Edouard TOUZAN – Samuel LAFORGE 06/05/19

M2103 – TP n°4

Filtrage avec IpTables

Table des matières

1. Filtrage sans état

1) On utilise la commande **iptables -L** :

```
 # iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     udp  --  anywhere             anywhere             udp dpt:domain
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:domain
ACCEPT     udp  --  anywhere             anywhere             udp dpt:bootps
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:bootps

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  anywhere             192.168.122.0/24     ctstate RELATED,ESTABLISHED
ACCEPT     all  --  192.168.122.0/24     anywhere
ACCEPT     all  --  anywhere             anywhere
REJECT     all  --  anywhere             anywhere             reject-with icmp-port-unreachable
REJECT     all  --  anywhere             anywhere             reject-with icmp-port-unreachable

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     udp  --  anywhere             anywhere             udp dpt:bootpc
```

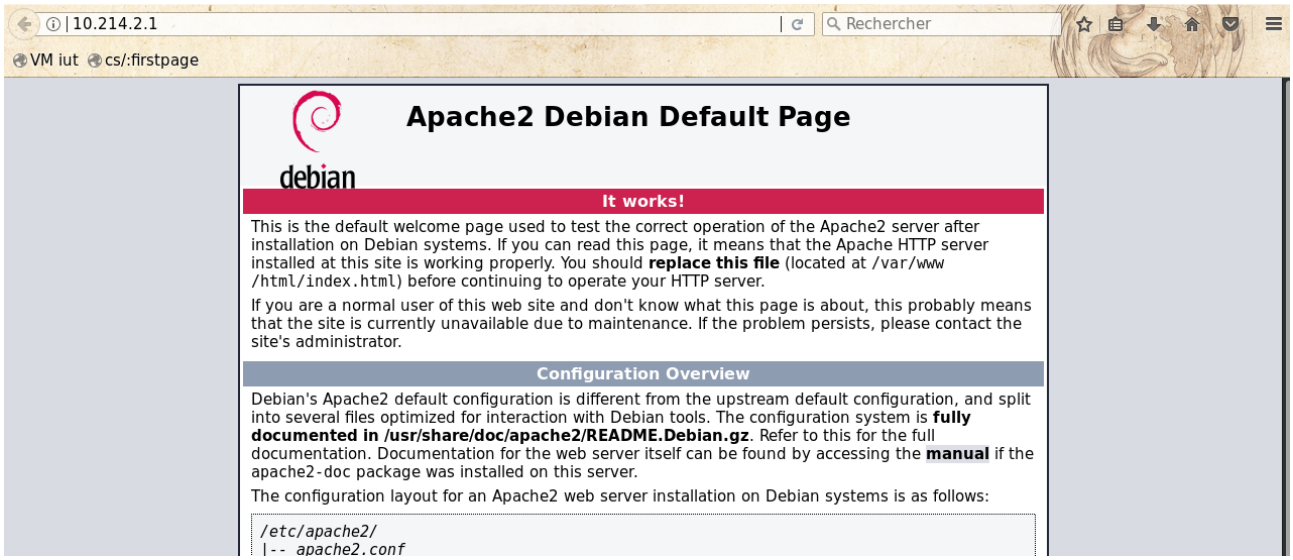
2) Pour démarrer le serveur apache sur la machine hôte :

```
# service apache2 start
```

Pour se connecter au serveur hôte depuis le client :

On se connecte au serveur en entrant l'adresse IP de la machine hôte dans un navigateur. Dans notre cas :

