

Dans le cadre de **SECURIDAY 2009**

SECURINETS



*Présente*

**Atelier : Mise en place d'une  
architecture sécurisée contre les  
attaques DDOS.**

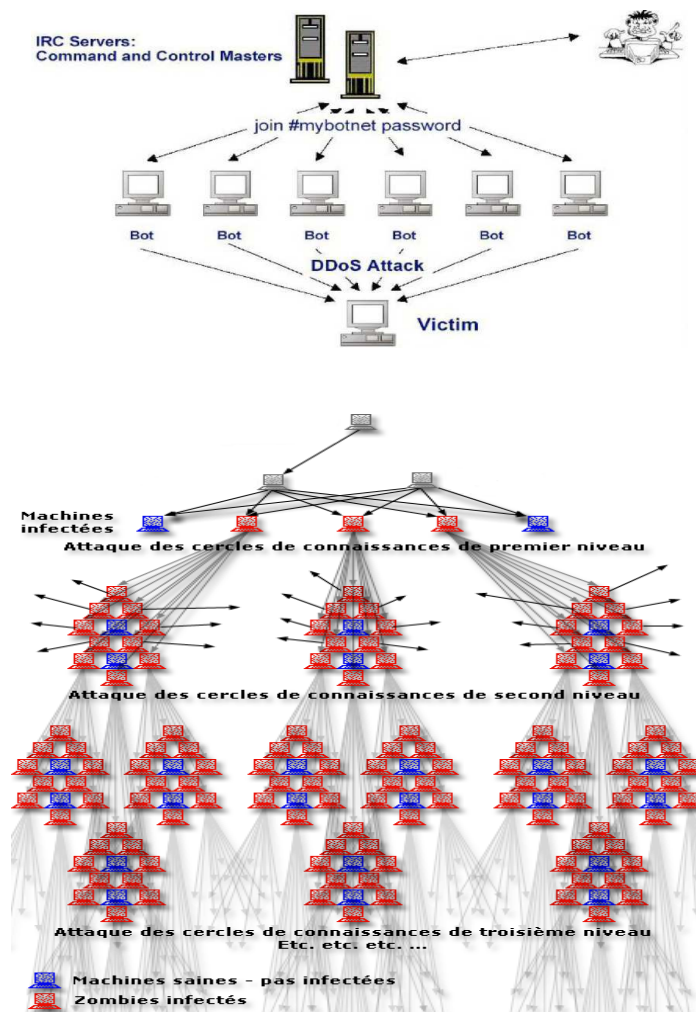
**Formateurs :**

- 1. Fitouri Abdelkrim**
- 2. Ghoulem Adel**
- 3. Yahia Marwen**
- 4. Zoghlami Oussema**

## 1. Présentation :

Un BotNet est un réseau de robots ou encore une grappe de Zombies (jusqu'à plusieurs centaines de milliers - 300.000 à 400.000 PCs), fonctionnant sous le contrôle d'un même pirate qui peut ainsi exploiter la somme des puissances de calcul et des largeurs de bandes passantes de tous "ses" Zombies. Ces BotNets peuvent être facilement utilisés afin de réaliser une attaque DDoS où tous les PCs commandés attaquent une seule destination.

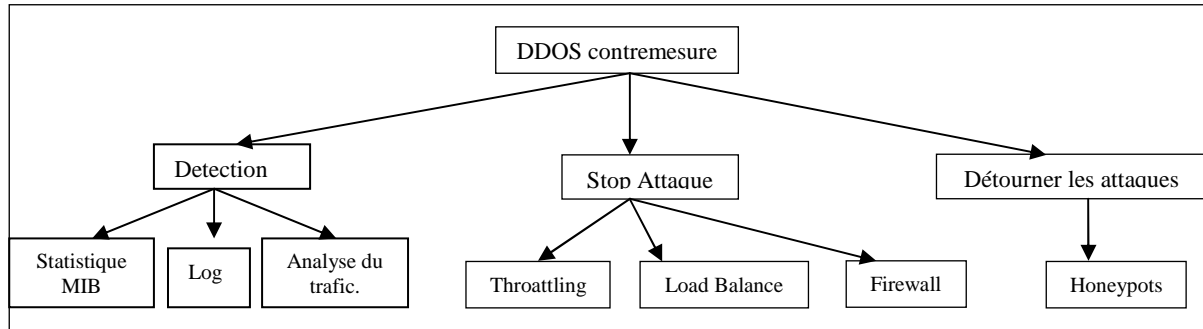
Une fois le premier bot implanté, le mécanisme de « recrutement de nouveaux hôtes candidats à l'infection » se fait à l'initiative des bots eux-mêmes où la plupart des bots, à l'aide d'une petite collection d'exploits, tentent constamment à compromettre davantage de machines afin de se propager.



