Protection contre les attaques de déni de service dans les réseaux IP

Référant : Osman SALEM

HOTTE Marion
LUTUN Quentin-Edouard
ASCOET Thomas

SOMMAIRE

Introduction

Définition Dos, DDoS : attaque d'un pirate, sur un serveur informatique, de façon à l'empêcher d'offrir le service pour lequel il est destiné.

Qui, où, support : les victimes du déni de service ne sont pas uniquement celles qui le subissent, les postes compromis (daemons et masters) et les postes clients qui n'arrivent pas à accéder aux services désirés sont également les victimes des pirates qui effectuent le DoS. De nos jours, le piratage peut être acquis aisément, l'attaquant peut donc être un utilisateur lambda tant que son poste est relié au réseau mondial.

Pourquoi (but) : but principal : que l'accès au serveur d'une entreprise devienne impossible aux clients, le but n'étant pas d'altérer les données contenues et échangées ni de voler des informations, mais plutôt de nuire à la réputation de l'entreprise en empêchant l'accès aux divers services fournis aux clients en provoquant un ralentissement significatif ou une saturation du système voire le crash du système. A l'origine, les pirates n'étaient intéressés que par la renommée d'avoir réussit à faire tomber un réseau. Aujourd'hui, la raison de ces attaques est le chantage, en effet ces criminels sont principalement motivés par l'argent.

Deux types de DoS :

- Deni de service par saturation : submerger une machine d'un grand nombre de requête afin qu'elle ne soit plus apte à répondre aux demandes des clients.
- Deni de service par exploitation de vulnérabilités : exploiter une faille du système dans le but de le rendre inutilisable.

Comment (principe) : envoyer une très grande quantité de paquets, dont la taille est relativement importante, en même temps, voire sur une longue période.

Le principe du Distributed Denial of Service (DDoS) consiste à utiliser une grande quantité de postes « Zombies », préalablement infectées par des « backdoors » ou « troyens », dans l'intention de paralyser la réponse du serveur attaqué. Les maîtres sont eux-mêmes reliés aux postes « daemons ». Le pirate se sert des postes maîtres pour contrôler les postes daemons qui effectueront l'attaque, sans cela, le pirate devrait se connecter lui-même à chaque daemons ce qui serait plus long à mettre en place, et plus facilement repérable.

Pour utiliser les masters et daemons, il est nécessaire d'exploiter des failles connues (FTP…). Le pirate se connecte aux masters en TCP pour préparer l'attaque, ces derniers envoient les commandes aux daemons en UDP.

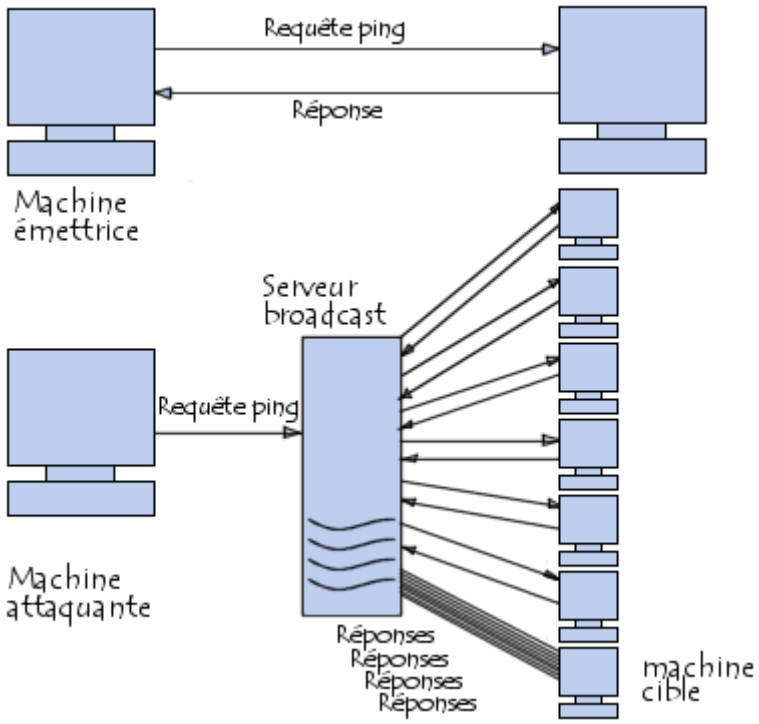
I. Historique

Les attaques par déni de service ont commencé dans les années 1980, ce n'est qu'en Aout 1992 qu'aurait eu lieu la première attaque de déni de service distribué dirigée contre les serveurs de l'Université du Minnesota, à la suite de quoi l'accès Internet de l'Université est resté bloqué pendant plus de 2 jours.

Historique des attaques de déni de service :

En décembre 1996 a eu lieu l'attaque Ping de la mort (Ping Of Death).

En juillet 1997, a eu lieu l'attaque Smurf qui, grâce à un serveur de diffusion (« broadcast »), duplique et envoie sur l'ensemble du réseau le message qu'il aura reçu de la machine attaquante. Par la suite, les machines du réseau répondent au serveur de diffusion qui redirigera ces réponses sur la machine cible.



Cette attaque fut suivie par l'attaque WinNuke qui provoquait un « blue screen » ou un « reboot » sur les postes Windows 95 et NT.
En octobre de la même année a eu lieu l'attaque Land dont le principe est l'usurpation d'adresse IP dans le but d'exploiter une faille du protocole TCP/IP dans les systèmes visés. Cette attaque consiste donc à envoyer dans les champs sources et destination des paquets IP exactement la même adresse et le même numéro de port ce qui avait pour conséquence de déstabiliser ou de faire tomber les systèmes vulnérables tels que les systèmes Windows 95, 98, NT 4.O, FreeBSD etc ...

Enfin, en décembre 1997, a été appliquée l'attaque Teardrop / Overdrop dont le principe est d'exploiter la fragmentation effectuée par le protocole IP (fragmentation des paquets, de taille trop importante, en plus petits paquets possédant tous un numéro d'identification et de séquence, à la réception, ils sont réassemblés grâce aux valeurs de décalage). On insère dans les paquets fragmentés de fausses informations de décalage qui, lors du réassemblage provoquaient des vides ou recoupements qui avaient pour conséquence une instabilité sur les stations Linux, Windows NT et 95.

En janvier 1998, l'attaque Bonk/Boink qui visait les stations Windows 95 et NT 4.0. Le principe est d'émettre une grande quantité de paquets UDP corrompus provoquant des blocages ou des plantages du système d'exploitation qui a été visé, suivie par l'attaque Fraggle qui inondait la cible de paquets UDP en utilisant une variante amplifiée de l'attaque Smurf. Suivit en juin 1998 de l'attaque Syndrop basée sur le Teardrop en TCP avec le bit SYN et possédant des champs invalides tels que le numéro de séquence par exemple. L'impact de cette attaque est le blocage les postes Windows NT4 SP3 par un Freeze (gèlés, blocage).

Le lundi 21 octobre 2002, une attaque de type Ping Flood bloque 9 des 13 serveurs DNS qui ont pour but le routage des requêtes de résolution de nom de domaine, rendant ainsi impossible l'accès à leurs ressources pendant trois heures. Les pirates ont pu grâce à un parc important de machines de générer un nombre de requêtes entre deux et trois fois supérieur à la capacité des treize serveurs visés.

Historique des attaques de déni de service distribué :

La première attaque DDOS qui fut médiatisée dans la presse s'est produite le 7 février 2000 contre Yahoo! L'attaque a empêché l'accès à son portail Internet pendant trois heures. Durant ces heures Yahoo aurait subi une perte d'environ 500 00$.
Le lendemain, ce sont Amazon.com, Buy.com, CNN et eBay ont été attaqués au moyen du déni de service distribué provoquant ainsi soit l'arrêt total, soit un fort ralentissement du fonctionnement de leur serveur. Sur les dix heures pendant lesquelles l'attaque a duré, Amazone aurait subi une perte de 600 000$, eBay est passé de 100 à 9.4% de disponibilité et CNN en dessous de 5% de la capacité habituelle. Le 9 février suivant, E Trade et ZDNet ont à leur tour été victimes de ce type d'attaques dont la conséquence fut l'inaccessibilité de leur portail.

En septembre 2001, un virus nommé Code Red infecte des milliers de systèmes, suivit par une seconde version, Code Red II, qui installait un agent effectuant le DDOS. Il paraîtrait que cette attaque aurait été lancée contre la Maison Blanche, cette dernière a, par la suite annoncée que les mesures nécessaires à la sécurité seraient entreprises.

Dès l'été de l'année suivante, c'est Internet qui subit une attaque DDOS à l'encontre de ses 13 serveurs racines. Cette attaque n'a duré qu'une heure mais elle aurait pu paralyser la totalité du réseau Internet.
En septembre 2002 est apparue la première version de Slapper qui, en seulement deux semaines, a contaminé plus de 13 000 serveurs Linux. Slapper utilise une faille de sécurité présente dans **le module OpenSSL1,** et y transfert un agent DDOS.

Le 25 janvier 2003, le ver SQL Slammer infecte des serveurs de bases de données MySQL de Microsoft qui étaient alors mal configurés. Seuls 4 des 13 serveurs racines ont été affectés ne provoquant qu'une réduction de 15% de la performance globale du réseau.

II. Algorithmes d'attaques existants

De très nombreuses méthodes permettent d'arriver à un DoS :

- Le SYN Flood consiste à saturer un serveur en envoyant un grand nombre de paquets TCP avec le flag SYN armé.
- L'UDP Flood consiste à saturer le trafic réseau en envoyant le plus grand nombre de paquets UDP à une machine.
- Le ping of death utilise aussi une faiblesse de certaines piles TCP/IP lors de la gestion de paquets ICMP trop volumineux.
- Le smurfing est aussi une attaque basée sur le protocole ICMP.
- Les bombes e-mail consistent à envoyer sur le réseau des mails trop volumineux.

Nous allons voir plus en détail, comment fonctionnent les différentes attaques proposées ci-dessous.

