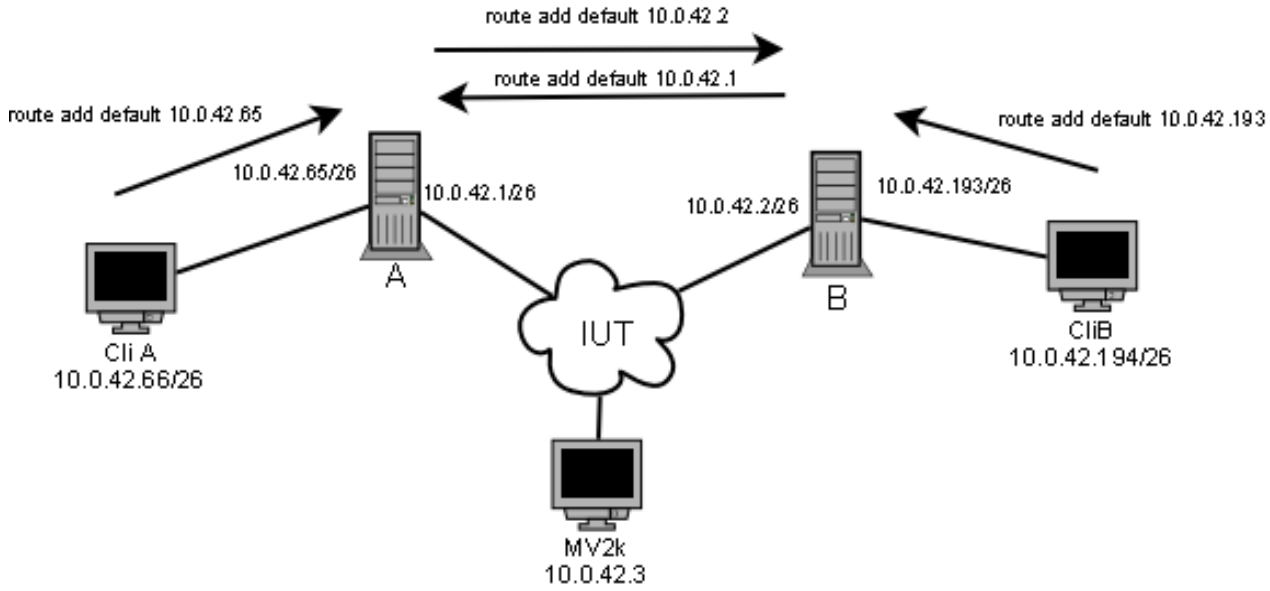
DOSSMANN Olivier
LEVAL Olivier

TP3: Pare-feu et IPSec

Les entreprises, du fait de la concurrence, ont besoin de protéger non seulement leur secret de fabrication, mais aussi leurs échanges via le réseau. Les grosses entreprises sont par ailleurs réparties sur plusieurs sites distants, ce qui augmente les chances d'intercepter les messages ou les données importantes entre les deux sites. Si la couche SSL permet des discussions sécurisées entre ordinateurs, elle ne permet pas pour autant de sécuriser l'ensemble des services puisque chacun doit posséder une implémentation de SSL.
Pour remédier à cela nous pouvons sécuriser le réseau via IPSec. Ce dernier est la mise en place d'un tunnel sécurisé entre deux réseaux. Etant situé plus bas que la couche SSL, tout les services des deux réseaux bénéficient de cette sécurisation des données. D'ailleurs c'est d'IPSec que se base VPN, preuve que c'est assez bien sécurisé.
Commençons donc ce TP dont le sujet principal est justement IPSec.

