Sécurité des réseaux informatiques TP : Synthèse

1 IPTABLES

1.1 Commandes utiles

- Démarrer un service :
  # service [nom du service] start
  Exemple :
  # service httpd start
- Visiter un site web en mode texte :
  # links [ip]
  Si un logiciel nécessaire n'est pas installé :
  # yum install [links | lynx | httpd | nom_du_logiciel]
- Configuration réseau en mode « graphique » :
  # system-config-network
  # service network restart ;pour appliquer les nouveaux paramètres
- Afficher les informations sur les paquets IP :
  # tcpdump –p ip proto 1
- Ne sniffer que ce qui arrive sur un port en particulier (ici : le 80) :
  # tcpdump –p port 80
- Sauvegarder la configuration des iptables pour le prochain redémarrage du service :
  # service iptables save
- Fichiers des associations de protocoles avec leur port :
  /etc/services
- Remettre à jour la liste des interfaces présentent sur CENTOS :
  # kudzu.
- Installer un script shell (.sh) :
  ./ « nom_du_script »
- Configurer une interface :
  # ifconfig eth0 « Add IP/Mask»
- Description et capture des paquets passant par l'interface eth0 sans résolution de nom et utilisant le protocole ICMP (1) :
  # tcpdump –i eth0 –n ip proto 1

1.2 Description

C'est un logiciel libre utilisé pour le protocole IPv4 afin de configurer les règles dans le pare-feu. Il est très basique et ressemble pas mal à d'autres outils du même genre (CheckPoint). Il n'est pas intéressant pour les grandes entreprises car lorsqu'il y a trop de règles, cela devient le foutoir. Il souffre du manque de standards. Il faut plusieurs outils pour créer et configurer des règles.

Il existe deux modes de fonctionnement :

1. Une machine linux pouvant envoyer et recevoir des données. Tous les services de base (RPC, NFS…) sont accessibles par quelqu'un qui accède à notre machine par le réseau (Add IP Publique). Il est donc recommandé d'arrêter tous les services inutiles (Ex : Si serveur mail, désactiver Web, DNS, DHCP…). Il faut également protéger les services actifs sur la machine qui va se connecter au réseau. Pour cela, on utilise un firewall.

Policy : - C'est la règle par défaut (la dernière prise en compte). ACCEPT pour iptable.
- Firewall : Lorsque l'on parle de policy, on parle de l'ensemble des règles du firewall.

Lorsque le firewall route des paquets, les chaines INPUT et OUTPUT sont inutiles. Ces services sont utilisés pour limiter l'accès aux données de la machine. L'ordre dans lequel on entre les différentes chaines de règles à une importance. En effet, lorsqu'un paquet est inspecté iptables va parcourir sa liste dans l'ordre où elles sont arrivées et dès qu'une chaine correspondante est trouvée, va l'exécuter et sortir de sa recherche de chaines. Ex : Si on bloque tous les paquets SSH en première règle et qu'ensuite on spécifie les adresses IP pouvant utiliser SSH, iptables lira la première chaine et verra qu'il doit rejeter les paquets SSH. Il n'ira donc jamais voir la deuxième règle.