A. Différentes attaques

Contexte :

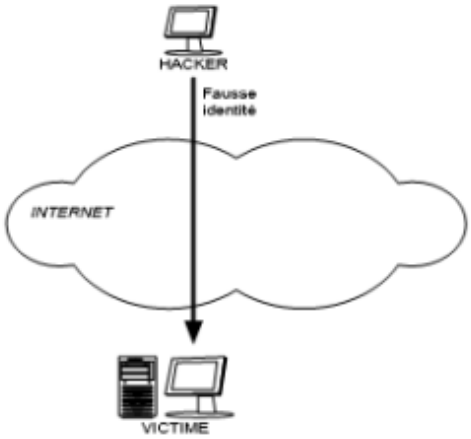
La réalisation d'un déni de service n'est pas très compliquée, mais pas moins efficace. Nous avons vu précédemment que la plupart du temps, elles sont réalisées avec succès car la plupart des dénis de service exploitent des failles liées au protocole TCP/IP. Les contre-mesures sont compliquées à mettre en place, d'autant plus que ce type d'attaque utilise les services et protocoles normaux d'Internet. La seule façon de s'en protéger est de détecter les comportements anormaux, ce qui implique notamment la vérification de l'intégrité des paquets, la surveillance du trafic, l'établissement de profils types et de seuils. Commençons tout d'abord par comprendre comment fonctionnent ces attaques afin de mieux les détecter ou de les contrer. Il existe de nombreux outils pour réaliser des attaques du type DoS et DDoS. La plupart d'entre eux sont conçus pour fonctionner avec les systèmes d'exploitation Unix et Linux, mais de plus en plus sont développés sous les systèmes Windows.

1. Attaques du type DoS

Logiciel	**Type d'attaques**
Hping	UDP/TCP/ICMP flooding, Smurf
Slowloris, NKiller	SYN-Flood over HTTP
LetDown	TCP/TCP SYN flooding
Sockstress	TCP Session flooding

Pour simuler les attaques du type Déni de Service, nous utiliserons l'outil « Hping ». Celui-ci permet de réaliser l'envoi de paquets via les protocoles TCP, UDP ou ICMP en modifiant leurs en-têtes. Il est disponible sous Unix, Linux, MacOS X et Windows.

Fonctionnement :



DoS

Dans ce type d'attaque, le hacker lance seul son attaque contre la victime. La plupart du temps, le hacker cache son identité réseau (adresse IP et ports UDP/TCP) en se faisant passer pour une, voire plusieurs autres machines (IP Spoofing). Ainsi, il ne peut pas être reconnu par la victime. L'IP Spoofing peut être également utilisé pour faire du DNS Spoofing.

Utilisation de l'outil « **Hping** » :

Attaque du type SYN FLOOD :

```
# hping –S –i u10 –p 80 –a @IP_MACHINE_USURPEE @IP_MACHINE_CIBLE
```

L'argument 'S' demande le positionnement du flag. Le second argument 'i' précise l'intervalle entre deux envois (ici en micro secondes). Le troisième 'p' spécifie le port destination. Le quatrième 'a' désigne une adresse source. Le dernier, enfin, donne l'adresse destination.

Attaque du type PING FLOOD (ICMP FLOODING) :

```
# hping -1 –i u10 @IP_MACHINE_CIBLE
```

L'argument -1 précise que les paquets se font en ICMP, ceux-ci sont par défaut des « echo request ».

Attaque du type SMURFING :

```
# hping -1 –i u10 –a @ IP_MACHINE_CIBLE  @IP_BROADCAST
```

Avantages : Il permet de générer des paquets TCP, UDP, ICMP et Raw IP contrairement à l'utilitaire ping(8) d'Unix, qui ne permet lui que d'envoyer des paquets ICMP. « Hping » est principalement utilisé comme un outil de sécurité, mais il peut être utilisé sous diverses formes comme :

- Test de firewall et du réseau
- Traceroute avancé
- Outil d'illustration, lors d'une formation à TCP/IP

Inconvénients :

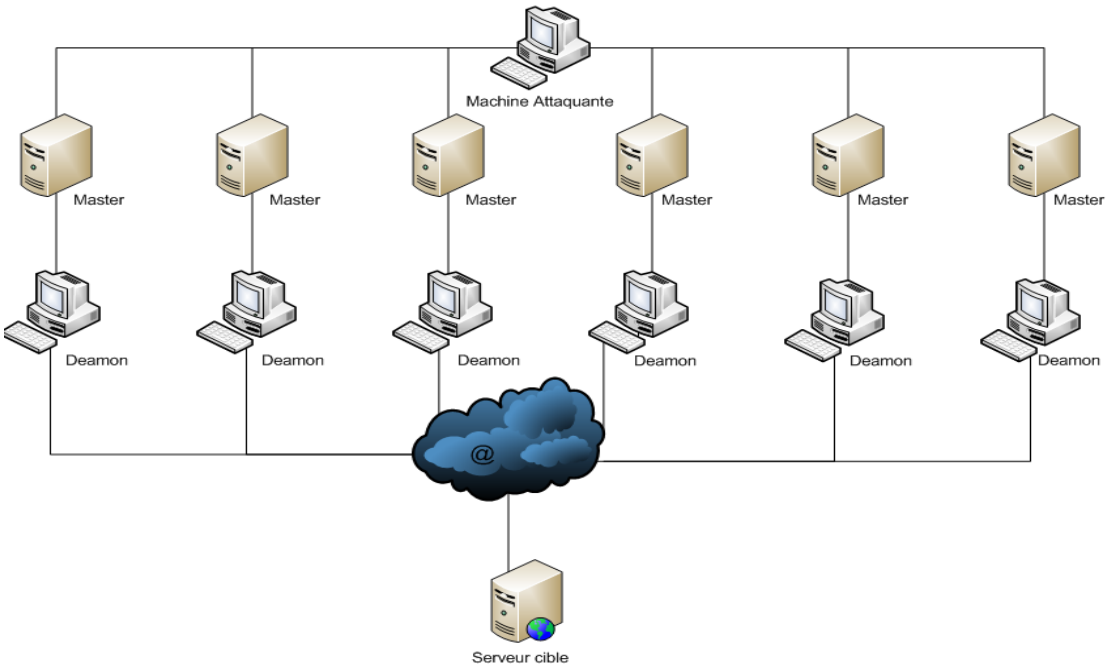
Les attaques ne proviennent que d'une seule machine, de ce faite les attaques du type PING Flood (ICMP flooding) ralentissent la machine mais pas suffisamment pour l'empêcher d'émettre les RST qui mettent fin à la connexion (la connexion est réinitialisée).

2. Attaques du type DDoS

Logiciel	**Type d'attaques**
Trinoo	UDP flooding
Tribe Flood Network (TFN) et TFN2k	UDP/TCP/TCP SYN flooding, Smurf
Stacheldraht	UDP/TCP/TCP SYN flooding, Smurf
Schaft	UDP/TCP/ICMP flooding
MStreamT	ACK flooding

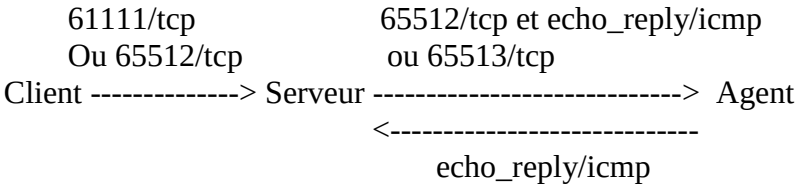
Compte tenu des performances actuelles des serveurs et de la généralisation des techniques de répartition de charge et de haute disponibilité, il devient de plus en plus difficile de réaliser une simple attaque DoS. De ce fait, les attaques du type Déni de service distribué sont de plus en plus prisées par les pirates informatiques car elles permettent de décupler les effets d'une attaque initiale. Malgré tout, elles restent complexes à mettre en œuvre. L'efficacité du déni de service étant liée au nombre de machines compromises, de telles attaques nécessitent donc au préalable une phase de corruption de machines sur Internet, afin d'y installer des agents, et de pouvoir plus tard utiliser cette armée d'attaquants.

Fonctionnement :



L'architecture ci-dessus comporte 4 types d'hôtes différents : le(s) client(s) (Hacker), le(s) serveur(s) (Master), les agents (Zombies) et enfin la cible (Victime). Le client, utilisé par l'attaquant, contrôle un ou plusieurs Serveurs, lesquels communiquent avec les Agents chargés de la mise en œuvre du déni de service. Pour éviter que les Agents et les Serveurs ne puissent être utilisés par autrui (administrateur légitime de l'hôte, autres pirates) toutes les commandes vers ces programmes nécessitent des mots de passe ou des informations supplémentaires. De plus, afin d'éviter le blocage des commandes par un outil de détection ou de filtrage, les communications peuvent être cryptées.

Nous utiliserons l'outil **Stacheldraht** (fil de fer barbelé en allemand), c'est un outil partiellement basé sur TFN (dont il reprend le code de l'Agent) mais qui a comme caractéristique de chiffrer toutes ses communications TCP. De plus, il comporte des instructions de compilation pour Linux et Solaris.



L'Agent, lorsqu'il démarre, cherche à contacter un certain nombre de serveurs sur le port TCP 65512 ou 65513 en envoyant un message ICMP echo_reply en clair avec comme identifiant « 666 » en décimal et « skillz » dans le champ de données. Si un Serveur est à l'écoute, il répond par un echo_reply avec un identifiant « 667 » et « ficken » en données. L'Agent procède alors à un test pour savoir s'il peut émettre des adresses sources complètement aléatoire en envoyant vers un Serveur un paquet ICMP echo_reply d'adresse source 3.3.3.3 avec son adresse réelle dans les données (identifiant 666). Si le message arrive au Serveur, il répond via echo_reply avec un identifiant à « 1000 » et « spoofworks » (l'usurpation fonctionne !) dans les données. En cas de succès, l'Agent fabriquera des adresses sources quelconques, sinon il se limitera à modifier l'octet de poids le plus faible (ce qui permet au moins à la cible d'identifier le réseau ou sous-réseau émetteur). Le Serveur stocke les Agents identifiés dans un fichier « .bc » chiffré à l'aide de l'algorithme symétrique BlowFish. De même, les Agents possèdent un fichier avec une liste de Serveurs (.ms par défaut) chiffré avec BlowFish ou bien se réfèrent à 2 Serveurs par défaut codés dans le binaire.

Le Client se connecte sur le port TCP 61111 ou 65512 d'un Serveur, s'authentifie avec un mot de passe et peut alors envoyer un certain nombre de commandes pouvant être transmises aux Agents via echo_reply/icmp (la commande est un code dans l'identifiant) :

- gestion de la liste de Serveurs des Agents (ajout, retrait) ;
- liste des Agents présents et morts après un test de la connexion TCP ;
- changement de la taille, des ports cibles, pour les paquets d'attaque ;
- attaques par saturation de requêtes UDP, TCP, ICMP sur plusieurs adresses ;
- ouverture d'un shell distant sur un port donné par les hôtes des Agents ;
- mise à jour de chaque Agent en utilisant le programme rcp sur la machine hôte.

