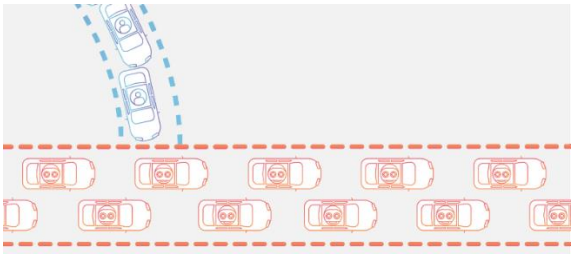
Simulations d'attaques

Nous allons simuler les attaques mentionnées dans le cahier des charges et nous essayerons de les capturer par Snort en activant des règles prédéfinies ou en créant les règles personnalisées. Pour éviter de recevoir beaucoup d'alertes, nous allons seulement activer les catégories des règles liées à l'attaque que nous allons effectuer.

DDoS (Distributed Denial of Service)

Une attaque DDoS ressemble à un embouteillage inattendu qui bloque une autoroute et empêche le trafic normal d'arriver à destination. DDoS (une attaque par déni de service distribué) est une tentative malveillante de perturber le trafic normal d'un serveur, service ou réseau en submergeant la cible.

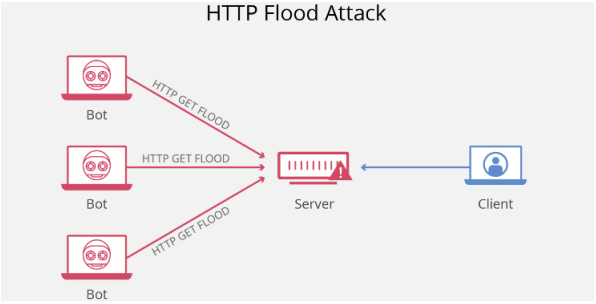
Les attaques DDoS sont exécutées avec des réseaux de machines connectées à Internet.



Ces réseaux sont constitués d'ordinateurs infectés par un logiciel malveillant qui permet au pirate de les contrôler à distance. Ces dispositifs individuels sont appelés « bots » (ou zombies), et un groupe de bots s'appelle un « botnet ».

DDoS http Flood

Une attaque http flood est un type d'attaque par déni de service distribué volumétrique conçu pour saturer un serveur ciblé de requêtes http. Une fois que la cible a été saturée de demandes et qu'elle est incapable de répondre au trafic normal, déni de service se produira pour les requêtes supplémentaires des utilisateurs existants.



Les attaques HTTP flood sont un type d'attaque DDoS de « couche 7 ».

Il existe deux variétés d'attaques http flood :

Attaque http GET : plusieurs ordinateurs envoient plusieurs requêtes d'images, de fichiers ou d'autres éléments vers un serveur ciblé. Lorsque la cible est inondée de requêtes, un déni de service se produit pour les requêtes supplémentaires provenant de sources de trafic légitimes.

Attaque http POST : lorsqu'un formulaire est soumis sur un site Web (requête POST), le serveur doit traiter la requête entrante et envoyer les données dans une base de données. Le traitement des données du formulaire et l'exécution des commandes sont intensif et cette attaque peut créer le déni du service.

Comment peut-on atténuer une HTTP flood ?

Une solution possible est de lancer un défi à la machine afin de vérifier s'il s'agit ou non d'un bot, un peu comme le test *captcha* que l'on trouve couramment lors de la création d'un compte en ligne.

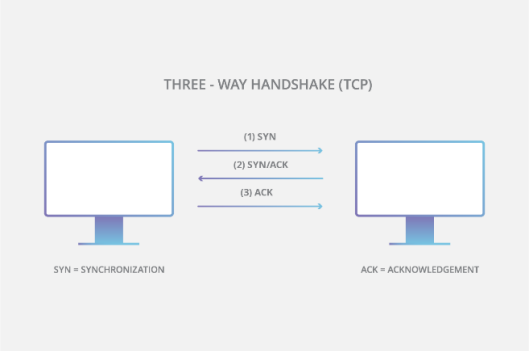
DDoS SYN flood

Une attaque SYN flood vise à rendre un serveur indisponible pour le trafic légitime en consommant toutes les ressources serveur disponibles. En envoyant des paquets de demande de connexion initiale (SYN : synchronize) le pirate submerge tous les ports disponibles sur un serveur ciblé, ce qui oblige le serveur à répondre lentement au trafic légitime ou l'empêche totalement de répondre.

