184.213*
Cette commande permet de définir le secret (*1234567890*) partagé ainsi que le partenaire (ip : 129.194.184.213) qui utilisera le même secret. On imagine facilement que plusieurs partenaires peuvent demander une connexion et qu'ils n'utilisent pas tous le même secret partagé.

*crypto IPSec transform-set myset esp-des esp-sha-hmac*
Le transform-set définit quel chiffrement sera utilisé ainsi que le type de hash.
Les types suivants sont disponibles :

| | |
|---|---|
| ah-md5-hmac | |
| ah-sha-hmac | |
| comp-lzs | compression de IP par LZS |
| esp-3des | (cipher 168 bits) |
| esp-des | (cipher 56 bits) |
| esp-md5-hmac | |
| esp-null | ESP transform w/o cipher |
| esp-sha-hmac | |

mode transport/tunnel

*crypto map mymap 10 ipsec-isakmp*
La map est le ''module'' qui une fois associée à une interface permettra de définir les conditions de connexion IPSEC. Le numéro permet de définir le niveau de priorité. Plus il est bas plus la map est prioritaire.

| | |
|---|---|
| match | liaison entre Access-list et map |
| set peer | affecte la map au peer (adresse ip) |
| set pfs | Active/Désactive la fonction PFS |
| set security-association | Définition des durées de vie |
| set transform-set | transform set à utiliser (il est possible d'en mettre plusieurs) |

*crypto dimamic-map*
Lorsqu'une map est définie en temps que dynamique, le routeur va s'adapter automatiquement au partenaire en créant une map temporaire. Une fois que la connection est établie, la map temporaire est effacée. Les dynamic-map ne sont utilisées que si aucune map n'a pu être satisfaite.

*access-list 100 permit ip 10.0.0.0* ***0.255.255.255*** *129.194.184.213 0.0.0.0*
Pour augmenter la sécurité, ainsi pour éviter que le routeur ne demande à chaque client potentiel de faire une connexion IPSec, on crée une access-list qui va permettre de définir quel trafic nécessitera une connexion IPSec.

On constate qu'après les adresses IP se trouve un autre champ (WildCard). Toutes les valeurs à 1 peuvent être modifiées alors que celles à zéro doivent être inchangées. Dans notre cas, les adresses valides sont 10.X.X.X ou X peut prendre n'importe quelle valeur.

*interface FastEthernet 0/0*
*crypto map mymap*
Il est important d'appliquer les règles définies plus haut à une interface.

*clear crypto isakmp*
*clear crypto sa*
Ces commandes suppriment les connexions IPSec et ISAKMP

Commande de debug :

Debug crypto ipsec
Debug crypto isakmp
Debug crypto key-exchange

Remarque : L'adresse 0.0.0.0 a été définie de façon à ce qu'elle valide toutes les adresses.

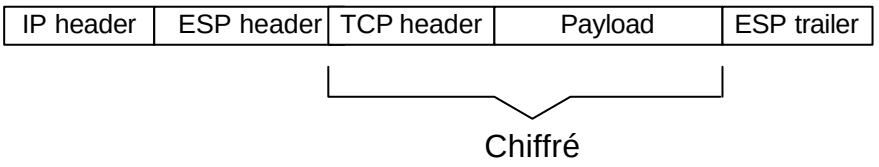
## 2 Description d'IPSec

Lors d'une connexion IPSec, une *Security Association* (SA) est définie. Elle va permettre de déterminer comment les paquets vont être sécurisés : Protocole, type de clé et leur durée de validité.

### 2.1 ESP, AH

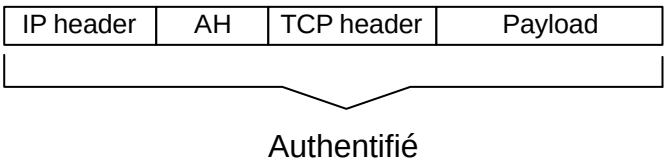
Deux possibilités d'utilisation d'IPSec sont possibles : ESP et AH

L'option ESP permet le chiffrement des données (voir figure ci-dessous)



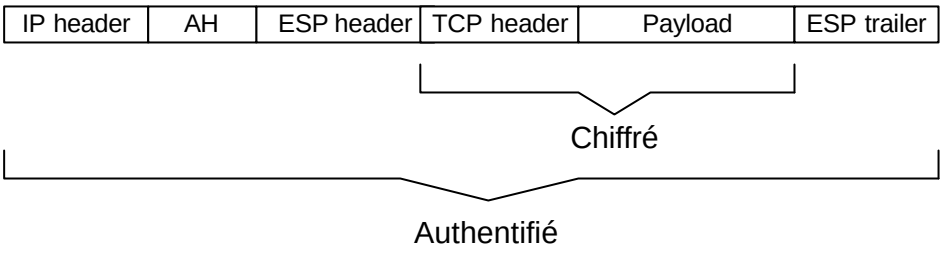

IPSec avec ESP en mode transport

L'option AH permet de faire une authentification. Cette option ne comprend pas de chiffrement.




IPSec avec AH en mode transport

Il est possible de combiner ces deux options afin d'avoir une authentification ainsi qu'un chiffrement. On obtient une trame sous la forme :




IPSec avec ESP,AH en mode transport

## 2.2 Types de chiffrements

La norme IPSec permet d'utiliser plusieurs types de chiffrement. Dans notre cas :
DES 56-bits
3DES 168-bits
deux types de *hash* sont disponibles :
md5 et sha1
deux méthodes d'authentification sont disponibles :
Secret partagé, RSA signature (clé publique, privée)

Le choix du chiffrement dépend des équipements (tous les équipements ne supportent pas tous les types de chiffrement) ainsi que les performances et le degré de sécurité désiré. Il faut savoir que les ressources du processeur demandées lors du chiffrement sont variables en fonction du type. Ces chiffrements sont plus ou moins complexes et c'est ce qui va déterminer leur capacité à ne pas être déchiffrés.
Le hash est un condensé fait avant le chiffrement des données. Ce condensé permet lors de la réception du message de vérifier que celui-ci n'a pas été modifié. En effet, il ne correspond qu'un seul hash par donnée.

## 2.3 Les clés

Il est possible d'utiliser deux types de clé :

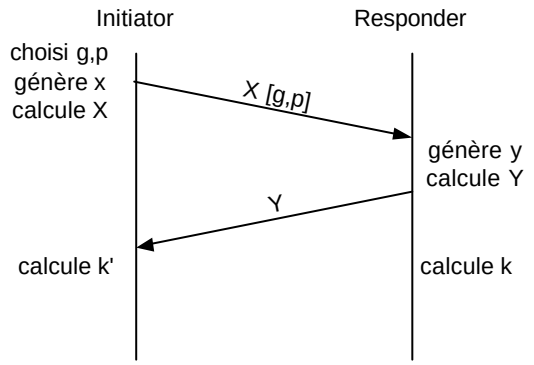
- le secret partagé : les deux équipements connaissent un même secret et l'utilisent pour permettre l'authentification. L'utilisation de ce type de clé est facile à installer mais est plus fragile à des attaques. Pour qu'un pirate puisse s'authentifier, il lui suffit de faire des demandes de connexion en essayant des mots de passe. Si le secret a mal été choisi, il devient facile pour lui de le retrouver. Le but de ce diplôme n'est pas d'aller dans les détails. Il existe plusieurs ouvrages dédiés à ce genre de problème et il pourrait être intéressant pour le lecteur voulant aller plus loin de se sensibiliser à ce genre de détail.
- Les certificats numériques : ceux-ci permettent une plus grande sécurité au détriment du temps de connexion. Il est plus difficile de mettre en oeuvre une architecture utilisant des certificats car il faut utiliser une

autorité de certification (CA) qui va permettre la vérification des authentifications.

### 2.4 Description de Internet Key Exchange (IKE) :

Dans un premier temps, nous expliquerons le fonctionnement d'un échange de clé avec la méthode du Diffie-Hellman.

Syntaxe :
p est un grand nombre premier
g est un entier
L'opérateur *mod* permet de faire une division entière.




L'initiator I choisit trois valeurs aléatoirement g, p et x. x est un entier. Il est recommandé de prendre une valeur sur 180 bits au minimum.
I calcule X ( $X = g^x \bmod p$ )
Il envoie X, g et p au responder R.
R génère y qui est aussi un entier sur 180 bits au minimum et calcule Y ($Y = g^y \bmod p$ ). Ensuite, R envoie Y a I.

A ce stade, I et R peuvent calculer la clé de session.
I : $k' = X^Y \bmod p$

R : $k = Y^X \bmod p$

Les clés sont identiques car :

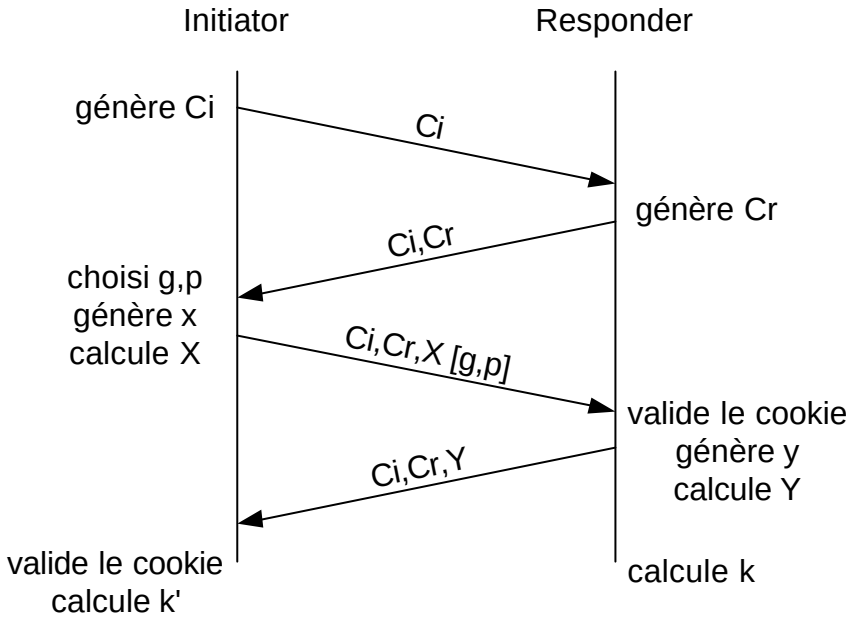
$$k = (g^y \bmod p)^x \bmod p = g^{xy} \bmod p = (g^x \bmod p)^y \bmod p = k'$$

On remarque tout de suite que cette méthode d'échange de clé est très sensible à deux types attaques : ''Denied of Service'' et ''Man in the Middle''
Une attaque du type ''Denied of Service'' consiste à envoyer des demandes de connexion de façon intensive avec plusieurs adresses IP source.
Une attaque du type ''Man in the Middle'' consiste à se placer entre les deux partenaires et à intercepter les messages qu'ils envoient afin de les lire et/ou d'en retransmettre d'autres.

Denied of Service (DoS)

L'échange Diffie-Hellman est très sensible à ce type d'attaque car le Responder doit à chaque demande de connexion calculer une exponentielle d'un nombre sur 180 bits au minimum. On voit tout de suite que si des demandes de connexions sont effectuées en grand nombre dans un laps de temps très court, le Responder va utiliser toutes ses ressources processeurs.

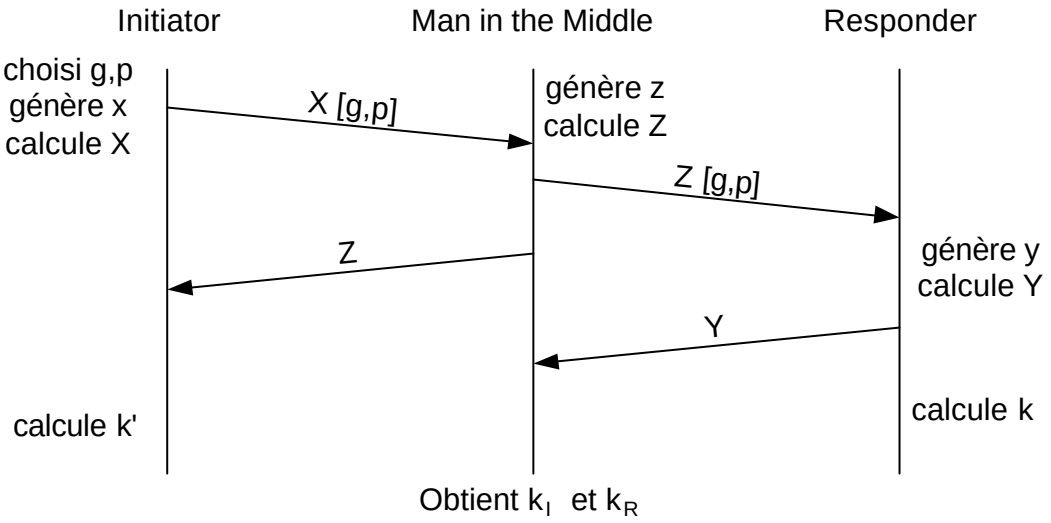
La solution qui a été proposée ne permet pas d'éliminer ce problème, mais permet de limiter ainsi que de compliquer une attaque DoS.
Avant de calculer les clés, l'Initiator et le Responder vont échanger des cookies.




I va générer aléatoirement son cookie Ci et l'envoyer à R. A la réception de Ci, R génère aléatoirement son cookie et renvoyer la paire de cookie.
La suite est identique à un échange standard de Diffie-Hellman, à la différence près, qu'à chaque échange la paire de cookies est transmise.
L'avantage est que la paire de cookies est associée de chaque coté à l'adresse du partenaire. De ce fait, si une attaque de type DoS est faite sans passer par l'échange des cookies, le Responder va jeter le paquet étant donné qu'il n'y a aucune relation entre les cookies et l'adresse du partenaire.
Si l'attaquant fait les deux premiers échanges de façon conformes et tente d'envoyer plusieurs fois le paquet 3, le Responder va détecter une erreur dans la suite logique de l'échange et va jeter les paquets.
L'utilisation des cookies rend la tache plus difficile à l'attaquant car il doit pour chaque paquet envoyé, être capable de recevoir le cookie du Responder afin de faire l'association correcte.

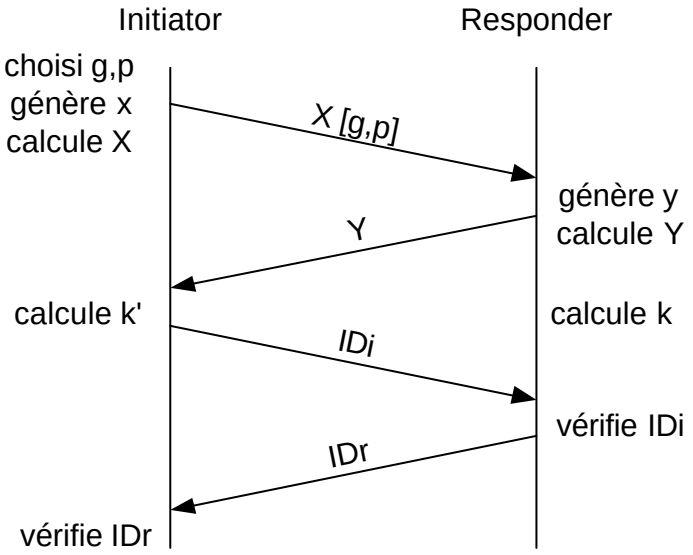
Man in the Middle :

Nous allons d'abord voir pourquoi une attaque de ce type est particulièrement dangereuse lors d'un échange Diffie-Hellman.




Le "Man in the Middle" M récupère X,g,p génère Z, l'envoie à R et à I. Après que R aye calculé Y, il l'envoie et M l'intercepte. Dans ce cas de figure, M a obtenu deux clés de session, une pour R et une pour I. Il peut donc déchiffrer et modifier les messages envoyés par I et R sans que ceux-ci ne se doutent de rien.

Pour parer cette attaque, I et R vont envoyer un ID (jeton d'identification). Il peut s'agire d'une adresse IP, d'un nom d'utilisateur, …



Quand l'ID est envoyé, il est chiffré avec la clé de session et avec la signature numérique de l'expéditeur. Dans le cas d'un secret partagé, à la place de la signature numérique, c'est le secret qui sera utilisé.
De cette façon, si une personne parvient à intercepter les échanges, elle ne pourra pas se faire passer par l'un des deux partenaires.

## 2.5 Internet Security Associations and Key Management Protocol (ISAKMP)

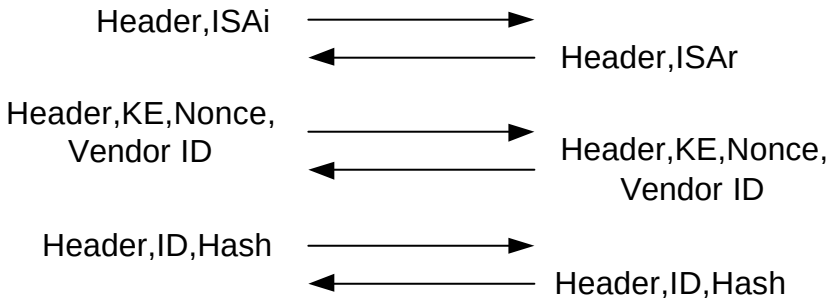
Avant chaque connexion ISPsec, il y a chaque fois un établissement d'une SA avec ISAKMP. C'est lors de cette phase que les clés seront échangées.
Une SA de ISAKMP n'a pas besoin d'avoir une durée de vie. Lorsque la connexion IPSec est établie, la SA se termine.
Il existe deux modes de connexion :

### 2.5.1 Phase 1

La phase 1 permet de créer une SA pour ISAKMP

#### 2.5.1.1 Mode normal

Comme on peut le voir sur la figure ci-dessous, il est constitué de six paquets.




Header : entête

ISA : Initiator Security Association
Dans cette partie, les deux partenaires vont définir quel type de chiffrement sera utilisé ainsi que le type de hash et s'ils utiliseront l'option ESP et/ou AH.
ISAi est envoyé par l'initiateur et envoie une suite de proposition.
ISAr est la réponse qui indique quelle proposition sera utilisée.

KE : Key Exchange
C'est l'échange des clés que nous avons vu dans le chapitre précédent.

