VPN – IPSec

Ce sujet a été initialement rédigé par Carlos Aguilar, Éric Asselin et Joris Barrier.

Environnement de travail

Cette section décrit comment travailler dans le cadre de ce TP en fonction de votre localisation physique ou virtuelle.

Travail à l'N7 ou sur machine personnelle GNU / linux

Pré-requis : QEMU + KVM

Si et seulement si machine N7 hors salles réseau / sécurité :

```
hote:~$ mkdir /work/monlogin
hote:~$ cd /work/monlogin
```

Dans tous les cas :

```
hote:~$ wget -c 'http://flash.enseeiht.fr/bemorgan/debian-10.9.0.tgz'
hote:~$ tar xvzf debian-10.9.0.tgz
debian-10.9.0/
debian-10.9.0/disk.qcow2
debian-10.9.0/CRED
debian-10.9.0/start.sh
hote:~$ cd debian-10.9.0 && ./start.sh
```

Réduire le terminal **sans le fermer**.

Travail sur machine personnelle non GNU / linux

Pré-requis : Oracle VirtualBox

Télécharger Oracle VirtualBox :

`https://www.virtualbox.org/wiki/Downloads`

Installer VirtualBox sur sa machine personnelle :

`https://www.virtualbox.org/manual/UserManual.html#installation`

Télécharger la machine virtuelle :

`http://flash.enseeiht.fr/bemorgan/debian-10.9.0-xfce.ova`

Importer la machine virtuelle :

`https://www.virtualbox.org/manual/UserManual.html#ovf-import-appliance`

Dans la section suivante, si vous utilisez VirtualBox, remplacez les ouvertures de terminaux + connexion SSH à la machine virtuelle `lxcland` par une simple ouverture de terminal directement sur la machine virtuelle `lxcland`.

Environnement de travail

La machine virtuelle `lxcland` contient 1 conteneur `lxc` par machine du réseau virtuel pour ce TP (hôte ou routeur). Cette machine virtuelle ne possède pas d'inter-

face graphique. Il s'agit de s'y connecter en SSH autant de fois que nécessaire pour s'attacher aux conteneurs des différents réseaux.

Démarrer les conteneurs

Ouvrir un nouveau terminal et se connecter en SSH à la machine virtuelle :

```
hote$ ssh -p 2224 n7@localhost
n7@lxcland:~$
```

Démarrer les conteneurs :

```
n7@lxcland$ cd tp-ipsec
n7@lxcland:~/tp-ipsec$ ./start.sh create
```

Démarrer les conteneurs :

```
n7@lxcland:~/tp-ipsec$ ./start.sh
```

Les conteneurs sont démarrés, vous pouvez vérifier leur présence avec :

```
n7@lxcland:~/tp-ipsec$ sudo lxc-ls
debian-ipsec host-a host-b router-a router-b
```

Les 4 conteneurs importants pour le TP sont les 4 derniers de cette liste.

S'associer aux conteneurs

Répéter cette pour les 4 conteneurs précédents : `hote-a, router-a, router-b, hote-b` :

```
# Ouvrir un terminal sur la machine hôte
hote$ ssh -p 2224 n7@localhost
n7@lxcland:~$ sudo lxc-attach host-a
root@host-a:/#
```

Vous avez donc maintenant 4 terminaux de travail sur la machine hôte qui sont connectés à la machine virtuelle de travail en SSH et associés aux conteneurs représentant nos 4 hôtes du réseau de ce TP.

Ouvrir wireshark sur la machine virtuelle lxcland

La dernière étape de préparation de ce TP est le lancement de wireshark sur les 3 `bridges` (switchs virtuel) de nos trois réseaux : a, router et b. Nous allons répeter les opérations suivantes pour les trois bridges : `br-a`, `br-router` et `br-b` :

```
# Ouvrir un terminal sur la machine hôte
hote$ ssh -p 2224 n7@localhost wireshark
```

Réduire le terminal **sans le fermer**. Sélectionner la capture sur le *bridge* `br-router`

Rédaction des solutions et récupération du travail

Ouvrez sur la machine hôte l'éditeur de texte de votre choix. Pour copier coller du texte et / ou des scripts de configurations, veuillez procéder de la façon suivante :