Pour toute la durée du TP nous suivrons la topologie du réseau suivante :



Nous sommes donc en présence de 3 sous réseaux préfixés par 10.0.42.X:
- de 0 à 63: Sous réseau liant A et B au sein du réseau de l'iut
- de 64 à 127: Sous réseau liant CliA et A
- de 128 à 191: Non utilisé
- de 192 à 255: Sous réseau liant CliB et B

Etape A : Création routeurs et configurations IP

Nous lançons deux machines virtuelles BSD munies d'IPSec dans le noyau (ceci nous évitera de recompiler le noyau pour la suite).
Le serveur A, que nous appellerons ServA, possède l'IP 10.0.42.1/26, tandis que le serveur B, que nous appellerons ServB, possède l'IP 10.0.42.2/26.
Les interfaces, sous Vmware, sont en mode *Bridge*.
Pour cela nous ajoutons dans le fichier /etc/rc.conf les lignes suivantes :

```
ifconfig_lnc0 = "inet 10.0.42.1/26" # ou 10.0.42.2/26
hostname = "serva" #ou servb selon ordinateur
defaultrouter="10.0.42.1" # ou 10.0.42.2
```

Les fichiers de configuration réglés sur les deux machines, nous les redémarrons à l'aide de la commande **reboot**.

Au redémarrage nous faisons un ping entre les deux machines : les deux machines pinguent (si pinguer est un verbe accepté du vocabulaire français).

Par la suite, nous testons l'UDP à l'aide de netcat, et plus particulièrement des commandes suivantes :

```
nc -ul 1234
ET
nc -u 10.0.42.1 1234
```

On fait une des commandes sur une machine, et l'autre commande sur l'autre machine. Après quoi nous tapons du texte, reçu de l'autre côté et inversement.
La capture des trames montre que tout passe en clair. L'utilisation du filtre **udp** sous Wireshark permet de cibler aux mieux les requêtes au travers du réseau et de séléctionner celles que nous voulons.

Etape B : Mise en place d'un pare - feu

Afin de pouvoir utiliser le pare - feu, il a fallu repasser la carte réseau connectée en NAT sur une

adresse locale avec la MH. Pour cela nous avons modifié les fichiers des machines virtuelles et ajouté une carte réseau avec les commandes suivantes :

```
Ethernet1.addressType = "generated"
Ethernet1.connectionType = "nat
```

Après redémarrage nous avons enfin une seconde carte que nous configurons de suite : *ifconfig lnc1 192.168.0.5.* Ensuite nous configurons SSH comme expliqué dans le TP2 (RootPermitLogon yes), nous le lançons sur le serveurs et utilisons winSCP sur la machine hôte pour placer le fichier ipfw.ko sur la MV.
Une fois le fichier sur la machine, on le met dans le dossier /boot/kernel, après quoi nous le chargeons à l'aide de la commande suivante :

```
kldload ipfw
```

Par défaut le pare – feu bloque toutes les circulations, mais pour permettre à UDP de passe nous faisons ceci :

```
ipfw add allow udp from me to any
ipfw add allow udp from any to me
```

Ce qui ajoute deux règles de niveau 1000 et 2000, que nous pouvons d'ailleurs voir à l'aide de **ipfw list**.

Concernant la question posée, sur la sécurisation du protocole UDP par le pare – feu, nous serions d'avis soit de poser une question plus précise, soit nous répondrions que oui si nous enlevons une des deux règles l'UDP ne passe plus et en somme ce serait comme s'il était sécurisé, mais que non par rapport à la sécurité clair/chiffré le pare – feu ne sécurise pas pour autant le contenu des messages transmis par l'UDP. Mais le pare – feu peut resteindre la discussion UDP seulement aux deux serveurs, par exemple :

ipfw add allow udp from 10.0.42.1 to me
ipfw add allow udp from me to 10.0.42.1

Ceci réduirait les chances que quelqu'un d'autres tente une communication UDP vers notre serveur ou inversement. Cela ne change pas le fait que les messages passent en clair.

Etape C : Mise en place d'un VPN IPSec en mode transport

Les machines étant des machines avec IPSEC, nous faisons confiance en notre professeur et sommes assurés qu'il contient bien ce qu'il faut pour IPSEC.
Nous allons donc mettre en place un VPN en mode transport, c'est à dire avec utilisation de AH au dessus du protocole ESP. Il faut donc configurer les deux serveurs, puis ajouter pour chaque serveur les SP (Security Policy) qui sont des règles mises en place pour accepter certaines communications.