Chaque Agent a la possibilité de créer un certain nombre de processus fils (1 par défaut et 15 maximums) pour attaquer séquentiellement l'ensemble des cibles.

Pour plus d'informations, cf. stacheldraht.analysis de D. Dittrich.

Utilisation de l'outil « **Stacheldraht** »:
Démarrage du Serveur(MASTER) :

```
# ./mserv
[*]-stacheldraht-[*] - forking in the background...
1 bcasts were successfully read in.
```

Démarrage de l'Agent :

```
# ./td
```

Connexion au Serveur(MASTER) :

```
./client @IP_MASTER
[*] stacheldraht [*]
(c) in 1999 by ...
trying to connect...
connection established.
-----------------------------------
enter the passphrase : sicken
-----------------------------------
entering interactive session.
*******************************
welcome to stacheldraht
*******************************
type .help if you are lame
stacheldraht(status: a!1 d!0)>
```

Si l'authentification avec le Serveur réussi, on accède alors à l'invite de commandes de ce dernier qui indique certaines informations comme le nombre d'agents qui sont censés être actifs ("a!") et inactifs ("d!"). On peut désormais envoyer un certain nombre de commandes d'administration (liste des Agents identifiés et vérification qu'ils sont actifs, arrêt des Agents, ...), ou encore demander un déni de service sur une ou plusieurs adresses IP.

Quelques commandes utiles :

.help : Liste toutes les commandes.

.mtimer : Définit la durée de l'attaque (en seconde).

.micmp : Lancer une attaque du type ICMP FLOODING.

.msyn : Lancer une attaque du type SYN FLOODING.

.mudp : Lancer une attaque du type UDP FLOODING.

.mack : Lancer une attaque du type TCP ACK FLOODING.

.msmurf : Lancer une attaque du type SMURF.

.mudns : Lancer une attaque du type UDP FLOODING sur le port 53 (DNS).

.mlist : Liste les machines infectées par le DoS.
.mstop all : Stop toutes les attaques en cours.

Exemple :

.micmp 192.168.0.1 : Lance une attaque DoS du type ICMP FLOODING vers la machine cible « 192.168.0.1 ».

Avantages :

- Effet multiplicateur par rapport à l'attaque DoS initiale.
- Possibilité de lancer une attaque massive de déni de service, coordonnée, par les machines infectées contre un ou plusieurs sites.
- Saturation de la bande passante du réseau en Amont.
- Avec le support d'authentification et de cryptage **BlowFish**, il est impossible de détecter les commandes passées, de cette manière le maitre est quasi-indétectable.

Inconvénient :

La mise en place est beaucoup plus complexe, elle nécessite au préalable une phase de corruption de machines sur Internet afin d'y installer des agents.

3. ARP Spoofing / ARP Poisoning

Comme nous l'avons dit précédemment, un pirate masque généralement son identité en falsifiant l'adresse IP source des paquets qu'il envoie. Il doit également s'assurer qu'il n'y aura pas de réponse de la part de la machine à qui on a usurpée l'adresse IP. En effet, si une machine dont l'IP a été usurpée venait à recevoir les SYN-ACK provenant de la victime, c'est à dire des paquets inattendus, elle y répondrait par un RST, ce qui mettrait fin à la connexion en attente et libérerait les ressources (empêchant donc l'attaque de se réaliser correctement). Ainsi un message RST est plus rapide qu'une négociation de fin de connexion.
Pour s'en assurer, on peut lancer une attaque du type SYN FLOODING pour inonder la machine cible avec une énorme quantité de SYN, tout en sachant que la machine usurpée est toujours active sur le réseau. Ainsi, on remarque que quelque soit le nombre de SYN envoyé, la cible parvient toujours à traiter les RST reçus et donc à libérer les ressources, ce qui prouve en effet qu'il faut réduire au silence la machine usurpée pour qu'elle n'envoie aucun RST. La solution idéale serait bien entendu de déconnecter la machine du réseau, mais il est bien évident qu'un pirate ne dispose généralement pas d'un accès physique à la machine. Les attaques du type PING FLOOD, c'est à dire l'envoi massif de requêtes ICMP pour saturer la machine ou encore le SMURFING, technique proche de la précédente, mise à part qu'elle met en jeu toutes les machines du réseau contre la machine usurpée, peuvent permettre d'une certaine façon la mise hors service. Malheureusement, ces attaques ralentissent la machine mais pas suffisamment pour l'empêcher d'émettre les RST. Le but recherché étant de surcharger au point de mettre hors service la machine. C'est là qu'intervient la technique de l'ARP Spoofing ou ARP Poisoning (empoisonnement de cache ARP), utilisant le protocole de résolution d'adresse ARP. Elle va permettre à l'attaquant de détourner des flux de communication transitant sur un réseau local commuté, lui permettant de les écouter, de les corrompre, mais aussi d'usurper une adresse IP ou de bloquer du trafic. Il s'agira donc ici d'empoisonner le cache ARP de la machine cible en attribuant une adresse mac invalide à l'adresse IP de la machine usurpée. Lorsque la cible envoie des paquets SYN-ACK, elle les envoie vers une adresse mac inexistante, et donc aucune machine ne peut répondre avec un paquet RST et mettre fin à la connexion semi-ouverte. Pour envoyer des paquets ARP reply à interval régulier, on pourra utiliser l'outil « send_arp », qui est un simple programme disponible sous Linux. Il envoie sur le réseau un paquet ARP paramétrable :
send_arp src_ip_addr src_hw_addr targ_ip_addr tar_hw_addr

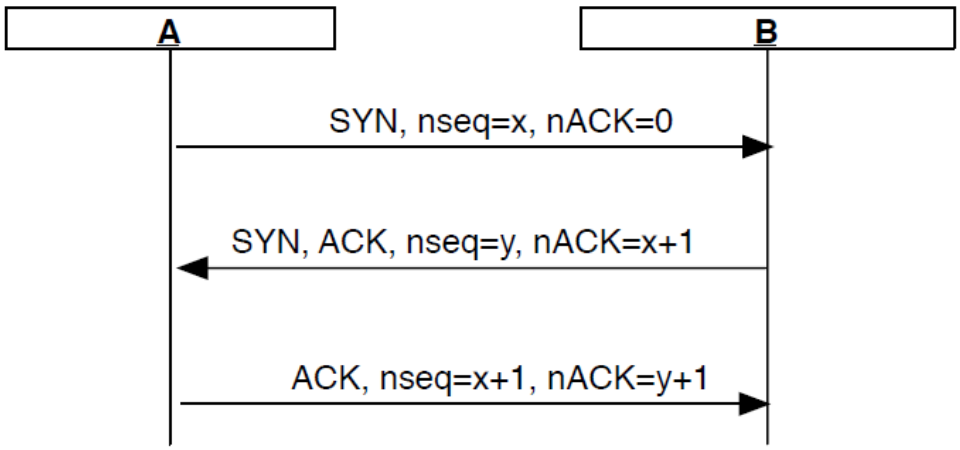
Exemple:
send_arp 192.168.0.1 00:e2:20:03:3f:d5 192.168.0.12 00:e0:30:9e:08:9a
Il devient donc très simple de rajouter une boucle afin qu'il envoie un tel paquet à interval régulier.
Il existe également d'autres alternatives au programme « send_arp » comme « Dsniff » ou encore « Ettercap » qui constituent une suite d'outils destinée à réaliser l'audit d'un réseau et divers tests de pénétration et qui permettent entre autres de mettre en pratique l'ARP Spoofing / ARP Poisoning.
Malgré tout, cette attaque est très limitée car elle se restreint au réseau local de la cible.

B. SYN Flooding

Principe :

Cette attaque utilise une faiblesse du protocole TCP en se basant sur l'envoi massif de demande d'ouverture de session SYN. L'objectif étant de saturer le nombre maximum de sessions TCP en cours. Ainsi, lorsque cette limite est atteinte, la cible ne pourra plus établir aucune session TCP causant une indisponibilité de toutes ces applications en écoute de port TCP.

Fonctionnement général :



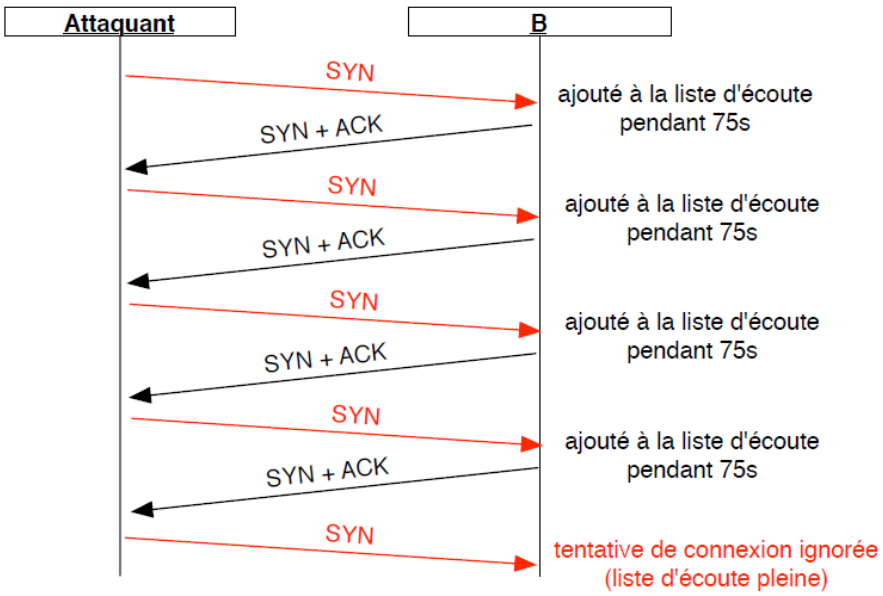
Remarque :

- Les numéros de séquence initiaux x et y sont choisis "aléatoirement".
- Un timer est déclanché après l'envoi d'un SYN.
- Si une réponse tarde trop à arriver (>75s), la connexion est abandonnée.

Se protéger :

- une bonne configuration des firewalls permet de détecter/limiter ce type d'attaque. Par exemple, on peut limiter le nombre de connexions TCP par seconde.

Fonctionnement :



Les SynCookies représentent une solution technique relativement efficace présentée dans le chapitre III, A. 3

C. UDP Flooding

Principe :

Cette attaque exploite le mode non connecté du protocole UDP. Elle consiste à générer une grande quantité de paquets UDP soit à destination d'une machine soit entre deux machines (Ex : Chargen Denial of Service Attack).