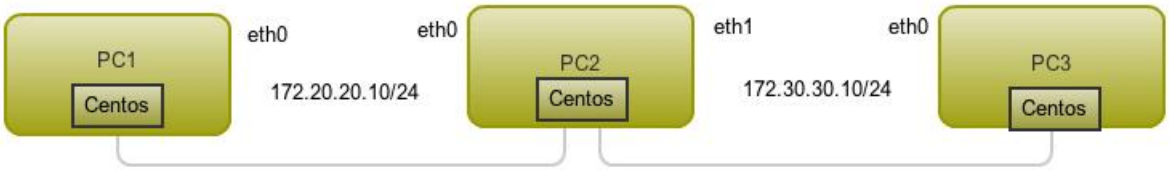
1.3 Commandes

- Voir le manuel contenant les commandes iptable :
  # man iptables
- Voir la configuration actuelle du firewall (Voir les policies appliquées sur les filtres) :
  # iptables –L
- La commande générale pour modifier les policies par défaut est :
  # iptables –P [INPUT | OUTPUT | FORWARD] [DROP ^ BLOCK]
  ➔ Le DROP signifie que si le paquet ne convient pas, on le jette sans prévenir. Le BLOCK est pareil, mais on prévient l'expéditeur. Dans la mesure du possible, il faut faire un BLOCK plutôt qu'un DROP dans un réseau local. Le DROP se fait sur l'interface externe du firewall, reliée à Internet.
- Une règle minimal (qui accepte tout) :
  # iptables –A INPUT –j ACCEPT
  ➔ -A et –j étant obligatoire.
- Pour retirer une règle que l'on a écrite plus tôt :
  # iptables –D OUTPUT –p icmp –j ACCEPT
  ➔On remplace le « A » par un « D ».
- Pour supprimer toutes les règles qui ont été définies précédemment (remise à zéro):
  # iptables –flush ou # iptables –F ou /etc/init.d/iptables stop
- Pour remettre le service DHCP :
  # service network restart
- Autres paramètres utilisables avec iptables :
  -i [input interface]
  -o [output interface]
  -s [source IP]
  -d [destination IP]
  --sport [source port]
  –-dport [destination port]

1.4 Exemple de configuration :

1. On commence par bloquer tout ce qui passe en entrée et en sortie sur l'interface réseau.
   # iptables -P INPUT DROP
   # iptables -P OUTPUT DROP
2. Ensuite, nous acceptons les « pings » provenant d'autres ordinateurs :
   # iptables -A INPUT -p icmp -j ACCEPT
3. Malheureusement, le « ping » ne pourra pas repartir. Pour qu'il puisse repartir, il faudra également ajouter :
   # iptables -A OUTPUT -p icmp -j ACCEPT
4. Il faut également autoriser les réponses sortantes pour les connexions entrantes acceptées précédemment (Statefull Packet) :
   # iptables -A OUTPUT -m state --state RELATED,ESTABLISHED –j ACCEPT
5. Nous devons ensuite autoriser les autres ordinateurs à accéder au serveur Web :
   # iptables –A INPUT –p tcp --dport 80 –j ACCEPT
   ➔ --dport est un paramètre supplémentaire pour –p et ne peut être utilisé qu'avec TCP et UDP.
6. On peut également autoriser notre propre machine linux à accéder au serveur Web mais cela n'a pas vraiment d'intérêt :
   # iptables –A OUTPUT –p tcp --sport 80 –j ACCEPT
7. Si on veut activer les requêtes DNS depuis notre machine linux :
   # iptables –A OUTPUT –p udp --dport 53 –j ACCEPT
8. Si on veut activer les requêtes DNS depuis l'extérieur :
   # iptables –A INPUT –p udp --dport 53 –j ACCEPT
9. Enfin, il est possible d'activer la prise de contrôle en SSH depuis une machine spécifique : (Évidemment, tous les autres exemples peuvent être pratiqués avec une machine spécifique).
   # iptables -A INPUT -p tcp --dport 22 -s 10.4.35.23 -j ACCEPT

1.5 Routage et transit sous Unix



Tous les PCs doivent pouvoir se pinguer. Pour cela, plusieurs choses sont nécessaires :

1. On doit rajouter une route entre PC1 et PC2 :
   # route add –net 172.30.30.0/24 gw 172.20.20.10
2. On doit rajouter une route entre PC3 et PC2 :
   # route add –net 172.20.20.0/24 gw 172.30.30.10
3. Pour voir les réseaux connectés à la machine, on rentre :
   # netstat -rn
4. Afin que tout fonctionne correctement, il faut que le routage sur PC2 soit actif :
   # cat /proc/sys/net/ipV4/ip_forward
   ➔ Il nous donnera la valeur de 0 si le routage n'est pas actif et de 1 s'il l'est.
5. Si le routage n'est pas actif, il faut l'activer :
   # echo 1 > /proc/sys/net/ipV4/ip_forward
6. Ensuite, on peut créer les règles sur le PC2 (Firewall) afin de permettre uniquement le trafic souhaité :
   # iptables –P FORWARD DROP
   ➔ Plus rien ne peut transiter.
   # iptables –A FORWARD –p icmp –i eth0 –j ACCEPT
   ➔ Règle autorisant PC1 à envoyé des echo request vers PC3. PC3 ne pouvant actuellement pas répondre.
   # iptables –A FORWARD –p icmp –i eth1 –m state –state RELATED,ESTABLISHED –j ACCEPT
   ➔ Permet à PC3 de répondre aux echos request. Utilisation du « Statefull packet ».