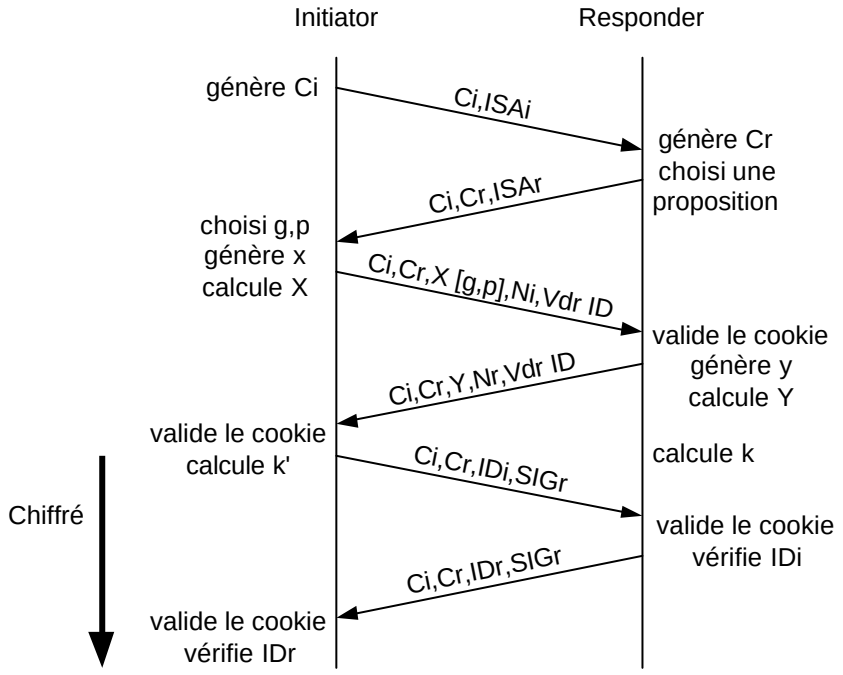
Nonce :
Le Nonce est un chiffre de 64-2048 bits choisi aléatoirement afin de rajouter de l'aléa lors de la conception des clés. Les routeurs Cisco utilisent un chiffre de 192 bits.

ID : Identification
C'est le même ID que dans le chapitre précédent.

Hash :
Permet de vérifier l'intégrité des données transmisent. Pour vérifier, les partenaires envoient des données de la transmission en utilisant la fonction de hash définie lors de l'échange. Il est possible que plusieurs échanges aient lieux.

Vendor ID :
Ce champ est difficile à définire, il contient des informations sur les partenaires (par exemple : Cisco, Sentinel, …) ce qui permet de savoir comment l'implémentation des clés est faite. Ce qui est envoyé, est un hash de la chaîne décrivant le Vendor plus un numéro de version.

Schéma détaillé de l'échange :




Remarque : SIGi,r correspond à la signature numérique. Lors d'un échange avec secret partagé, c'est un hash qui est envoyé.

Sur cette base, trois clés supplémentaires sont générées ($SKEYD_x$) :
$SKEYD_a$ : est utilisée pour l'authentification.

$$SKEYID_a = hashfunct(SKEYID, SKEYID_d, k|, C_i|, C_r|1)$$

$SKEYD_e$ : est utilisée pour chiffrer les données transmises.

$$SKEYID_e = hashfunct(SKEYID, SKEYID_a, k|, C_i|, C_r|2)$$

$SKEYD_d$ : est utilisée pour générer d'autre clé si nécessaire.

$$SKEYID_d = hashfunct(SKEYID, k|, C_i|, C_r|0)$$

*hashfunct* a été négocié lors des échanges précédents.
Les Hash utilisés pour l'authentification sont :

$$HASH_i = hashfunct(SKEYID_a, X|, Y|, C_i|, C_r|, ISA_i|ID_i)$$

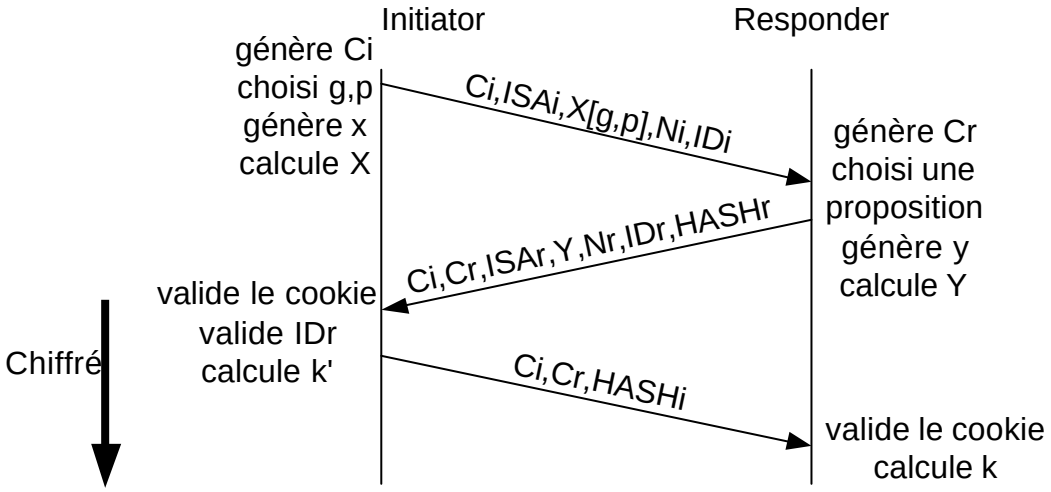
$$HASH_d = hashfunct(SKEYID_a, X|, Y|, C_i|, C_r|, ISA_d|ID_d)$$

SKEYID est calculé à l'aide des informations obtenues lors de l'échange Diffie-Hellman.
Dans les cas d'une utilisation avec un secret partagé on a :

$$SKEYID = hashfunct(k, N_i|N_r)$$

#### 2.5.1.2 Le mode Agressif

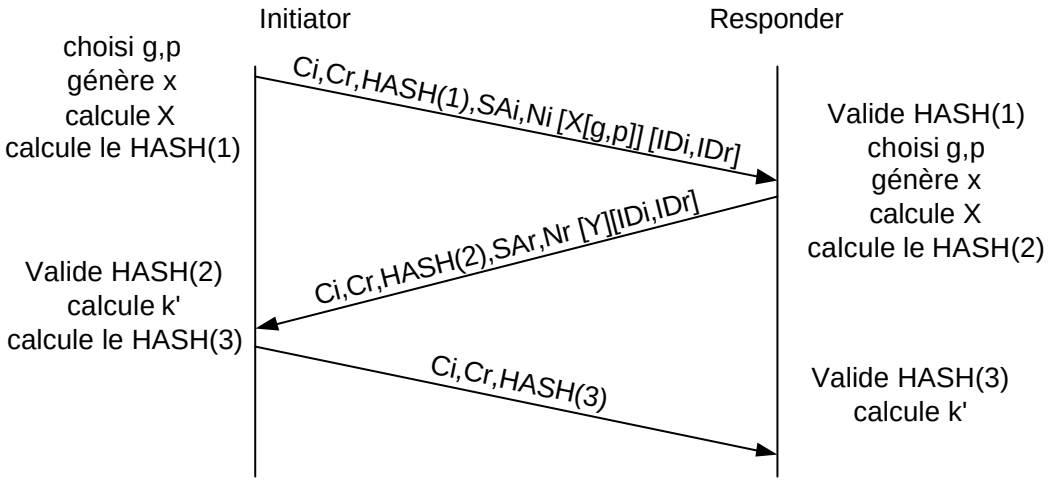
Ce mode a pour but de diminuer le temps de connexion. Pour cela, seul trois paquets sont transmis.




Cette méthode ne peut pas être utilisée si un l'authentification se fait avec des clés publiques.
Le danger de cette méthode est que les ID ne peuvent pas être chiffrés.

### 2.5.2 Phase 2

La phase 2 permet de créer la SA pour IPSEC. Cette phase est entièrement chiffrée avec les clés définient dans la phase 1.




Si la fonction PFS est activée, un nouvel échange Diffie-Hellman a lieu (les valeurs entre [ ] sont envoyées)..

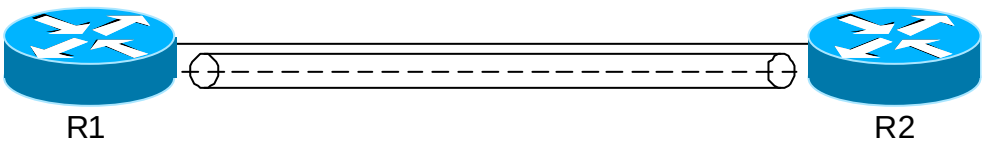
Le HASH est utilisé pour l'authentification des données.
SA : Proposition relative à IPSec (ESP, AH).

## 2.6 Fonction Perfect Forward Secrecy (PFS)

Cette fonction est une sécurité supplémentaire. C'est-à-dire que lors de l'échange de données, des clés supplémentaires sont générées à intervalle régulier et sont utilisées pour chiffrer le message.
Pour plus de détail, à l'adresse :
http://www.skip.org/spec/epsf.html

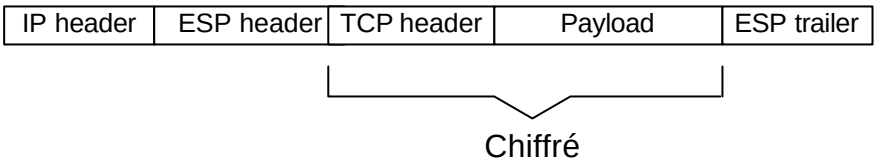
## 2.7 Mode transport

Ce mode est utilisé pour des connexions de host a host ou router to router




Cette figure est un peu simpliste mais permet de bien comprendre que la connexion se fait de bout en bout. Ce type de configuration est très peu utilisé car les informations ne sont utiles qu'aux routeurs et ne sont donc pas routées. On peut imaginer utiliser une telle configuration pour des protocoles tel que RIP ou TFTP.

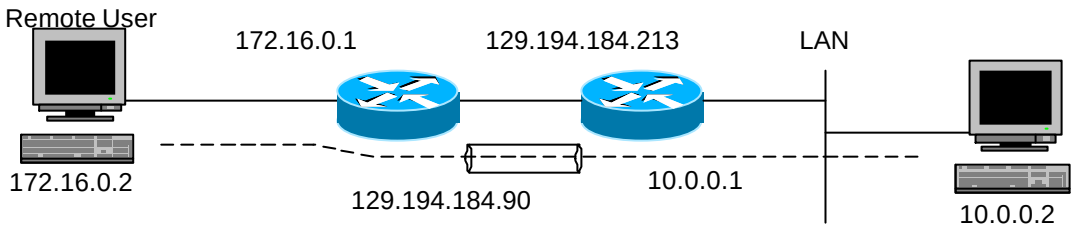
Les paquets IP envoyés ont la structure :



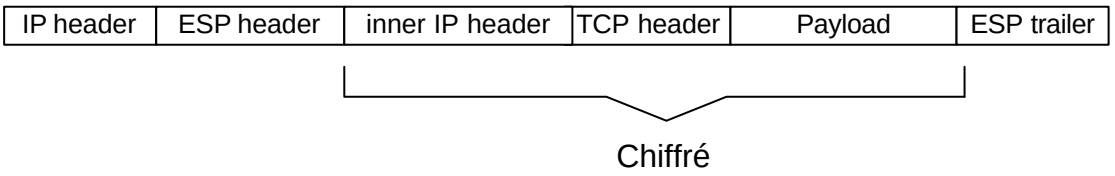

Le champ ESP contient les champs importants suivant :
Security Parameters Index (SPI) : permet d'identifié une SA
Numéro de séquence : est incrémenté de un à chaque échange de paquet sécurisé

### 2.8 Mode tunnel

Ce mode est utilisé lorsqu'un équipement veut atteindre un réseau distant.




Dans cette configuration, les paquets IPSec ont la forme :




Le premier champ IP a pour source le Remote User (172.16.0.2) et pour destination le deuxième routeur (129.194.184.213)
Le routeur va décoder le paquet IPSec et le renvoyer en utilisant le champ inner IP header (S : 172.16.0.2 D: 10.0.0.2). Le paquet qui va transiter dans le LAN aura une structure IP standard.

SSH Sentinel 1.1 b3 (amélioration)

Cette version de SSH Sentinel permet de définir la durée de vie des clés ainsi que la durée de vie de la connexion IPSec.

Avec une configuration de ce type, on peut se demander si la fonction NAT est utile ou non. Etant donné la modification des entêtes des paquets, le Nat

n'est pas nécessaire . Cependant, si les machines des sous réseaux veulent sortire (Internet), il faudra utiliser la fonction NAT.

## 3. Configuration

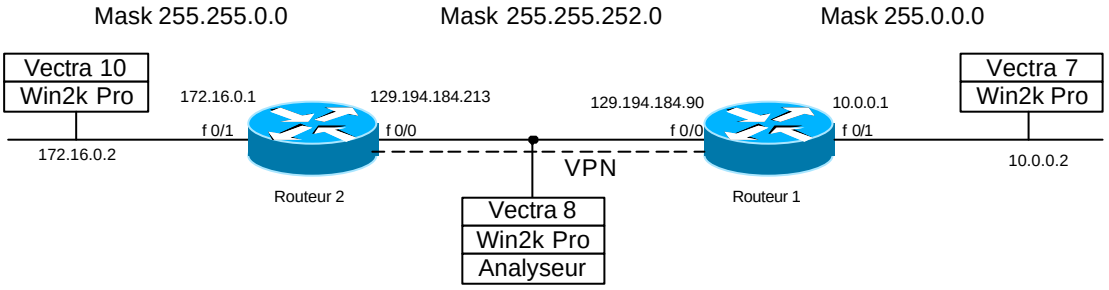
Il est important de bien savoir à quoi servent les commandes Cisco ainsi que le fonctionnement d'IPSec avant de se lancer dans la configuration du routeur. Il faut aussi prévoir un temps d'adaptation à la configuration en mode console.
Ici, on trouvera des parties de configuration. Celles-ci seront commentées quand vous trouverez ''//''

### 3.1 Secret partager

Dans un premier temps, nous étudierons les échanges avec un secret partagé car cela nous permet d'éviter d'éventuels problèmes avec une entité de certification (CA).

#### 3.1.1 Configuration Routeur to Routeur mode tunnel

Cette configuration peut être utilisée si on veut créer une connexion sécurisée entre deux sites distants.




Cette configuration permet d'obtenir de bonne base pour la suite. Les deux routeurs Cisco utilisant le même module IPSec, on est sur de la compatibilité. Donc si la connexion ne se fait pas, il s'agit uniquement d'une erreur de configuration.
Attention les noms entre les modules doivent concorder. Si ce n'est pas le cas, le routeur Cisco ne donnera pas de message d'erreur et la connexion ne se fera pas.
Configuration utilisée sur le routeur 2 (voir annexe 1 et 2)

```
crypto cisco key-timeout 1
```
//définit la durée de vie de la clé en minute

```
crypto isakmp policy 1
 encryption des
 hash sha
 authentication pre-share
```
//authentification avec secret partagé.
```
 group 2
```

```
crypto isakmp key 1234567890 address 129.194.184.90
```
//défini le secret partagé et à quel partenaire il est associé

```
crypto ipsec security-association lifetime seconds 120
```
//durée de vie de la SA

```
crypto ipsec transform-set myset esp-des esp-sha-hmac
```
//types de chiffrement qui seront utilisés

```
crypto map test 10 ipsec-isakmp
 set peer 129.194.184.90
 set transform-set myset
 match address 101
```
// la crypto map est associée à un partenaire, une méthode de chiffrement, une règle d'accès

```
interface FastEthernet0/0
 ip address 129.194.184.213 255.255.252.0
 no ip directed-broadcast
 duplex auto
 speed auto
 crypto map test
```
// la map définie ci-dessus est associée à une interface

```
access-list 101 permit ip 172.16.0.0 0.0.0.255 10.0.0.0
0.0.0.255
```

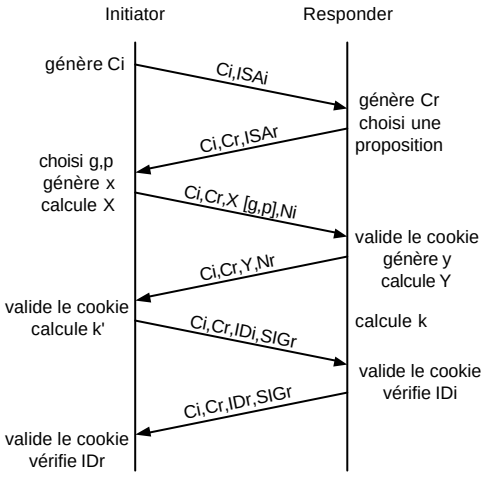
Les access-list agissent comme des filtres qui vont permettre de définire quel trafic sera chiffré. Il est possible de définire des filtres très précis. Par exemple : *access-list 101 permit tcp X.X.X.X 0.0.0.0 Y.Y.Y.Y 0.0.0.0 eq 80*
Cette règle ne chiffrera que le trafic de X -> Y avec un port de 80 comme destination. Il est possible d'aller plus loin encore avec en vérifiant s'il y a le bit TCP SYN, ACK,… Il est aussi possible de refuser un trafic avec *deny* à la place de *permit*.
Ce systeme permet de définir avec précision le trafic à chiffrer. Ainsi, en ne chiffrant que ce qui est nécéssaire, on peut diminuer la charge sur le processeur du routeur.

On remarquera que la connexion prend un temps non négligeable (15 secondes) lors de la première requête après le boot du routeur. Il faudrait voir pourquoi il y a un tel délai. Aucune information tant sur le site de Cisco que dans les *news group* n'est disponible.

Le routeur 1 n'a pas été configuré avec les même *time-out* afin de vérifier que lors de la connexion, le partenaire ayant la configuration la plus restrictive prend le dessus.
L'utilisation des délais est intéressante car elle permet de voir les routeurs Cisco faire un nouvel échange de clé en cours de connexion.

Exemple de capture d'une connexion Routeur to Routeur.
Le Routeur 2 (initiator) fait une demande de connexion au Routeur 1 (responder).