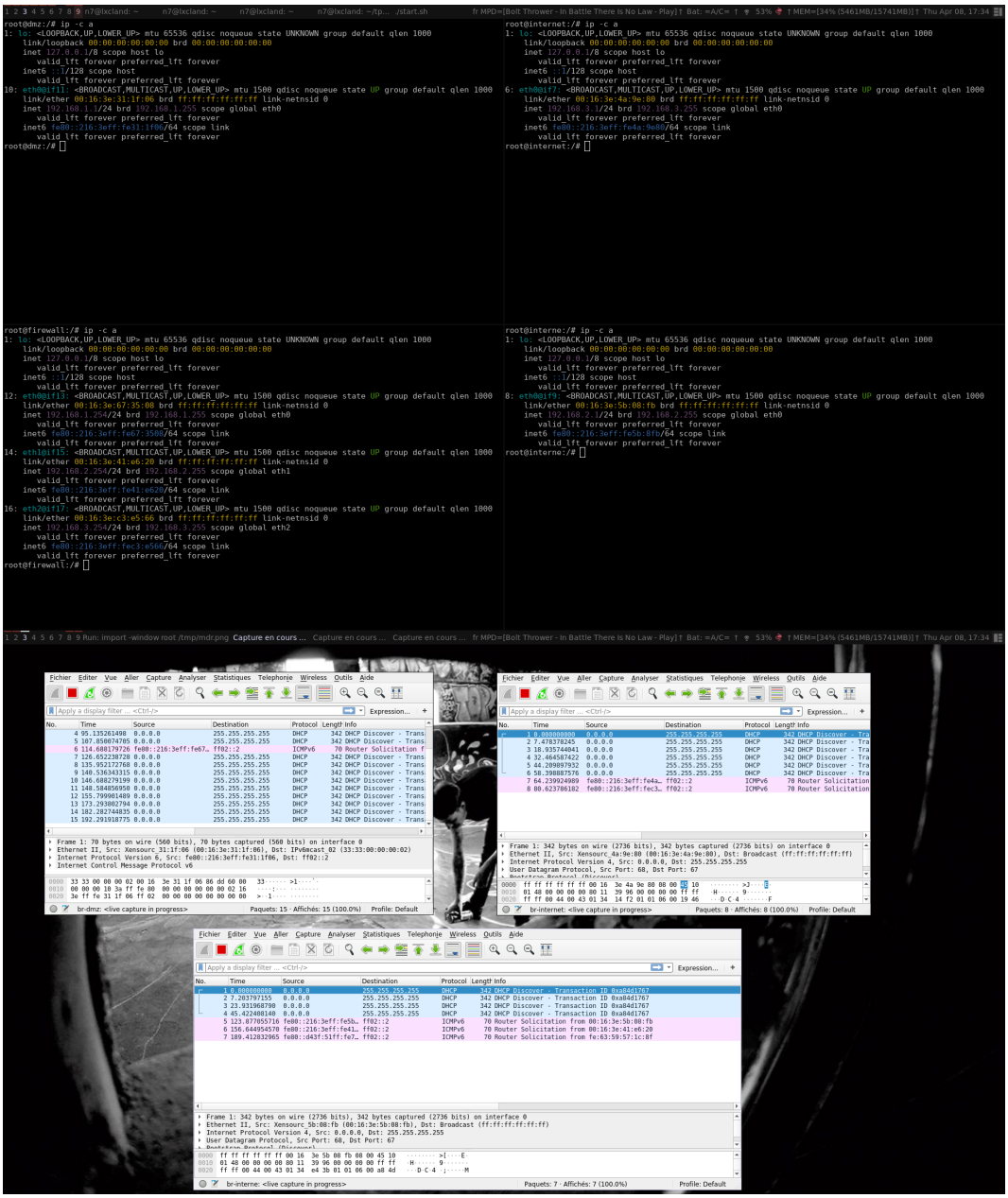
```
# Copier le texte à insérer dans le conteneur
root@firewall:/# cd /root/
root@firewall:~# cat > ipsec-tools.conf
# Coller le texte est frapper Ctrl+d (^D) ce qui va inscrire le fichier
```

Pour récupérer votre travail depuis le disque de la machine virtuelle hôte à posteriori. Admettons que le fichier `ipsec-tools.conf` soit dans .

```
hote$ scp -P 2224 n7@localhost:ipsec-tools.conf .
```

Environnement attendu

Voici ce que vous devriez obtenir une fois que tous les terminaux et wiresharks sont lancés. Vous devez zoomer... Les réseau et conteneurs représentés dans cette image sont ceux du TP *firewall* mais le contept est identique.



Objectifs

L'objectif de ces séances est de comprendre en quoi le protocole IPSec permet de créer des réseaux privés virtuels (VPN) en assurant la sécurité des communications entre deux entités via un canal non-sécurisé.

C'EST PARTI !