Pour les deux serveurs nous avons :

```
# setkey -c
 add 10.0.42.1 10.0.42.2 ah 1000 -m transport -A hmac-sha1 "ANTICONSTITUTIONELLE" ;
 add 10.0.42.1 10.0.42.2 esp 2000 -m transport -E des-cbc "MOTPASSE" ;
 add 10.0.42.2 10.0.42.1 ah 3000 -m transport -A hmac-sha1 "ANTICONSTITUTIONELLE" ;
 add 10.0.42.2 10.0.42.1 esp 4000 -m transport -E des-cbc "MOTPASSE" ;
```

Puis pour le Serveur A :

```
 A# setkey -c
  spdadd 10.0.42.1 10.0.42.2 any -P out ipsec
  esp/transport/10.0.42.1-10.0.42.2/require ;
  spdadd 10.0.42.1 10.0.42.2 any -P out ipsec
  ah/transport/10.0.42.1-10.0.42.2/require ;
```

Et finalement le serveur B :

```
 B# setkey -c
  spdadd 10.0.42.2 10.0.42.1 any -P out ipsec
  esp/transport/10.0.42.2-10.0.42.1/require ;
  spdadd 10.0.42.2 10.0.42.1 any -P out ipsec
  ah/transport/10.0.42.2-10.0.42.1/require ;
```

Nous enlevons les anciennes règles du pare – feu sur les paquets UDP, et n'acceptons que les paquets du protocole ESP, ceci permet donc de n'accepter que les paquets du protocole ESP, paquets qui de toute manière sont chiffrés avec une clé secrète.
Nous procédons donc ainsi :

```
ipfw list
```

```
ipfw delete 1000
ipfw delete 2000
ipfw add allow esp from me to any
ipfw add allow esp from any to me
```

La communication entre A et B est donc sécurisée puisque A s'authentifie à B (protocole AH), après quoi ils échangent leurs messages avec chiffrement par le protocole ESP.
Le service UDP bénéficie de cette sécurité, puisque les trames que nous avons capturées ne rendent pas compte du contenu des messages. Et comme nous avons enlevé l'UDP du pare – feu pour ne passer que par l'ESP, c'est donc que les paquets UDP résultants de la commande nc -ul 1234 passent bien par le protocole ESP qui est reçu à l'autre bout du tunnel.

Cf. Image en pièce jointe pour les captures de trames. **01-CaptureIPSecTransport.PNG**.

Etape D : Communication entre deux réseaux locaux

Nous utilisons à nouveau deux machins BSD/IPSec pour le TP.
L'une des machines cliente se nommera cliA, avec pour IP 10.0.42.66, et l'autre cliB, avec pour IP 10.0.42.194. Évidemment elles sont chacune reliées comme expliqué sur le schéma du début de TP.
Les interfaces sont en NAT, et nous ajoutons dans le fichier /etc/rc.conf la ligne suivante :

```
defaultrouter = "10.0.42.65" # ou 10.0.42.193 pour le cliB
```

Les serveurs ont chacun une carte supplémentaire en NAT, nous leur ajoutons donc une IP statique :

```
ifconfig_lnc1 = "inet 10.0.42.65/26" # ou 10.0.42.66/26 pour le serveur B
```

Il faut veiller à ce que les serveurs fassent bien passerelle vers un autre réseau, pour cela nous ajoutons les lignes suivantes aux serveurs A et B :

```
gateway_enable = "YES"
defaultrouter="10.0.42.2" # ou 10.0.42.1 pour le serveur B
```

L'IP Forwarding doit être activé, pour cela nous ajoutons une ligne dans le fichier /etc/sysctl.conf :

```
net.inet.ip.forwarding=1
```

Nous redémarrons bien évidemment toutes les machines pour assurer le bon fonctionnement de toutes ces modifications, avant cela nous n'oublions pas de décharger le module ipfw du noyau, bien qu'après redémarrage il devrait être déchargé :

```
kldunload ipfw
```

Tout devrait fonctionner, un ping entre chacune des machines devrait en être la preuve.
Le schéma du début de TP montre déjà le schéma final, espérons qu'il soit suffisant.