## 2. Sécurisation :

La sécurisation d'une attaque BotNet DDOS est peut-être effectuée par plusieurs méthodes.

Ci-dessus quelques méthodes connues :

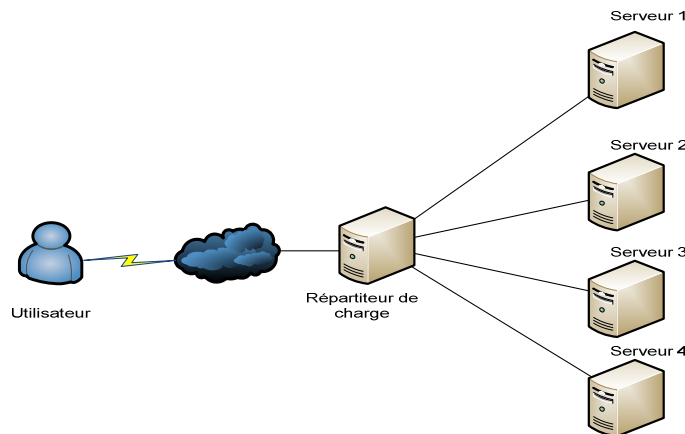


Dans notre atelier nous avons traité quelques méthodes qui stoppent les attaques.

### a) Load Balance :

C'est la répartition de charge ou l'équilibrage de charge, c'est une technique utilisée en informatique pour distribuer un travail entre plusieurs ordinateurs ou serveurs ce qui permet de réduire l'effet de DDOS.

Le principe de base consiste à interposer entre les Bots et le pool de ressources un dispositif (le répartiteur) qui connaît l'état d'occupation de chaque ressource et qui est capable de diriger les paquets vers la ressource la moins occupée, ou la plus facilement accessible et il peut définir aussi un taux d'utilisation si on le dépasse on bloque le trafic.



**b) Throttling :**

Cette méthode consiste à bloquer le trafic supérieur à un seuil prédéfini. Ce seuil est ajusté pour que le trafic maximal soit sans aucun danger sur la machine, la valeur du seuil est en fonction de débit et de ressource utilisé.

**c) Firewall (ou rejet de paquet) :**

Un pare-feu est un élément du réseau informatique, logiciel et/ou matériel, qui a pour fonction de faire respecter la politique de sécurité du réseau, celle-ci définissant quels sont les types de communication autorisés ou interdits.

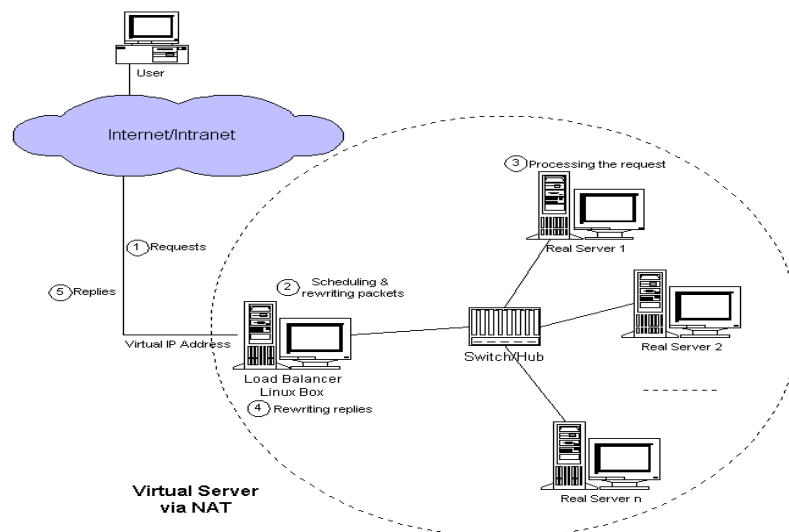
Cette méthode permet de bloquer les paquets par l'adresse source ou destination, port, protocole, état de la connexion,....