Afin de simplifier la lecture, certains champs ne représentant pas un grand intérêt pour les explications ont été effacés. Les commentaires commencent par ‘’//’’. La capture complète se trouve à l’annexe 3




rappel de l’échange ISAKPM

**Src MAC Addr** **Dst MAC Addr** Protocol Src Other Addr Dst Other Addr Type Other Addr
**Routeur 2** **Routeur 1** ISAKMP 129.194.184.213 129.194.184.90 IP

+ UDP: Src Port: ISAKMP, (500); Dst Port: ISAKMP (500); Length = 80 (0x50)
ISAKMP: Major Version: 1 Minor Version: 0 Length: 72
ISAKMP: Initiator cookie = **D3 0A 7A D8 DE 01 DA 4C**//Cookie de l’initiator
ISAKMP: Responder cookie = 00 00 00 00 00 00 00 00//Cookie du responder (n’a pas encore été définit)
ISAKMP: **Next payload = Security Association** //prochain échange à faire
ISAKMP: Major version = **1** (0x1) //
ISAKMP: Minor version = **0** (0x0) //version de protocole supportée
ISAKMP: Exchange type = Identity Protection
ISAKMP: Flags summary = 0 (0x0)
ISAKMP: .....0 = **Payloads are not encrypted** //les clés ne sont pas encore définies
ISAKMP: Message ID = 0 (0x0)
ISAKMP: Length = 72 (0x48)
ISAKMP: **Payload type = Security Association** //type de l’échange
ISAKMP: Next payload = None
ISAKMP: Reserved = 0 (0x0)
ISAKMP: Payload length = 44 (0x2C)
ISAKMP: DOI = 1 (0x1)
ISAKMP: Situation = SIT_INDENTITY_ONLY
ISAKMP: Labeled domain identifier = 32 (0x20)
ISAKMP: Secrecy length = 257 (0x101)
ISAKMP: ERROR: Reserved = 1 (0x1), it should = 0 (0x0)
ISAKMP: Secrecy level = 00 00 00 18 01 01 00 00 80 01 00 01 80 02 00 02 80 04 00 02...

Ce premier paquet correspond dans le model théorique au premier échange. C’est à dire que l’initiateur fait une demande de connexion et envoie son cookie.

Src MAC Addr Dst MAC Addr Protocol Src Other Addr Dst Other Addr Type Other Addr
Routeur 1 Routeur 2 ISAKMP 129.194.184.90 129.194.184.213 IP

+ UDP: Src Port: ISAKMP, (500); Dst Port: ISAKMP (500); Length = 80 (0x50)
ISAKMP: Major Version: 1 Minor Version: 0 Length: 72
ISAKMP: Initiator cookie = **D3 0A 7A D8 DE 01 DA 4C**//Cookie de l’initiator
ISAKMP: Responder cookie = **47 7A DA B1 FF 8F BF BE**//Cookie du responder
ISAKMP: **Next payload = Security Association** //prochain échange à faire
ISAKMP: Major version = 1 (0x1) //
ISAKMP: Minor version = 0 (0x0) //version de protocole supportée
ISAKMP: Exchange type = Identity Protection

```
    ISAKMP: Flags summary = 0 (0x0)
      ISAKMP: .......0 = Payloads are not encrypted     //les clés ne sont pas encore définies
    ISAKMP: Message ID = 0 (0x0)
    ISAKMP: Length = 72 (0x48)
    ISAKMP: Payload type = Security Association        //type de l'échange
      ISAKMP: Next payload = None
      ISAKMP: Reserved = 0 (0x0)
      ISAKMP: Payload length = 44 (0x2C)
      ISAKMP: DOI = 1 (0x1)
      ISAKMP: Situation = SIT_INDENTITY_ONLY
```

Dans cet échange, le routeur 1 renvoie le cookie du routeur 2 ainsi que son cookie.

```
Src MAC Addr   Dst MAC Addr   Protocol   Src Other Addr    Dst Other Addr    Type Other Addr
Routeur 2      Routeur 1      ISAKMP     129.194.184.213   129.194.184.90    IP

+ UDP: Src Port: ISAKMP, (500); Dst Port: ISAKMP (500); Length = 212 (0xD4)
  ISAKMP: Major Version: 1  Minor Version: 0  Length: 204
    ISAKMP: Initiator cookie = D3 0A 7A D8 DE 01 DA 4C          //
    ISAKMP: Responder cookie = 47 7A DA B1 FF 8F BF BE         //les cookies
    ISAKMP: Next payload = Key Exchange
    ISAKMP: Exchange type = Identity Protection
    ISAKMP: Flags summary = 0 (0x0)
      ISAKMP: .......0 = Payloads are not encrypted     //les clés ne sont pas encore définies
    ISAKMP: Message ID = 0 (0x0)
    ISAKMP: Length = 204 (0xCC)
    ISAKMP: Payload type = Key Exchange                //débute l'échange des clés X[g,p]
      ISAKMP: Next payload = Nonce
      ISAKMP: Reserved = 0 (0x0)
      ISAKMP: Payload length = 132 (0x84)
      ISAKMP: Key exchange data = 71 BC 52 90 2F 4C D0 A6 9B A2 01 10 45 B9 86 B7 39 A2...
    ISAKMP: Payload type = Nonce                       //le nonce Ni est envoyé
      ISAKMP: Next payload = Vendor ID
      ISAKMP: Reserved = 0 (0x0)
      ISAKMP: Payload length = 24 (0x18)
      ISAKMP: Nonce data = 3E 3A 51 A9 34 60 15 2B B5 DC D2 C9 2F A6 70 A2 54 CB 97 DA
    ISAKMP: Payload type = Vendor ID                   //définition du fabriquant pour l'implémentation
      ISAKMP: Next payload = None
      ISAKMP: Reserved = 0 (0x0)
      ISAKMP: Payload length = 20 (0x14)
      ISAKMP: Vendor ID = 26 CD DD C5 DE 00 DA 4C A8 7C B1 61 A5 C6 A0 54

Src MAC Addr   Dst MAC Addr   Protocol   Src Other Addr    Dst Other Addr    Type Other Addr
Routeur 1      Routeur 2      ISAKMP     129.194.184.90    129.194.184.213   IP

+ UDP: Src Port: ISAKMP, (500); Dst Port: ISAKMP (500); Length = 212 (0xD4)
  ISAKMP: Major Version: 1  Minor Version: 0  Length: 204
    ISAKMP: Initiator cookie = D3 0A 7A D8 DE 01 DA 4C
    ISAKMP: Responder cookie = 47 7A DA B1 FF 8F BF BE
    ISAKMP: Next payload = Key Exchange
    ISAKMP: Exchange type = Identity Protection
    ISAKMP: Flags summary = 0 (0x0)
      ISAKMP: .......0 = Payloads are not encrypted     // les clés ne sont pas encore définies
    ISAKMP: Message ID = 0 (0x0)
    ISAKMP: Length = 204 (0xCC)
    ISAKMP: Payload type = Key Exchange                //échange des clés Y
      ISAKMP: Next payload = Nonce
      ISAKMP: Reserved = 0 (0x0)
      ISAKMP: Payload length = 132 (0x84)
      ISAKMP: Key exchange data = 73 41 01 62 83 FE 0F 51 06 E2 18 EC 84 32 0C 43 01 16...
    ISAKMP: Payload type = Nonce                       //le nonce Nr est envoyé
      ISAKMP: Next payload = Vendor ID
      ISAKMP: Reserved = 0 (0x0)
      ISAKMP: Payload length = 24 (0x18)
      ISAKMP: Nonce data = 62 55 60 81 4A E2 02 59 79 D3 EB 69 B2 6F 5C 5A E6 5C 6E 74
    ISAKMP: Payload type = Vendor ID
      ISAKMP: Next payload = None
```

```
    ISAKMP: Reserved = 0 (0x0)
    ISAKMP: Payload length = 20 (0x14)
    ISAKMP: Vendor ID = B2 BD 7D AC FF 8E BF BE 73 9D 46 3A E9 DD 48 E8
```

```
Src MAC Addr   Dst MAC Addr   Protocol    Src Other Addr     Dst Other Addr     Type Other Addr
Routeur 2      Routeur 1       ISAKMP   129.194.184.213   129.194.184.90     IP

+ UDP: Src Port: ISAKMP, (500); Dst Port: ISAKMP (500); Length = 76 (0x4C)
 ISAKMP: Major Version: 1  Minor Version: 0  Length: 68
   ISAKMP: Initiator cookie = D3 0A 7A D8 DE 01 DA 4C
   ISAKMP: Responder cookie = 47 7A DA B1 FF 8F BF BE
   ISAKMP: Next payload = Identification
   ISAKMP: Exchange type = Identity Protection
   ISAKMP: Flags summary = 1 (0x1)
     ISAKMP: .......1 = Payloads are encrypted          //les clés sont échangées et les paquets chiffrés
   ISAKMP: Message ID = 0 (0x0)
   ISAKMP: Length = 68 (0x44)
   ISAKMP: Payload type = Identification              //envoie IDi le reste est chiffré d'où les incohérences
     ISAKMP: Next payload = 0x75
     ISAKMP: ERROR: Reserved = 246 (0xF6), it should = 0 (0x0)
     ISAKMP: Payload length = 39010 (0x9862)
     ISAKMP: ID type = 0xB3
     ISAKMP: Protocol ID = 226 (0xE2)
     ISAKMP: Port = 20693 (0x50D5)
     ISAKMP: Identification data = 84 7A 28 63 09 1B F5 C5 D7 38 8B E0 A2 E9 EF CC 42 77...
```

```
Src MAC Addr   Dst MAC Addr   Protocol   Src Other Addr    Dst Other Addr    Type Other Addr
Routeur 1      Routeur 2       ISAKMP   129.194.184.90    129.194.184.213   IP

+ UDP: Src Port: ISAKMP, (500); Dst Port: ISAKMP (500); Length = 76 (0x4C)
 ISAKMP: Major Version: 1  Minor Version: 0  Length: 68
   ISAKMP: Initiator cookie = D3 0A 7A D8 DE 01 DA 4C
   ISAKMP: Responder cookie = 47 7A DA B1 FF 8F BF BE
   ISAKMP: Next payload = Identification
   ISAKMP: Exchange type = Identity Protection
   ISAKMP: Flags summary = 1 (0x1)
     ISAKMP: .......1 = Payloads are encrypted
   ISAKMP: Message ID = 0 (0x0)
   ISAKMP: Length = 68 (0x44)
   ISAKMP: Payload type = Identification              //envoie IDr le reste est chiffré d'où les incohérences
     ISAKMP: Next payload = 0x64
     ISAKMP: ERROR: Reserved = 104 (0x68), it should = 0 (0x0)
     ISAKMP: Payload length = 28843 (0x70AB)
     ISAKMP: ID type = 0xC9
     ISAKMP: Protocol ID = 30 (0x1E)
     ISAKMP: Port = 29092 (0x71A4)
     ISAKMP: Identification data = 46 2D C8 DF B9 CE AD 11 E5 C4 49 0F 2D 6A EA AE 2C B2...
```

Les trois paquets qui suivent sont un acquittement des donnés transmises lors de l'échange des clés. L'intégrité des données est vérifiée avec la fonction de *hash* définie lors de l'échange (SHA1, MD5). Dans ce cas de figure SHA1 a été utilisée.

```
Src MAC Addr   Dst MAC Addr   Protocol    Src Other Addr     Dst Other Addr     Type Other Addr
Routeur 2      Routeur 1       ISAKMP   129.194.184.213   129.194.184.90     IP

+ UDP: Src Port: ISAKMP, (500); Dst Port: ISAKMP (500); Length = 180 (0xB4)
 ISAKMP: Major Version: 1  Minor Version: 0  Length: 172
   ISAKMP: Initiator cookie = D3 0A 7A D8 DE 01 DA 4C
   ISAKMP: Responder cookie = 47 7A DA B1 FF 8F BF BE
   ISAKMP: Next payload = Hash
   ISAKMP: Flags summary = 1 (0x1)
     ISAKMP: .......1 = Payloads are encrypted
   ISAKMP: Message ID = 403159536 (0x1807B9F0)
   ISAKMP: Payload type = Hash
```

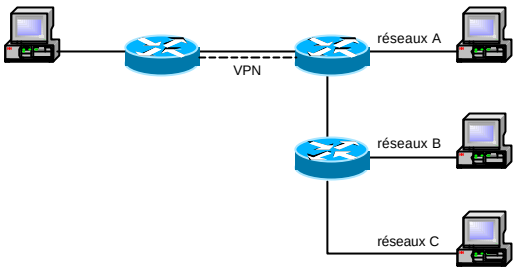
```
        ISAKMP: Next payload = 0xD5
        ISAKMP: ERROR: Reserved = 195 (0xC3), it should = 0 (0x0)
        ISAKMP: Payload length = 37342 (0x91DE)
        ISAKMP: Hash data = 5A 82 20 2B 09 87 02 FC FF B0 2F 14 4A 18 11 92 F6 20 E7 9F B6...

Src MAC Addr    Dst MAC Addr    Protocol    Src Other Addr     Dst Other Addr     Type Other Addr
Routeur 1           Routeur 2           ISAKMP   129.194.184.90     129.194.184.213    IP

+ UDP: Src Port: ISAKMP, (500); Dst Port: ISAKMP (500); Length = 180 (0xB4)
  ISAKMP: Major Version: 1  Minor Version: 0  Length: 172
     ISAKMP: Initiator cookie = D3 0A 7A D8 DE 01 DA 4C
     ISAKMP: Responder cookie = 47 7A DA B1 FF 8F BF BE
     ISAKMP: Next payload = Hash
     ISAKMP: Flags summary = 1 (0x1)
        ISAKMP: .......1 = Payloads are encrypted
     ISAKMP: Message ID = 403159536 (0x1807B9F0)
     ISAKMP: Payload type = Hash
        ISAKMP: Next payload = 0x43
        ISAKMP: ERROR: Reserved = 196 (0xC4), it should = 0 (0x0)
        ISAKMP: Payload length = 56671 (0xDD5F)
        ISAKMP: Hash data = D3 B8 AA D6 E2 84 EC D0 BB 6C F5 31 D2 63 F4 9F 9E 59 64 99 5F...

Src MAC Addr    Dst MAC Addr    Protocol    Src Other Addr     Dst Other Addr     Type Other Addr
Routeur 2           Routeur 1           ISAKMP   129.194.184.213    129.194.184.90     IP

+ UDP: Src Port: ISAKMP, (500); Dst Port: ISAKMP (500); Length = 68 (0x44)
  ISAKMP: Major Version: 1  Minor Version: 0  Length: 60
     ISAKMP: Initiator cookie = D3 0A 7A D8 DE 01 DA 4C
     ISAKMP: Responder cookie = 47 7A DA B1 FF 8F BF BE
     ISAKMP: Next payload = Hash
     ISAKMP: Flags summary = 1 (0x1)
        ISAKMP: .......1 = Payloads are encrypted
     ISAKMP: Message ID = 403159536 (0x1807B9F0)
     ISAKMP: Payload type = Hash
        ISAKMP: Next payload = 0x79
        ISAKMP: ERROR: Reserved = 69 (0x45), it should = 0 (0x0)
        ISAKMP: Payload length = 11466 (0x2CCA)
        ISAKMP: Hash data = A1 E1 65 36 72 0B 28 23 05 45 7B A5 D2 8B 49 87 6B A3 29 1C D6...
```

Quand ces échanges sont finis, la connexion est sécurisée et les paquets suivant ont la forme:

```
Src MAC Addr    Dst MAC Addr    Protocol    Src Other Addr     Dst Other Addr     Type Other Addr
Routeur 2           Routeur 1           ESP         129.194.184.213    129.194.184.90     IP

+ IP: ID = 0xF; Proto = 0x32; Len: 112
  ESP: SPI = 0xDC702EF, Seq = 0x1
     ESP: Security Parameters Index = 231146223 (0xDC702EF)
     ESP: Sequence Number = 1 (0x1)
     ESP: Rest of Frame: Number of data bytes remaining = 84 (0x0054)

Src MAC Addr    Dst MAC Addr    Protocol    Src Other Addr     Dst Other Addr     Type Other Addr
Routeur 1           Routeur 2           ESP         129.194.184.90     129.194.184.213    IP

+ IP: ID = 0xDC; Proto = 0x32; Len: 112
  ESP: SPI = 0x18DB01D2, Seq = 0x1
     ESP: Security Parameters Index = 417006034 (0x18DB01D2)
     ESP: Sequence Number = 1 (0x1)
     ESP: Rest of Frame: Number of data bytes remaining = 84 (0x0054)
```

On voit que le contenu de ces paquets est chiffré.
Ils ont été générés suite à une commande ping.

Remarque : Dans un cas réel, les configurations sont bien plus complexes, avec de multiple sous-réseaux.



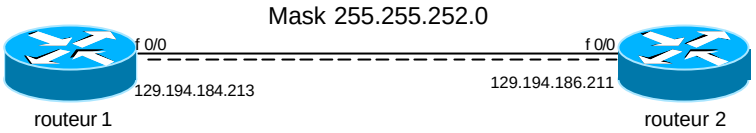

La question qui se pose alors est de savoir si une connexion VPN peut satisfaire un échange entre tous les sous-réseaux. Et comment gérer ce qui sera chiffré et ce qui ne le sera pas.. Pour cela, on utilise les access-list : soit en chiffrant tout avec la valeurs 0.0.0.0 (any), soit en énumérant chaque sous-réseaux et les données à chiffrer.
D'un point de vue sécurité, il est préférable de tout chiffrer. Cependant, une tel configuration peut entraîner un diminution des performances (trop de chiffrement). Il faut donc avant de mettre un configuration en place, réfléchire à l'impacte quelle peut avoir.

Le temps de connexion est de ~1,6 secondes, ce qui peut poser passablement de problèmes, si la SA se termine. Pendant ces 1,6 secondes, aucune information ne peut passer d'un sous-réseaux à l'autre. Pour parer ce problème, il existe une commande (*isakmp* ***keepalive***). Cette commande permet de garder la SA active en envoyant des paquets avant le time-out de la SA. Il est dit que cette commande peut avoir une grande incidence sur les performances du routeur. Malheureusement, par manque de temps, je n'ai pas eu le temps de faire des tests pour voir si l'impacte de cette commande est vraiment important.

Si des paquets passent d'un routeur à l'autre. Ceux-ci ne sont pas chiffrés. Ceci vient de la façon dont l'acces-list a été définie. (annexe 15)

### 3.1.2 Configuration Routeur to Routeur mode transport

Cette configuration est utilisée pour masquer des échanges tel que RIP, TFTP car en interceptant les paquets RIP, il est possible d'obtenir des informations sur la topologie du réseau.