L'attaque SYN flood fonctionne en exploitant le processus d'établissement de liaison d'une connexion TCP.

La connexion TCP est composée de trois processus distincts :

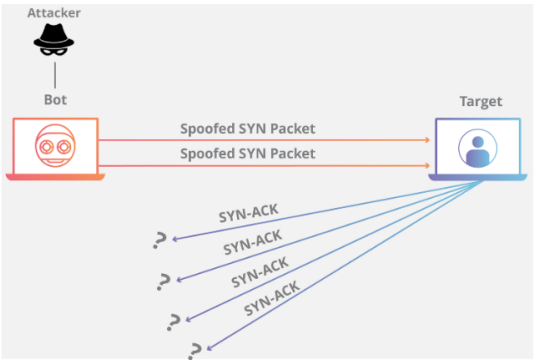
1. Le client envoie un paquet SYN au serveur afin d'établir la connexion.
2. Le serveur répond à ce paquet avec un paquet SYN/ACK, afin d'accuser réception de la communication.
3. Le client renvoie un paquet ACK pour accuser réception du paquet provenant du serveur.



Après avoir terminé ses étapes avec succès la connexion TCP est établi et ouverte pour envoyer et recevoir des données.

Pour créer un déni de service par l'attaque SYN flood :

1. Le pirate envoie un volume élevé de paquets SYN au serveur ciblé.
2. Le serveur répond ensuite à chacune des demandes de connexion et laisse un port ouvert prêt à recevoir la réponse.
3. Pendant que le serveur attend le dernier paquet ACK qui n'arrive jamais, le pirate continue d'envoyer plus de paquets SYN. Chaque paquet SYN oblige le serveur de créer une nouvelle connexion de port ouverte et une fois que tous les ports disponibles ont été utilisés, le serveur ne peut plus fonctionner normalement.



Trois manières pour la création d'une attaque SYN flood existe :

1. **Attaque directe :** une attaque SYN flood où l'adresse IP n'est pas usurpée est connue sous le nom d'attaque directe. Dans cette attaque, le pirate ne masque pas du tout son adresse IP. Cette méthode est rarement (voire jamais) utilisée, car les mesures d'atténuation sont relativement simples : il suffit de bloquer l'adresse IP de chaque système malveillant.

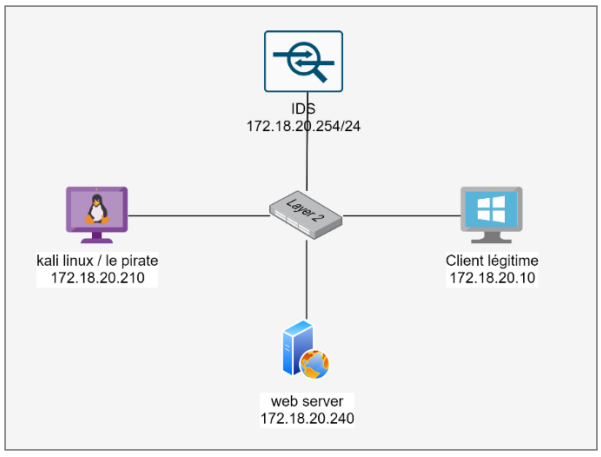
2. **Attaque par usurpation d'identité :** Un utilisateur malveillant peut également usurper l'adresse IP de chaque paquet SYN qu'il envoie afin d'entraver les efforts d'atténuation et de rendre son identité plus difficile à découvrir. Bien que les paquets puissent être usurpés, ils peuvent potentiellement être retracés jusqu'à leur source.
3. **Attaque distribuée (DDoS) :** si une attaque est créée à l'aide d'un botnet, la probabilité de pouvoir trouver la source de l'attaque est faible. Un pirate peut aussi faire en sorte que chaque périphérique distribué usurpe également les adresses IP à partir desquelles il envoie des paquets.