**3. Outil utilisé :**

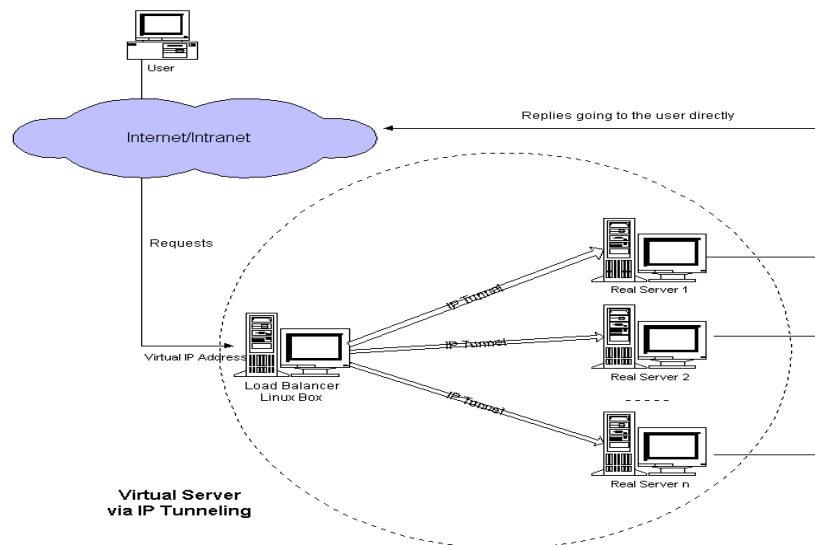
**a) LVS (linux virtuel server) :**

LVS est un serveur qui fait le basculement de charge. Il dispose de trois méthodes différentes de transmission de paquets :

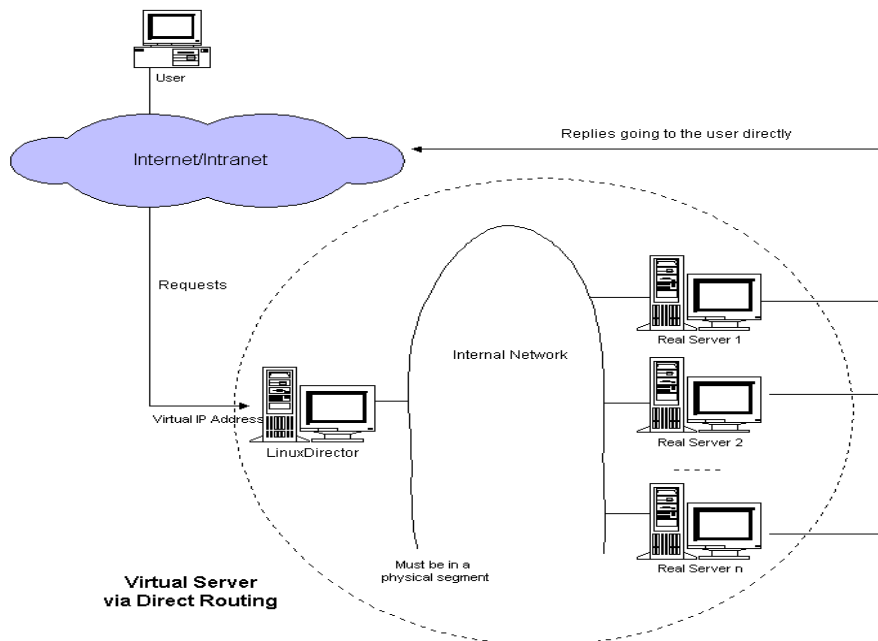
- Network Adresse Translation (NAT) : Une méthode de manipulation du port source et du port de destination et de l'adresse d'un paquet. L'utilisation la plus courante est de celle de l'IP masquerading qui est souvent utilisé pour permettre à des réseaux privés d'accéder à Internet. Dans le contexte de la couche 4 de commutation, les paquets sont reçus des utilisateurs finaux et le port de destination et l'adresse IP sont modifiés pour que le vrai serveur soit choisi. C'est-à-dire le serveur LVS modifie le couple adresse/port afin de joindre la destination.