1 Introduction

La technologie des réseaux privés virtuels (VPN) repose sur la possibilité d'émettre des données sensibles sur un réseau public tel qu'Internet, et de les acheminer jusqu'au destinataire en assurant leur sécurité. À l'origine, la pile de protocoles TCP/IP a été conçue pour offrir un maximum de disponibilité sans prendre des mesures fortes pour assurer la sécurité de l'information. Des mécanismes supplémentaires doivent donc être mise en oeuvre pour garantir l'authentification des interlocuteurs ainsi que l'authenticité, l'intégrité et la confidentialité des messages transmis.

Si l'on souhaite sécuriser une communication, il est possible d'avoir recours à des tunnels SSH. Cependant, l'utilisation d'un tel procédé ajoute un sur-coût considérable et diminue la charge utile. C'est pourquoi nous allons nous intéresser à d'autres moyens tels que IPSec pour mettre en oeuvre des architectures réseaux dont les communications doivent être sécurisées.

2 Pré-configuration

Réalisez le schéma de la figure 1 qui servira tout au long des exercices.

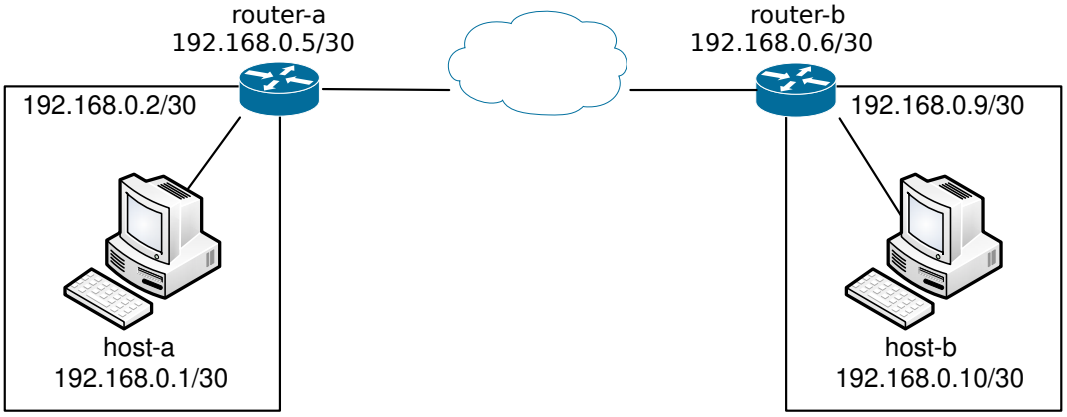


FIGURE 1 – Schéma complet du TP.

3 IPSec

3.1 Définition du protocole

"IPsec est conçu pour donner une sécurité de haute qualité, basée sur la cryptographie, sans problème d'interopérabilité, pour IPv4 et IPv6. L'ensemble des services de sécurité proposé inclut le contrôle d'accès, l'intégrité en mode non connecté, l'authentification de l'origine des données, la protection contre le rejeu (une forme d'intégrité), la confidentialité (chiffrement), et une certaine confidentialité sur le flux

de trafic. Ces services sont offerts au niveau de la couche IP, offrant de la protection à la couche IP et aux couches supérieures."

RFC 2401, traduction libre

Un des avantages d'IPSec est qu'il permet de sécuriser toute communication ayant recours à IP. IPSec utilise trois protocoles pour fournir cette sécurité :

— **Authentication Header (AH)** fournit l'intégrité et l'authentification de l'origine des données sur l'ensemble du paquet.

— **Encapsulating Security Payload (ESP)** fournit la confidentialité à l'aide du chiffrement ainsi qu'une authentification des données, à l'exception de l'entête IP.

— **Internet Key Exchange (IKE)** fournit le mécanisme d'échange de clés.

Ces protocoles peuvent être combinés ou utilisés seuls. Les protocoles AH et ESP sont alors utilisés conjointement afin de créer une communication sécurisée entre différentes entités. Le protocole IKE est utilisé pour simplifier la distribution des clés servant au chiffrement et à l'authentification.

3.1.1 AH en mode transport

L'entête AH est inséré dans le paquet IP originel, entre l'entête IP et l'entête de la couche transport (TCP ou UDP), tel que présenté dans la figure 2.

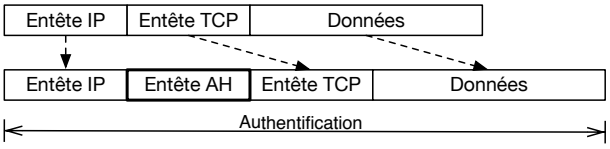


FIGURE 2 – Paquet IP avec AH en mode transport.

3.1.2 AH en mode tunnel

Le mode tunnel nécessite la création d'un nouvel entête IP, après lequel sont placés l'entête AH puis le paquet IP originel, tel que présenté dans la figure 3.