Dans les routeurs Cisco, le mode transport n'est pas activé par défaut. Quand le mode transport est activé, le routeur accepte soit une connexion mode transport soit mode tunnel. En mode tunnel, un connexion en mode transport n'est pas possible.

J'ai utilisé une configuration avec option AH uniquement. Cela me permet de voir si les paquets ont la trame vue dans la théorie.

Configuration du routeur :

```
crypto isakmp policy 1
 authentication pre-share
 group 2

crypto isakmp key 1234567890 address 129.194.186.211

crypto ipsec transform-set myset ah-md5-hmac
mode transport
// c'est ici que le mode transport est activé.

crypto map test 10 ipsec-isakmp
 set peer 129.194.186.211
 set transform-set myset
 match address 101

interface FastEthernet0/0
 ip address 129.194.184.213 255.255.252.0
 crypto map test
 no shutdown

ip route 0.0.0.0 0.0.0.0 129.194.186.211

access-list 101 permit ip  129.194.184.213 0.0.0.0
                           129.194.186.211 0.0.0.0
```

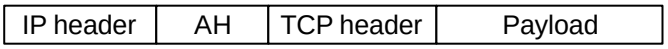
L'échange ISAKMP n'est pas très intéressant, il garde la même structure que pour une connexion en mode tunnel.

J'ai utilisé la commande ping depuis le routeur afin de simuler un trafic routeur to routeur. On peut très bien imaginer que les paquets qui passent sont des paquets RIP.

On obtient dans la capture :

```
Frame    Time        Src MAC Addr    Dst MAC Addr    Src Other Addr     Dst Other Addr     Type Other Addr
10       6.299058    Routeur 2       Routeur         129.194.184.213    129.194.186.211    IP

+ Frame: Base frame properties
+ ETHERNET: ETYPE = 0x0800 : Protocol = IP:  DOD Internet Protocol
+ IP: ID = 0x6; Proto = 0x33; Len: 124
+ AH: Protocol = ICMP, SPI = 0x119A0116, Seq = 0x1
+ ICMP: Echo: From 129.194.184.213 To 129.194.186.211

Frame  Time       Src MAC Addr   Dst MAC Addr  Src Other Addr      Dst Other Addr     Type Other Addr
11     6.299058   Routeur 1      Routeur 2     129.194.186.211     129.194.184.213    IP

+ Frame: Base frame properties
+ ETHERNET: ETYPE = 0x0800 : Protocol = IP:  DOD Internet Protocol
+ IP: ID = 0x6; Proto = 0x33; Len: 124
+ AH: Protocol = ICMP, SPI = 0x14251F9A, Seq = 0x1
+ ICMP: Echo Reply: To 129.194.184.213 From 129.194.186.211
```

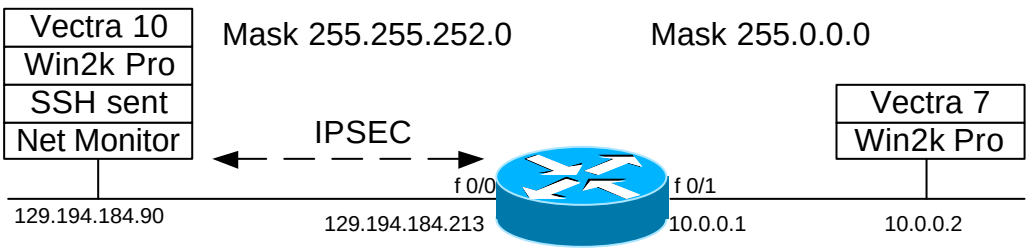
Sur cette capture, on voit bien que la trame correspond à la théorie :




La capture complète se trouve à l'annexe 4.

### 3.1.3 Connexion Host to Routeur

Nous avons utilisé SSH Sentinel 1.1 bêta 3 avec pilot 1.2.0.15 du coté client. L'amélioration la plus évidente est qu'il est possible de définire la durée de vie des clés ainsi que la connexion IPSec.




Cette configuration m'a posé passablement de problème du fait qu'il n'est pas possible de savoir avec exactitude ce dont SSH Sentinel a besoin pour établir la connexion.
En essayant de faire une configuration du même type que pour la connexion Routeur to Routeur, je me suis heurté à un problème. Je n'ai pas pu définir

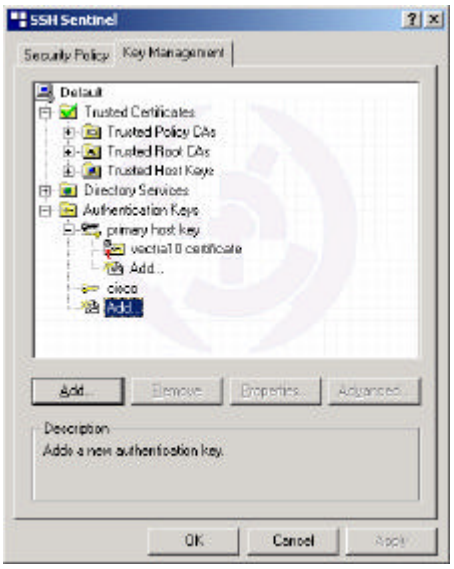
quelle était la cause exacte de l'erreur. Cependant, l'échange des clés se fait sans problème et c'est après que les deux équipements n'arrivent plus à s'entendre car selon Sentinel le champ Vendor ID lui est inconnu.
En regardant dans les log du routeur Cisco, on n'obtient pas d'information permettant de résoudre le problème.
En regardant dans les log de SSH Sentinel, on a une information supplémentaire qui est que le routeur Cisco utilise un mauvais champ ID.
La solution à ce problème, est d'utiliser les dynamic-map. Mais en utilisant ces map, il devient difficile de savoir ce que les partenaires se sont échangés.

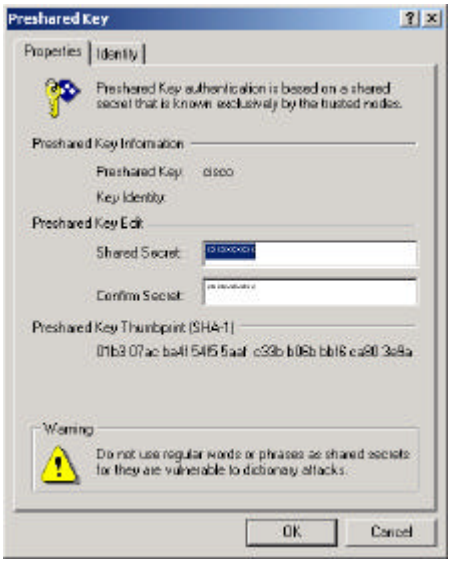
La configuration avec les dynamic-map nous donne : (annexe 5)

```
crypto isakmp enable
crypto isakmp identity address
crypto isakmp policy 1
 encryption des
 hash sha
 authentication pre-share
 group 2
crypto isakmp key 1234567890 address 129.194.184.213

crypto ipsec transform-set myset esp-des esp-sha-hmac

crypto dynamic-map mydynamap 10
 set transform-set myset

crypto map mymap 10 ipsec-isakmp dynamic mydynamap
// ici, on utilise une map dynamique afin que le routeur s'adapte
automatiquement à l'hôte

interface FastEthernet 0/0
 no shutdown
 description connected to Internet
 crypto map mymap
 ip address 129.194.184.90 255.255.252.0

interface FastEthernet 0/1
 no shutdown
 description connected to EthernetLAN_1
 ip address 10.0.0.1 255.0.0.0

ip route 0.0.0.0 0.0.0.0 129.194.184.213
```

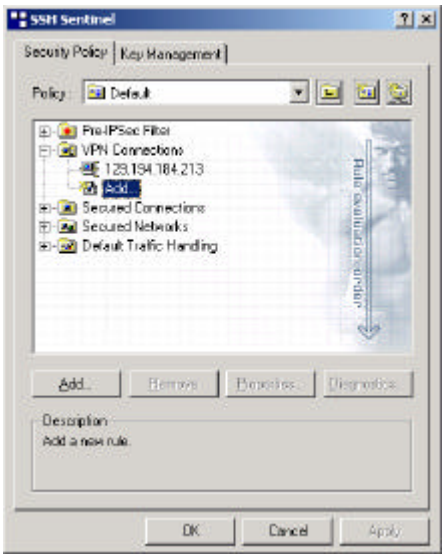
Configuration de SSH Sentinel :

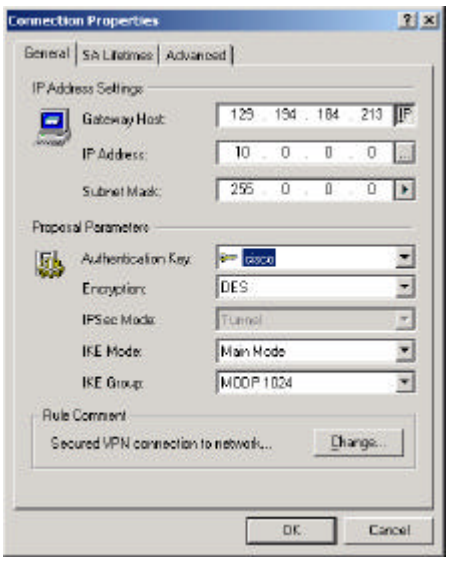
D'abord, il faut définir la clé du secret partagé. Pour cela, cliquez sur add du menu Authentication Key et choisir Create new pre-shared key. Il est important que le secret soit le même entre le routeur Cisco et SSH Sentinel.




Il est possible de modifier le secret après avoir créé l'objet.




Dans cette configuration, nous avons une connexion en mode tunnel.

L'échange a la même structure que l'échange vu au paragraphe 3.1.1.
On notera quand même une différence dans le premier paquet. Cette différence vient du fait que SSH Sentinel fait plusieurs propositions pour la connection.

Capture : (annexe 6)

```
Src MAC Addr    Dst MAC Addr    Protocol    Src Other Addr     Dst Other Addr     Type Other Addr
SSH_Sentinel     Routeur             ISAKMP   129.194.184.90    129.194.184.213    IP

+ UDP: Src Port: ISAKMP, (500); Dst Port: ISAKMP (500); Length = 1332 (0x534)
 ISAKMP: Major Version: 1  Minor Version: 0  Length: 1324
    ISAKMP: Initiator cookie = 5C F5 BE FA CA 00 00 00
    ISAKMP: Responder cookie = 00 00 00 00 00 00 00 00
    ISAKMP: Next payload = Security Association
    ISAKMP: Exchange type = Identity Protection
  + ISAKMP: Flags summary = 0 (0x0)
    ISAKMP: Message ID = 0 (0x0)
    ISAKMP: Length = 1324 (0x52C)
    ISAKMP: Payload type = Security Association          // on constate que le payload est plus
       ISAKMP: Next payload = Vendor ID                 // grand que lors de l'échange R to R
       ISAKMP: Reserved = 0 (0x0)                       // car Sentinel fait plusieurs propositions
       ISAKMP: Payload length = 1276 (0x4FC)
       ISAKMP: DOI = 1 (0x1)
       ISAKMP: Situation = SIT_INDENTITY_ONLY
       ISAKMP: Labeled domain identifier = 1264 (0x4F0)
       ISAKMP: Secrecy length = 1 (0x1)
       ISAKMP: ERROR: Reserved = 2078 (0x81E), it should = 0 (0x0)
       ISAKMP: Secrecy level = 5C F5 BE FA CA 00 00 00 03 00 00 28 00 01 00 00 80 01 00 01...
       ISAKMP: Secrecy length(in bits) = 32779 (0x800B)
       ISAKMP: ERROR: Reserved = 1 (0x1), it should = 0 (0x0)
       ISAKMP: Secrecy catagory = 80 0C 00 78 03 00 00 28 06 01 00 00 80 01 00 06 80 02 00...
       ISAKMP: Integrity length = 32779 (0x800B)
       ISAKMP: ERROR: Reserved = 1 (0x1), it should = 0 (0x0)
       ISAKMP: Integrity level = 80 0C 00 78 03 00 00 2C 0F 01 00 00 80 01 00 03 80 0E 00...
```

Suite à d'autres tests, il m'a été possible de configurer le routeur sans les dynamic map.
Pour cela, j'ai installé le pilot de Sentinel 1.2.0.15. Il reste néanmoins un problème avec le champ Vendor ID et la connexion ne s'effectue que si c'est SSH-Sentinel qui lance la connexion.

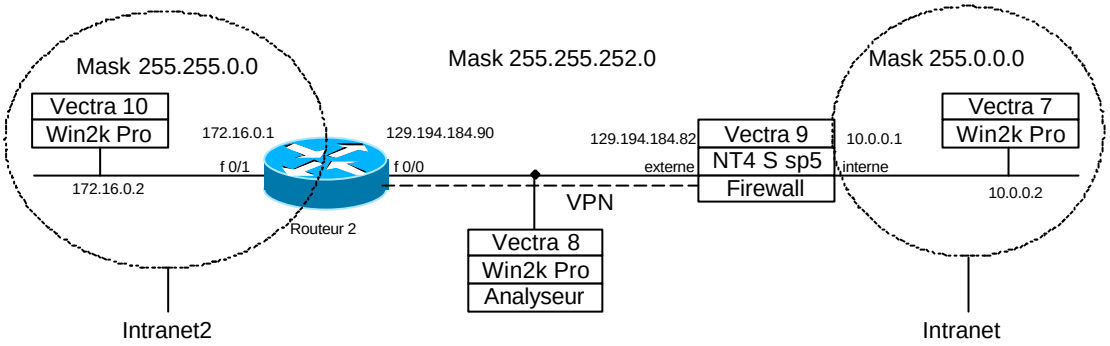
On obtient pour la configuration du routeur :

```
crypto isakmp policy 1
 encryption des
 hash sha
 authentication pre-share
 group 2
crypto isakmp key 1234567890 address 129.194.184.90
//secret partagé
crypto ipsec transform-set myset esp-des esp-sha-hmac

crypto map 10 ipsec-isakmp
match address 101
set transform-set myset
set peer 129.194.184.90 //addresse du partenaire

interface FastEthernet 0/0
crypto map mymap
ip address 129.194.184.213 255.255.252.0

interface FastEthernet 0/1
 no shutdown
ip address 172.16.0.1 255.255.0.0
```

```
ip route 0.0.0.0 0.0.0.0 129.194.184.90

access-list 101 permit ip 172.16.0.0     0.0.255.255
                          129.194.184.90 0.0.0.0
//autorise les transactions entre le sous-réseau 172.16.X.X et SSH Sentinel
```

### 3.1.4 Connexion Routeur to Firewall

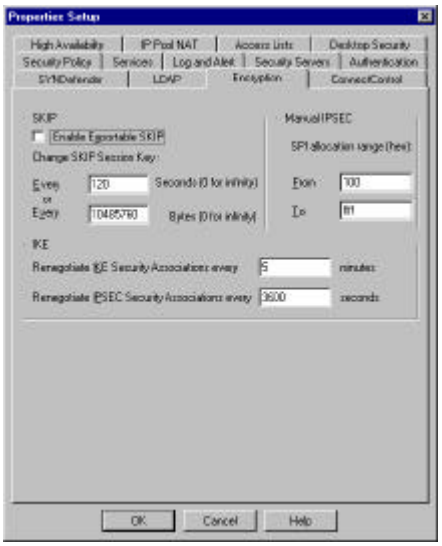
Cette configuration a pour but de tester la compatibilité entre les différents équipements. Pour ce faire, nous allons utiliser la configuration suivante :




Au niveau de Cisco, la configuration n'a rien de particulier. Elle est pratiquement identique à la configuration du paragraphe 3.1.1 Routeur to Routeur.
Pour cette raison, cette partie n'est pas détaillée. La configuration se trouve à l'annexe 7.

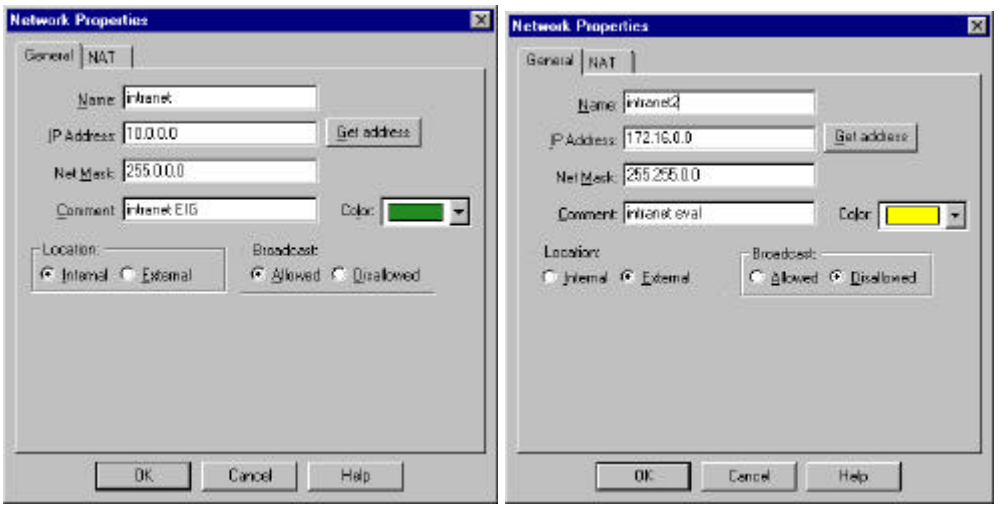
Sur le firewall Checkpoint, il va falloir définir quatre objets :
Deux du type Network pour intranet et intranet2
Le firewall
Le routeur Cisco.