Etape E : Mise en place d'un VPN IPSec en mode tunnel

Tout d'abord mettons à vide le SP (Security Policy et les règles que nous avions mises en place tout à l'heure. Pour cela nous tapons :

setkey -FP && setkey -F
setkey -D #pour voir s'il reste toujours quelque chose

Ensuite nous tapons ceci :
sur le serveur A :
A# setkey -c
add 10.0.42.2 10.0.42.1 esp 2000 -m any -E des-cbc "MOTPASSE" ;
add 10.0.42.1 10.0.42.2 esp 4000 -m any -E des-cbc "MOTPASSE" ;
spdadd 10.0.42.64/26 10.0.42.192/26 any -P out ipsec esp/tunnel/10.0.42.64-10.0.42.192/require ;
spdadd 10.0.42.192/26 10.0.42.64/26 any -P in ipsec esp/tunnel/10.0.42.192-10.0.42.64/require ;

sur le serveur B :
B# setkey -c
add 10.0.42.2 10.0.42.1 esp 2000 -m any -E des-cbc "MOTPASSE" ;
add 10.0.42.1 10.0.42.2 esp 4000 -m any -E des-cbc "MOTPASSE" ;
spdadd 10.0.42.64/26 10.0.42.192/26 any -P in ipsec esp/tunnel/10.0.42.64-10.0.42.192/require ;
spdadd 10.0.42.192/26 10.0.42.64/26 any -P out ipsec esp/tunnel/10.0.42.192-10.0.42.64/require ;

Les captures de trames sont jointes avec le courriel, elles ont pour nom :
- 02-netcastIPSecTunnel.PNG
- 03-pingIPSECTunnel.PNG

Les captures d'écran prouvent à la fois, qu'après utilisation des commandes précédentes, le client A peut pinguer le client B, qu'ils peuvent discuter en UDP avec netcat, mais également que ce que Wireshark capture sont des paquets dont l'origine et la destination sont 10.0.42.1 et 10.0.42.2, ce qui ne donne donc pas la réelle destination du paquet ! Par ailleurs la sécurité est prise en compte pour l'ensemble des services et paquets transitant entre les deux réseaux, une solution de choix !

Par ailleurs, pour éviter de configurer les clés manuellement, il faudrait mettre en place un IKE, ou Internet Key Exchange, de cette façon nous devrions donc créer un tunnel sécurisé à clé privé et clé publique, le temps d'échanger la clé secrète entre les deux serveurs. Cette clé secrète pourrait alors changer fréquemment, avantage pour la sécurité globale du VPN.

Concernant le pare – feu, forcément nous mettrions ipfw ! Pour empêcher un ping il faut bloquer l'icmp, pour empêcher le tracroute, nous bloquerions le TCP. Et donc il faudrait autoriser l'UDP. Pour cela nous ferions :

```
ipfw add allow esp from me to any
ipfw add allow esp from any to me
ipfw add allow udp from me to any
ipfw add allow udp from any to me

ipfw add deny icmp from me to any
ipfw add deny icmp from any to me
ipfw add deny tcp from me to any
ipfw add deny tcp from any to me
```

Ce devrait alors fonctionner.

Conclusion

Finalement l'utilisation d'un VPN permet de joindre deux réseaux en mettant un point d'honneur à la sécurité. Par diverses couches et protocoles de transport, chiffrement et authentification il permet de répondre aux mieux aux besoins de la sécurité, avec ce petit plus qui est que même après réception d'un paquet avant son arrivée nous ne sachions pas à qui est destiné le paquet ni ce qu'il contient. Contrairement à SSH qui ne s'occupe que de chiffrer le message mais pas les destinataires, VPN sur IPSec est un plus puisque ni les destinataires ni le message ne sont visibles en clair!
Cependant la mise en place est compliquée quand nous ne connaissons pas, elle passe mieux une fois quelques tests réussis et un résultat satisfaisant.