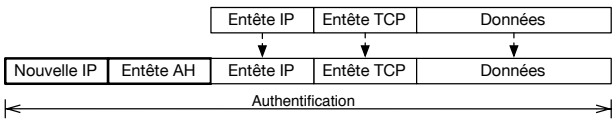


FIGURE 3 – Paquet IP avec AH en mode tunnel.

3.1.3 ESP en mode transport

Le paquet ESP est créé en insérant l'entête ESP dans le paquet IP originel et en plaçant la terminaison et les données d'authentification à la fin du paquet, tel que présenté dans la figure 4.

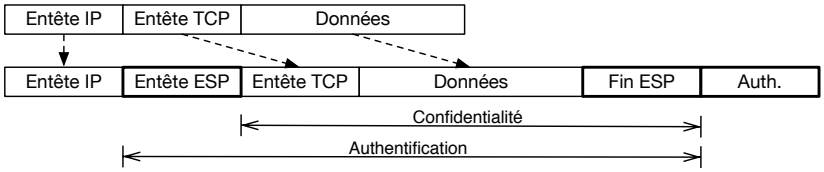


FIGURE 4 – Paquet IP avec ESP en mode transport.

3.1.4 ESP en mode tunnel

Le paquet IP originel est entièrement encapsulé dans un nouveau paquet. Un nouvel entête IP est créé suivi de l'entête ESP et du paquet originel. Puis la terminaison ESP et les données d'authentification y sont annexés, tel que présenté dans la figure 5.

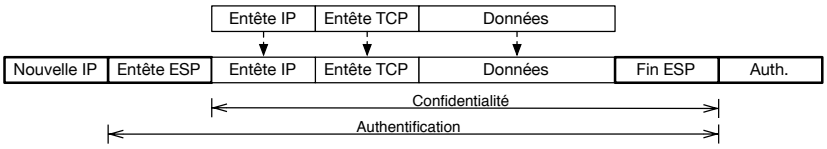


FIGURE 5 – Paquet IP avec ESP en mode tunnel.

3.2 Exercice 1 - Entêtes IPSec

Dans ce premier exercice, la représentation du schéma de la figure 6 permet de bien comprendre le fonctionnement d'IPSec, comment les protocoles AH et ESP interagissent ainsi que d'évaluer leurs capacités à répondre aux besoins d'authentification, d'intégrité et de confidentialité.

Afin de sécuriser une communication, il est impératif de connaître quelles sont les adresses IP concernées, dans quel sens la communication doit être sécurisée et avec quel entête. IPSec appelle cela une ***Security Association*** **(SA)**. Il est important de noter que chaque SA est unidirectionnelle et que pour une communication bidirectionnelle, il faut spécifier deux SAs. Ensuite il faut établir dans quels cas il faut utiliser les SA pour sécuriser les messages, c'est ce qu'on appelle un politique de sécurité ou ***SP (pour Security Policy)***. En résumer, une SP spécifie ce que l'on veut et une SA, comment on le veut.

On utilisera le package Debian **ipsec-tools**, qui contient tout le nécessaire. Pour configurer le package **ipsec-tools**, il suffit d'éditer le fichier **/etc/ipsec-tools.conf** et d'y inclure des commandes **setkey**. Vous pouvez consulter le manuel avec la commande :

```
man setkey
```

Dans le cadre de cet exercice, nous utiliserons principalement les quatre commandes suivantes :

— **flush** : nettoie les SA
— **spdflush** : nettoie les SP
— **add** : définie une SA
— **spdadd** : définie une SP

FIGURE 6 – Exercice 1 - Schéma de l'architecture réseau.

3.2.1 Authentification et intégrité avec AH

Il a été vu en cours qu'une adresse IP n'est pas suffisante pour garantir l'identité d'un hôte grâce à l'attaque IP Spoofing qui consiste à usurper l'adresse IP d'un hôte existant et potentiellement de confiance. Bien que des règles de pare-feu permettent d'éviter dans une certaine mesure cette attaque entre différents réseaux, ces règles n'y peuvent rien lors d'une connexion point-à-point.

Les manipulations suivantes démontrent comment le protocole AH d'IPSec permet de répondre aux besoins d'authentification et d'intégrité.

1. Utilisez le schéma de la figure 6 et assurez-vous de pouvoir communiquer avec la commande `netcat`. À l'aide de la commande `tcpdump` ou Wireshark, remarquez le format des entêtes de protocoles ainsi que les messages échangés en texte clair.