2 NAT

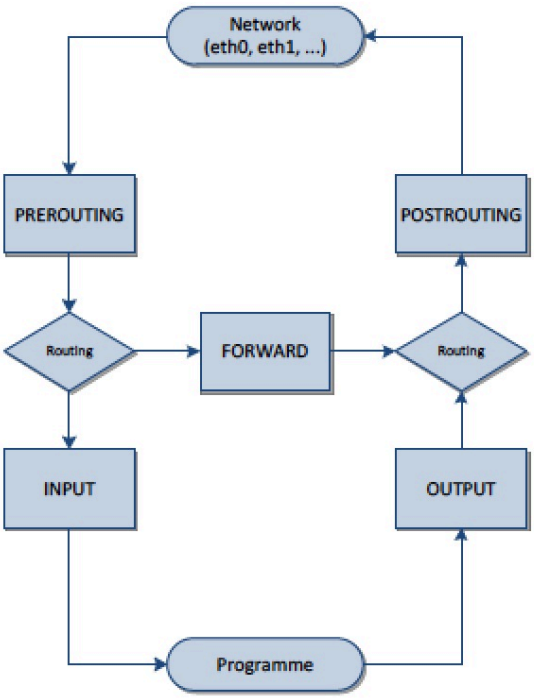
2.1 Description

Pour configurer du NAT sur CENTOS, on utilise la commande « -t nat ». La traduction d'adresse est gérée avec les chaînes PREROUTING, POSTROUTING et OUTPUT. Dans la chaîne PREROUTING (avant routage), on ne peut modifier que l'adresse de destination. L'adresse source est conservée. On fait donc du DNAT. On dit qu'on fait du "NAT destination". Dans la chaîne POSTROUTING, (après routage) on ne peut modifier que l'adresse source. L'adresse de destination est conservée. On fait donc du SNAT. On dit qu'on fait du "NAT source".

De l'intérieur du LAN vers l'extérieur : POSTROUTING.
De l'extérieur vers le LAN : PREROUTING.
INPUT et OUTPUT sont pour le firewall. Lorsque l'on ne fait que le traverser, c'est du FORWARD.

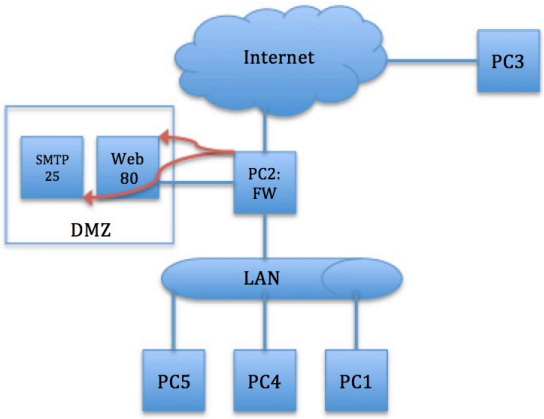
2.2 Commandes

- Pour voir les policies appliquées à propos du NAT :
  # iptables –t nat -L
- Permet de masquer l'IP source avec eth1 comme interface sortie du routeur (NAT effectué avant d'aller sur Internet et après le routage interne) :
  # iptables -t nat -A POSTROUTING -o eth1 -j MASQUERADE
  ➔ L'astuce est de ne pas fournir d'interface de sortie. Cela permettra d'activer le masquage pour tous les paquets natés.
- Il est possible de spécifier nous même l'IP source en plus, grâce au SNAT :
  # iptables -t nat -A POSTROUTING -o eth1 -j SNAT --to-source 172.30.30.X
- L'option « to-source » permet de spécifier un port. Elle permet aussi de spécifier un interval d'IP ou de ports :
  # ... --to-source 172.30.30.140-172.30.30.150:22-80
- Résumé et ordre de passage des paquets dans les différentes chaines de iptables :



2.3 Exemple de configuration

Le but de cet exemple est depuis un PC situé sur Internet (PC3), accéder au service tournant sur une machine se trouvant dans le LAN en passant par un firewall faisant du NAT. Sur le schéma, les flèches rouges montrent quels ports sont ouverts sur le Firewall.



1. On démarre le service web sur le PC1 :

service httpd start

2. On configure le PC2 (Firewall) pour qu'il redirige tout ce qui lui arrive sur son interface eth1 et qui est destiné au port 80 vers le PC1 :

iptables -t nat -A PREROUTING -i eth1 –p tcp –dport 80 –j DNAT –-to 172.20.20.X(PC1)

➔ Cela doit être fait car PC3 ne connaît que l'adresse IP de PC2 et envoi donc ces paquets sur cette IP. Or, il faut que les requêtes de PC3 arrivent sur PC1. On NAT donc l'adresse de destination (PC2 => PC1).