Dans un premier temps, il faut définire les valeurs générales pour une connexion chiffrée. policy -> properties



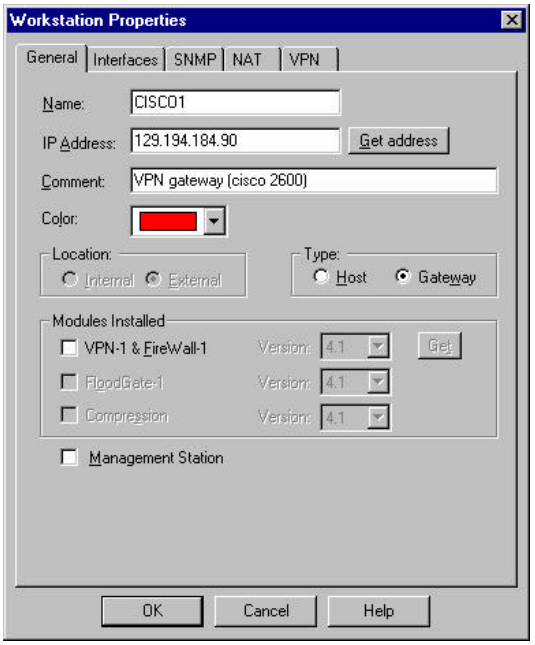

Les champs IKE permettent de définire la durée de vie des clés ainsi que la durée de vie de la connexion IPSec

Création des deux objets Network :
Depuis Manage -> Network Object… , choisir New -> Network.
Ces deux objets représentent les deux réseaux du shéma ci-dessus. Il convient de les compléter de la façon suivante :



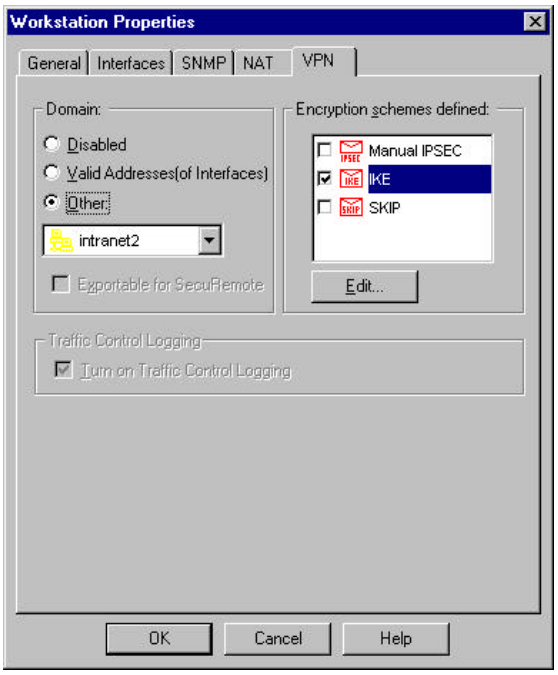

La fonction NAT ne nous intéresse pas dans ce cas de figure.

Création du routeur.
Depuis Manage -> Network Object… , choisir New -> WorkStation



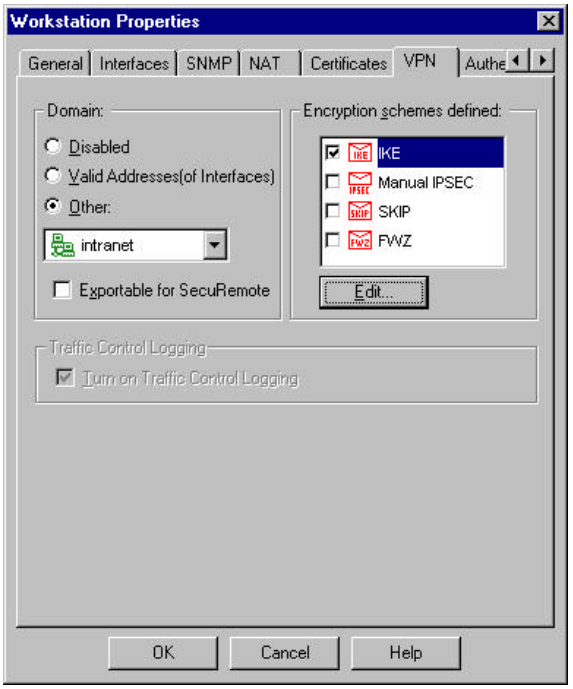

Il est important de cocher la case Getway.

En allant sur l'onglet VPN, nous allons pouvoir définir quel type de protocole va être utilisé pour les connexions sécurisées.



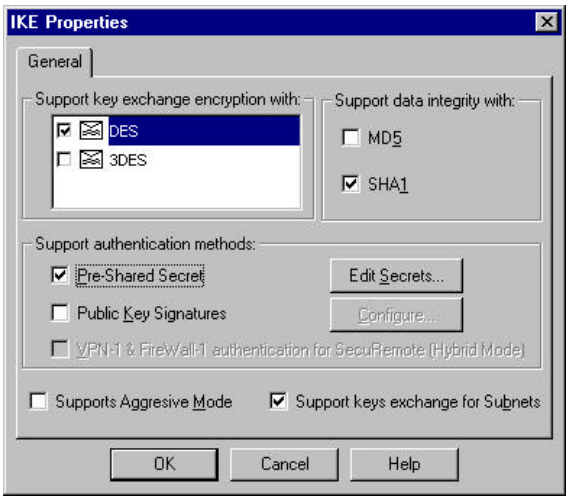

Le champ domaine permet d'indiquer à quel domaine le routeur est rattaché.
En cliquant sur edit, on voit qu'il est possible de choisir le type de codage ainsi que le type de hash qui seront utilisés lors des connexion IPSec.
Il faut cocher la case Pre-Shared Secret et éditer celui-ci.

Normalement, l'objet firewall existe déjà. Pour ce qui est de la création de l'objet firewall, se reporter au document Projet VPN de F. Truphème.

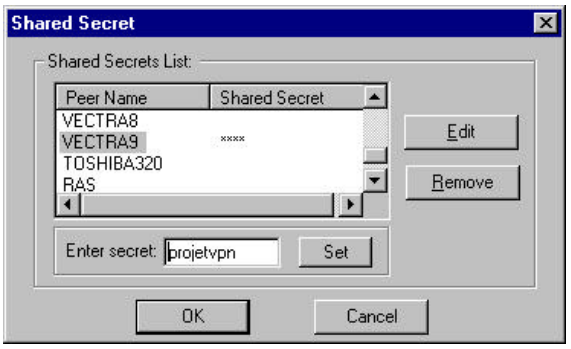



En sélectionnant l'objet firewall, depuis l'oglet VPN, il faut définir dans quel domaine se trouve le firewall et édité le shéma de chiffrement IKE.

La configuration IKE est identique tant pour la Cisco que pour le firewall.



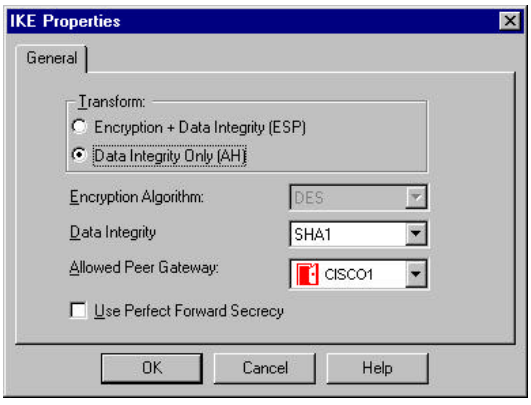

Dans cette fenêtre, nous pouvons définire le type de chiffrement utilisé et le type de hash.




En cliquant sur Edit Secret, on obtient une fentêtre qui contient tous les partenaires possible. Il suffit donc de sélectionner le firewall et d'entrer le sercet partagé.

Une fois les objets créés, il faut définir les règles du firewall.


Pour le mode Encrypt, il faut éditer les règles pour le protocole IKE en double cliquant sur Encrypt. Choisir IKE.




Ici, on peut définire si on utilise AH ou ESP + AH.
Il faut définir quel objet sera le partenaire pour la connexion.

Il est important de rajouter une route dans le firewall :
Route add 172.16.0.0 mask 255.255.252.0 129.194.184.213
Avec 176.16.0.0 le sous réseau à atteindre et 129.194.184.213 l'adresse du routeur pour atteindre ce sous réseau.
Cette règle est nécessaire car le routeur exécute dans l'ordre :

Routage
Chiffrement
NAT

Imaginons que Vectra 10 ping Vectra 7.
On pose : A = adresse de Vectra 10 172.16.0.2
B = adresse de Vectra 7 10.0.0.2
C = adresse du routeur 129.194.184.90
D = adresse du firewall 129.194.184.82
Avant de passer le routeur :
Src : A Dest : B
Entre le routeur et le firewall :
Src : C Dest : D iner src : A iner dest : B
Derrière le firewall :
Src : A Dest : B
Retour du ping :
Src : B Dest : A
Entre le routeur et le firewall :
Src : D Dest : C iner src : B iner dest : A
Ici on peut penser qu'il n'y a aucun problème, mais étant donné que le routage intervient avant le chiffrement, l'adresse qui sera utilisée sera B. Donc le paquet sera envoyé au routeur par défaut. Ça ne pose pas forcément un problème, mais étant donné que je me trouvait en îlot, le routeur par défaut ne pouvait être atteint.

Problèmes de configuration pouvant intervenir :
NAT et l'option *Support keys exchange for Subnets*
Généralement, on utilise la fonction NAT avec les firewalls car les sous-réseaux derrière utilise un adressage privé afin d'obtenir une meilleur securité.
L'option *Support keys exchange for Subnets* permet, lors d'une connexion VPN, de ne pas utiliser le NAT uniquement lorsque les paquets transitent d'un coté à l'autre du VPN. Si l'option n'est pas activée, le firewall utilisera la fonction NAT et il faudra modifier les access-list du routeur.

Cas 1 : Pas de NAT
Si le NAT n'est pas utilisé, il n'est pas utile d'activé *Support keys exchange for Subnets*. L'access-list du routeur est :

```
access-list 101 permit ip 172.16.0.0 0.0.0.255 10.0.0.0 0.0.0.255
```

Cas 2 : NAT et Support keys exchange for Subnets activé
Pour le routeur, on obtient la même règle que le cas 1. Mais on a un échange de clé supplémentaire.

Cas 3 : NAT activé *Support keys exchange for Subnets* désactivé
Cette configuration est aussi possible cependant, il faut modifié l'access-list :

```
access-list 101 permit ip 172.16.0.0 0.0.0.255 10.0.0.0 0.0.0.255
access-list 101 permit ip 172.16.0.0 0.0.0.255 129.194.184.82 0.0.0.0
```

Remarque: la connexion ne s'effectue que si c'est le firewall qui lance la connexion. Malheureusement, le *debug* du routeur ainsi que les logs du firewall ne m'ont pas permis de découvrir l'erreur.

Analyse de protocole :

| Frame | Time | Src MAC Addr | Dst MAC Addr | Protocol | Src Other Addr | Dst Other Addr |
|---|---|---|---|---|---|---|
| 1 | 5.788323 | Routeur2 | Firewall | ESP | 129.194.184.90 | 129.194.184.82 |
| 2 | 5.998626 | Routeur2 | Firewall | ARP_RARP | | |
| 3 | 5.998626 | Firewall | Routeur2 | ISAKMP | 129.194.184.82 | 129.194.184.90 |
| 4 | 6.128813 | Routeur2 | Firewall | ISAKMP | 129.194.184.90 | 129.194.184.82 |
| 5 | 6.138827 | Firewall | Routeur2 | ISAKMP | 129.194.184.82 | 129.194.184.90 |
| 6 | 6.309072 | Routeur2 | Firewall | ISAKMP | 129.194.184.90 | 129.194.184.82 |
| 7 | 6.319087 | Firewall | Routeur2 | ISAKMP | 129.194.184.82 | 129.194.184.90 |
| 8 | 6.329101 | Routeur2 | Firewall | ISAKMP | 129.194.184.90 | 129.194.184.82 |
| 9 | 6.349130 | Firewall | Routeur2 | ISAKMP | 129.194.184.82 | 129.194.184.90 |
| 10 | 12.247611 | Routeur2 | Firewall | ESP | 129.194.184.90 | 129.194.184.82 |
| 11 | 12.367784 | Firewall | Routeur2 | ISAKMP | 129.194.184.82 | 129.194.184.90 |
| 12 | 19.508051 | Routeur2 | Firewall | ISAKMP | 129.194.184.90 | 129.194.184.82 |
| 13 | 19.538095 | Firewall | Routeur2 | ISAKMP | 129.194.184.82 | 129.194.184.90 |
| 14 | 19.558123 | Routeur2 | Firewall | ISAKMP | 129.194.184.90 | 129.194.184.82 |
| 15 | 24.264891 | Routeur2 | Firewall | ESP | 129.194.184.90 | 129.194.184.82 |
| 16 | 24.385064 | Firewall | Routeur2 | ISAKMP | 129.194.184.82 | 129.194.184.90 |
| 17 | 24.645439 | Routeur2 | Firewall | ISAKMP | 129.194.184.90 | 129.194.184.82 |
| 18 | 24.645439 | Firewall | Routeur2 | ISAKMP | 129.194.184.82 | 129.194.184.90 |
| 19 | 24.795655 | Firewall | Routeur2 | ISAKMP | 129.194.184.82 | 129.194.184.90 |
| 20 | 24.795655 | Routeur2 | Firewall | ISAKMP | 129.194.184.90 | 129.194.184.82 |
| 21 | 24.895799 | Firewall | Routeur2 | ISAKMP | 129.194.184.82 | 129.194.184.90 |
| 22 | 24.895799 | Routeur2 | Firewall | ISAKMP | 129.194.184.90 | 129.194.184.82 |
| 23 | 31.345072 | Routeur2 | Firewall | ESP | 129.194.184.90 | 129.194.184.82 |
| 24 | 31.345072 | Firewall | Routeur2 | ESP | 129.194.184.82 | 129.194.184.90 |
| 25 | 32.346512 | Routeur2 | Firewall | ESP | 129.194.184.90 | 129.194.184.82 |
| 26 | 32.346512 | Firewall | Routeur2 | ESP | 129.194.184.82 | 129.194.184.90 |

Dans ce cas, *Support keys exchange for Subnets* a été activé, c'est pourquoi on a deux négociation.
On constate que dans cet échange, le routeur et le firewall n'arrivent pas à se mettre d'accord tout de suite.
La phase 1 de ISAKMP fonctionne correctement, mais la phase 2 pose problème.
Le mode debug du routeur nous permet de confirmer qu'il y a bien une erreur. Il m'a été possible de définire avec exactitude la source de l'erreur. L'utilisation du champ SPI par le firewall n'est pas correcte. Prenons le cas de deux connexions consécutive : La première est détruite (time out) et on lance la deuxième, l'échange est entièrement refait (phase 1 et 2). Cependant, lors de la deuxième connexion, le champ SPI utilisé par le firewall et faux. Il utilise la valeurs de la première connexion. C'est pourquoi on a une resynchronisation (la phase 2 est refaite).

Du coté du firewall, on peut voir dans le fichier log :

| Date | Time | Origin | Type | Action | Source | Destinati.. | Proto. | Rule | SrcKeyID | DstKeyID |
|---|---|---|---|---|---|---|---|---|---|---|
| 23Oct2001 | 11:29:12 | VECTRA9 | log | key install | CISCO1 | VECTRA9 | | | | |
| 23Oct2001 | 11:29:12 | VECTRA9 | log | key install | CISCO1 | VECTRA9 | ip | 0 | 0x0724107e | 0xee773f90 |
| 23Oct2001 | 11:29:18 | VECTRA9 | log | key install | VECTRA9 | CISCO1 | ip | 0 | 0xee773f91 | 0x09540a04 |
| 23Oct2001 | 11:29:18 | VECTRA9 | log | decrypt | 172.16.0.2 | Vectra7 | icmp | 3 | 0x09540a04 | |
| 23Oct2001 | 11:29:19 | VECTRA9 | log | encrypt | Vectra7 | 172.16.0.2 | icmp | 2 | | 0x09540a04 |

| Product | Info. |
|---|---|
| VPN-1 & FireWall-1 Module | IKE Log: Phase 1 completion: DES/SHA1/Pre shared secrets Negotiation Id: 2a5e2284edac7b23-fae51b3e20b8547b |
| VPN-1 & FireWall-1 Module | scheme: IKE methods: Combined ESP: DES + SHA1 (phase 2 completion) for subnet: 172.16.0.0 (mask= 255.255.255.0) and for subnet: 10.0.0.0 (mask= 255.255.255.0) |
| VPN-1 & FireWall-1 Module | scheme: IKE methods: Combined ESP: DES + SHA1 (phase 2 completion) for subnet: 10.0.0.0 (mask= 255.0.0.0) and for subnet: 172.16.0.0 (mask= 255.255.0.0) |
| VPN-1 & FireWall-1 Module | icmp-type 8 icmp-code 0 scheme: IKE methods: Combined ESP: DES + SHA1 |
| VPN-1 & FireWall-1 Module | icmp-type 0 icmp-code 0 scheme: IKE methods: Combined ESP: DES + SHA1 |

Les trois premières lignes correspondent à l'établissement des clés.
Les deux derniers sont le passage de ping.

Une autre configuration a été testée : établissement avec AH uniquement.
Pour cette configuration, on rencontre le même problème lors de la phase 2 de l'établissement des clés. (voir annexe 8)
Avec l'option AH, il est possible de voir l'imbrication des protocole.