Conséquences :

- DoS : congestion du réseau et saturation des ressources des deux hôtes victimes ;
- Congestion généralement plus importante qu'avec le TCP Flooding car l'UDP ne possède pas de mécanisme de contrôle de congestion ;
- Les paquets UDP sont prioritaires sur les paquets TCP ;
- La totalité de la bande passante peut être saturée : effondrement de la totalité du réseau.

Se protéger :

- Configurer les firewalls pour limiter le trafic UDP.
- Désactiver si possible certains services comme echo et chargen.

Rappel sur UDP :

- C'est un protocole de la couche transport.
- C'est un service sans connexion et non fiable : les paquets UDP peuvent se doubler ou se perdre.
- Aucun service supplémentaire n'est ajouté par rapport à IP.

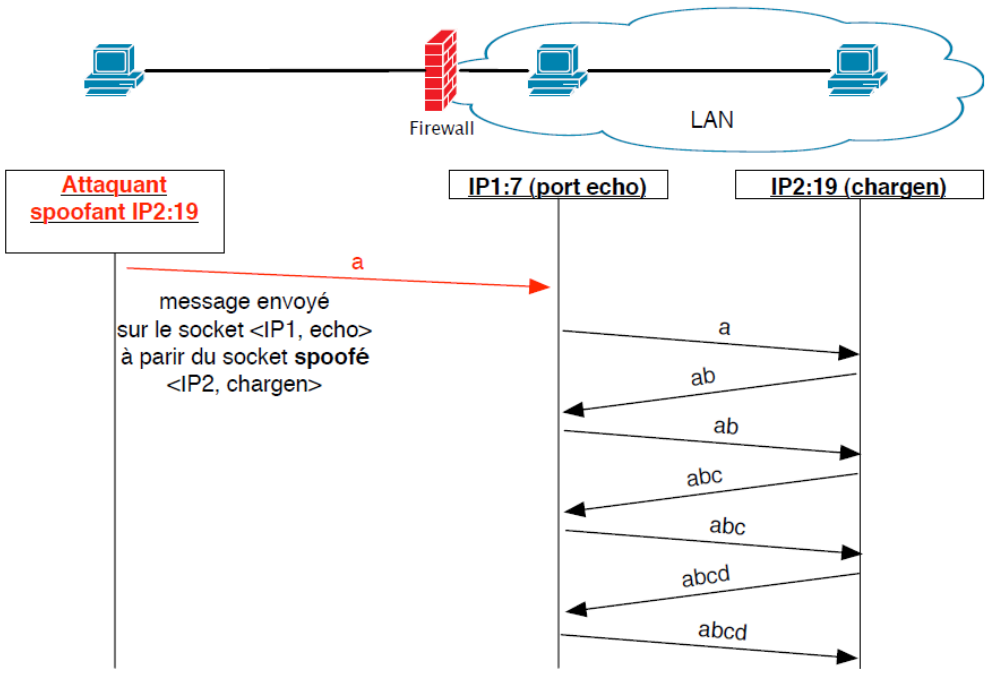
En-tête d'un paquet UDP :

31	16 15	0
UDP SOURCE PORT	UDP DESTINATION PORT	
MESSAGE LENGTH	CHECKSUM	
DATA		
....		
...		

Faiblesse d'UDP :

- Aucun champ n'est chiffré dans une en-tête UDP
- Il n'y a pas de service d'authentification

Fonctionnement d'UDP Flooding :



D. Attaque par fragmentation

Principe :

Les dénis de service de type Packet Fragment utilisent des faiblesses dans l'implémentation de certaines piles TCP/IP au niveau de la défragmentation IP (réassemblage des fragments IP). Une des attaques les plus connues utilisant ce principe est le Teardrop.

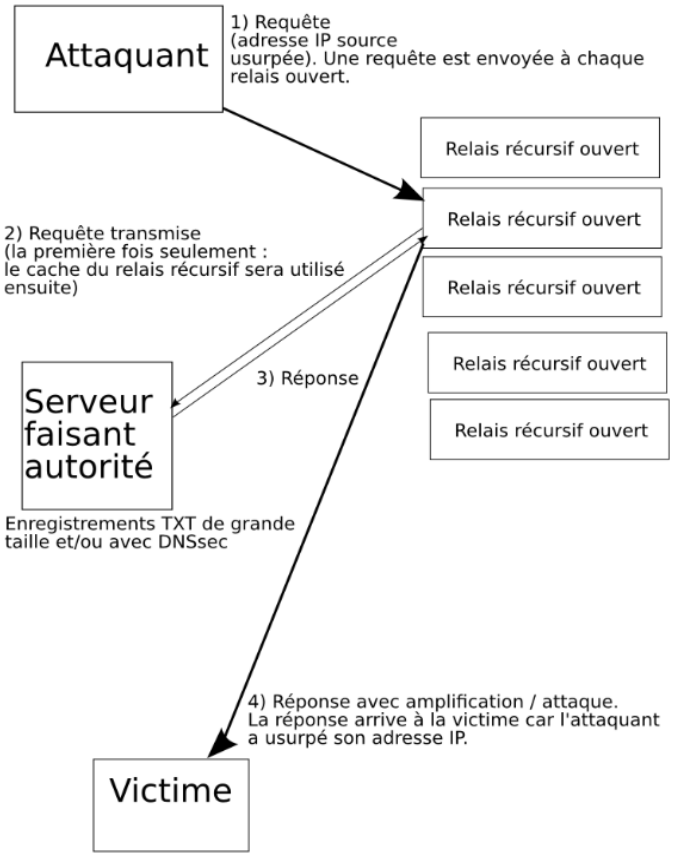
Conséquences :

Certains cas de réassemblage non prévus entraînent un crash de la machine et donc un déni de service.

Se protéger :

Installer si possible une implémentation de la pile TCP/IP résistant à cette faille.

Fonctionnement :



L'astuce est de modifier les numéros de séquence afin de générer des blancs ou des recouvrements lors du réassemblage par la pile IP cible.

Aujourd'hui, comme pour le ping de la mort, cette technique n'est plus viable du fait que les piles IP ont toutes évoluées.

E. Pind of death

Cette technique d'attaque est dépassée, mais elle a fait ses preuves à l'époque. Elle exploitait une faiblesse dans l'implémentation de la plupart des piles IP en envoyant un paquet ICMP d'une taille non conforme (supérieur à 64 octets). Ceci avait pour effet de planter directement la pile IP attaquée.

Cependant, comme pour l'attaque par fragmentation, cette technique n'est plus viable du faite que les piles IP ont toutes évoluées. Nous pouvons donc discuter en datagramme ICMP de grande taille sans aucun souci.

L'attaque Ping de la mort est souvent confondue en pensant qu'elle est basée sur le fait de saturer une bande passante en ICMP. Ce n'est pas le cas, car ce principe est appliqué par l'attaque Ping Flood et non pas par le Ping de la mort.

F. Smurffing

Principe : Cette attaque utilise

- Le protocole ICMP,
- De l'IP spoofing,
- Parfois le broadcast.

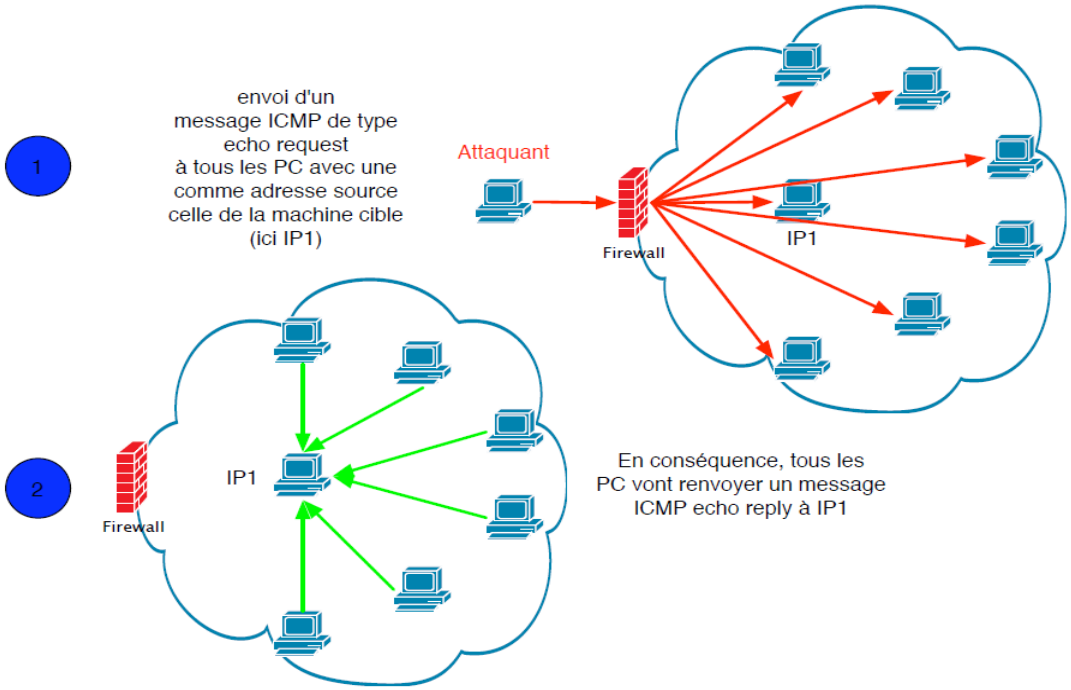
Conséquences :

- Denis de service (dans le cas de smurfing par exemple).
- Interception de paquets.

Se protéger :

- Configurer les firewalls pour limiter le trafic ICMP par seconde ;
- Configurer les firewalls pour bloquer les ping ;
- Interdire le broadcast.

Fonctionnement :



Quelques types d'erreur :

- type 8 (Echo Request) : test si la machine cible est opérationnelle
- type 0 (Echo Reply) : en réponse au paquet ICMP de type 8
- type 11 (Time Exceeded) : exploité par traceroute

Les messages ICMP "Time exceeded" ou "Destination unreachable" peuvent forcer un ordinateur à casser ses connexions en cours (celles concernées par le message). Si un attaquant envoie de manière répétée de faux messages de ce type, il peut causer un DoS.

Les messages ICMP "Redirect" (normalement utilisés par les routeurs) permettent de détourner un flux de données en spécifiant la route que doit emprunter les paquets. Deux restrictions : l'attaquant doit être sur le même réseau local, et d'autre part, une connexion entre l'attaquant et la victime doit déjà exister.