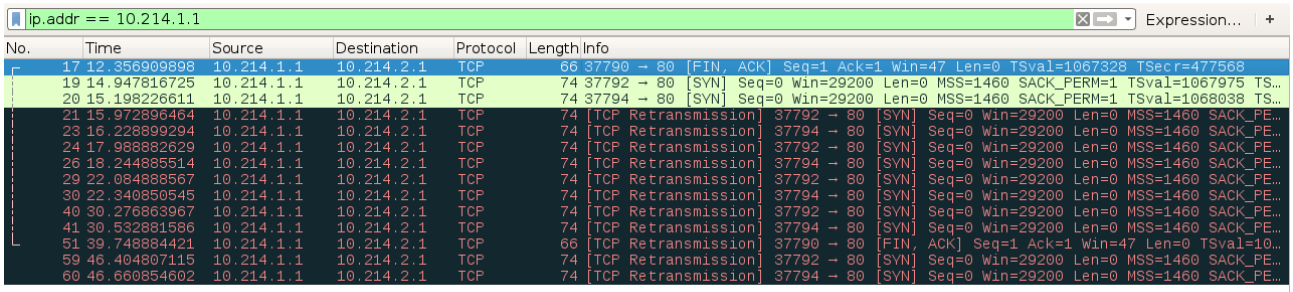
```
10.214.2.1
```



3) Pour empêcher le trafic de traverser les chaînes, on utilise les commandes suivantes :

```
# iptables -P OUTPUT DROP
# iptables -P INPUT DROP
```

En utilisant la méthode de connexion de le question 2, on ne peut plus accéder à la page depuis un client. Voici la capture Wireshark sur le poste hôte (identique au client) :



Le client envoie des requêtes au serveur, sans qu'il ne lui réponde jamais.

4) Pour accepter la réception des paquets sur le serveur apache, on utilise la commande :

```
# iptables -A INPUT -i eno -p tcp --dport 80 -j ACCEPT
```

Cette commande ne permettant ne permettant d'accepter uniquement les paquets en entrée, et pas en sortie.

M2103 – TP n°4
Filtrage avec IpTables

5) La capture de trame relevée confirme nos attentes :

```
441 402.705668110 10.214.1.1   10.214.2.1   TCP      74 37798 → 80 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=1164915 TS…
442 402.955603344 10.214.1.1   10.214.2.1   TCP      74 37800 → 80 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=1164977 TS…
443 403.717081594 10.214.1.1   10.214.2.1   TCP      74 [TCP Retransmission] 37798 → 80 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PE…
444 403.973008963 10.214.1.1   10.214.2.1   TCP      74 [TCP Retransmission] 37800 → 80 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PE…
446 405.733102781 10.214.1.1   10.214.2.1   TCP      74 [TCP Retransmission] 37798 → 80 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PE…
447 405.989077174 10.214.1.1   10.214.2.1   TCP      74 [TCP Retransmission] 37800 → 80 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PE…
454 409.925108020 10.214.1.1   10.214.2.1   TCP      74 [TCP Retransmission] 37798 → 80 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PE…
456 410.181005228 10.214.1.1   10.214.2.1   TCP      74 [TCP Retransmission] 37800 → 80 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PE…
461 418.117108410 10.214.1.1   10.214.2.1   TCP      74 [TCP Retransmission] 37798 → 80 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PE…
462 418.373081382 10.214.1.1   10.214.2.1   TCP      74 [TCP Retransmission] 37800 → 80 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PE…
478 434.245121206 10.214.1.1   10.214.2.1   TCP      74 [TCP Retransmission] 37798 → 80 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PE…
479 434.501145000 10.214.1.1   10.214.2.1   TCP      74 [TCP Retransmission] 37800 → 80 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PE…
```

Illustration 1: Capture de trames avec entrée de paquets acceptée

6) Pour autoriser les paquets sortant sur la machine serveur, on utilise la commande suivante :

```
# iptables -A OUTPUT -p tcp -j ACCEPT
```

7) On utilise la commande suivante pour interdire les connexions entrantes :

```
# iptables -A INPUT -i eno -p tcp --dport 80 -j DROP
```

J'ai aussi essayé :

```
# iptables -A INPUT -i eno -p tcp --dport 80 -j RETURN
```

et

```
# iptables -A INPUT -i eno -p tcp --dport 80 -j REJECT
```

Malgré cela, le client peut toujours se connecter au serveur.