   ```
   serveur: netcat -vl ip_serveur port
   client: netcat -v ip_serveur port
   ```

2. Chargez avec la commande `modprobe` les modules `ah4`, `esp4` et `ipcomp`, nécessaire au bon fonctionnement des mécanismes cryptographiques de IPsec.
3. Pour configurer le protocole AH d'IPSec, il faut tout d'abord générer des clés que seuls les hôtes autorisés connaîtront. Il s'agit d'une clé de 256 bits, c'est-à-dire 32 octets, à l'aide de la commande :

```
dd if=/dev/urandom count=1 bs=32 | xxd -ps
```

4. Configurer IPSec sur les deux hôtes tel qu'illustré dans le listing suivant (**attention, il faut inscrire chaque commande sur une seule ligne, les commentaires sont à titre indicatif seulement**) et démarrer IPSec sur les deux hôtes avec la commande :

```
sudo setkey -f fichier_de_conf
```

Listing 1 – ipsec-tools.conf

```
#!/usr/sbin/setkey -f

# nettoyage des SA et de la SPD
flush;
spdflush;

# Security Association (SA)
add 192.168.0.5     # Source
    192.168.0.6     # Destination
    ah              # Protocol
    0x200           # Security Parameter Index (SPI)
    -A hmac-sha256  # Tag d'integrite + fonction de hashage
    0xffffffff...ffffffff; # cle 256 bits (32 octets)

add 192.168.0.6     # Source
    192.168.0.5     # Destination
    ah              # Protocol
    0x300           # Security Parameter Index (SPI)
    -A hmac-sha256  # Tag d'integrite + fonction de hashage
    0xffffffff...ffffffff; # cle 256 bits (32 octets)

# Security Policy (SP)
spdadd
    192.168.0.5     # Source range
    192.168.0.6     # Destination range
    any             # Upper-layer protocol
    -P out ipsec    # Policy (direction)
    ah/transport//require;  # How to process the packet

spdadd
    192.168.0.6     # Source range
    192.168.0.5     # Destination range
    any             # Upper-layer protocol
    -P in ipsec     # Policy (direction)
    ah/transport//require;  # How to process the packet
```

5. Communiquez avec `netcat` et vérifiez avec Wireshark. Quelles sont vos observations ?

6. Modifiez les clés sur une des deux machines et testez la communication. Quelles sont vos observations ?

3.2.2 Confidentialité avec ESP

Le vol de données sensibles est l'une des plus grandes inquiétudes des entreprises. L'isolement total des systèmes informatique est aujourd'hui presque impossible et l'écoute passive de canaux de communication plutôt facile à mettre en œuvre. Le protocole ESP d'IPSec permet de chiffrer une communication entre deux hôtes et ainsi garantir la confidentialité des données échangées.

1. Pour configurer le protocol ESP d'IPSec, il faut tout d'abord générer des clés partagées. Dans ce cas-ci, il s'agit d'une clé de 160 bits, c'est-à-dire 20 octets, à l'aide de la commande :

```
dd if=/dev/random count=1 bs=20 | xxd -ps
```

2. Configurer IPSec sur les deux hôtes tel qu'illustré dans le listing suivant :

Listing 2 – ipsec-tools.conf

```
#!/usr/sbin/setkey -f

# netoyage des SA et de la SPD
flush;
spdflush;

# Security Association (SA)
add 192.168.0.5     # Source
    192.168.0.6     # Destination
    esp             # Protocol
    0x201           # Security Parameter Index (SPI)
    -E aes-ctr      # Crypto Algorithm
    0xffffffffffffffffffffffffffffffffffffffff; # cle 160 bits

add 192.168.0.6     # Source
    192.168.0.5     # Destination
    esp             # Protocol
    0x301           # Security Parameter Index (SPI)
    -E aes-ctr      # Crypto Algorithm
    0xffffffffffffffffffffffffffffffffffffffff; # cle 160 bits

# Security Policy (SP)
spdadd
    192.168.0.5     # Source range
    192.168.0.6     # Destination range
    any             # Upper-layer protocol
    -P out ipsec    # Policy (direction)
    esp/transport//require;  # How to process the packet

spdadd
```

```
192.168.0.6   # Source range
192.168.0.5   # Destination range
any             # Upper-layer protocol
-P in ipsec     # Policy (direction)
esp/transport//require;  # How to process the packet
```

3. Communiquez avec `netcat` et vérifiez avec Wireshark. Quelles sont vos observations ?
4. Modifiez les clés sur une des deux machines et testez la communication. Quelles sont vos observations ?
5. Combinez AH et ESP, communiquez avec `netcat` et vérifiez avec Wireshark. Comment sont combinés les entêtes ?

3.3 IPSec et le protocole *Internet Key Exchange* (IKE)

Nous avons vu dans l'exercice précédant que toute la sécurité fournie par IPSec tient du secret que sont les clés. Alors dès qu'un attaquant les obtient, il peut non seulement usurper l'identité d'un des hôtes mais également déchiffrer les données capturées lors d'une communication entre deux hôtes légitimes.