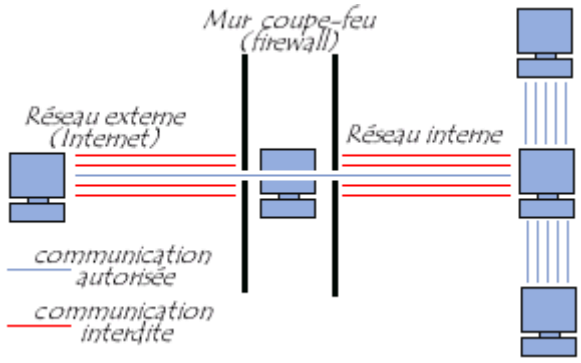
III. Logiciels et algorithmes de défense

Nous avons vu précédemment qu'il était important de se protéger contre des attaques toujours plus massives et efficaces. Il est donc important de bien configurer notre firewall et de comprendre comment le hacker effectue ses attaques. Voici plusieurs logiciels et algorithmes permettant de détecter les intrusions.

Qu'est-ce qu'un pare-feu?

Un **pare-feu** (appelé aussi *coupe-feu* ou **firewall** en anglais), est un système permettant de protéger un ordinateur ou un réseau d'ordinateurs des intrusions provenant d'un réseau tiers (notamment internet). Le pare-feu est un système permettant de filtrer les paquets de données échangés avec le réseau, il s'agit ainsi d'une passerelle filtrante comportant les interfaces réseau suivante :

- une interface pour le réseau à protéger (réseau interne) ;
- une interface pour le réseau externe.



Le système firewall est un système logiciel, reposant parfois sur un matériel réseau dédié, constituant un intermédiaire entre le réseau local (ou la machine locale) et un ou plusieurs réseaux externes. Il est possible de mettre un système pare-feu sur n'importe quelle machine et avec n'importe quel système pourvu que :

- La machine soit suffisamment puissante pour traiter le trafic ;
- Le système soit sécurisé ;
- Aucun autre service autre que le service de filtrage de paquets ne fonctionne sur le serveur.

A. IDS / IPS

Contexte :
Nous avons vu précédemment qu'il était très simple de mettre en œuvre une attaque du type Déni de Service ou encore de façon Distribuée sans avoir de réelles connaissances techniques sur les protocoles réseau et en programmation puisque des outils prêts à l'emploi existent. Il est tout de même très difficile de lutter contre ce type d'attaque, compte tenu de la complexité de la mise en œuvre de moyens pour la distinguer du trafic légitime. Le seul moyen de s'en prémunir consisterait donc à agir en amont, en implémentant des techniques de prévention, détection, traçage et suppression des attaques dans le réseau. Il faut rappeler que la sélection d'une technique de prévention dépend des contraintes techniques provenant du réseau existant, du type de client et du type d'attaque que l'on souhaite prévenir.

Prévention contre les Déni de Service de type Applicatifs :
Pour rappel, ce genre d'attaque vise à exploiter les vulnérabilités d'une application comme par exemple les dépassements de tampon (*Buffer Overflow*) ou encore la saturation de ressources CPU. Il exploite les faiblesses de la conception d'un protocole ou de son implémentation, il est donc nécessaire pour éviter ce genre d'attaque de mettre à jour le système régulièrement pour corriger les éventuelles failles logicielles ou systèmes. Mais le problème majeur qui se pose est souvent celui de la mauvaise configuration des services par l'administrateur qui constitue la principale cause des dénis de service applicatifs. En effet, malheureusement l'erreur est humaine et il est difficile de penser à tout pour mettre en place une configuration bien sécurisée.

Un moyen efficace de se protéger des attaques du type Déni de Service est d'utiliser un système de d'analyse et de détection d'intrusion *IDS (Intrusion Detection System)* ainsi qu'un système de prévention *IPS (Intrusion Protection System).*

1. Les IDS

Les systèmes de détection d'intrusion (ou encore IDS) permettent de détecter de manière furtive les activités anormales ou suspectes et permettent ainsi d'avoir une action de prévention sur les risques d'intrusion en écoutant le trafic réseau.

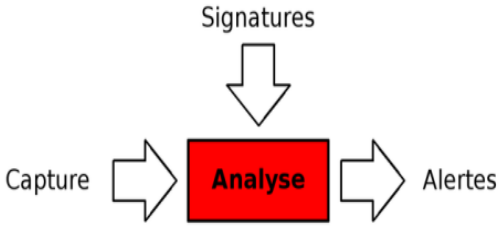
Il existe trois grands types d'IDS bien distincts :

- Les NIDS (*Network Based Intrusion Detection System*).
- Les HIDS (*HostBased Intrusion Detection System*).
- Les IDS hybrides à la fois NIDS et HIDS.

a) Les NIDS (Network Based Intrusion Detection System)

Le NIDS analyse et contrôle le trafic réseau, cherchant d'éventuelles traces d'attaques en générant des alertes lorsque des paquets suspects sont détectés.

Il fonctionne de trois manières différentes :



- En mode Capture (ou encore Sniffer) qui permet de récupérer tout le trafic réseau en temps réel. Il utilise pour cela la bibliothèque standard de capture de paquet *libpcap*. La bibliothèque de capture de paquets *Packet Capture Library* est portée sur quasiment toutes les plateformes, ce qui permet en général aux IDS d'être compatible sur toutes les plates formes. Par exemple, sous Linux, le fonctionnement de la capture d'un NIDS est de copier tout paquet arrivant au niveau de la couche liaison de données du système d'exploitation. Une fois ce paquet copié, il lui est appliqué un filtre BPF (Berkley Packet Filter), nécessaire à l'affinage de ce que l'IDS cherche à récupérer comme information. Il est aussi à noter que le trafic analysé n'est pas forcément égal à celui du trafic entrant, étant donné que la *libpcap* agit à une couche en dessous du pare-feu (qui agit au niveau réseau).

- En mode Analyse dont on distingue deux approches complémentaires :

1. L'analyse par Signatures qui permet de la même manière qu'un Antivirus de rechercher des formes d'événements caractérisant une attaque à partir de bibliothèques de signatures : Exemple: « /bin/sh » vers un serveur web.

Avantages :
- Très puissant pour reconnaître une attaque sans générer un trop grand nombre de faux-positifs.
- Diagnostique rapidement l'utilisation d'une technique d'attaque.

Inconvénients :
- Impossibilité de détecter des attaques non connues ou récentes ➔ Mise à jour constante !
- Problème d'évasion.

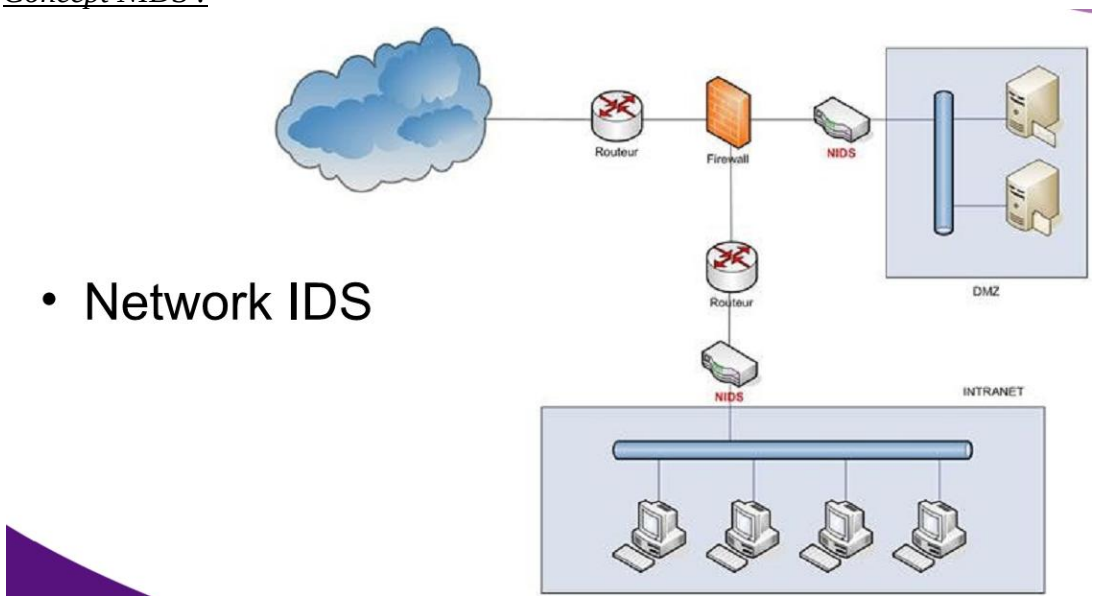
2. L'analyse Comportementales qui permet de rechercher des formes d'évènements liés à des activités anormales : Exemple: Un stagiaire utilise des outils de hack sur le réseau de la société.

Avantages :
- Détection d'anomalies permettant de détecter un comportement non attendu.
- Détection des symptômes d'une attaque avant l'application de l'attaque elle-même.

Inconvénients :
- Quantité importante de fausses alertes.
- Phase d'apprentissage très fastidieuse pour caractériser un comportement anormal.
- En mode Alertes (ou encore Générateur de Log) qui permet de journaliser les tentatives d'attaques ou d'intrusions généralement stockées dans le gestionnaire de

journalisation du système Syslog. Il est également possible de formater les alertes sous la norme IDMEF (pour *Intrusion Detection Message Exchange Format*) afin de les rendre compatible avec d'autres logiciels de sécurité. Ainsi, celles-ci peuvent être remontées par email par la plupart des IDS.

Concept NIDS :



NIDS - Emplacement solution n°1

- Port span : Le switch fait une copie de tout trafic d'un port ou d'un vlan vers le port du NIDS.

Avantages :

- Facilité d'installation et de configuration.
- Pas de modification de l'architecture existante.

Inconvénients :

- Saturation si contrôle de plusieurs ports.

NIDS - Emplacement solution n°2

- Hub : Le NIDS écoute tout le trafic qui transite par le Hub.
-

Avantages :

- Facilité d'installation.
- Aucun matériel supplémentaire.
- Solution peu chère.

Inconvénients :

- Connexion Half duplex.
- Débit mauvais.

NIDS - Emplacement solution n°3

- Tap : Similaire au Hub. Sépare le flux selon la direction (entrant ou sortant).

Avantages :

- Résiste aux coupures de courant.
- Pas de changement de topologie.
- Prise en charge de plusieurs ports (➔12).

Inconvénients:

- Prix.