Regardons la liste des règles actives :

```
# iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     udp  --  anywhere             anywhere             udp dpt:domain
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:domain
ACCEPT     udp  --  anywhere             anywhere             udp dpt:bootps
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:bootps
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:http
ACCEPT     tcp  --  anywhere             anywhere
DROP       tcp  --  anywhere             anywhere             tcp dpt:http
RETURN     tcp  --  anywhere             anywhere             tcp dpt:http

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  anywhere             192.168.122.0/24     ctstate RELATED,ESTABLISHED
ACCEPT     all  --  192.168.122.0/24     anywhere
ACCEPT     all  --  anywhere             anywhere
REJECT     all  --  anywhere             anywhere             reject-with icmp-port-unreachable
REJECT     all  --  anywhere             anywhere             reject-with icmp-port-unreachable

Chain OUTPUT (policy DROP)
target     prot opt source               destination
ACCEPT     udp  --  anywhere             anywhere             udp dpt:bootpc
ACCEPT     tcp  --  anywhere             anywhere
```

Comme les paquets sortants sont toujours autorisés, le client a toujours accès au site apache.

8) Pour supprimer la 1ère règle :

```
# iptables -D INPUT 1
```

Avant :

```
target     prot opt source               destination
ACCEPT     udp  --  anywhere             anywhere            udp dpt:domain
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:domain
ACCEPT     udp  --  anywhere             anywhere            udp dpt:bootps
```

Après :

```
target     prot opt source               destination
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:domain
ACCEPT     udp  --  anywhere             anywhere            udp dpt:bootps
```

2. REJECT ou DROP

1) Pour effacer toutes les règles d'un coup, on utilise la commande suivante :

```
# iptables -F

root@214-2 : /home/test
# iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy DROP)
target     prot opt source               destination
```

2) On autorise le trafic sur INPUT et OUTPUT :

```
# iptables -P INPUT ACCEPT

root@214-2 : /home/test
# iptables -P OUTPUT ACCEPT

root@214-2 : /home/test
# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
```

M2103 – TP n°4
Filtrage avec IpTables

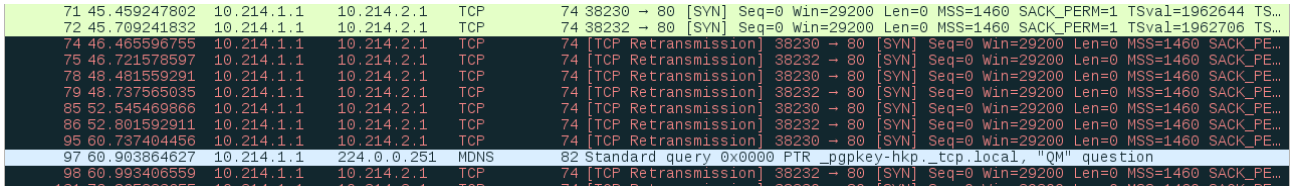
```
target     prot opt source               destination
```

3) On utilise DROP pour interdire le trafic web :

```
# iptables -A INPUT -i eno1 -p tcp --dport 80 -j DROP
```

4) Manipulation réussie, le client externe n'a pas accès au site web.

5) Capture Wireshark identique aux deux postes :



On remarque que seulement le poste client envoie des paquets TCP. Le serveur ne répond jamais.

6) On utilise la commande **nmap** pour analyser les ports de la machine serveur :

```
# nmap 10.214.2.1

Starting Nmap 7.40 ( https://nmap.org ) at 2019-05-06 14:48 CEST
Nmap scan report for 214-2 (10.214.2.1)
Host is up (0.0000050s latency).
Not shown: 997 closed ports
PORT    STATE SERVICE
80/tcp  filtered http
111/tcp open  rpcbind
902/tcp open  iss-realsecure
```

En vert, le port 80 apparaît comme filtrant le protocole http. Il est donc fermé.