Dans un premier temps, nous allons configurer le protocole IKE qui permet de négocier dynamiquement les **SA** à partir d'un secret partagé, appelé *pre-shared key* (**PSK**). Le cas est très similaire à l'exercice précédent, à l'exception qu'un attaquant ne peut pas déchiffrer des données capturées entre deux hôtes légitimes. Par contre, s'il est en possession des clés, il pourra toujours usurper l'identité d'un hôte. Dans un deuxième temps, nous verrons comment régler en partie ce problème avec une paire de clés publique/privée et une autorité de certification (**CA**).

3.3.1 Introduction à IKE avec un secret partagé

Le protocole IKE fonctionne en 2 phases différentes. Dans ce TP nous nous concentrons sur le mode ***main*** de la première phase. La première phase, permettant l'authentification mutuelle, consiste en l'échange de 6 messages dans lesquels l'initiateur propose plusieurs suites de protection. Chaque suite définie l'algorithme de chiffrement, l'algorithme de hachage, la méthode d'authentification ainsi que le groupe Diffie-Hellman (pour partager un secret) et des attributs optionnels. La seconde phase, appelée ***quick mode***, va permettre de générer les **Security Association** (SA) qui vont protéger les échanges de données entre les deux extrémités. Au même titre que la phase 1, une négociation des suites de sécurité est effectuée.
Dans ce TP, nous utilisons les paramètres de phase suivants :

Phase 1 :
— Mode main
— Algorithme de hachage SHA-1
— Chiffrement avec 3DES
— Clé partagée de 20 octets
— Groupe Diffie-Hellman 2
— Durée de vie de 24h

Phase 2 :
— Adresses IP publiques et ports
— Authentification HMAC MD5
— Chiffrement avec 3DES
— Groupe Diffie-Hellman 5
— Durée de vie de 8h

3.4 Exercice 2 : Tunnel IPSec avec clé partagée

Nous avons vu dans la première partie de ce TP comment sécuriser une communication entre deux machines en utilisant les protocoles AH et ESP. Afin d'utiliser le protocole IKE, nous utiliserons l'outil **racoon**. Le manuel est disponible avec les commandes :

```
man racoon
man racoon.conf
```

Pour réaliser cette exercice, nous allons utiliser le schéma de l'exercice précédent, c'est-à-dire celui de la figure 6.

1. Afin d'établir votre secret partagé, générez une clé de 20 octets tel que dans l'exercice précédent et placez la clé dans le fichier `/etc/racoon/psk.txt` avec l'adresse IP de chaque hôte et ce, sur les deux machines. Attention aux droits du fichier `psk.txt`, ils doivent être `600` ou `-rw-------`.
2. Puisque les **SA** seront maintenant négociés dynamiquement avec l'outils `racoon`, il faut commenter la section des **SA** dans les fichiers `ipsec-tools.conf` récupérés de l'exercice précédent et garder seulement la section des **SP** que vous devrez modifier pour utiliser le mode **tunnel**.
3. Il faut ensuite configurer les éléments des phases de sécurité qui seront utilisés par IPSec et le protocole IKE. Le fichier `/etc/racoon/racoon.conf` contient tous les paramètres liés aux négociations de **SA** entre les deux hôtes.
   Le fichier `/etc/racoon/racoon.conf` suivant est donné en guise d'exemple et vous aidera à inscrire les bons éléments de configuration :

Listing 3 – racoon.conf exemple

```
log notify;

#Chemin du fichier contenant les cles partagees
path pre_shared_key "/etc/racoon/psk.txt";

# Parametres de phase 1
# Adresse IP du site distant
remote 192.168.0.6 {

    exchange_mode main; #le mode d'utilisation
```

```
    proposal {
        encryption_algorithm 3des; # algorithme de chiffrement
        hash_algorithm sha1; # algorithme de hashage
        authentication_method pre_shared_key; # methode authentification
        dh_group 2; # groupe Diffie-Hellman
        lifetime time 24 hours; # duree de vie des cles
    }
}

# Parametres de phase 2
# On indique les sous-reseaux et les protocoles vises
# ou anonymous pour toutes les connexions
sainfo address 192.168.0.5 any address 192.168.0.6 any
{
    pfs_group 5; # groupe Diffie-Hellman
    encryption_algorithm 3des; # algorithme de chiffrement
    authentication_algorithm hmac_md5; # algorithme de hashage
    lifetime time 8 hours; # duree de vie des cles
    compression_algorithm deflate; # algorithme de compression
}
```

4. Lancez IPSec avec les commandes `setkey -f ipsec-tools.conf` et `racoon -f racoon.conf`. En cas de difficultés, vous pouvez lancer racoon avec l'option -F (pour foreground) et -d pour augmenter le niveau de debug (jusqu'à -ddd pour le niveau maximum).
5. Communiquez avec `netcat` et vérifiez avec Wireshark. Quelles sont vos observations ? La communication est-elle chiffrée ? Comment sont utilisées les entêtes ?