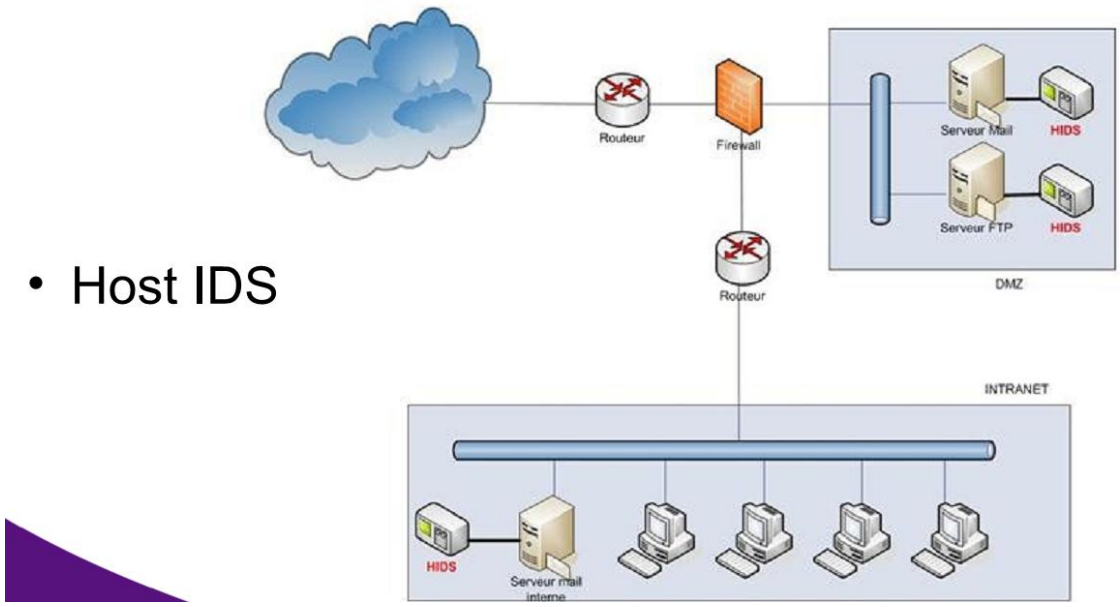
En Résumé :

Le NIDS permet donc de surveiller tout le flux réseau d'une société sans impacter sur la charge du réseau. En effet, celui constitue un élément passif du réseau écoutant le trafic, ce qui ne perturbe pas le réseau. De plus, il est mono-tâche et apparaît invisible aux yeux des pirates. Malgré tout, il demande des ressources nécessaires importantes notamment aux niveaux du CPU et de la mémoire en traitant un grand nombre d'informations à la fois. De plus, certains paquets ne seront pas analysés, c'est pourquoi on préférera plutôt une solution Hardware (plus rapide) que Software. Ainsi, les solutions du type VPN et SSL (par exemple), compliquent l'analyse. En effet le NIDS est incapable d'analyser le trafic chiffré. Enfin, il est également nécessaire de tenir à jour la base de données des signatures pour que le NIDS soit efficace aux attaques.

b) Les HIDS (HostBased Intrusion Detection System)

Les HIDS analyse et contrôle des informations contenues sur un équipement précis (ex: un serveur). Ainsi, contrairement à un NIDS, le HIDS récupère les informations qui lui sont données par le matériel ou le système d'exploitation. Il y a pour cela plusieurs approches : signatures, comportement (statistiques) ou délimitation du périmètre avec un système d'ACL.

Concept HIDS :



• Host IDS

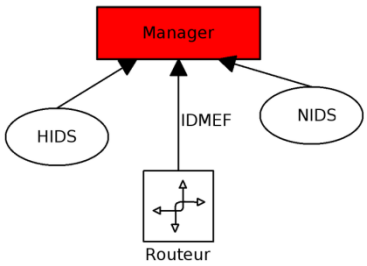
Avantages :

- Permet de traiter des événements locaux au système qu'un NIDS ne peut pas voir.
- Peut analyser le trafic chiffré (ex: ssh).
- Pas d'impact sur l'utilisation d'un réseau « switché ».
- Permet de détecter les Rootkit (chevaux de Troie).

Inconvénients :

- Difficulté d'administration (exemple avec Tripwire et Aide, fichier de politique à configurer…).
- Risque en cas de compromission locale.
- Les attaques de type DoS (Déni de service) et DDoS (déni de service distribué) peuvent gêner l'analyse locale d'un HIDS.
- CPU de la sonde = CPU du serveur !

c) ***Les IDS hybrides à la fois NIDS et HIDS***



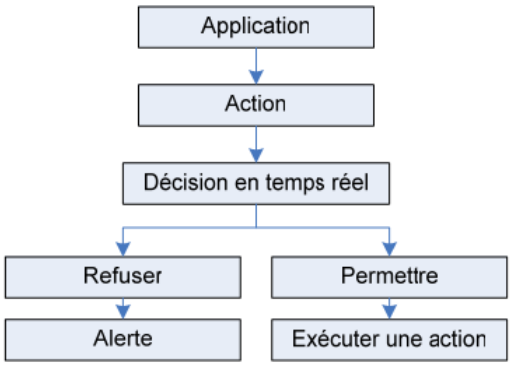
Les IDS hybrides sont basés sur une architecture distribuée, où chaque composant unifie son format d'envoi d'alerte (typiquement IDMEF) permettant à des composants divers de communiquer et d'extraire des alertes plus pertinentes.

Les avantages des IDS hybrides sont multiples :

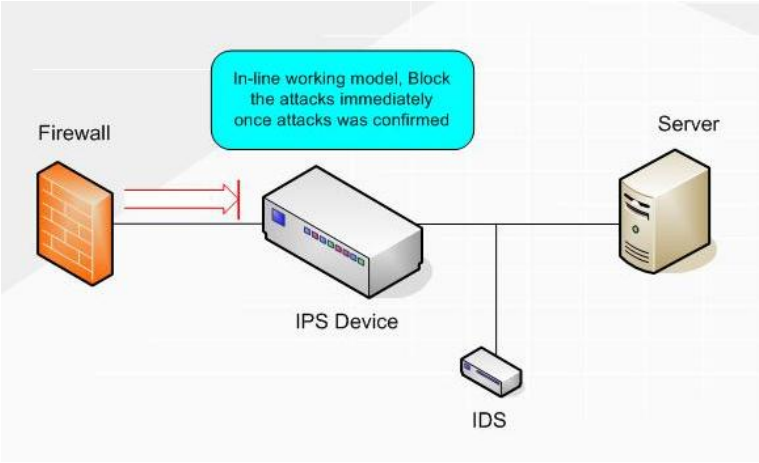
- Moins de faux positifs
- Meilleure corrélation
- Possibilité de réaction sur les analyseurs

2. Les IPS

Les IPS, contrairement aux IDS (sécurité passive) classiques, constituent une sécurité active pour filtrer et bloquer les flux, ajoutant à cela la défense proactive et la prévention des intrusions sur le réseau/hôte. Avant toute action, une décision en temps réel est exécutée (l'activité est comparée à un ensemble de règles). Si l'action est conforme à l'ensemble de règles, la permission de l'exécuter sera accordée et l'action sera exécutée. Si l'action est illégale (c'est-à-dire si le programme demande des données ou veut les changer alors que cette action ne lui est pas permise), une alarme est donnée. Dans la plupart des cas, les autres détecteurs du réseau (ou une console centrale) en seront aussi informés dans le but d'empêcher les autres ordinateurs d'ouvrir ou d'exécuter des fichiers spécifiques. Le diagramme ci-dessous illustre le fonctionnement d'un IPS.



Concept IPS :



Fonctionnalités IPS:

- Le comportement de l'application est analysé et noté (quelles données sont normalement demandées, avec quels programmes elle interagit, quelles ressources sont requises, etc.).
- L'interception d'appels au système : avant qu'un appel au système (rootkit) soit accepté, il doit être complètement vérifié (par exemple, quel programme a demandé l'appel au système, sous quelles autorisations d'utilisateur tourne le processus -root...-, à quoi l'appel système essaie-t-il d'accéder, etc.). Cette fonctionnalité permet la surveillance des essais de modification d'importants fichiers du système ou de la configuration.
- Lorsqu'une attaque est détectée, l'alerte peut aller d'une simple entrée dans un journal à un blocage de ressources.
- L'interaction avec les équipements réseaux tels que : firewall, IDS, VPN, anti-virus, anti-spam, etc.
- D'autres fonctionnalités sont possibles, comme la compréhension des réseaux IP (architecture, protocoles, etc.), la maîtrise des sondes réseau/analyse des logs, la défense des fonctions vitales du réseau, la vitesse d'analyse et un mode "stateful inspection".

Limites IPS :

Les principales limites et contraintes des IPS à ce jour semblent être leur mise en place délicate, leur administration rebutante, la possibilité de bloquer tout le réseau en cas de fausse alerte, ainsi que l'inexistence d'un standard actuel. Il est également capable de stopper des attaques réseau et applicatives les plus courantes aujourd'hui. C'est pour toutes ces raisons, que les IDS sont beaucoup plus utilisées que les IPS, bien que certains IDS soient maintenant dotées de fonction permettant de réagir et donc débloquer certaines attaques. Malgré tout, pour les entreprises, un IPS constitue, un investissement pour la sécurité plus judicieux qu'un simple IDS.

Les différents types IDS/IPS connus :

Voici un lien d'un tableau résumant les différents types d'IDS/IPS :
Cf. Tableau

A l'origine écrit par Martin Roesch, Snort est la solution la plus répandue pour les parties IDS et IPS. C'est d'une part le fait qu'il soit libre (publié sous licence GNU GPL) et donc que le fonctionnement soit accessible au plus grand nombre et d'autre part sa modularité qui permet à tous de participer à l'augmentation du nombre de pré-processeurs, de règles et de modules de sortie disponibles. Cependant, la recherche de trames qui circulent sur le réseau peut être longue et fastidieuse, mais si l'effort est récompensé, cela vaut sûrement la peine.

Snort fonctionne en 4 modes :

- Sniffer (snort -vde) : Comme tcpdump et WireShark, permettant d'écouter le trafic réseau en des points stratégiques.
- Générateur de log (snort -vde -l ./log) : Permettant le débogage des attaques en cours ou passées.
- NIDS (Intrusion Detection System) : Détecteur d'anomalies permettant de capter des intrusions (Sécurité Passive).
- IPS (Intrusion Prevention System) : Permettant la prévention des intrusions sur le réseau (Sécurité Active) ➔ Snort_Inline.