- **Direct Routing:** Les paquets des utilisateurs sont envoyés directement au serveur réel. Les paquets IP ne sont pas modifiés, le serveur réel doit être configuré pour accepter le trafic contenant l'adresse IP du serveur virtuel. Cela peut être fait en utilisant une interface de filtrage de paquets ou de rediriger le trafic adressé au serveur virtuel de l'adresse IP d'un port local. Le serveur réel envoie leurs réponses directement à l'utilisateur. Ainsi, le serveur virtuel n'a pas besoin d'être dans la voie de retour des réponses.



- **IP-IP Encapsulation (Tunnelling) :** Il permet à des paquets adressés à une adresse IP d'être redirigés vers une autre adresse, éventuellement sur un autre réseau. Le principal avantage de l'utilisation de tunnels est que les vrais serveurs peuvent être sur des réseaux différents.



**b) Netfilter/iptables :**

Netfilter permet de faire beaucoup de choses en matière de filtrage de paquets et de translation d'adresses, ce qui fait de Linux 2.6 un outil de choix pour la réalisation de passerelles entre réseaux privés et l'Internet. Il permet de:

- Effectuer des filtrages de paquets, principalement pour assurer des fonctions de Firewall. On pourra par exemple interdire à tous les paquets venant de l'Internet et s'adressant au port 80 (HTTP) de passer. Notre serveur APACHE est un serveur Intranet et ne doit pas être accessible depuis l'extérieur.
- Effectuer des opérations de NAT (Network Address Translation) Ces fonctions sont particulièrement utiles lorsque l'on veut faire communiquer tout ou partie d'un réseau privé, monté avec des adresses IP privées (192.168.x.x par exemple) avec l'Internet.
- Effectuer des opérations de marquage des paquets, pour leur appliquer un traitement particulier. Ces fonctionnalités sont particulièrement intéressantes sur une passerelle de réseau d'entreprise.

**4. Installation et configuration :**

Remarque :

Dans toute cette partie nous avons utilisé Ubuntu 8.10, version du noyau 2.6.x.

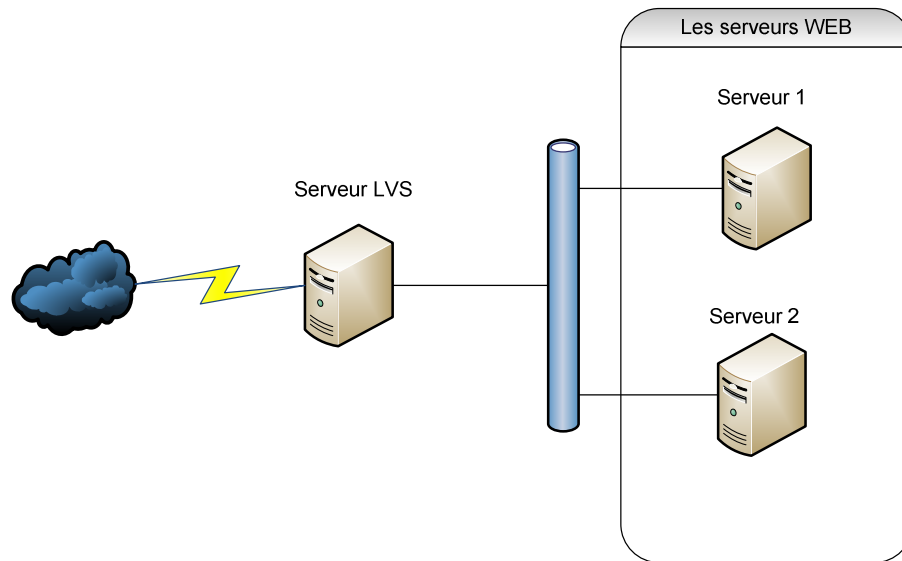
**a) Basculement de charge (load balance) :**

- Architecture choisi :

Parmi les 3 architectures supportées par LVS (les tunnels, NAT, Routage direct), notre choix s'est orienté vers l'architecture qui utilise un NAT. Cette architecture nous semble être la moins coûteuse et la plus sécurisée mais elle a aussi des inconvénients. En fait, l'utilisation de la translation d'adresse peut introduire un délai important dans les communications.