3. Ensuite, on peut vérifier que les requêtes de PC3 vont bien jusque PC1 :

tcpdump –np port 80

➔ On peut également regarder le trafic internet d'un seul hôte :

tcpdump –np port 80 and host 172.20.20.X

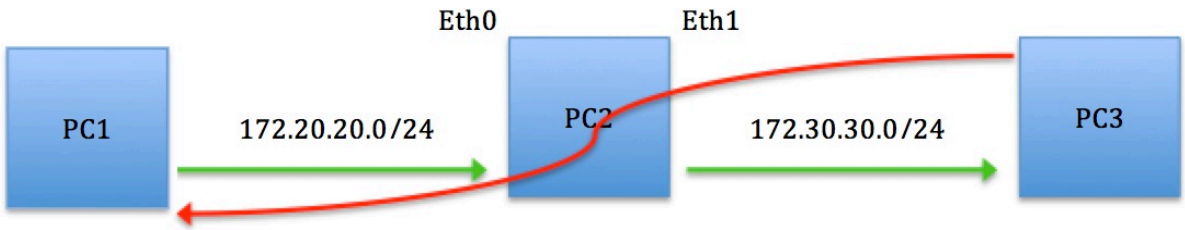
4. Il faut maintenant autoriser les paquets que PC3 a envoyés :

iptables -A FORWARD –p tcp –-dport 80 –i eth1 –j ACCEPT

5. Et autoriser les réponses que lui envoi PC1 :

iptables –A FORWARD –m state –-state RELATED,ESTABLISHED –j ACCEPT

2.4 Exemple type EXAMEN :



I. PC1 est un serveur web et peut donc être accédé par n'importe qui.

1. Autoriser le routage sur PC2 :

cat /proc/sys/net/ipV4/ip_forward

echo 1 > /proc/sys/net/ipV4/ip_forward

2. Bloquer tout le trafic (PC2 également) /!\ Seulement par écrit, pas dans les TPs ! :

iptables –P INPUT DROP

iptables –P OUTPUT DROP

iptables –P FORWARD DROP

3. Configurer PC2 :

   a. Configuration du NATing :

   # iptables –t nat –A PREROUTING –i eth1 –p tcp –-dport 80 –j DNAT --to 172.20.20.X (IP PC1).

   b. Autoriser les flux vers le serveur web de PC1. Les réponses ne sont pas encore autorisées :

   # iptables –A FORWARD -d 172.20.20.X (IP PC1) –p tcp –-dport 80 –j ACCEPT.

   c. Autoriser les réponses en générales :

   d. # iptables –A FORWARD –m state –-state RELATED,ESTABLISHED –j ACCEPT.

II. PC1 est un client web et peut donc accéder à n'importe quel site web. ➔ DNS naté + HTTP.

1. Autoriser le routage sur PC2 :

```
# cat /proc/sys/net/ipV4/ip_forward
# echo 1 > /proc/sys/net/ipV4/ip_forward
```

2. Bloquer tout le trafic (PC2 également) /!\ Seulement par écrit, pas dans les TPs ! :

```
# iptables –P INPUT DROP
# iptables –P OUTPUT DROP
# iptables –P FORWARD DROP
```

3. Configurer PC2 :

   a. Configuration du NATing :

   ```
   # iptables –t nat –A POSTROUTING –p tcp –-dport 80 –s (source) 172.20.20.0/24 –j MASQUERADE.
   # iptables –t nat –A POSTROUTING –p tcp –-dport 80 –o (interface de sortie) eth1 –j MASQUERADE.
   ```

   ➔ L'option MASQUERADE permet à toutes les adresses IP privées d'utiliser la même IP publique. L'adresse IP du routeur sera la seule utilisée comme adresse IP externe. C'est en fait la manière de faire du SNAT (Source NATing). On rentrera la commande SNAT lorsqu'il sera question d'IP logiques. (#ifconfig eth0:1 172.30.30.2/24

   b. Configuration du DNS :

   ```
   # iptables –t nat –A POSTROUTING –p udp –-dport 53 –s (source) 172.20.20.0/24 –j MASQUERADE.
   # iptables –t nat –A POSTROUTING –p udp –-dport 53 –o (interface de sortie) eth1 –j MASQUERADE.
   ```

   c. Pour les deux commandes précédentes, on remplace l'IP de PC1 par l'IP de Eth1 de PC2.