L'architecture de Snort est composée comme suit :

- Noyau de base : au démarrage, ce noyau charge un ensemble de règles, compile, optimise et classe celles-ci. Durant l'exécution, le rôle principal du noyau est la capture de paquets.
- Une série de pré-processeurs, ceux-ci améliorent les possibilités de SNORT en matière d'analyse et de recomposition du trafic capturé. Ils reçoivent les paquets directement capturés, éventuellement les retravaillent puis les fournissent au moteur de recherche de signatures.

- Une série d'analyses est ensuite appliquée aux paquets. Ces analyses se composent principalement de comparaisons de différents champs des headers des protocoles (IP, ICMP, TCP et UDP) par rapport à des valeurs précises.
- Après la détection d'intrusion, une série de « output plugins » permet de traiter cette intrusion de plusieurs manières : envoie vers un fichier log, envoie d'un message d'alerte vers un serveur Syslog ou encore stocker cette intrusion dans une base de données SQL.

Réglages de Snort :

Tous les paramètres à régler de Snort se trouvent dans le fichier "snort.conf" situé dans le répertoire "/etc/snort". Dans ce fichier, une quantité importante d'information est fournie en commentaire "#" pour permettre à l'utilisateur de connaître l'utilité de chaque section, de chaque variable et être capable de faire une configuration correcte. Nous allons maintenant décrire les sections les plus importantes du fichier « snort.conf ».

Réglage des variables réseau

La configuration d'un Système de Détection d'Intrusion sur un réseau dépend naturellement de la topologie de ce réseau, de ses plages d'adresses, des serveurs dont il est équipé, etc. Par défaut, Snort va analyser tous les paquets de toutes les adresses IP du réseau. Ce choix de réglage se fait en utilisant la variable HOME_NET réglée comme suit : var HOME_NET any. Si l'utilisateur, ne veut analyser que les paquets de certaines adresses IP, il peut rentrer des intervalles d'adresses (ex : 192.168.2.0/24), des IP particulières, etc.

```
#Valeur pour un réseau en 192.168.0.x
var HOME_NET 192.168.0.0/24
#Valeur par défaut
#var HOME_NET any
#Valeur pour deux réseaux utilisant chacun une classe d'adresse
#var HOME_NET [10.0.0.0/24,192.168.0.0/24]
```

Choix des règles

Cette section de "snort.conf" est une suite d' « include » de fichiers de règles. En effet, les règles pour un certain type d'attaques sont regroupées dans un même fichier au nom explicite avec l'extension « .rules » dans le répertoire « rules », par exemple : attack-responses.rules, ftp.rules. Pour que Snort utilise un fichier de règles, il suffit que cette ligne soit sans commentaire. Pour omettre un fichier de règles, il faut commenter la ligne avec un « # » devant.

```
#On inclut les fichiers de règles qui nous intéresse
#Personal Rules
include $RULE_PATH/perso.rules

#EmergingThreats Rules
include $RULE_PATH/emerging-attack_response.rules
include $RULE_PATH/emerging-botcc-BLOCK.rules
include $RULE_PATH/emerging-botcc.rules
#include $RULE_PATH/emerging-compromised-BLOCK.rules
#include $RULE_PATH/emerging-compromised.rules
include $RULE_PATH/emerging-dos.rules
```

Définition de règle Snort :

Les règles de SNORT sont composées de deux parties distinctes : le header et les options.

- Le header permet de spécifier le type d'alerte à générer (alert, log…) et d'indiquer les champs de base nécessaires au filtrage : le protocole (TCP, UDP ou ICMP), l'orientation du trafic auquel la règle s'applique (Unidirectionnel ou Bidirectionnel) ainsi que les adresses IP et ports sources et destination.
- Les options, spécifiées entre parenthèses, permettent d'affiner l'analyse, en décomposant la signature en différentes valeurs à observer parmi certains champs du header ou parmi les données.

L'entête de règle contient l'information qui défini le "qui, où, et quoi" d'un paquet, ainsi que quoi faire dans l'événement où le paquet avec tous les attributs indiqués dans la règle devrait se présenter. Le premier élément dans une règle est l'action de règle. L'action de règle dit à Snort quoi faire quand il trouve un paquet qui correspond aux critères de la règle. Il y a cinq actions accessibles par défaut dans Snort, alert, log, pass, activate, et dynamic.

- alert - génère une alerte en utilisant la méthode d'alerte sélectionnée, et alors journalise le paquet
- log - journalise le paquet
- pass - ignore le paquet
- activate - alerte et alors active une autre règle dynamic
- dynamic - reste passive jusqu'à être activée par une règle activate, alors agit comme une règle log

Les options de règle forment le cœur du moteur de détection d'intrusion de Snort, combinant facilité d'utilisation, puissance et flexibilité. Toutes les options de règle de Snort sont séparées les unes des autres par un caractère point virgule ";". Les mots clés des options de règle sont séparés de leurs arguments avec un caractère deux points ":". Ci-dessous les différentes options de règle disponibles dans Snort :

- msg - affiche un message dans les alertes et journalise les paquets
- ttl - teste la valeur du champ TTL de l'entête IP
- ipoption - regarde les champs des options IP pour des codes spécifiques
- fragbits - teste les bits de fragmentation de l'entête IP
- dsize - teste la taille de la charge du paquet contre une valeur
- flags - teste les drapeaux TCP pour certaines valeurs
- seq - teste le champ TCP de numéro de séquence pour une valeur spécifique
- ack - teste le champ TCP d'acquittement pour une valeur spécifiée
- itype - teste le champ type ICMP contre une valeur spécifiée
- icode - teste le champ code ICMP contre une valeur spécifiée
- icmp_id - teste la champ ICMP ECHO ID contre une valeur spécifiée
- icmp_seq - teste le numéro de séquence ECHO ICMP contre une valeur spécifique
- content - recherche un motif dans la charge d'un paquet
- content-list - recherche un ensemble de motifs dans la charge d'un paquet
- offset - modifie l'option content, fixe le décalage du début de la tentative de correspondance de motif
- session - affiche l'information de la couche applicative pour la session donnée
- sid – identifiant de signature contenu dans la base de signature

Exemple de règle contre le SYN FLOOD sur le protocole HTTP(80):

La technique de détection des attaques du type SYN FLOOD peut s'avérer difficilement applicable. Il faut donc chercher une nouvelle orientation. Il s'agit de trouver ce qui caractérise un Syn Flood « frauduleux » (<> Syn Flood créé par une grande affluence de clients) ; Il est donc préférable plutôt que détecter les effets immédiats détecter les effets secondaires.

Dans certains cas de figure, le pirate ne peut empêcher complètement la machine usurpée de répondre. Il reste donc certains paquets RST trahissant la présence d'une attaque. La règle suivante détecte donc un seuil anormal de paquet RST :

```
alert tcp any any -> 192.168.0.1 80 (msg:"RST"; flow:stateless; flags:R; threshold: type both, track by_dst, count 100, seconds 20; sid:1000003;)
```

De même, si le pirate emploie une adresse qui ne soit pas présente sur le réseau, certains routeurs sont configurés pour répondre par un paquet ICMP de type « host unreachable » (il se peut que le routeur réponde aussi avec des RST, ce qui empêche la bonne réalisation de l'attaque, néanmoins il peut être intéressant de détecter cette tentative) :

```
alert icmp any any -> any any (msg:"Host unreachable"; itype: 3; icode: 1; threshold: type both, track by_dst, count 100, seconds 20; sid:1000002;)
```

Snort en Mode Sniffer :

```
#snort -dev PROTO
```

-d : Pour Imprimer à l'écran les données dans les paquets.
-e : Imprimer à l'écran l'entête Ethernet(couche liaison).
-v : Verbose mode, pour l'affichage des informations contenus dans les entêtes TCP/IP.

PROTO : Imprime uniquement les messages du protocole PROTO (i.e. IP, TCP, UDP, ICMP)

Snort en Mode Générateur de log :

#snort -dev -l ./log : Pour l'enregistrement des paquets dans un fichier.
-l : Répertoire pour enregistrer le fichier ("./log" dans l'exemple).

Snort en Mode NIDS (Intrusion Detection System) :

Démarrage de Snort :

```
#/etc/init.d/snort start
```

Arret de Snort :

```
#/etc/init.d/snort stop
```

Redémarrage de Snort :

```
#/etc/init.d/snort restart
```

Options SNORT :

Cf. Options

Mise à jour de Snort :

Une fois Snort installé, il est nécessaire d'installer les règles de signature Snort et de les maintenir à jour. Pour cela, nous pouvons utiliser le programme **oinkmaster.**

#oinkmaster -o /etc/snort/rules -b /etc/snort/backup 2>&1

La dernière instruction ci-dessus signifie que nous lancons le script perl oinkmaster, les règles sont placées dans le dossier /etc/snort/rules et si il y a des changements dans les nouvelles règles, les règles courante vont être sauvegardées dans le dossier /etc/snort/backup.

➔A Noter que le téléchargement des règles de signature Snort nécessite un abonnement payant, néanmoins il existe une alternative proposé par une communauté libre et dynamique: http://www.bleedingsnort.com
La différence par rapport aux règles officielles Snort fournit par la société Sourcefire est que, avec les règles bleedingsnort, les règles sont disponibles gratuitement immédiatement après leur création.

➔ Pour faciliter la lecture des rapports générés par Snort, il existe des solutions front-end web comme SnortSnarf, ACID ou encore BASE qui sont des interfaces graphiques écrites en PHP utilisée pour afficher les logs générés par l'IDS Snort et envoyés dans la base de données.

3. SYN Cookie / SYN Cache / SYN Proxy

Les « SynCookies » :

La technique de détection des connexions semi-ouvertes peut s'avérer difficilement applicable, il faut donc chercher d'autres alternatives pour contrer les attaques SYN FLOODING. Le problème est qu'à chaque fois qu'un paquet SYN est reçu, les informations concernant la connexion sont stockées dans une file. Une fois que l'attaquant à envoyé un nombre suffisant de paquets SYN la file se rempli complètement et de nouvelles connexions ne sont plus possibles, ce qui engendre l'indisponibilité du service. Il faudrait donc supprimer cette file d'attente. Le serveur n'a en fait pas besoin de stocker les informations de la connexion, contenues dans le paquet SYN (adresse IP et port du client et du serveur), vu qu'elles sont également présentes dans le paquet ACK envoyé par le client. La solution revient donc à utiliser les « SynCookies ». En effet, afin d'éviter de stocker les informations de la connexion dans une file d'attente, le serveur va se servir du réseau comme zone mémoire. Ainsi, il va envoyer au client les informations dont il a besoin pour établir la connexion. Le client les lui retournera, puis il vérifiera si le *numéro d'acquittement* du paquet envoyé par le client passe un test de sécurité spécifique pour créer et établir la connexion.