# S E C U R I N E T S

Club de la sécurité informatique  
I N S A T



- Installation :

Il faut s'assurer que les modules suivants sont activés sur le noyau de linux avant de commencer l'installation.

```
$cd /usr/src/linux-headers-2.6.xx
```

```
$Sudo make menuconfig
```

Networking-→

Networking option -----→

IP virtual server support (EXPERIMENTAL) --->

```
-- IP virtual server support (EXPERIMENTAL)
[ ] IP virtual server debugging
(12) IPVS connection table size (the Nth power of 2)
*** IPVS transport protocol load balancing support ***
[*] TCP load balancing support
[*] UDP load balancing support
[*] ESP load balancing support
[*] AH load balancing support
*** IPVS scheduler ***
<M> round-robin scheduling
<M> weighted round-robin scheduling
<M> least-connection scheduling
<M> weighted least-connection scheduling
<M> locality-based least-connection scheduling
<M> locality-based least-connection with replication scheduling
<M> destination hashing scheduling
<M> source hashing scheduling
<M> shortest expected delay scheduling
<M> never queue scheduling
*** IPVS application helper ***
<M> FTP protocol helper
```

On recompile le noyau:

```
$Sudo make
```

Installation de la commande ipvsadmin qui va nous permettre d'ajouter des serveurs :

```
$Sudo apt-get install ldirectord ipvsadm
```

Il faut ensuite modifier le fichier /etc/sysctl.conf, on enlève « # » devant la ligne :

```
net.ipv4.ip_forward = 1
```

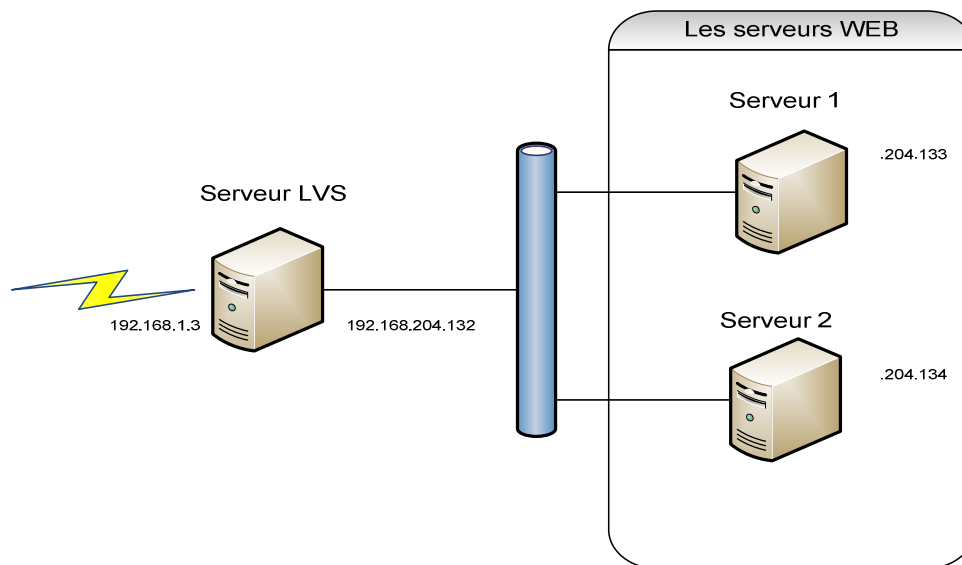
Validation la modification avec :

```
$Sudo sysctl -p
```

- Mise en oeuvre

Nous allons maintenant faire du basculement de charge entre 2 serveurs Web. La répartition de serveurs peut garantir une haute disponibilité du service, l'amélioration des performances et peut être un moyen efficace pour contrer les attaques de Type DOS (dans notre cas http flood)

Le serveur LVS utilisée contient 2 interfaces réseau, une interface pour le sous réseau contenant les serveur web, une autre interface pour le réseau extérieur.