2.5 Quelques liens utiles

http://www.inetdoc.net/guides/iptables-tutorial/targets.html
http://fr.wikipedia.org/wiki/Iptables

3 Monitoring/Scanning réseau

3.1 Nmap

3.1.1 Le manuel de Nmap

Le manuel décrit les différentes fonctions que Nmap peut remplir et donne également les commandes utilisables et leurs fonctions. Plus loin, il décrit les différentes façons de travailler avec Nmap. Nmap est un scanner de ports open source conçu pour détecter les ports ouverts, identifier les services hébergés et obtenir des informations sur le système d'exploitation d'un ordinateur distant.

3.1.2 Commandes

- Voici un exemple de commande nmap. Elle permet d'effectuer un scan sur TCP d'une machine distante. –sT spécifie un scan TCP. –p spécifie une liste de port à scanner, ici tous. On aurait pu effectuer un scan sur le réseau en entier en indiquant l'adresse réseau ainsi que son masque derrière.
  # nmap –sT –p 1-65535 « IP »
- Permet d'effectuer un scan UDP.
  # nmap –sU « IP »
- Désactive la résolution DNS inverse
  # nmap –n « IP »
- Permet de scanner la plage d'adresses IP passée en paramètre et lister toutes les machines. Il indiquera à côté de chaque adresse IP, celles qui ont été scannées et celles qui ne l'ont pas été.
  # nmap -sL « ip »-« mask »
- Permet de pinguer la plage d'adresses IP passée en paramètre et indiquer à côté de chaque adresse, celles qui sont UP et celle qui sont DOWN.
  # nmap -sP « ip »-« mask »
- Permet de scanner la machine afin de connaître le système d'exploitation qui tourne sur la machine. Grâce à cette commande, il nous montre aussi combien de ports sont fermés ainsi que lesquels sont ouverts. Dans mon cas, le TCP 22 (SSH), TCP 111 (rcpbind) et TCP 841 (Inconnu) sont ouverts. Il sait dire que c'est un linux mais pas exactement quelle version.
  # nmap -O « ip »
- Il est possible, si Nmap ne parvient pas à déterminer le système d'exploitation, de lui demander une liste de systèmes probablement exécutés sur la machine distante.
  # nmap –O –oscan-guess « IP »
- La méthode de « FIN » scan consiste à envoyer un paquet avec l'indicateur FIN, c'est-à-dire une demande de coupure de connexion. Comme aucune connexion n'est en cours, ce paquet n'a pas lieu d'être. Le système doit renvoyer un reset si le port est fermé et ignorer ce paquet si le port est ouvert.
  # nmap -sF -v "IP"
- La méthode « XmasTree » scan permet d'envoyer un paquet TCP avec tous ses flags activés à une machine distante. Ensuite, si la machine distante à son port fermé, elle répondra par un RST (Permet de rejeter la connexion). Si la machine distante à son port ouvert, elle ne répondra pas. Cela permet de passer les firewalls non-stateful. Heureusement, de nos jours ce type de paquet est directement détecté par les IDS.
  # nmap -sX -v « IP »

- La méthode « NULL » scan permet d'envoyer un paquet qui, ne comporte aucun indicateur. Le système distant doit se comporter comme dans le cas précédent, sauf s'il s'agit d'un Windows.
  # nmap -sN -v "IP"
- Quelques ports connus :
  - 111 : RPC (Uniquement disponible sous windows).
  - 139 : netbios
  - 445 : smb over TCP.

3.1.3 Conclusion

Nmap envoi des paquets SYN sur le port d'un hôte pour des requêtes TCP. Lorsqu'il reçoit un SYN ACK, il sait que le port de la machine est ouvert. Lorsqu'il ne reçoit rien, c'est qu'il y a un firewall entre la machine exécutant Nmap et la machine cible. Si il obtient un reset, c'est que le port de la machine hôte est fermé. Lorsque l'hôte répond avec un SYN ACK, on peut essayer de trouver quel système d'exploitation tourne sur la machine cible.
Il envoi des SYN mais pas de ACK pour ne pas être loggué sur le serveur. Dans ce cas là, il peut rester discret.