3.5 Exercice 3 : Tunnel entre deux sous-réseaux relié par deux routeurs

Passons à un cas d'utilisation plus réel. Le schéma de la figure 7 représente le cas d'une entreprise dont les bureaux sont situés sur deux sites distants. Afin de réduire le coût des communications entre les sites, l'entreprise a opté pour la mise en place d'un tunnel sécurisé sur le réseau Internet publique.

Suite à un rapport d'analyse des algorithmes de cryptographie, l'entreprise vous demande d'utiliser AES-128 pour le chiffrement et SHA-256 afin de sécuriser le tunnel. De plus, elle vous demande de vous assurer que seul le trafic entre les deux sous-réseaux privés soit sécurisé par le tunnel.

1. Assurez-vous tout d'abord que les hôtes peuvent communiquer normalement.
2. Renseignez le fichier `racoon.conf` avec les paramètres de configuration décrits dans la mise en situation.
3. Renseignez le fichier `ipsec-tools.conf` afin de modifier les adresses IP des hôtes et de préciser les acteurs participant au tunnel.

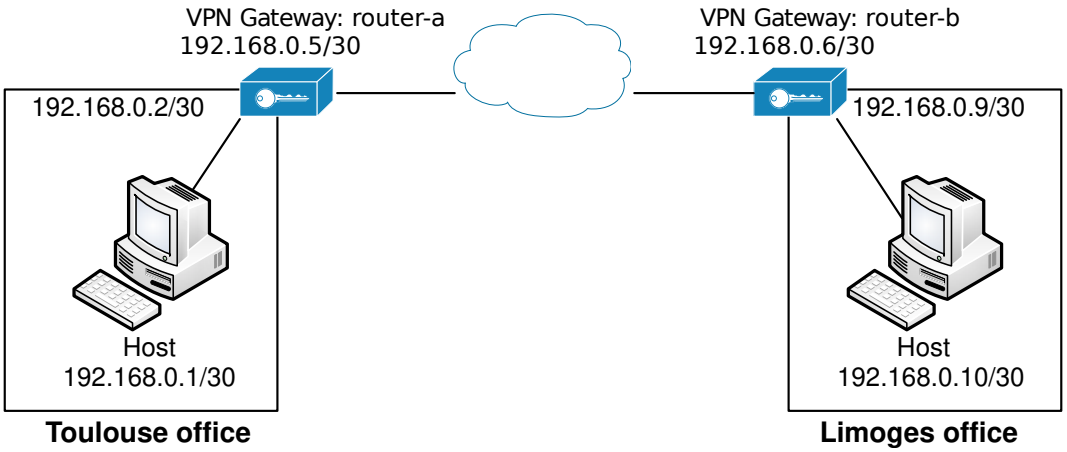


FIGURE 7 – Schéma des exercices 3 et 4.

4. Lancez **setkey** et **racoon** puis assurez-vous que les deux hôtes peuvent communiquer. Vérifiez également avec Wireshark que la communication est bien chiffrée.
5. (Optionel) Changez la configuration de **racoon** pour chiffrer seulement le trafic tcp sur le port 80. Quelles-sont vos déductions ?
6. (Optionel) Changez la configuration de **racoon** pour chiffrer tout le trafic mais dans un sens seulement. Pouvez-vous imaginer des cas d'utilisation où cela serait pertinent ?

3.6 Exercice 4 : Tunnel IPSec avec certificat

Jusqu'à maintenant nous avons fait passer le secret partagé d'un ordinateur à l'autre par nos propres moyens ce qui est ni pratique ni sécurisé. De plus, on pourrait arguer que l'utilisation d'un secret partagé pose, par construction, un problème de sécurité. En effet, il « suffirait » de pirater l'une des gateway du VPN pour connaître le secret de toutes les gateways.

Question : *Quels sont les problèmes rencontrés lors du partage des clés ? Quels sont les problèmes de sécurité qu'une telle architecture peut poser ?*

Afin de sécuriser davantage l'échange de secret, il est possible de construire un tunnel IPSec en se basant sur des certificats. Chaque routeur génère une bi-clé d'un système de chiffrement asymétrique (comme RSA) puis remplit une demande de certificat qui, accompagnée de la clé publique précédemment produite, est soumise à une autorité de certification (CA). Si cette dernière considère la demande légitime, elle signe les informations et la clé publique du routeur. Dans notre cas c'est l'une des deux gateway que jouera le rôle de CA, elle signera, sont certificat et celui de l'autre gateway.