- ✓ ipvsadm :

La commande ipvsadm permet de faire le mappage entre les serveurs réels et serveur virtuelle avec le numéro de port, cette commande permet aussi d'ajouter des serveurs réels, des services, de spécifier un type d'ordonnement, une métrique pour chaque serveur, type de balancement de charges...

- ✓ Type d'ordonnement :

Divers types d'ordonnements sont possible avec LVS :

- Round-robin (-s rr) : Permet de diviser les requêtes équitablement entre les serveurs.

**S E C U R I N E T S**  
**Club de la sécurité informatique**  
**I N S A T**

- Weighted round robine : (-s wr) : permet de diviser les requête entre les serveurs suivant une métrique (poids)
- Least-connection (-s lc) : Répartie les requêtes selon le nombre de clients connectés au serveur.
- Locality based with Least-connection (-s lbic): Permet d'envoyer les requêtes à des serveurs appartenant au même sous réseau que le client.
- Shortest expected delay (-s sed) : Permet d'envoyer les requêtes au serveurs avec le délai le plus minimal en suivant cette règle :  $(C_i + 1/U_i)$  avec  $C_i$  la bande passante,  $U_i$  la métrique.
- ....

Suivant les performances et le niveau de sécurité attendu on choisit un algorithme d'ordonnement.

Remarque :

Pour ajouter une métrique à chaque serveur on utilise l'option (-w poids) poids peut appartenir à un intervalle entre 0 a 65535 (0 pour dire inutilisable).

✓ Seuil max/min :

- Upper threshold (-u-threshold ou -x) : La valeur max de connexion simultanée au delà de laquelle un serveur n'accepte plus de connexion. (par défaut 0 aucune limite).
- Lower threshold (-l-threshold ou -y) : La valeur minimale au-dessus de laquelle un serveur commence a accepté des connexions.

✓ Type de protocole :

LVS utilise les protocoles de la couche 4 pour le basculement de charge :

- Tcp (-t)
- Udp (- u)
- Marquage de paquet firewall (-f) : les paquets peuvent être marqués par les firewalls (comme netfilter) ensuite routés par LVS suivant le champ TOS

Exemple de configuration :

```
$sudo ipvsadm -A -t 192.168.1.3:81 -s rr
```

Pour ajouter un service.

```
$sudo ipvsadm -a -t 192.168.1.3:81 -r 192.168.204.128:80 -m  
$sudo ipvsadm -a -t 192.168.1.3:81 -r 192.168.204.129:80 -m
```

Pour associer les services à des serveurs réels.

Remarque :

On utilise « -A » pour ajouter un service et « -a » pour ajouter un serveur réel.

L'option « -r » pour spécifier un serveur réel.

L'option « -m » pour dire qu'on utilise le NAT (masquarading), on peut utiliser « -g » pour le routage direct (gatewaying) ou « -i » pour les tunnelles (ip tunnelling).

Autre configuration (sorry server):

```
$sudo ipvsadm -A -t 192.168.1.3:81
$sudo ipvsadm -a -t 192.168.1.3:81 -r 192.168.204.128:80 -m -x 15 -y 1
$sudo ipvsadm -a -t 192.168.1.3:81 -r 192.168.204.129:80 -m -y 14
```

Une fois que le premier serveur atteint 15 connections, les paquets seront redirigés vers le « sorry server » qui affichera un message « désolé, serveur plein essayer plus tard ».

• **Sécurisation de LVS:**

Lvs est aussi conçu pour faire face à des attaques de type déni de service (DOS).

Pour chaque paquet reçu LVS crée une entrée dans la mémoire de taille 124-128 octets dans laquelle il enregistre des informations utiles. L'attaque consiste à bombarder le serveur de paquet pour provoquer un débordement de mémoire (plus d'espace mémoire disponible).