3.2 Nessus

3.2.1 Description

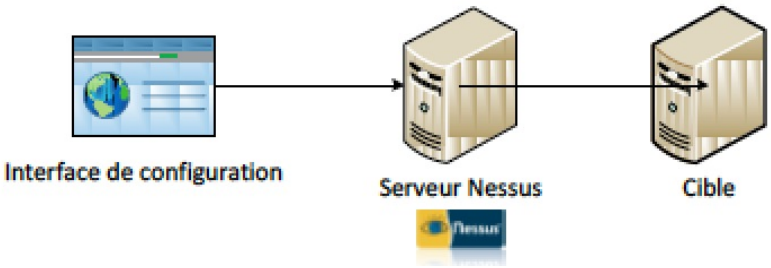
Nessus est un outil de sécurité informatique. Il permet de scanner des hôtes, mais également de détecter des vulnérabilités à propos des services installés. Pour ce faire, il se base sur une base de connaissance de vulnérabilités connues. Ceci inclut, entre autres :

- les services vulnérables à des attaques permettant la prise de contrôle de la machine, l'accès à des informations sensibles (lecture de fichiers confidentiels par exemple), des dénis de service...
- les fautes de configuration (relais de messagerie ouvert par exemple)
- les patchs de sécurité non appliqués, que les failles corrigées soient exploitables ou non dans la configuration testée
- les mots de passe par défaut, quelques mots de passe communs, et l'absence de mots de passe sur certains comptes systèmes. Nessus peut aussi appeler le programme externe Hydra pour attaquer les mots de passe à l'aide d'un dictionnaire.
- les services jugés *faibles* (on suggère par exemple de remplacer Telnet par SSH)
- les dénis de service contre la pile TCP/IP

3.2.2 Fonctionnement

Nessus détecte les machines vivantes sur un réseau, balaie les ports ouverts, identifie les services actifs, leurs versions, puis tente diverses attaques.

Nessus se divise en deux parties : *nessusd* qui est un daemon (service) exécutant les requêtes ainsi que la communication avec la cible, et *nessus*, une application client qui récupère les données et affiche le résultat. Ce découpage est classique, le daemon tournant avec des privilèges élevés (root) alors que l'interface graphique, plus complexe et donc vulnérable, tourne sous l'identité d'un utilisateur non privilégié. Le rôle de client peut être réalisé par n'importe quel navigateur web (par défaut : Internet Explorer). Ce client se connecte à l'interface de configuration du serveur Nessus via le port 8843.



3.3 Snort

3.3.1 Description

Snort est un système de détection d'intrusion (NIDS). Il est capable d'effectuer en temps réel des analyses de trafic et de logger les paquets sur un réseau IP. Il peut effectuer des analyses de protocole, recherche/correspondance de contenu et peut être utilisé pour détecter une grande variété d'attaques et de sondes comme des dépassements de buffers, scans, attaques sur des CGI, sondesSMB, essai d'OS fingerprintings et bien plus. Pour effectuer ces analyses, il se fonde sur des règles.

Snort possède trois modes de fonctionnement :
1. **Mode écoute :** il fonctionne comme un sniffer pour observer le trafic en temps réel.
2. **Mode Log :** il permet de capturer et enregistrer les paquets dans un fichier de log, en spécifiant des règles de filtrage.
3. **Mode IDS :** snort se base sur un ensemble de règles pour détecter des schémas de trafic et générer des alertes.