L'objectif de l'exercice est de modifier l'architecture précédente afin d'utiliser des certificats en lieu et place des clés partagées.

1. Suivez *scrupuleusement* l'annexe 4.1 afin de générer un certificat pour les deux routeurs et de les placer dans les bons dossiers.
2. Au début de vos fichiers `racoon.conf` ajoutez la ligne suivante pour préciser le dossier contenant les certificats :

   ```
   path certificate /etc/racoon/certs
   ```

   Puis utilisez le manuel de racoon.conf section *Remotes Nodes Specification* pour modifier votre fichier afin d'utiliser les certificats. Vous devez ajouter les élémnts suivants la partie suivant `remote` du fichier afin de spécifier :

   (a) l'identifiant du routeur ; (`my_identifier asn1dn`)

   (b) l'identifiant de l'autre routeur ; (voir *peer*)

   (c) le type de certificat du routeur son nom et la clé ;

   (d) le type de certificat de l'autre routeur et son nom ;

   Dans la partie `proposal` vous devez seulement modifier la méthode d'authentification (de type signature RSA).
3. Vérifiez la connexion entre les machines 192.168.0.1 et 192.168.0.9 avec les outils `ping`, `netcat` et `ssh`.
4. Observez avec Wireshark ou `tcpdump` que le trafic entre les gateway est bien chiffré.

Astuce : En cas de problème essayez d'ajouter `verify_cert off` à `racoon.conf`.

4 Annexes

4.1 Génération des certificats

Les clés et certificats numériques des différentes parties (autorité de certification et les 2 gateways) vont simplement, dans le cadre du TP, être générés sur un des 2 gateway. Ils seront générés dans le répertoire `/etc/racoon/certs` sur la gateway que nous appellerons gateway 1.

1. Créez le répertoire `/etc/racoon/certs` s'il n'existe pas.
2. Générez le certificat numérique de l'autorité de certification à l'aide de la commande :

   ```
   /usr/lib/ssl/misc/CA.pl -newca
   ```

   Cette commande génère notamment le fichier `demoCA/cacert.pem`. Cette commande vous posera un certain nombre de questions aux-quelles vous répondrez avec l'aide de l'enseignant.
   Attention : Appuyez directement sur la touche entrée à la première question.
3. Générez le certificat numérique de la gateway 1 à l'aide des commandes :

   ```
   /usr/lib/ssl/misc/CA.pl -newreq
   /usr/lib/ssl/misc/CA.pl -sign
   ```

   La première commande génère le fichier `newreq.pem` et la clé privée chiffrée dans `newkey.pem`. La seconde crée véritablement le certificat en le signant à l'aide de la clé privée de l'autorité de certification. Vous devrez également répondre à des questions.
4. Le certificat généré s'appelle `newcert.pem`. Renommez-le par exemple en `certR1.pem`.
5. Vous devez également générer dans un fichier la clé privée en clair. Pour cela, exécutez la commande :

   ```
   openssl rsa -in newkey.pem -out keyR1.pem
   ```

   Le fichier contenant la clé est donc `keyR1.pem`.
6. Générez le certificat de la seconde gateway de la même manière et nommez-le `certR2.pem`. De même, générez la clé privée dans le fichier `keyR2.pem`.
7. Dans le répertoire `/etc/racoon/certs`, recopiez `demoCA/cacert.pem`. Exécutez ensuite la commande :

   ```
   ln -s cacert.pem `openssl x509 -noout -hash < cacert.pem`.0
   ```

Attention c'est un .0 (chiffre zero) et non un .O (lettre « o » majuscule).

8. Sur la gateway 1, dans le répertoire `/etc/racoon/certs`, vous devez donc avoir : `certR1.pem, certR2.pem, keyR1.pem, cacert.pem` ainsi que le lien généré par la commande ci-dessus.
9. De même, sur la gateway 2, vous devez avoir `certR1.pem, certR2.pem, keyR2.pem, cacert.pem` et le lien. Il vous faut donc recopier depuis la gateway 1 sur la gateway 2, les 3 premiers fichiers et générer le lien directement sur la gateway 2.

4.2 Rappel de commandes

4.2.1 Utilitaires

```
netcat -v -l [-u -n] ip_addr port
```

4.2.2 Transférer un fichier

```
Dest: netcat -l ip_addr port > nom_du_ficher
Src: cat nom_du_ficher | netcat ip_addr port
```

4.2.3 Flush complet de setkey

```
sudo setkey -F -P
```