

Securinets



Club de la sécurité informatique
INSAT



SFD 2011

[NetFilter - IPTables]

Chef Atelier : Amina MSEDDE (RT5)
Noureddine BLAGUI (RT5)
Asma JERBI (RT5)
Jihene BOUCHTIBA (RT4)
Chayma BEN HAHA (RT4)

Objectif de l'atelier

Dans cet atelier, nous allons nous initier à l'utilisation des commandes IPTables et ce en sécurisant un serveur contre les attaques DDOS, SynFlood, Brute Force et en adoptant une politique de sécurité bien précise qui déterminera les paquets autorisés à entrer et à sortir de notre serveur.

1. Qu'est-ce que Netfilter ?

Netfilter est une couche logicielle intégrée dans le kernel de Linux, c'est à dire le cœur même du système d'exploitation Linux. Elle se situe plus particulièrement au niveau des couches réseaux IP, et se caractérise sous la forme de 5 hooks ("crochets") placés à 5 endroits stratégiques du système réseau du kernel.

Lorsque qu'un paquet IP arrive à un de ces hook, il va subir un certain nombre de vérifications à travers les règles qui constituent la chaîne associée au hook en question. En fonction du résultat de ces vérifications, une action (appelée "cible" par la suite) sera décidée pour lui : suppression du paquet, acceptation, modification, etc...

Ce que sait faire Netfilter:

- * D'effectuer des filtrages de paquets, principalement pour assurer des fonctions de Firewall.
- * D'effectuer des opérations de NAT (Network Address Translation)
- * D'effectuer des opérations de marquage des paquets, pour leur appliquer un traitement spécial.

2. Qu'est-ce que IPTables ?

Lorsque l'on parle de firewall sous Linux, iptables est la commande à connaître.

Iptables est installé en standard sur la plupart des distributions basées sur un kernel 2.4.x, il permet à l'administrateur de configurer Netfilter, en lui indiquant quelles sont les règles à utiliser pour les différentes tables et chaînes.

3. Les tables, chaînes et cibles :

a. Les Tables

Il existe trois tables qui vont servir à contenir des règles de "filtrage" :

Table	Description
Filter	Cette table permet de filtrer les paquets. Typiquement ce sera pour les accepter ou non.
Nat	Avec cette table, on peut réaliser des translations d'adresse (ou de ports). Ceci sera notamment utile pour partager une connexion.
Mangle	Elle sert pour modifier les en-têtes des paquets. On la rencontrera parfois pour marquer des paquets afin que d'autres applications puissent les reconnaître.

b. Les chaînes

Les chaînes sont des ensembles de règles que nous allons écrire dans chaque table. Ces chaînes vont permettre d'identifier des paquets qui correspondent à certains critères.

Chaîne	Table	Description
PREROUTING	nat, mangle	Par cette chaîne passeront les paquets entrant dans la machine avant routage.
INPUT	filter	Cette chaîne traitera les paquets entrants avant qu'ils ne soient passés aux couches supérieures (les applications).
FORWARD	filter	Ce sont les paquets uniquement transmis par la machine sans que les applications n'en aient connaissance.
OUTPUT	filter, nat, mangle	Cette chaîne sera appelée pour des paquets envoyés par des programmes présents sur la machine.
POSTROUTING	nat	Les paquets prêts à être envoyés (soit transmis, soit générés) seront pris en charge par cette chaîne.

c. Les cibles

Le dernier élément important est la notion de cible. Il s'agit du traitement que l'on décide d'appliquer au paquet. C'est la cible qui se chargera de faire les opérations nécessaires.

Voici les cibles prédéfinies les plus courantes :

Cible	Description
ACCEPT	Les paquets envoyés vers cette cible seront tout simplement acceptés et pourront poursuivre leur cheminement au travers des couches réseaux.
DROP	Cette cible permet de jeter des paquets qui seront donc ignorés.
REJECT	Permet d'envoyer une réponse à l'émetteur pour lui signaler que son paquet a été refusé.
LOG	Demande au noyau d'enregistrer des informations sur le paquet courant. Cela se fera généralement dans le fichier <code>/var/log/messages</code> (selon la configuration du programme <code>syslogd</code>).