Format des paquets :

| IP header | AH | inner IP header | TCP header | Payload |
|---|---|---|---|---|

Dans le paquet ci-dessous : Vectra10 de intranet2 ping Vectra7 de intranet

**IP**: ID = 0x14; Proto = 0x33; Len: 104 //IP header
IP: Version = 4 (0x4)
IP: Header Length = 20 (0x14)
IP: Precedence = Routine
IP: Type of Service = Normal Service
IP: Total Length = 104 (0x68)
IP: Identification = 20 (0x14)
+ IP: Flags Summary = 0 (0x0)
IP: Fragment Offset = 0 (0x0) bytes
IP: Time to Live = 255 (0xFF)
IP: Protocol = 0x33
IP: Checksum = 0x471D
IP: Source Address = **129.194.184.90** //Adresses IP du routeur et du Firewall
IP: Destination Address = **129.194.184.82**
IP: Data: Number of data bytes remaining = 84 (0x0054)
**AH**: Protocol = IPv4, SPI = 0xE2640934, Seq = 0x1 //
AH: Next Header = IPv4 - Internet Protocol version 4
AH: Payload Length = 6 (0x6)
AH: Reserved = 0 (0x0)
AH: Security Parameters Index = 3798206772 (0xE2640934)
AH: Sequence Number = 1 (0x1)
AH: Authentication Data: Number of data bytes remaining = 12 (0x000C)
**IP**: ID = 0x1967; Proto = ICMP; Len: 60
IP: Version = 4 (0x4)
IP: Header Length = 20 (0x14)
IP: Precedence = Routine

```
    IP: Type of Service = Normal Service
    IP: Total Length = 60 (0x3C)
    IP: Identification = 6503 (0x1967)
  + IP: Flags Summary = 0 (0x0)
    IP: Fragment Offset = 0 (0x0) bytes
    IP: Time to Live = 127 (0x7F)
    IP: Protocol = ICMP - Internet Control Message
    IP: Checksum = 0x6C46
    IP: Source Address = 172.16.0.2
    IP: Destination Address = 10.0.0.2
    IP: Data: Number of data bytes remaining = 40 (0x0028)
+ ICMP: Echo: From 172.16.00.02 To 10.00.00.02
```

### 3.2 Certificat

L'utilisation de Certificat, permet d'utiliser des clés asymétriques. Les clés permettent de ne pas utiliser de secret partagé pour l'authentfication.
Le certificat permet d'associer une clé publique à une entité (personne, machine). Voir mémoire de D. Cotte

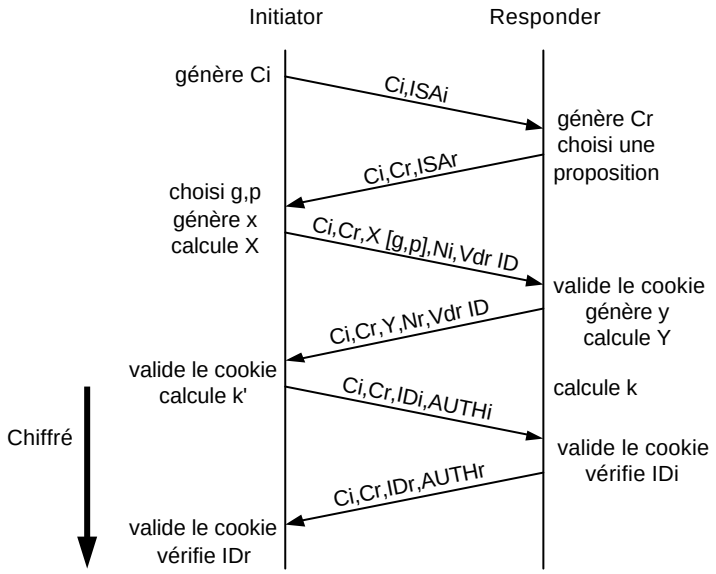
Cisco a développé un protocole (SCEP Simple Certificate Enrollement Protocol) permettant de transmettre un certificat ainsi que faire des enregistrements *on ligne*.

Les requêtes SCEP ont la forme :
**GET** *chemin_du_CGI* ?**Operation=** *opération_à_effectuer* **&message** *message_à_transmettre*
Le chemin du CGI est l'emplacement de pkiclient.exe sur le serveur
Les opérations possibles sont :

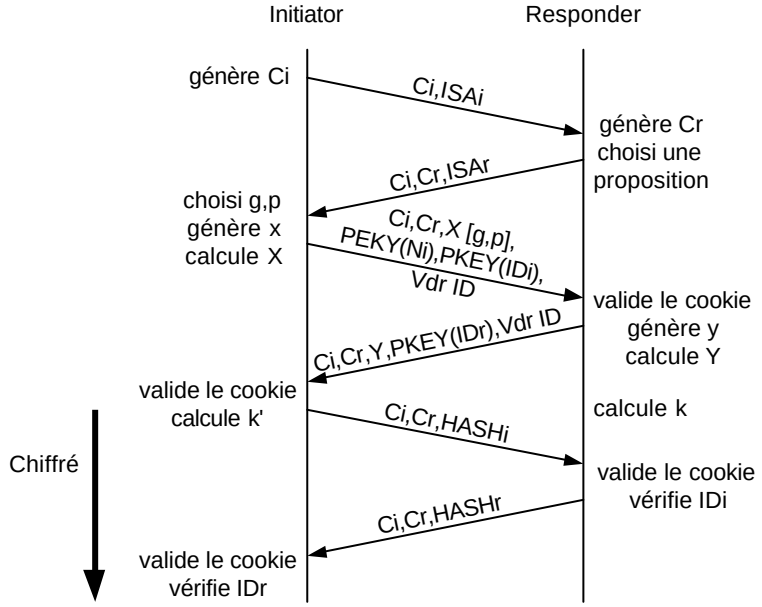
- *GetCACert* pour obtenir le certificat du CA. Le message contient le nom de la CA qui a publié le certificat.
- *GetCACertChain* pour obtenir la chaîne de certification. Le message contient le nom de la CA qui a publié le certificat.
- *PKIOperation* utilisé pour les requêtes de certificat (enregistrement) ou de liste de révocation. Le message contient des messages PKI (PKCS#10, PKCS#7,…) au format base 64

Lorsque l'on utilise des certificats pour l'authentification, l'échange ISAKMP est légèrement modifié. On obtient :

1. Si l'authentification est faite avec les signatures :




AUTHi,r sont les messages contenants les signatures
2. Si l'authentification est faite par chiffrement :




avec PKEY() le chiffrement avec la clé privée.

Commande relative au certificat dans un routeur Cisco :

L'utilisation d'une CA implique l'utilisation d'url. C'est pourquoi, il faut définir un DNS. Si la CA n'est pas dans le DNS, il est possible d'associer manuellement un nom à une adresse IP.

*ip name-server adresse_IP_du_DNS*

*ip host nom adresse_IP*

*ip domain-name nom_du_domaine*
nom du domaine dans lequel le routeur se trouve.

*crypto isakmp policy 1*
*authentication rsa-sig*
utilise les signatures pour s'authentifier
*authentication rsa-encr*
chiffre les nonces pour s'authentifier
Cette méthode est très intéressante car elle permet de faire un authentification tout en chiffrant une valeur utilisée pour la génération des clés.

*crypto ca identity nom_du_certificat_de_la_CA*
Cette commande définit une nouvelle CA. Il est important de spécifier le nom du certificat de la CA. Si ce nom n'est pas spécifié correctement, les requêtes SCEP n'aboutiront pas.

enrollment url http://*adresse_du_CA*
adresse url de la CA

crl optional
la liste de révocation sera utilisée, mais si elle n'existe pas, le routeur continuera les authentifications.

*crypto key generate rsa*
génère la paire de clés du routeur (de 360 à 2048 bits)

*crypto ca authenticate nom_du_certificat_de_la_CA*
demande le certificat du CA (utilisation de SCEP)
Suite à cette commande, le routeur affiche le fingerprint du certificat reçu. Il est important de vérifier s'il correspond avec le fingerprint du certificat du CA.

*crypto ca enroll nom_du_certificat_de_la_CA*
Enregistrement au près du CA en utilisant les clés générées.
lorsque cette commande est entrée, plusieurs questions sont posées

*Password:* // le password est utilisé pour la CRL
*Re-enter password:*

*% The subject name in the certificate will be: Routeur2.td.unige.ch*
// Nom du certificat
*% Include the router serial number in the subject name? [yes/no]:* y
*% The serial number in the certificate will be: ED29C094*
// insertion du N° de série dans le certificat
*% Include an IP address in the subject name? [yes/no]:* y
*Interface:* fastethernet0/0
// insertion de l'addresse IP. L'interface doit être spécifiée.
*Request certificate from CA? [yes/no]:* y
*% Certificate request sent to Certificate Authority*

Ensuite, quand le certificat est validé, il revient et le routeur affiche le fingerprint. Il est important de le comparer avec celui sur la CA afin de vérifier qu'il n'y aie pas d'erreurs.

commande de *debug* permettant de voir les échanges relatifs au certificat.
debug crypto pki message
debug crypto pki transaction

La commande *Show run* permet de voir les certificats présents dans le routeur.

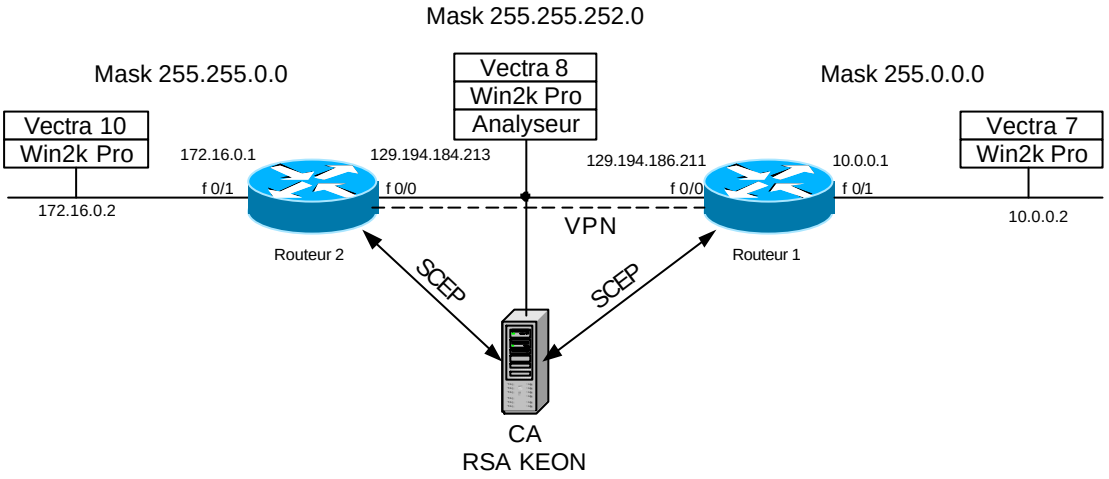
Le clock est important. Après un boot, le routeur est à l'heure : 0 :0 :0 lundi 1 mars 1993 !
Ceci a pour effet que les certificats seront invalides car la date d'émission précède l'heure courante du routeur.

Dans un premier temps, il faut que le routeur obtienne un certificat signé par une CA ainsi que son certificat.
Nous voulions utiliser la CA de Microsoft.
Par défaut, celle-ci n'implémente pas le protocole SCEP. Il a donc fallut l'installer avec le fichier cepsetup.exe fournit par Microsoft. Celui-ci installe le fichier mscep.dll. Le routeur doit pour utiliser SCEP, envoyer les requêtes à cette dll.
Par défaut lors de l'installation, le chemin pour y excéder est :
http://*nom_de_la_CA*/certsrv/mscep/mscep.dll.
Malheureusement, la CA ne répond pas correctement au requêtes du routeur. Suite à ceci, j'ai pu constater, grâce à des recherches sur des forums, que les versions antérieur à 5.131.2195.1 contiennent des bugs. Malgré une mise à jour, la CA ne fonctionne pas.

Suite à ces problèmes, j'ai utilisé la CA RSA KEON. Celle-ci fonctionne correctement et m'a permis d'obtenir un certificat avec le protocole SCEP.
Remarque : par défaut, lorsqu'on veut se connecter avec le protocole SCEP, le client recherche le fichier pkiclient.exe sur le serveur. C'est ce fichier qui permet l'intégration de SCEP.

### 3.2.1 Connexion Routeur to Routeur

Faire fonctionner les routeurs avec des certificats comme méthode d'authentification n'est pas très compliqué une fois que les problèmes liés au CA sont réglés.




Configuration du routeur :

```
crypto isakmp policy 1
 authentication rsa-sig

crypto ipsec transform-set myset esp-des esp-sha-hmac

crypto map test 10 ipsec-isakmp
 set peer 129.194.186.211
 set transform-set myset
 match address 101

ip host ca.cotte 129.194.187.55

crypto ca identity rootscep
 enrollment url http://ca.cotte:446  //le port 446 est utilisé par RSA Keon
                                       pour le protocol SCEP
 crl optional

crypto key gen rsa

crypto ca authenticate rootscep

crypto ca enroll rootscep
```

Configuration complète : voir annexe 9

Analyse : Paquet 3

```
Src MAC Addr    Dst MAC Addr    Protocol      Src Other Addr      Dst Other Addr     Type Other Addr
Routeur 2          Routeur 1          ISAKMP      129.194.184.213    129.194.186.211    IP

+ UDP: Src Port: ISAKMP, (500); Dst Port: ISAKMP (500); Length = 318 (0x13E)
  ISAKMP: Major Version: 1  Minor Version: 0  Length: 310
     ISAKMP: Initiator cookie = 27 CA 46 CD E8 55 EC F6
     ISAKMP: Responder cookie = 46 95 41 C9 51 A1 E1 D2
     ISAKMP: Next payload = Key Exchange
     ISAKMP: Major version = 1 (0x1)
     ISAKMP: Minor version = 0 (0x0)
     ISAKMP: Exchange type = Identity Protection
    + ISAKMP: Flags summary = 0 (0x0)
     ISAKMP: Message ID = 0 (0x0)
     ISAKMP: Length = 310 (0x136)
     ISAKMP: Payload type = Key Exchange
        ISAKMP: Next payload = Nonce
        ISAKMP: Reserved = 0 (0x0)
        ISAKMP: Payload length = 100 (0x64)
        ISAKMP: Key exchange data = 6A 67 21 EA 5D 51 6D E1 52 7D 63 76 F8 A1 77 0B 44 9A...
     ISAKMP: Payload type = Nonce
        ISAKMP: Next payload = Certificate Request
        ISAKMP: Reserved = 0 (0x0)
        ISAKMP: Payload length = 24 (0x18)
        ISAKMP: Nonce data = 68 B7 CE 72 6B DD DA 00 11 29 5F FA AA 60 DF E1 8B 1F DC B4
     ISAKMP: Payload type = Certificate Request
        ISAKMP: Next payload = Vendor ID
        ISAKMP: Reserved = 0 (0x0)
        ISAKMP: Payload length = 138 (0x8A)
        ISAKMP: Certificate type = X.509 Certificate - Signature
        ISAKMP: Certificate authority = 30 81 82 31 0B 30 09 06 03 55 04 06 13 02 43 48 31...
     ISAKMP: Payload type = Vendor ID
        ISAKMP: Next payload = None
        ISAKMP: Reserved = 0 (0x0)
        ISAKMP: Payload length = 20 (0x14)
        ISAKMP: Vendor ID = D2 0D E1 D0 E8 54 EC F6 53 2F 2F 89 53 F0 E1 32
```

On voit dans ce paquet le champ *Certificate Request* ainsi que la signature l'authentification.

On constatera aussi que la taille des paquets suivants est bien plus grande que lors d'un échange avec secret partagé.

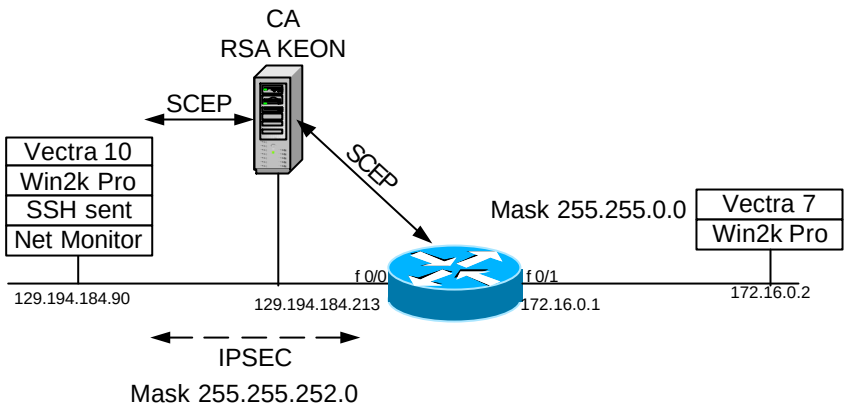
On peut aussi constater une différence entre le mode *rsa-sig* et *rsa-encr*
Avec *rsa*-sig, seul le champ est en fait la signature.
Avec rsa-encr, le champ nonce est chiffré. On peut le constater par sa taille qui passe de 192 bits à 1024 bits (taille des clés).

Remarque : la connexion avec des certificats, prend plus de temps. C'est du au fait que le chiffrement et le déchiffrement, avec des clés symétriques, est bien plus long qu'avec une clé symétrique. La taille des paquets joue aussi un rôle. Etant donné que les clés font 1024 bits, chaque champ chiffré a une taille de 1024 bits.

Détail de la connexion à l'annexe 11 pour rsa-sig et 12 pour rsa-encr

Dans l'annexe 10, on a la capture de la demande de certificat du routeur avec le protocole SCEP.

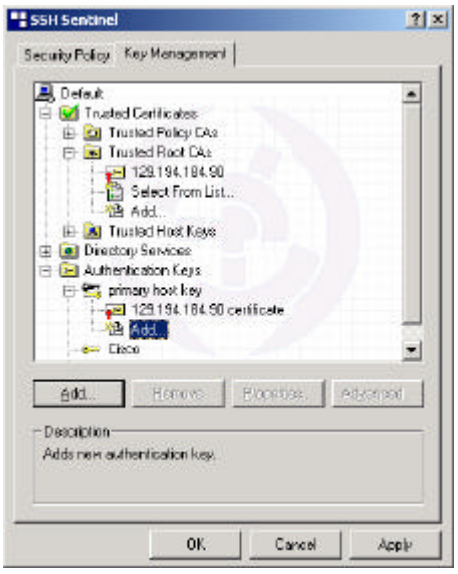
### 3.2.2 Connexion Host to Routeur

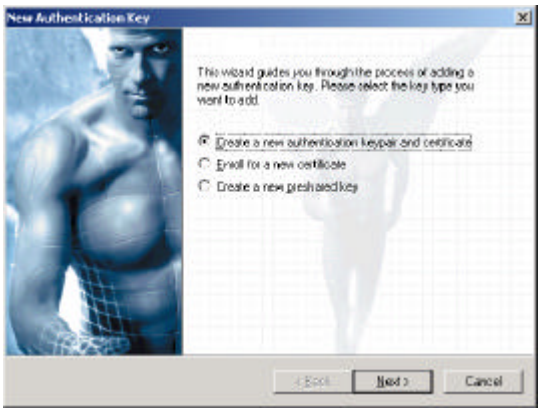


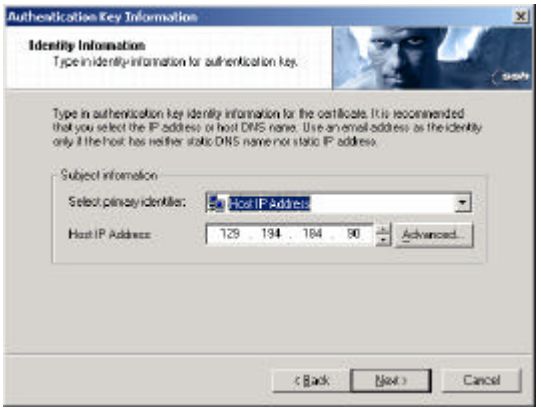

Sur le PC Host, j'ai utilisé SSH Sentinel 1.1 beta trois avec Pilot 1.2.0.15
La difficulté a été d'obtenir le certificat root de la CA par le protocole SCEP.
En effet, Sentinel ne reconnaissait pas le format du certificat. Plusieurs essais on été fait en utilisant tous les formats de certificat reconnu par Sentinel en vain.
Le problème a été exposé au support technique de SSH Sentinel qui nous a répondu que le programme est très restrictif au niveau des caractères dans les champs des certificats et que ce défaut serait corrigé dans la version 1.3.
Un nouveau certificat a donc été créé en n'utilisant que les caractères ''imprimable''.
Suite à cette modification, Sentinel a pu obtenir le certificat root du CA.
Cependant, la demande d'enregistrement ne s'effectue pas. En cherchant dans les FAQ de SSH Sentinel, il est expliqué que des améliorations seront apportées dans les prochaines versions.