1.1 **Mode écoute :**
Pour afficher les en-têtes de paquets TCP/IP, utiliser la commandes ./snort -v. Pour afficher en plus les informations de la couche applicative, utilisez : ./snort -vd. L'option e permet également d'obtenir les informations de la couche Liaison de Données.
1.2 **Mode log :**
Pour logger les paquets, il suffit d'indiquer le répertoire où il faut stocker les fichiers de log, avec l'option -l : ./snort -dev -l ./log. Il peut être également utile d'informer le logiciel de l'adresse du réseau local. Cela s'effectue avec la commande -h.
1.3 **Mode IDS :**
Pour lancer snort en mode détection d'intrusion, il faut lui donner un fichier de configuration en paramètre. Cela s'effectue simplement avec la commande suivante : ./snort -dev -l ./log -h 192.168.1.0/24 -c /etc/snort/snort.conf. Le fichier /etc/snort/snort.conf est un exemple de fichier de configuration sur lequel vous pouvez vous baser. N'hésitez pas à aller y jeter un coup d'oeil. Le fichier de configuration indique à snort où trouver les règles sur lesquelles il va se baser pour effectuer ses analyses. Voici un exemple de règle : alert tcp any any -> 192.168.1.0/24 111 (content:"|00 01 86 a5|"; msg: "mountd access";).
Cette règle définit d'abord une action (ici, alert), puis, le protocole transport des paquets à considérer. Ensuite, de part et d'autre de la flèche se trouvent la source et la destination du segment tcp. Dans ce cas-ci, la source peut être n'importe quelle adresse IP, et n'importe quel numéro de port. Par contre, la règle s'applique aux paquets à destination du réseau 192.168.1.0/24, sur le port 111. Ensuite, entre parenthèses, viennent les options de la règle.
Ici, la première option « content » indique un contenu spécifique à rechercher dans chaque paquet. La seconde, msg, indique le message à afficher lorsque la règle est déclenchée.

3.3.2 Commandes

1. Il faut d'abord désactiver le Firewall et mettre toutes les règles en ACCEPT.
    - Iptables –F= efface les regles ds le fw
    - Iptables –P FORWARD ACCEPT
    - Iptables –P INPUT ACCEPT
    - Iptables –P OUTPUT ACCEPT

2. Il faut ensuite matcher tous les paquets dans ce réseau (on peut mettre any any).
   - Ip src port srcip dest port dest option(flags :tcp)
   - Regle snort : alert tcp 10.0..0.0/24 22 -> 10.1.0.2 80 (content : « Bouh » ;msg : « attaque bouh détecter » ;Sid :1000001 ;)

     Udp

     Icmp
3. Ensuite, on lance snort
   - Snort –d –l ./log –c /etc/snort/snort.conf
4. Enfin, on peut observer les alertes émises par snort dans le dossier de log créé précédemment.
5. Exemple d'alerte snort.
   - alert tcp 10.1.0.2/24 80 -> 10.2.0.2 80 (content : »text/html » ;msq : »n importe » ;sid :1000001)

3.3.3 Astuces

- Ajouter le répertoire « C:\snort\bin\ » au PATH de Windows. Cela permettra de travailler plus facilement avec cet utilitaire en ligne de commande.

  Ordinateur > Propriétés > Paramètres systèmes avancés > Variables d'environnement > Path > Modifier > Ajouter : » ;C:\snort\bin\ » à la fin de la chaine.

3.4 Commandes utiles

- Obtenir l'adresse MAC d'un hôte via son IP :

nbtstat -a (Windows)

nmblookup -a (Linux)

- Afficher les ports actifs sur sa machine :

netstat –n (Windows)

netstat –a (Linux)

- Il est bon de rappeler qu'il existe maximum 65534 ports car le port 0 est inutilisé. Les ports 0 à 1024 sont utilisés par des services connus et sont donc réservés.

3.5 Quelques liens utiles

http://nmap.org/book/man.html ou http://nmap.org/man/fr/

http://blog.nicolargo.com/2007/08/nmap-le-scanneur-de-reseau.html

http://fr.wikipedia.org/wiki/Nmap

http://static.tenable.com/documentation/nessus_4.4_user_guide.pdf

http://fr.wikipedia.org/wiki/Nessus_%28logiciel%29

http://fr.wikipedia.org/wiki/Snort

4 Certificats

4.1 SSL

SSL : Secure Sockets Layer = TLS Transport Layer Security
Un certificat SSL est utilisé pour identifier une entité physique ou morale mais également pour chiffrer des échanges. La création de certificats utilise l'outil « openssl » sous Centos.
Il y a trois critères pour vérifier le certificat :

- **Date** : Validité temporelle
- **Nom** : CN=…
- **Trust** : Signé par une organisation connue et de confiance.
- **(CRL = Certificate Revocation List)** : C'est la liste des certificats qui ont été révoqués ou qui ne sont plus valables et qui ne sont donc plus digne de confiance. Parfois, les CRL provoquent un délai de propagation des informations de révocations.

Le but du certificat est de fournir de la sécurité à plusieurs niveaux :

- Authentification du serveur
- Confidentialité des données échangées (sessions chiffrées)
- Intégrité des données échangées
- Authentification (forte) du client avec utilisation du certificat ➔ Optionnel
- Transparence

On protège donc une clé avec un mot de passe. Clé qui sert elle-même à protéger le certificat.