4. Réalisation de l'atelier

a. Sécuriser un serveur HTTP

Dans un premier temps, nous avons vidé les tables prédéfinies et créées par l'utilisateur pour commencer sur une base saine et vierge. :

```
iptables -t filter -F
```

```
iptables -t filter -X
```

A ce stade, on a commencé à définir les règles, nous avons interdit tout le trafic de la table filter. Cette règle va être la politique « par défaut » gérant le cas où aucune règle de la chaîne ne s'applique au paquet concerné.

```
iptables -t filter -P INPUT DROP
```

```
iptables -t filter -P FORWARD DROP
```

```
iptables -t filter -P OUTPUT DROP
```

Nous avons aussi, supprimé tous les paquets mal formés.

```
iptables -t filter -A INPUT -m state --state INVALID -j DROP
```

```
iptables -t filter -A OUTPUT -m state --state INVALID -j DROP
```

Nous avons Autorisé le loopback

```
iptables -t filter -A INPUT -i lo -j ACCEPT
```

```
iptables -t filter -A OUTPUT -o lo -j ACCEPT
```

Maintenant, nous avons commencé à sélectionner les paquets qui vont être autorisés à entrer ou sortir de notre réseau(paquets icmp,http,dns,ftp...).

SSH:

```
iptables -t filter -A INPUT -p tcp --dport 2222 -j ACCEPT
iptables -t filter -A OUTPUT -p tcp --dport 2222 -j ACCEPT
```

DNS In/Out

```
iptables -t filter -A OUTPUT -p tcp --dport 53 -j ACCEPT
iptables -t filter -A OUTPUT -p udp --dport 53 -j ACCEPT
iptables -t filter -A INPUT -p tcp --dport 53 -j ACCEPT
iptables -t filter -A INPUT -p udp --dport 53 -j ACCEPT
```

HTTP + HTTPS Out

```
iptables -t filter -A OUTPUT -p tcp --dport 80 -j ACCEPT
iptables -t filter -A OUTPUT -p tcp --dport 443 -j ACCEPT
```

HTTP + HTTPS In

```
iptables -t filter -A INPUT -p tcp --dport 80 -j ACCEPT
iptables -t filter -A INPUT -p tcp --dport 443 -j ACCEPT
iptables -t filter -A INPUT -p tcp --dport 8443 -j ACCEPT
```

Et, nous avons autorisé les paquets appartenant à une connexion déjà établie.

```
iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
```

Enfin, nous avons redirigé tout autre paquet entrant ou sortant à : /var/log/messages en se basant sur la cible LOG pour pouvoir les consulter après.

```
iptables -t filter -A INPUT -j LOG
iptables -t filter -A OUTPUT -j LOG
```

b. Sécuriser le serveur contre DDOS

i. Qu'est-ce que le DDOS

Attaque par déni de service distribuée (Distributed Denial of Service attack ou DDoS attack)

Un serveur informatique doit traiter plusieurs requêtes dans un court laps de temps. Lorsque le serveur est incapable de traiter toutes les requêtes qu'il reçoit, il y a "déni de service".

Une attaque DDoS (une attaque par déni de service distribuée) consiste à bombarder un serveur cible de requêtes par plusieurs ordinateurs simultanément, de sorte à ce que celui-ci, complètement saturé, soit incapable de répondre.