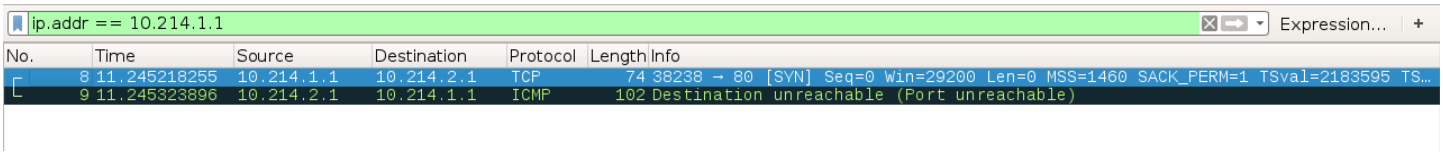
7) On supprime la règle précédente :

```
# iptables -D INPUT 1
```

8) On utilise REJECT pour interdire le trafic web :

```
# iptables -A INPUT -i eno1 -p tcp --dport 80 -j REJECT
```

9) Voici la capture wireshark associée à la tentative de connexion du client au serveur :



Cette fois ci, on obtient un unique message d'erreur ICMP indiquant que le port 80 n'est pas atteignable par le client.

10) Le port 80 est dans le même état qu'à la question 6.

3. Filtrage avec état

1) On supprime tous les règles :

```
# iptables -F
```

2) On paramètres les chaînes pour bloquer la trafic entrant et sortant :

```
# iptables -P INPUT DROP
# iptables -P OUTPUT DROP
```

3) Le client ne peut pas se connecter au serveur.

4) Pour vérifier que le module est déjà chargé, on le recherche dans la liste des modules en cours :

```
# lsmod | grep conntrack
nf_conntrack_ipv4      16384  1
nf_defrag_ipv4         16384  1 nf_conntrack_ipv4
xt_conntrack           16384  0
nf_conntrack          114688  5
nf_conntrack_ipv4,nf_nat_masquerade_ipv4,xt_conntrack,nf_nat_ipv4,nf_nat
x_tables               36864  11
ipt_REJECT,iptable_mangle,ip_tables,ebtables,iptable_filter,xt_tcpudp,ipt_MASQUERADE,xt_CHE
CKSUM,ip6table_filter,xt_conntrack,ip6_tables
```

Pour activer le suivi de connexion sun°1 de la chaîne INPUT

? F
aites-le et
c
onnectez-vous au serveur web à partir d'une autre mac
r conntrack on utilise la commande :

```
iptables -A INPUT -m conntrack –ctstate NEW,RELATED,ESTABLISHED -i eno1 -p tcp –dport
80 -j ACCEPT
```

4. Gestion des règles

1) Lorsque l'on redémarre la machine, toutes les règles précédemment enregistrées sont supprimées.

2) Pour sauvegarder les règles actuelles :

```
iptables-save > /chemin_du_fichier/Mes_regles_iptables
```

3) On supprime toutes les règles de toutes les chaînes :

```
iptables -F
```

4) On restaure les règles sauvegardées précédemment :

```
# iptables-restore /home/test/Bureau/Mes_regles_iptables
```

5) La commande **iptables -L** nous permet de vérifier que les règles ont bien été supprimées puis restaurées.

6) Voici le programme bash que nous allons utiliser pour automatiser le processus :

```
!/bin/bash
# iptables-restore /home/test/Bureau/Mes_regles_iptables
iptables -P INPUT DROP
iptables -P OUTPUT DROP
```

5. Journalisation

1) L'option LOG du mode journalisation permet de logger tous les évènements.

2) Pour logger les trames entrantes, on utilise :

```
# iptables -A INPUT -p tcp -j LOG --log-prefix «trame_entrante_pour_Port_80 » --dport 80
```

3) Les logs sont stockés dans */var/log/messages*

4) L'intérêt de **–log-prefix** est nommer tous les paquets entrants par le port 80 avec un nom commun permettant de facilement les identifier.