4.1.1 Commandes

1. Création de la clé privée RSA qui servira à valider le certificat. Cette commande va générer un fichier contenant une clé de 1024 bits chiffré avec DES3.
   openssl genrsa –des3 –out Server.key
2. Il faut ensuite entrer un mot de passe
   centos
3. Création du fichier contenant les informations sur le propriétaire du certificat. C'est la clé publique qui sera envoyée aux machines souhaitant se connecter à notre serveur.
   openssl req –new –key Server.key –out Server.csr
4. On doit ensuite entrer plusieurs informations.
   - *On remet la clé privée : centos*
   - *On indique le pays avec 2 lettres : BE*
   - *On indique la province : Brabant-Wallon*
   - *On indique la ville : Louvain-la-Neuve*
   - *On indique la compagnie : ephec*
   - *On indique le nom de l'organisation de section : TI*
   - *On indique le Common Name qui peut être son propre nom : Lamarche*
   - *On indique son adresse E-mail : test@test.com*
   - *On ajout un mot de passe standard : test1*
   - *On ajoute un nom de compagnie (optionnel) : RDV*

5. Génération du certificat. Il sera chiffré à l'aide de notre clé privée crée précédemment. Il sera valable 365 jours.
   openssl x509 –req –days 365 –in Server.csr –signkey Server.key –out Server.crt
6. On insère ensuite le mot de passe de la clé secrète.
   centos

4.1.1.1 Association à un serveur Apache

7. Il faut ensuite vérifier que les fichiers de configuration SSL pour apache sont présents et dans ce cas, les configurer. Ils se trouvent dans /etc/httpd/conf/httpd.conf ou /etc/httpd/conf.d/ssl.conf
   SSLEngine : Il doit être sur « on »
   SSLCertificateFile : Il faut le faire pointer vers le nouveau certificat (Server.crt). Ex : /root/Server.crt
   SSLCertificateKeyFile : Il faut le faire pointer vers le fichier de la clé privée (Server.key). Ex : /root/Server.key
   ➔ *Lorsque l'on redémarrera Apache, on devra entrer le mot de passe de la clé privée (centos).* ***Service httpd restart***
8. Pour tester, il faut que l'on se connecte au serveur Apache. Pour cela, on va sur chrome, on rentre l'adresse IP de la machine UNIX avec https devant et là, on est sur notre serveur. Ex : https://10.4.35.X Ensuite, on peut voir les informations à propos de notre certificat en cliquant sur le cadenas.
9. Pour se connecter en telnet en rajoutant une couche SSL à un serveur web, il faut effectuer une commande spéciale qui permet de se mettre en mode client. Dans ces cas là, on récupère le certificat puis on fait des connexions telnet normales.
   openssl s_client –connect 10.4.35.X :443

4.1.1.2 Commandes utiles

- La fonction GET /HTTP/1.0 envoi une requête sur le service Apache qui se trouve derrière SSL.
- x509 : C'est un utilitaire de certificat. Il peut être utilisé pour afficher des informations relatives aux certificats, convertir les certificats sous diverses formes, signer les certificats ou modifier les paramètres de confiance du certificat.

4.1.2 SSL avec Reverse Proxy

1. Créer le nouveau certificat
2. Configurer le Reverse Proxy
   a) Ajouter les nouveaux chemins du certificat dans ssl.conf avec SSLCertificateFile et SSLCertificateKeyFile
   b) Ajouter à la configuration d'Apache (/etc/httpd/conf/httpd.conf)
      ProxyPass / http://ps953/
      ProxyPassReverse / http://ps953/
      Dans le cas du Reverse Proxy, il remplacera l'URL absolue par une URL relative pour permettre au client de se connecter au bon domaine (ephec.be) lorsqu'il aura une réponse 302 (demande d'authentification) du serveur.

c) Tester en entrant https://10.4.35.X dans chrome. Ensuite, il faut ajouter le certificat comme étant de confiance.

3. Changer le domaine dans lequel un cookie est actif.
   ProxyPassReverseCookieDomain ephec.local ephec.be
   [set_cookie : cookie_name=xyz123 ;domain=ephec.local ;path =/]
4.

4.4 Quelques liens utiles

http://www.linux-france.org/prj/edu/archinet/systeme/ch24s03.html
http://fr.wikipedia.org/wiki/OpenSSL