Comment atténuer une attaque SYN flood ?

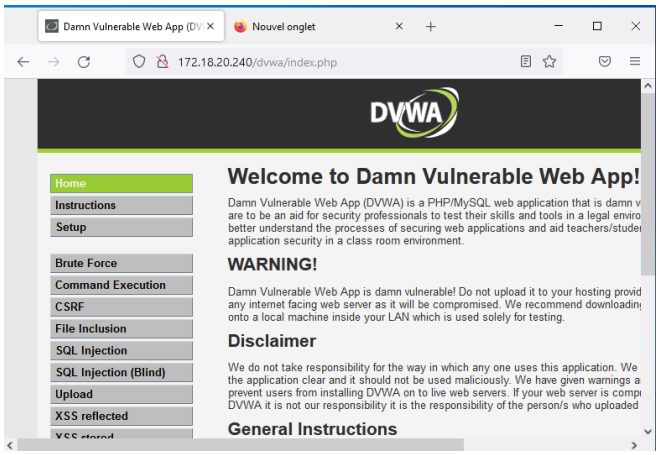
1. Augmenter la file d'attente du backlog : Chaque système d'exploitation sur un appareil ciblé dispose d'un certain nombre de connexions semi-ouvertes qu'il autorise. Il est possible de répondre aux volumes élevés de paquets SYN en augmentant le nombre maximal de connexions semi-ouvertes possibles que le système d'exploitation autorisera (le backlog).
2. Ecraser la connexion semi-ouverte la plus ancienne une fois le backlog rempli.

Simulation d'une attaque DDoS SYN flood

Pour simuler l'attaque SYN flood, nous allons utiliser une machine kali en tant que pirate. Une machine Metasploitable 2 en tant que serveur web et une machine Windows 10 en tant que client légitime. Nous allons également installer Snort sur une Pfsense. Voici le schéma :



Le serveur web est disponible depuis la machine cliente sans problème :



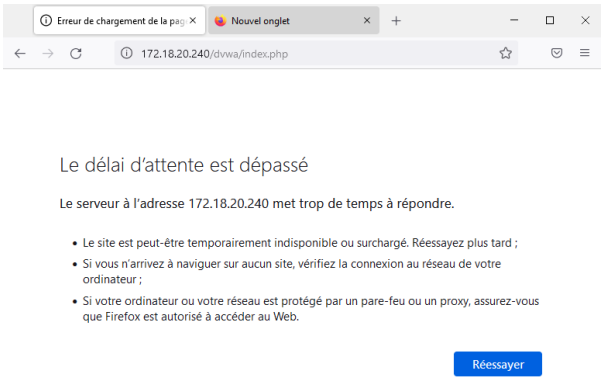
Sur la Kali, nous allons utiliser un outil qui s'appelle **hping3** qui nous permet d'envoyer les paquets ICMP/UDP/TCP et d'afficher les réponses de la cible comme le programme ping le fait avec les réponses ICMP. Nous allons utiliser la commande suivante pour créer une attaque SYN flood vers le serveur web :

sudo hping3 -c 15000 -d 120 -S -w 64 -p 80 --flood --rand-source 172.18.20.240

```
┌──(kali㉿kali)-[~]
└─$ sudo hping3 -c 15000 -d 120 -S -w 64 -p 80 --flood --rand-source 172.18.20.240
HPING 172.18.20.240 (eth0 172.18.20.240): S set, 40 headers + 120 data bytes
hping in flood mode, no replies will be shown
```