Demmande du certificat root :
Cette procédure devrait aussi permettre l'enregistrement du certificat.

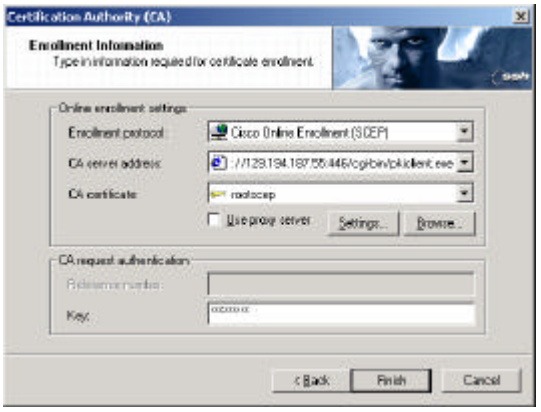
Dans SSH Sentinel, cliquez sur add. Comme nous voulons créé un certificat et l'enregistrer, il faut selectionner : *Create a new keypair and certificate*



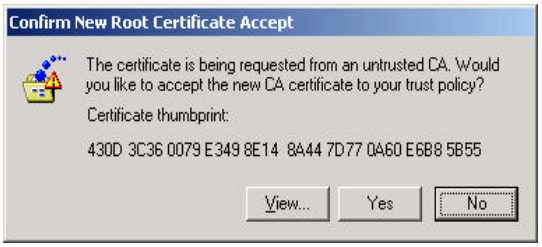




Ensuite, il nous est demandé de donner le nom qui sera utilisé pour le certificat.

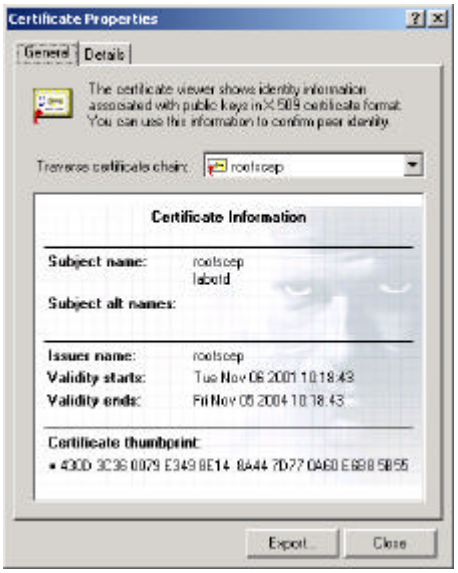



Cette fenêtre permet de définire comment le certificat sera délivré (SCEP, PKCS#10). Dans notre cas, nous utiliserons SCEP.
Dans le champ url, il faut donner l'emplacement du fichier pkiclient.exe qui permettra la connexion SCEP.
Il faut aussi définire le nom du certificat que l'on veut obtenir. Dans

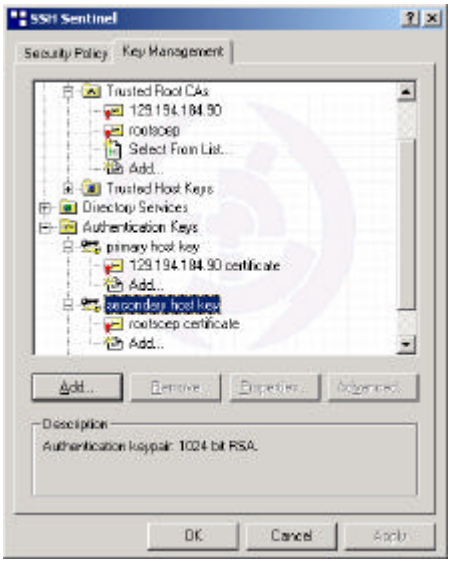
notre cas : rootscep. Très important, le champ key, il permet de définir un mot de passe qui sera utilisé pour la révocation éventuelle du certificat. Si le champ est vide, le certificat ne sera pas délivré.




Dans un premier temps, nous obtenons le certificat de la CA. Il est possible de le visualiser. Il est important de vérifier le champ thumbprint afin de s'assurer que le certificat n'a pas subit de modification.
Pour vérifier ce champ, il faut contacter l'administrateur de la CA.

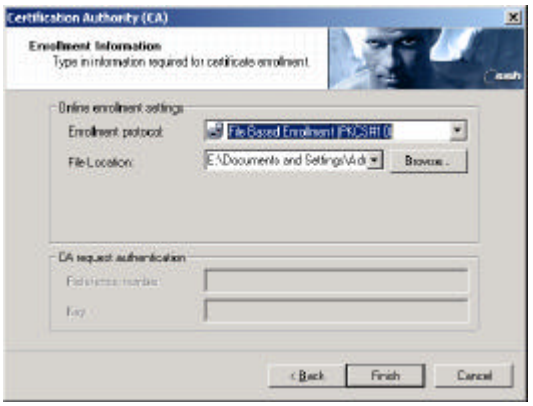



Visualisation du certificat Root de la CA




On voit que deux nouveaux certificats sont apparus. Un dans le champ *Trusted Root CAs* et un dans *Authentication Keys*.

Il est possible d'obtenir le certificat sans passer par SCEP. Pour cela, refaire comme précédement mais au lieu de choisir SCEP, il faut choisir File Based *Enrollment* (PKCS#10)



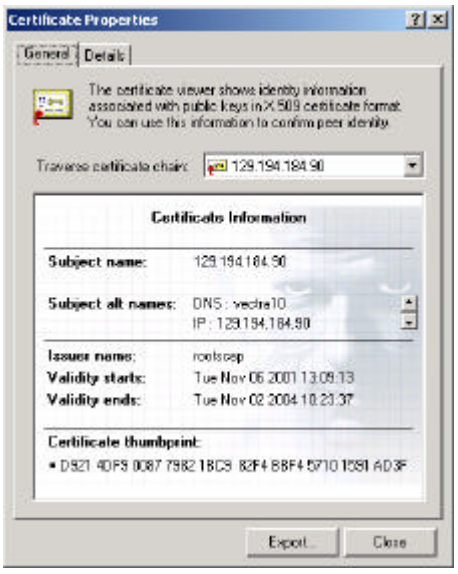

La requête de certificat sera enregistrée dans un ficher. En cliquant sur *browse* il est possible de définir le nom ainsi que l'emplacement.

On constatera que le certificat n'est pas validé. Il est en attente. Il faut pour pouvoir l'utiliser, qu'il soit valider par la CA.

Il faut donc se connecter au CA et demander un enregistrement par PKCS#10.
Il est necéssaire de choisir le même certificat root que celui du routeur.

Ouvrir le fichier PKCS#10 avec un éditeur de texte (notepad) et copier le contenu dans le champ prévu à cet effet sur la page du CA.

Lorsque ceci est fait, la CA doit valider la requête. Quand le certificat est validé, il faut le copier sur la machine ayant SSH Sentinel afin de l'importer.
Pour importer le certificat, il suffit de cliquer avec le bouton droit sur le champ ''Secondary host key'' et de choisir ''import new Auth.Key''. Là, il faut sélectionner le certificat copié du CA.



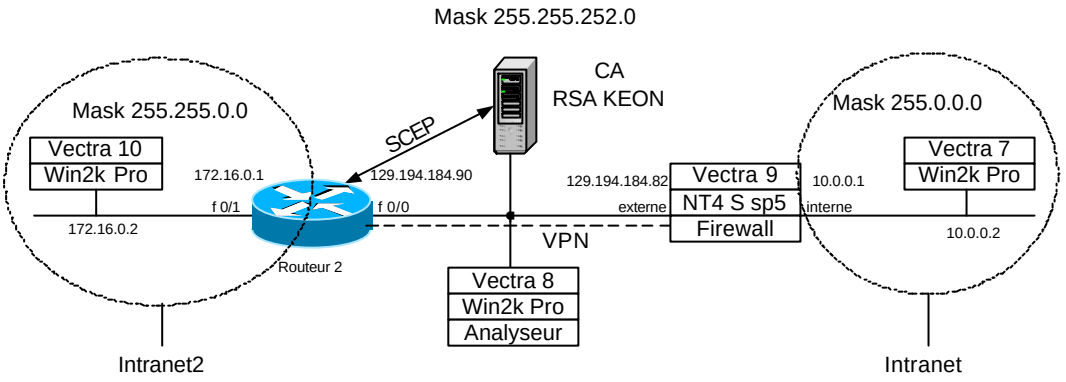

Lorsque tout ceci est fait, il est possible de visionner le certificat en double cliquant dessus.

Pour la connexion avec le routeur, le principe est le même que pour un secret partagé sauf que dans ce cas la clé est un certificat.

La configuration du routeur se trouve a l'annexe 13

Au niveau du principe des échanges, il n'y a pas de différence avec une connexion Routeur to Routeur. C'est pour cette raison qu'il n'y a pas d'explication.

### 3.2.3 Connexion Routeur to Firewall



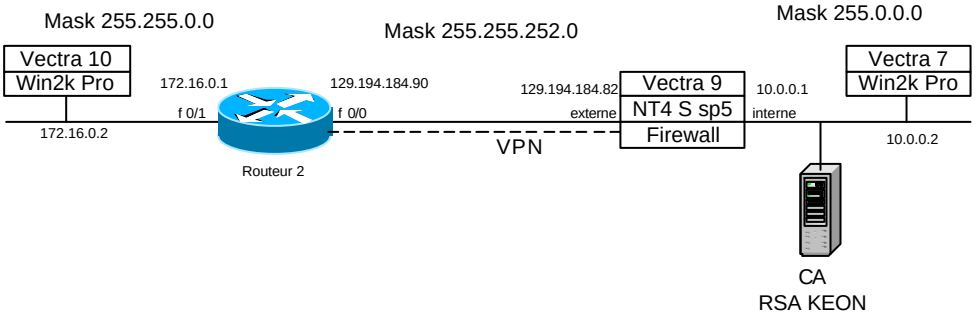

Le Firewall Checkpoint ne permet pas d'utiliser le protocole SCEP.

Il faut d'abord installer le certificat sur le firewall (voir Projet VPN par F. Trupème)

Il faut installer le certificat sur le routeur comme fait précédemment.
La configuration et les problèmes pouvant intervenir sont les même que pour un secret partagé. Pour la configuration, voir annexe 14
Il faut faire attention au heure à cause la date d'émission du certificat et l'heure courante du firewall qui peuvent être différentes.

**Remarque : Le firewall Checkpoint ne supporte que le mode rsa-enc.**

On peut imaginé que la CA Keon soit située derrière le firewall.

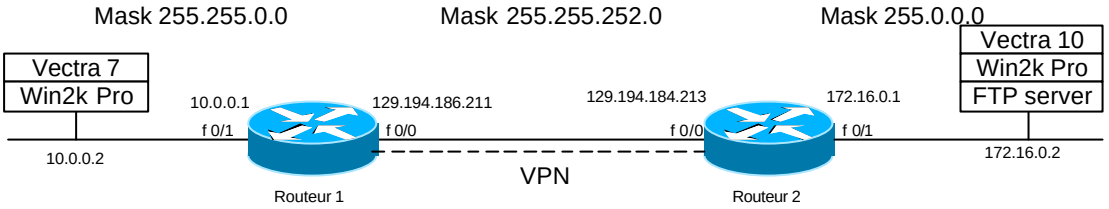


Ce cas s'approche plus de la réalité.
Si la CA est derrière le firewall, il faut inclure une règle permettant au routeur d'atteindre la CA sans demander une authentification. Dans notre cas :

Source : Routeur Destination : CA Keon port : udp 446 accept

## 4. Tests entre différents codages

Les tests on été effectués avec la configuration suivante :




Le serveur FTP utilisé est BulletProof v2.15.
Machine utilisée : HP Vectra pentium III 500 MHz 128 Mo ram
Les temps représentent un transfert FTP d'un fichier de 11 MB. La connexion IPSec est déjà établie.

| AH | ESP | HASH | Temps [s] | Débit moyen [KB/sec] |
|---|---|---|---|---|
| - | DES | - | 57.3 | 192 |
| - | 3DES | - | 119 | 93 |
| - | DES | MD5 | 67,4 | 163 |
| - | DES | SHA1 | 76,6 | 143 |
| - | 3DES | MD5 | 132.3 | 83 |
| - | 3DES | SHA1 | 139,5 | 79 |
| SHA1 | - | - | 82 | 135 |
| MD5 | - | - | 25 | 440 |
| SHA1 | DES | SHA1 | 138 | 80 |
| SHA1 | 3DES | SHA1 | 193 | 57 |

Temps de connexion : Avec secret partagé, il faut compter 1,6 seconde
Avec certificat, il faut compter 7,2 secondes
Les temps de connexion ne dépendent pas des types de codage choisi.
Avec cette mesure, on voit bien que le chiffrement et déchiffrement avec un système de clé publique/privée prend plus de temps.

Remarque : Il ne m'a pas été possible de faire le test sans IPSec, car le débit était de ~200 KB/sec. Ce débit est inférieur à certain chiffrement ce qui n'est pas normal. Je n'ai pas encore trouvé la source de ce problème. Suivant la cause, il faudra remettre en question les résultats trouvés.

## 5. Conclusion

IPSec est un protocole très intéressant. Cependant, il est très complexe et la mise en œuvre de connexion n'est pas toujours évidente.
Les routeurs Cisco ne permettent pas une installation de connexions IPSec sans savoir au préalable comment fonctionne le protocole. Leur configuration est relativement simple et varie très peu suivant le partenaire avec lequel il va établire la connexion IPSec.
Le mode debug du routeur est très intéressant. Il permet de voir beaucoup de chose.
SSH Sentinel est relativement simple à utiliser, mais la version beta 3 contient encore quelques bugs. Le *debug* est très complet et peut-être même trop, les niveau de détail sont soit trop élevés soit insuffisants.
Dans l'ensemble la norme IPSec est bien définie ce qui permet une bonne interopérabilité. Cependant, on voit que certaines implémentations sont plus souples que d'autre (connexion dans un sens mais pas dans l'autre).
Il n'est pas toujours facile de vouloir debuger une connexion dû fait que les problème interviennent dans la partie chiffrée de l'échange.

## 6. Remerciements :

A Eric Jenny et Gérald Litzistorf pour leurs rigueurs et l'intérêt à ce travail.

A Daniel Stocco pour sont aide et le temps qu'il a consacré dans le carde de ce projet.

A mes collègues diplômants pour l'ambiance conviviale qu'il ont fait régner dans le laboratoire. Et en particulier Denis Cotte pour sa précieuse aide dans le cadre de l'utilisation de la CA Keon

A ma famille pour le soutient moral qu'il m'ont apporté.

BORSA Sébastien

## 7. Bibliographie :

Les documents fournis avec les routeurs :
Introduction to Cisco Routeur Configuration
Software Configuration Guide
Ces deux ouvrages m'ont permis de me familiariser avec l'équipement Cisco.

IPSec de Naganand Doraswamy – Dan Harkins edition PTR/PH
Ce livre explique tout le fonctionnement d'IPSec.

L'article : Methods and protocols for Secure Key Negotiation Using IKE par Michael S. Borella parut dans IEEE Network Juillet/Aout 2000.
Cet article explique en détail l'échange des clés IKE et ISAKMP. (très intéressant)

IPSec Network Security sur le site de Cisco à l'adresse :
http://www.cisco.com/univercd/cc/td/doc/product/software/ios113ed/113t/113t_3/ipsec.htm
Ce document explique tous les paramètres relatif à Cisco et propose des débuts de configuration. (très utile)

Configuring IPSec sur le site de Cisco à l'adresse :
http://www.cisco.com/univercd/cc/td/doc/product/software/ios121/121cgcr/secur_c/scprt4/scdipsec.htm

Certification Authority Interoperability sur le site de Cisco à l'adresse :
http://www.cisco.com/univercd/cc/td/product/software/ios113ed/113t/113t_3/interop.html

Explication de SCEP :
http://www.vpnc.org/draft-nourse-scep

Forums sur le site de google.com
Les news groups :
Comp.dcom.sys.cisco
Comp.dcom.vpn

Le *technical support* de SSH Sentinel:
sentinel-support@ssh.com

Borsa Sébastien
Laboratoire de transmissions de données

TE 3
22 novembre 2001

# Personal Firewall

Table des matières: page



Annexe :

# 1. Introduction

Un firewall est un logiciel permettant de filtrer les informations qui entrent et sortent d'un réseau ou d'une machine. Les filtres sont définis par des règles.
Par exemple : **source** any **destination** any accept port TCP 80.
Cette règle permet de laisser passer tous les paquets utilisant le port TCP 80.
Les firewall personnels sont apparus avec les technologies permettant aux utilisateurs d'obtenir de grand débit (ADSL). Ces firewall ont aussi la possibilité de bloquer les programmes voulant accéder au réseau.