ii. Comment avons-nous sécurisé notre serveur contre le DDOS

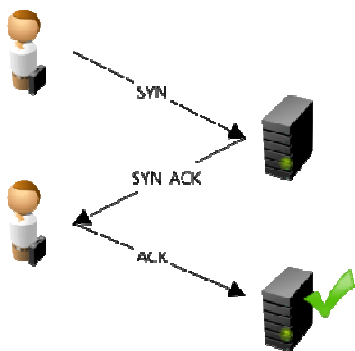
Nous avons autorisé les ping entrants, mais limite leur nombre à 1ping /s



c. Sécuriser le serveur contre Syn Flood

i. Qu'est-ce que le SynFlood

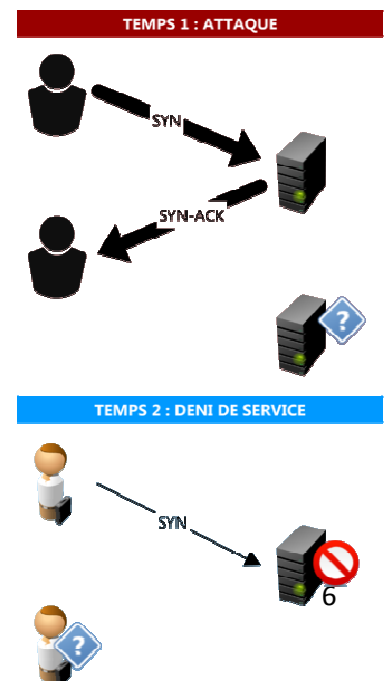
Le SYN flood est une attaque informatique visant à atteindre un déni de service. Elle s'applique dans le cadre du protocole TCP et consiste à envoyer une succession de requêtes SYN vers la cible.



Lorsqu'un client établit une connexion à un serveur, le client envoie une requête SYN, le serveur répond alors par un paquet SYN/ACK et enfin le client valide la connexion par un paquet ACK (acknowledgment, qui signifie accord ou remerciement).

ement).

Une connexion TCP ne peut s'établir que lorsque ces 3 étapes ont été franchies.



L'attaque SYN consiste à envoyer un grand nombre de requêtes SYN à un hôte avec une adresse IP source inexistante ou invalide. Ainsi, il est impossible que la machine cible reçoive un paquet ACK.

Les machines vulnérables aux attaques SYN mettent en file d'attente, dans une structure de données en mémoire, les connexions ainsi ouvertes, et attendent de recevoir un paquet ACK.

ii. Comment protéger notre serveur du SynFlood

On crée une nouvelle chaîne utilisateur du nom de syn-securinets

```
iptables -N syn-securinets
```

Si on dépasse 1 demande de connexion par seconde (après avoir dépassé une limite initiale de 25 demandes la tentative est marquée dans le Log précédé de '[SYN FLOOD max) atteint]:'

```
iptables -A syn-securinets -s 0.0.0.0 -m limit --limit 1/s -j ACCEPT
iptables -A syn-securinets -m limit --limit 1/s -j LOG --log-prefix '[SYN FLOOD max
atteint]:'
iptables -A syn-securinets -p tcp --syn -m limit --limit 1/s -j ACCEPT
iptables -A INPUT -m limit --limit 1/s --limit-burst 25 -j syn-securinets
```

Si la première contrainte est satisfaite on passe à notre règle syn-securinets pour bloquer l'IP des attaquants.

```
iptables -A syn-securinets -j DROP
```

d. Sécuriser le serveur contre le brute force

i. Qu'est-ce que le Brute Force

Les mots de passe sont stockés généralement cryptés dans un fichier. Pour obtenir un mot de passe, il suffit de récupérer ce fichier et de lancer un logiciel de brute force cracking. Ce procédé consiste à tester de façon exhaustive toutes les combinaisons possibles de caractères (alphanumériques + symboles), de manière à trouver au moins un mot de passe valide.

ii. Comment protéger notre serveur du Brute Force utilisant IPTables

Afin de remédier à cette attaque, nous compterons le nombre de connexion qu'une adresse IP aura fait en 600 millisecondes, et, nous n'autoriserons plus les paquets issues d'adresses IP ayant effectué plus que 2 connexions en ce laps de temps.

```
iptables -I INPUT -p tcp --dport 22 -i eth0 -m state --state NEW -m recent --set
iptables -I INPUT -p tcp --dport 22 -i eth0 -m state --state NEW -m recent --update --
seconds 600 \ --hitcount 2 -j DROP
```

5. Rendre ce filtrage automatique

```
sudo /etc/init.d/iptables save
```