Pour activer les SYN-cookies sous Linux :

```
echo 1 > /proc/sys/net/ipv4/tcp_syncookies
```

Pour les désactiver :

```
echo 0 > /proc/sys/net/ipv4/tcp_syncookies
```

Les SynCookies présentent quelques désavantages :
Tout d'abord comme les options de TCP ne sont pas sauvegardées lors de la demande de connexion, certaines ne seront pas prises en compte. De plus, ils peuvent également poser

des problèmes de sécurité, ainsi un attaquant peut ainsi tenter un ACK flood en espérant tomber sur quelques cookies, complétant ainsi une connexion et consommant tout de même des ressources. Enfin, les SynCookies utilisent un peu plus le CPU qu'une connexion ne les utilisant pas, vu qu'ils doivent générer un numéro de séquence en utilisant certaines informations du paquet SYN (adresses IP, ports). Mais cela reste négligeable.

Le « SynCache » :
Contrairement à Linux qui utilise très classiquement les SynCookies comme une solution de secours suite à un flood détecté sur des queues classiques, les systèmes comme FreeBSD ont décidé de modifier leur système stockant les états afin de le rendre plus résistant avant de passer en dernier recours à un mécanisme de SynCookies. FreeBSD utilise désormais un SynCache pour conserver l'état d'une connexion jusqu'à sa complétion. Il a donc été décidé de limiter fortement les données conservées en utilisant une table de hashage, en lieu et place de la classique liste linéaire par sockets, avec les sockets indexées par bucket via des listes chaînées. Ainsi, en cas de détection de flood, la plus ancienne entrée (par bucket ou sur la totalité du cache) est supprimée, avant de basculer sur les SynCookies. Ce mécanisme permet à la fois de diminuer l'occupation mémoire et d'accélérer le parcours de la structure de données.

N.B : En jargon informatique, on appelle "bucket" ("boîte à confettis"), la "boîte" dans laquelle les bits disparaissent lorsqu'ils se perdent durant la transmission de données.

Le « SynProxy » :

Normalement, lorsqu'un client ouvre une connexion TCP vers un serveur, le par-feu relaie les paquets d'ouverture au fur et à mesure qu'ils arrivent. Ce dernier peut également agir en tant que mandataire (proxy). Dans ce cas, aucun paquet n'est transmis au serveur avant que le client n'ait terminé l'échange initial. Ainsi, grâce à cette méthode, le serveur est protégé contre les attaques SYN FLOOD.

➔Toutes les connexions à destination du serveur HTTP seront mandatées par le par-feu.

Malheureusement, l'option *synproxy state* n'est disponible (par défaut) uniquement sous Paketfilter (FreeBSD) et non sur Netfilter (Linux); Néanmoins, il est possible de bénéficier de l'option sous Netfilter en ajoutant des extensions à ce dernier.

B. NETFILTER

Définition :

Netfilter est un module du noyau Linux qui offre la possibilité de contrôler, modifier et filtrer les paquets IP, et de suivre les connexions. Il fournit ainsi les fonctions de pare-feu, de partage de connexions internet et d'autorisation du trafic réseau. Iptables est l'interface en "ligne de commande" permettant de configurer Netfilter.

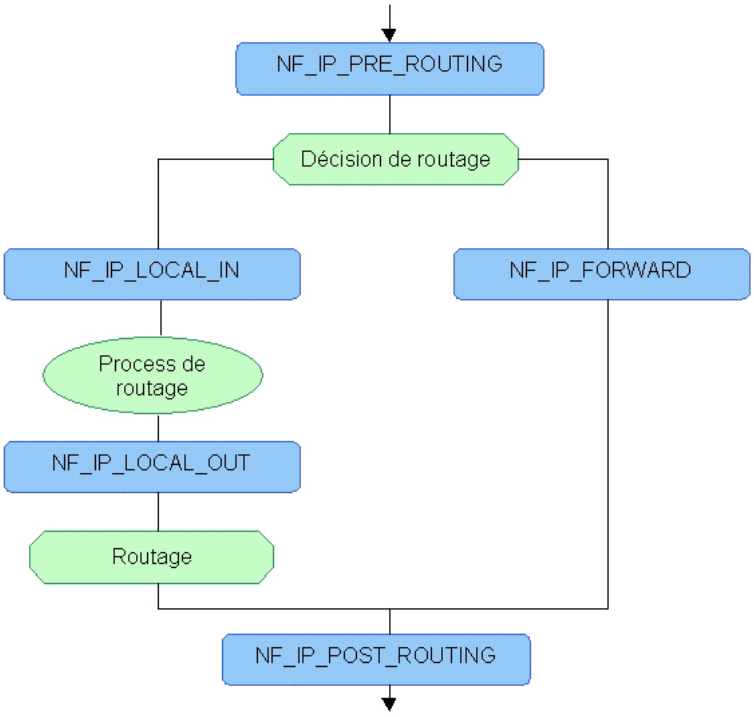
Fonctionnalité :

Netfilter se présente comme une série de 5 "hooks" (points d'accrochage), sur lesquels des modules de traitement des paquets vont se greffer. Ces points sont:

- NF_IP_PRE_ROUTING
- NF_IP_LOCAL_IN
- NF_IP_FORWARD
- NF_IP_POSTROUTING
- NF_IP_LOCAL_OUT

La branche gauche représente le trajet des paquets qui entrent et qui sortent vers et depuis un processus local (SMB, FTP, HTTP etc.)

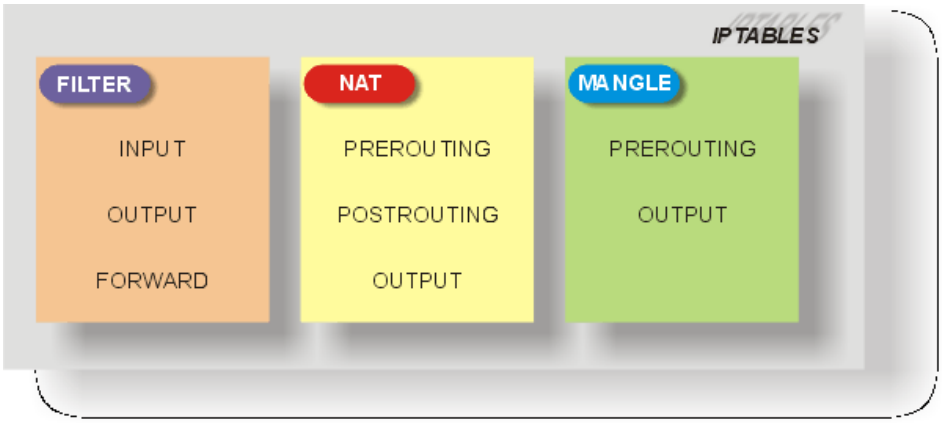
La branche de droite représente le trajet des paquets qui traversent notre passerelle dans sa fonction de routeur.



A travers ces cinq points d'insertion, Netfilter être capable :

- D'effectuer des filtrages de paquets, principalement pour assurer des fonctions de Firewall. Nous pourrons, 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.
- D'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.
- D'effectuer des opérations de marquage des paquets, pour leur appliquer un traitement spécial. Ces fonctionnalités sont particulièrement intéressantes sur une passerelle de réseau d'entreprise, un peu moins pour notre réseau domestique.

Les tables et leurs chaînes :
Il existe trois tables qui vont servir à contenir des règles de "filtrage":



Exemples de manipulations avec IPtables :
Pour commencer, nous allons tout fermer au niveau de la passerelle dans la table "filter".

Nous vidons les chaînes :
iptables –F

Nous supprimons d'éventuelles chaînes personnelles :
iptables -X

Nous les faisons pointer par défaut sur DROP
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP

C. CUSUM

CUSUM est un algorithme de détection de changement de modèle dans des processus stochastiques (étude des phénomènes aléatoires dépendant du temps).

Les machines victimes d'une attaque de type TCP/SYN flooding pourraient ainsi être détectées en appliquant un test de détection de ruptures correspondant au nombre de paquets SYN reçus par chaque machine destination présente dans le trafic. Cependant, un réseau typique d'opérateur fait circuler plusieurs dizaines de milliers de connexions par seconde ; les analyser toutes devient rapidement un problème difficile, au vu de la quantité massive de données à traiter.
C'est pourquoi, une méthode de réduction de dimension doit être préalablement employée.

A chaque pas de temps, les statistiques du CUSUM sont comparées à un seuil. Une alarme est déclenchée lorsque les valeurs dépassent ce seuil.

D. ALGORITHME ADAPTATIF DE SEUIL

L'algorithme adaptatif de seuil est un l'algorithme robuste et simple. Il se fonde sur l'examen de l'analyse du trafic pendant un intervalle de temps. Si le nombre de paquets SYN excède un seuil particulier il est probable qu'une attaque soit en cours.
Afin de prendre en compte des variations dû aux habitudes des utilisateurs, la valeur du seuil est placée de manière adaptative basée sur une évaluation du nombre moyen de paquets SYN.

IV. Conclusion

Le Déni de Service est en augmentation, il est donc nécessaire d'investir dans la sécurité interne en créant un poste ou service dédié au sein de l'entreprise. Leurs deux objectifs principaux devront être la prévention et le curatif. Le premier sera de mener une veille active sur les nouvelles attaques et vulnérabilités. Le second objectif, le curatif, devra procéder à un suivi régulier des différentes mises à jour et être très réactif sur la surveillance des événements relatés en supervision de l'architecture sécurité.

La réalité du risque est donc évidente, dans ce cas, il peut être intéressant de contracter une police d'assurances couvrant les pertes éventuelles engendrées par une attaque de ce type.

Tout comme les virus, pour pouvoir contrer les différentes attaques, il faut d'abord que celle-ci soit créée. On ne combat pas quelque chose qui n'a pas encore été créé. Il faut avoir été « infecté » une première fois pour pouvoir analyser et stopper ces attaques. C'est le rôle de plusieurs logiciels décrits dans les précédents chapitres.

V. Index

http://fr.wikipedia.org/wiki/Attaque_par_d%C3%A9ni_de_service#Programmes_disponibles_sur_Internet
http://www.authsecu.com/dos-attaque-deny-of-service/dos-attaque-deny-of-service.php#Les_principales_attaques
http://www.securiteinfo.com/attaques/hacking/ddos.shtml
http://www.securiteinfo.com/attaques/hacking/dos.shtml
http://cr.yp.to/syncookies.html
http://gd.tuwien.ac.at/www.hping.org/manpage.html
http://en.wikipedia.org/wiki/Stacheldraht