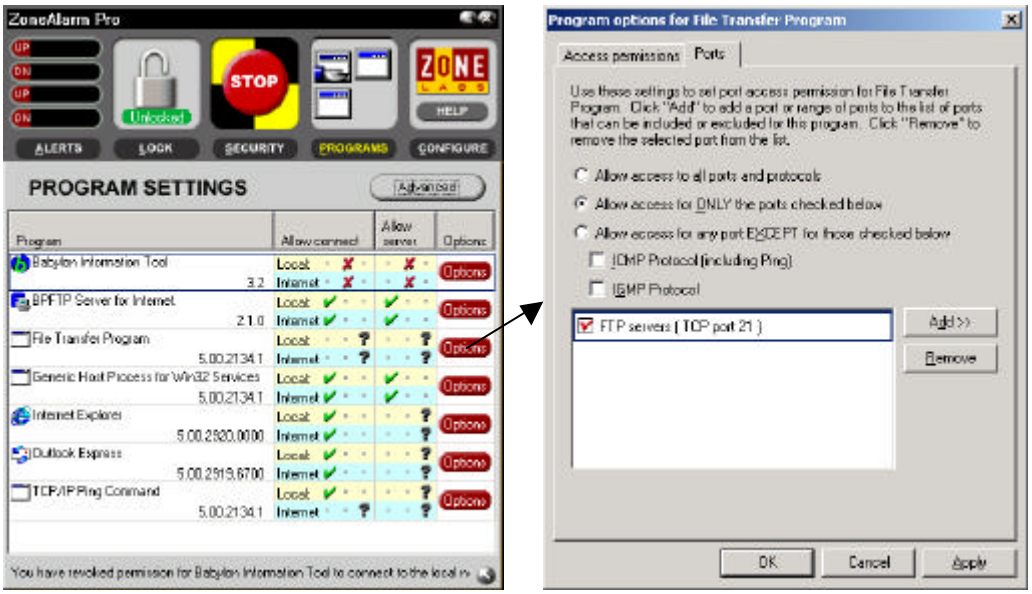
# 2. Philosophie

## 2.1 ZoneAlarm

Le principe consiste à définir une configuration globale pour la machine. Puis lorsque les programmes tentent d'accéder au réseau (il est possible de les insérer manuellement), ils sont entrés dans une liste et l'utilisateur choisit si le programme est autorisé à accéder au réseau. Il est possible de définir une règle par défaut : le nouveau programme sera automatiquement accepté, refusé ou c'est à l'utilisateur de choisir.
Chaque programme peut accéder au réseau sur n'importe quel port. Il faut, à chaque nouveau programme entré, définir quel port il peut utiliser.
ZoneAlarm distingue deux réseaux :

Intranet : Réseau local. Il est possible que le réseau local n'existe pas. Pour définir le réseau local, ZoneAlarm utilise l'adresse IP de la machine et le *subnet mask*. Il est possible par la suite de définir aussi d'autre sous réseaux ou machine comme faisant partie du réseau local.
Internet : comprend tout ce qui ne fait pas partie du réseau local.

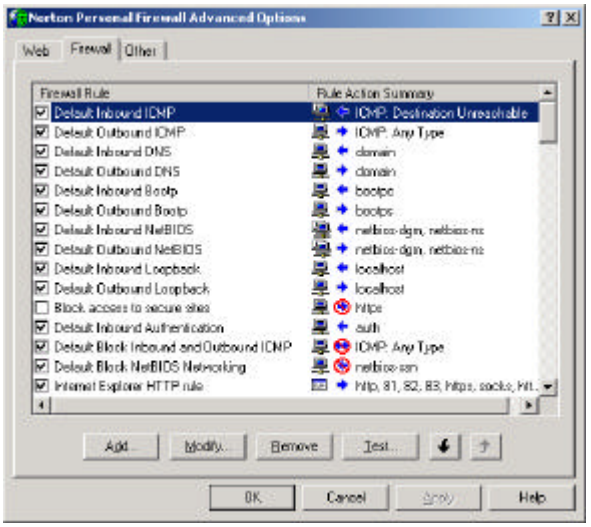
Ce système permet de définire des règles pour Internet et le réseau local.
Par exemple : en local on autorise NetBios et ftp alors que pour des connexions Internet ceci n'est pas possible.




ZoneAlarm présente aussi une fonctionnalité intéressante qui est la protection des règles par un mot de passe. En plus de ne pas pouvoir autoriser un programme à accéder au réseau, il n'est pas possible de désactiver ZoneAlarm sans ce mot de passe à moins de tuer les tâches Zapro.exe et vsmon.exe.
Un filtrage des pièces jointes à un e-mail est aussi présent. Il permet uniquement de mettre le fichier en quarantaine en modifiant l'extension du fichier à risque. Cette méthode permet lorsqu'on ouvre un mail de ne pas voir la pièce jointe s'exécuté automatiquement.

## 2.2 Norton personal firewall

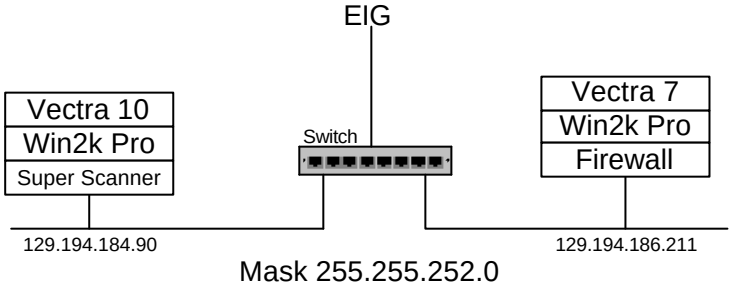
Norton personal firewall utilise une suite de règles comme on a l'habitude de le voir dans les firewalls. On a ici une priorité des règles suivant l'ordre dans lesquels quel elles apparaissent. On peut à tout moment ajouter une règle, soit pour **tous** les programmes, soit pour **un seul**, en choisissant les ports TCP/UDP et définir sa priorité en la plaçant dans plus ou moins haut dans la liste.



A la différence de ZoneAlarm, Norton personal firewall ne bloque pas automatiquement un programme mais peut générer des règles pour le laisser passer.
Lorsqu'un programme veut sortir, il regarde si il existe une règle bloquant l'accès. S'il n'en n'existe pas, il va proposer l'installation de règles, soit pour permettre soit pour bloquer l'accès au programme en question.
Pour les programmes du type serveur, Norton ne fait la détection que lors d'une tentative de connexion sur la machine.
Norton contient des fichiers logs très détaillés permettant de connaître le nombre de connexion fait sur un port avec un host, le nombre de bytes reçus et envoyés, et le numéro de la règle qui a été utilisée.
Le firewall permet aussi de faire un filtrage des éléments ActiveX, applet Java et cookies suivant la page consultée.

## 3. Scan Port

**Scénario :** Vectra 10 scan les ports de Vectra 7 afin de trouver une faille.

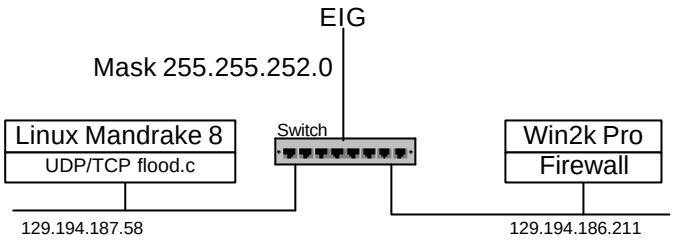



**But :** Vérifier que les règles définies pour ZoneAlarm et pour Norton Personal Firewall fonctionnent correctement et que seul les ports ouverts par le firewall sont disponibles.
Pour faire ce test, j'ai utilisé le logiciel SuperScan 3.00, qui m'a permit de scanner tous les ports (1-65535)

**Résultat :** Comme on pouvait l'attendre, les deux Firewall bloquent bien les ports verrouillés.

## 4. Déni de service (DoS)

**Scénario :** une machine envoie des paquets UDP ou TCP sur la machine contenant le firewall afin de faire tomber la machine ou de la rendre inutilisable.




La machine avec le firewall est une HP Vectra, pIII 500 MHz, 128 Mb RAM

**But :** Mesurer les ressources que les firewall utilisent lorsqu'ils subissent un DoS et vérifier qu'ils ne tombent pas.

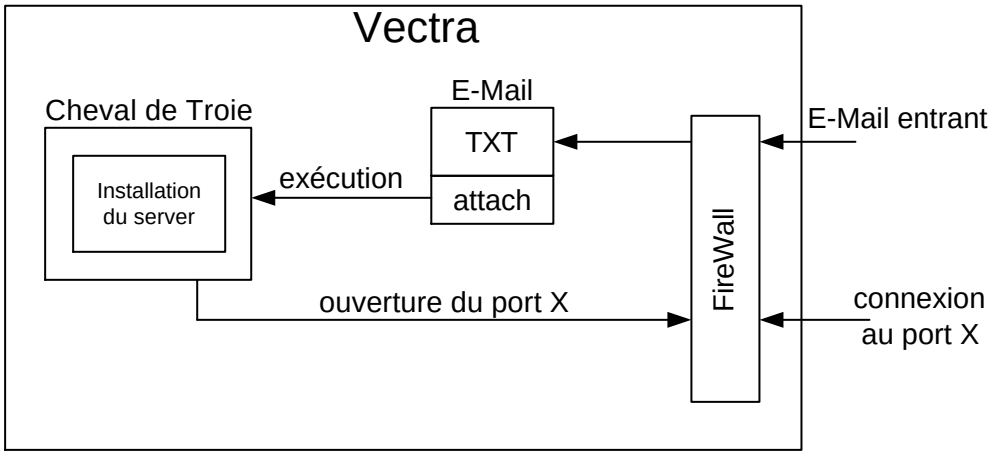
Le programme utilisé pour faire le DoS est un flooder UDP. Ce programme fonctionne sous Linux et permet de générer un trafic important. J'ai refait le même scénario avec un flood TCP. Pour le code source, voir annexes 1 et 2

Pour mesurer les résultats, j'ai utilisé le *CPU Usage* du *Task Manager* de Windows 2000.

**Résultat :** Norton personal firewall ne supporte pas les deux DoS et la machine est bloquée.
ZoneAlarm demande ~60 % des ressources avec UDP et la machine est bloquée avec TCP.

**5. Contamination par un cheval de Troie**

**Scénario :** Un mail est envoyé. Il contient un cheval de Troie qui va permettre à un pirate de s'infiltrer dans la machine. Pour cela, le cheval de Troie installe un serveur sur la machine. Ce serveur ouvre un port et attend une demande de connexion.



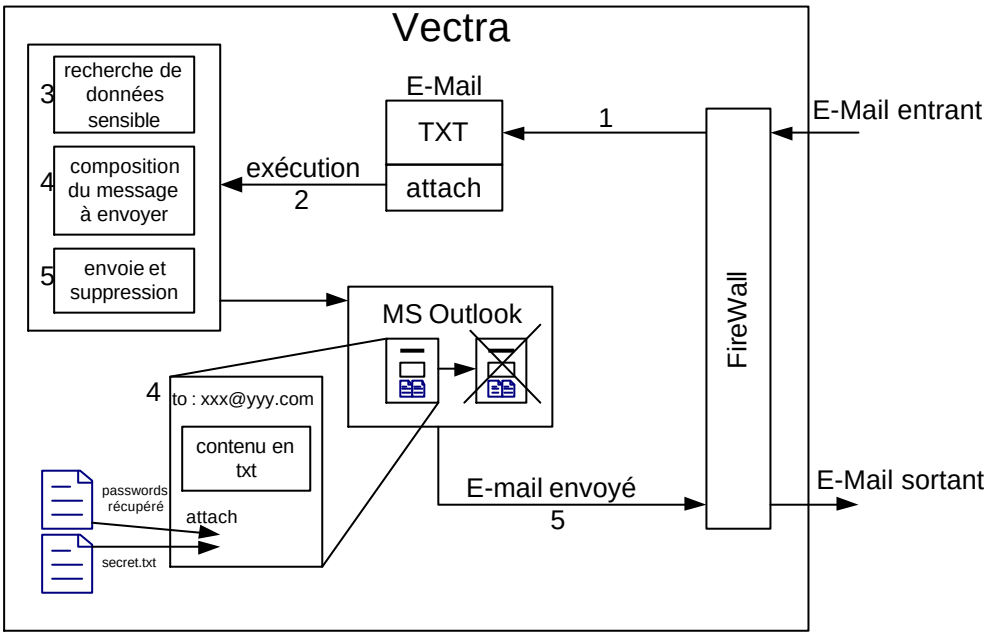

**But :** vérifier que les programmes ne peuvent pas sortir sans autorisation.
Pour faire ce test, j'ai utilisé NetBus pro 2.01. Ce cheval de Troie est connu par les anti-virus. Mais pour ce test, on suppose que les antivirus ne le connaissent pas encore et qu'il a réussit à s'implanter sur la machine.

**Résultat :**

ZoneAlarm : Le programme a été bloqué sans problème.
Norton Firewall : Le programme a été bloqué. On notera que Netbus est bloqué par une règle par défaut et qu'il a fallu la désactiver pour faire le test.

## 6. Utilisation de Microsoft Outlook

**Scénario:** Un mail contenant un programme est reçu. Le programme un fois exécuté, effectue des recherches de mot de passe, prend des informations confidentielles. Une fois les informations récoltées, il utilise Microsoft Outlook pour envoyer un e-mail contenant les informations récoltées.




**But :** montrer qu'il est possible à des logiciels d'utiliser Microsoft Outlook, ou d'autre programme, pour envoyer des mails. Microsoft appelle ça : les automations.

Ceci pose un gros problème de sécurité car Outlook étant le programme de messagerie par défaut, il est normal qu'il puisse accéder au réseau. Donc si un programme utilise Outlook pour envoyer un mail, la firewall ne bloquera pas le message. Il est possible d'utiliser d'autre logiciel ayant accès au réseau tel que Internet Explorer.

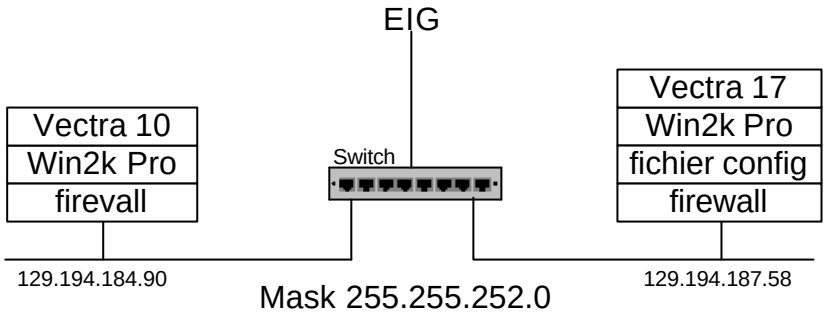
Remarque : Il est possible de faire une mise à jour des programmes afin de bloquer les automations.

Pour démontrer cette faille, j'ai utilisé un programme en visual basic (source voir annexe 3)

**Résultat :** ZoneAlarm et Norton Firewall n'ont pas arrêté le message.

## 7. Gestion centralisée

**Scénario :** Plusieurs machines ont le firewall installer et vont utilisé les fichiers de configuration se trouvant sur une machine.




Les fichiers de configurations sont dans un répertoire partagé (Internet logs) sur Vectra 17

**But :** permettre une configuration centralisée des règles, log, alerte.

**ZoneAlarm**

Pour faire ce test, il a fallut modifier la base de registre.
Dans les chemins :
HKEY_LOCAL_MACHINE\SOFTWARE\Zone labs\MiniLog
HKEY_LOCAL_MACHINE\SOFTWARE\Zone labs\TrueVector\LocalStoreDir
HKEY_LOCAL_MACHINE\SOFTWARE\Zone labs\TrueVector\LogStoreDir
Modifier les chemins c:\winnt\Internet logs\ en \\adresse de la machine\rep. partagé\
Dans notre cas : \\129.194.187.58\Internet Logs\

Les fichiers de configuration sont par défaut dans le répertoire :
c:\WINNT\Internet Logs

Il semblerait que l'ouverture des fichiers de configuration est exclusive ce qui ne permet pas à plusieurs programmes d'utiliser simultanément un même fichier de configuration. Cependant, il est tout à fait possible de mettre les fichiers de configuration et logs sur une autre machine. La mise de tous les fichiers de configuration sur une machine peut-être utile malgré le fait qu'il faille un fichier par machine. En effet, si un virus veut modifier les fichiers de configuration, il doit atteindre le réseau. Or la configuration de ZoneAlarm est sensée le bloquer.

Il n'est pas possible de faire une copie du fichier de configuration pour le mettre sur une autre machine.
Au niveau des alertes, il n'est pas possible de les exporter sur une autre machine.

**Norton**

Il n'est pas possible d'exporter les fichiers de configuration, le firewall référence les règle dans la base de registre de la machine. De plus, la configuration dans la base de registre est relativement complexe, ce qui ne facilite pas la tâche.

**Résultats :** Norton personal firewall et ZoneAlarm ne permettent pas une gestion centralisée de la configuration.

## 8. Conclusion

Dans l'ensemble, ZoneAlarm et Norton personal firewall permettent de mieux protéger les machines contre les attaques extérieures et contre les chevaux de Troie.
Norton firewall permet plus de possibilité quant à l'édition des règles, mais ce système devient très vite complexe à gérer. Les règles automatiques inexistantes ne pose pas de problème à priori car contrairement a ZoneAlarm, il est possible de définir des règles s'appliquant à tous les programmes.
D'un point de vue plus subjectif, j'ai trouvé que la méthode de configuration de ZoneAlarm est très simple.
Norton personal firewall, avec son système de règle, va peut-être un peu trop loin ce qui complique passablement la configuration.
Il faut rappeler que ces firewalls sont destinés à se trouver sur des machines de client et ne sont pas fait pour protéger un réseau.

## 9. Tableau comparatif

| Produit | ZoneAlarm Pro 2.6.357 | Norton personal firewall 2000 |
|---|---|---|
| Fabriquant | Zonelabs | Symantec |
| Site WEB | http://www.zonelabs.com | http://www.symantec.com |
| Système | Windows 9x, Me, NT4 sp3, 2000, XP | Windows 9x, Me, NT4 sp3, 2000 |
| Processeur | 386 (rec 486) | pentium 133 MHz |
| Mémoire | 8 Mb | 32 Mb pour NT et 2000 (rec. 48 Mb), 24 Mb pour 9x. Me (rec. 32 Mb) |
| Espace disque | 3 Mb | 10 Mb |
| Protection Scan Port | oui | oui |
| Occupation du CPU par un DoS UDP | ~60% | Machine bloquée |
| Occupation du CPU par un DoS TCP | Machine bloquée | Machine bloquée |
| Bloque un cheval de Troie | oui | oui |
| Bloque les Automations | non | non |
| Protection par mot de passe | oui | non |
| Fichiers log | oui | oui |
| Protection applet Java | non (version pro and Internet cleanUp) | oui |
| Protection ActiveX | non (version pro and Internet cleanUp) | oui |
| Scan mail entrant | oui | non |
| Une installation pour tous les users | oui | oui |
| Envoie des alertes sur un serveur | non | oui |
| Mise des logs sur un serveur | oui | non |
| Une configuration pour X machines | non | non |

Les tests ont été effectués sur des machines HP Vectra, pIII 500 MHz, 128Mb RAM avec Windows 2000