LVS implémente 3 stratégies pour contrer ce type d'attaque :

➤ Suppression d'entrée (drop\_entry) :

si la taille de mémoire utilisée atteint un certain seuil fixé, des entrée seront supprimer de la mémoire (aleatoirement).

Pour fixer un seuil, en nombre de page,on utilise le fichier

```
/proc/sys/net/ipv4/vs/amemthresh
```

Pour activer le mode drop\_entry on remplace 0 par 3 dans le fichier :

```
/proc/sys/net/ipv4/vs/drop_entry
```

➤ Rejet de paquet (drop\_packet):

Rejette les paquets si on atteint un taux proche d'une valeur fixé, avec :

Taux =amemthresh/ (amemthresh-mémoire utilisée)

On fixe le taux dans le fichier :

```
/proc/sys/net/ipv4/vs/am_droprate
```

Pour activer le mode drop\_packet on remplace 0 par 3 dans le fichier :

```
/proc/sys/net/ipv4/vs/drop_packet
```

➤ Tcp sécurisé (Secure tcp):

Utilise des tables de transition d'état (tcp:SYN,SYN+ACK,ACK,RST.....) et des timeout de chaque état. On modifie les timeouts des états pour que les paquets suspects, comme un SYN+ACK sans ACK qui le suit, ne remplissent la mémoire.

Pour activer le mode secure\_tcp on remplace 0 par 3 dans le fichier :

```
/proc/sys/net/ipv4/vs/secure_tcp
```

Les timeouts peuvent être modifié à partir des fichiers:

```
/proc/sys/net/ipv4/vs/timeout_*
```

## b) Firewall:

Dans cette partie nous allons présenter quelques règles Netfilter qui peuvent être utiles lors d'une attaque DDOS.

➤ Attaque Ping of death :

```
iptables -I INPUT -p icmp --icmp-type echo-request -m length --length 32:65535 -j DROP
```

Elle permet de rejeter les paquets dont la taille est supérieure à 32 octets.

➤ Ping flooding :

```
iptables -I INPUT -p ICMP -m icmp --icmp-type echo-request -m limit --limit 10/minute -j ACCEPT
```

Permet d'accepter un seul paquet ICMP par minute.

```
iptables -I INPUT -p icmp -m recent --set
```

```
iptables -I INPUT -p icmp -m recent --update --seconds 60 --hitcount 5 -j DROP
```

Cette commande permet de créer une liste des adresses IP qui ont accédé récemment.

--set ajoute l'adresse ip à la liste « recente list »

--update teste si l'adresse ip est présente dans la liste ou pas.

--seconds --hitcount : si l'adresse est présente dans la liste alors la règle rejettera le ping que si il correspond au argument de seconde et hitcount c'est-à-dire 5 paquets chaque 60 seconde.

➤ SYN flood

```
iptables -I INPUT -p tcp --tcp-flags syn --dport 22 -m recent --set
```

```
iptables -I INPUT -p tcp --tcp-flags syn --dport 22 -m recent --update --seconds 60 --hitcount 4 -j DROP
```

## **Conclusion:**

Ce type d'attaque reste très difficile à contrer ou à éviter : il s'agit donc d'une menace que beaucoup craignent. En effet, cette attaque est très dévastatrice, et ne provient plus seulement d'une seule machine, mais d'un réseau tout entier. Sachant le nombre de machines non sécurisées présentes sur Internet, on peut imaginer l'ampleur d'une telle attaque.

Il n'est donc pas évident de s'en protéger étant donné que l'identité des attaquants change souvent et que le temps nécessaire pour organiser une protection adéquate est bien souvent supérieur au temps nécessaire pour mettre à mal la victime. Il est donc primordial de localiser l'initiateur et de repérer sa signature. La détection d'un trafic suspect peut servir de prévention.

## **Bibliographie :**

- Man ipvsadm
- <http://linuxgazette.net/108/odonovan.html>
- <http://www.linuxvirtualserver.org/Documents.html>
- <http://www.fifi.org/doc/iptables-dev/html/netfilter-extensions-HOWTO.html>
- <http://www.student.montefiore.ulg.ac.be/%7Ebleuwart/Rapport.tar.gz>
- CEHv6 Module 14 Denial of Service