Dans cette ligne de commande : Kali envoie 15000 paquets (-c 15000) avec la taille de 120 octets (-d 120) chacun. Nous spécifions que le flag SYN (-S) droit être activé avec la taille de la fenêtre de 64 (-w 64). Nous ciblons l'attaque vers le serveur web sur le port 80 (-p 80) et utilisons le flag --flood pour envoyer les paquets le plus vite possible. Le flag --rand-source (random source) génère des adresses IP usurpée pour déguiser la source réelle et éviter la détection et en même temp arrêter les paquets SYN-ACK réponses de la victime d'arriver au pirate. Cela empêchera la machine du pirate de répondre avec le paquet ACK puisqu'il n'a jamais reçu SYN-ACK.

Après avoir lancé la commande, nous allons retenter d'accéder à la page web et nous voyons qu'il n'est plus accessible :



Visualiser les paquets SYN par Wireshark

Nous pouvons capturer les paquets transmis dans le réseau pour vérifier les paquets SYN créés par le pirate :

No.	Time	Source	Destination	Protocol	Length	Info
28	6.255741	139.71.237.198	172.18.20.240	TCP	174	1537 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]
29	6.255741	152.5.205.43	172.18.20.240	TCP	174	1538 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]
30	6.255741	48.61.56.43	172.18.20.240	TCP	174	1539 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]
31	6.255741	21.88.233.5	172.18.20.240	TCP	174	1540 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]
32	6.255741	185.114.134.192	172.18.20.240	TCP	174	1541 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]
33	6.255741	186.68.30.42	172.18.20.240	TCP	174	1542 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]
34	6.255741	19.205.239.36	172.18.20.240	TCP	174	1543 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]
35	6.255741	5.140.193.237	172.18.20.240	TCP	174	1544 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]
36	6.255741	110.88.54.211	172.18.20.240	TCP	174	1545 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]
37	6.255741	182.90.156.78	172.18.20.240	TCP	174	1546 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]
38	6.255741	189.209.55.196	172.18.20.240	TCP	174	1547 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]
39	6.255741	213.188.19.156	172.18.20.240	TCP	174	1548 → 80 [SYN] Seq=0 Win=64 Len=120 [TCP segment of a reassembled PDU]

On voit bien tous les paquets SYN envoyés depuis la machine du pirate vers le serveur web. Pour l'adresse source il n'y a pas d'adresse IP du pirate mais des adresses IP usurpées aléatoires. On peut aussi voir d'autres paramétrages dans le champ info.

Capturer l'attaque par Snort

On relance l'attaque mais cette fois on active le service IDS sur le port LAN de notre pare-feu :

	Interface	Snort Status	Pattern Match	Blocking Mode	Description	Actions
☐	LAN (em1)	✓ ↻ ■	AC-BNFA	DISABLED	LAN	✎ ⧉ 🗑

Dans l'onglet Alert on voit les alertes créées à cause de cette attaque :

39 Entries in Active Log

Date	Action	Pri	Proto	Class	Source IP	SPort	Destination IP	DPort	GID:SID	Description
2022-09-29 13:08:20	⚠	2	TCP	Misc Attack	169.249.61.157	9203	172.18.20.240	80	1:2400017	ET DROP Spamhaus DROP Listed Traffic Inbound group 18
2022-09-29 13:08:20	⚠	2	TCP	Misc Attack	196.10.66.252	8444	172.18.20.240	80	1:2400023	ET DROP Spamhaus DROP Listed Traffic Inbound group 24
2022-09-29 13:08:20	⚠	2	TCP	Misc Attack	137.55.191.132	8295	172.18.20.240	80	1:2400009	ET DROP Spamhaus DROP Listed Traffic Inbound group 10
2022-09-29 13:08:20	⚠	2	TCP	Misc Attack	196.55.193.245	8258	172.18.20.240	80	1:2400023	ET DROP Spamhaus DROP Listed Traffic Inbound group 24
2022-09-29 13:08:20	⚠	2	TCP	Misc Attack	86.106.110.86	7727	172.18.20.240	80	1:2400003	ET DROP Spamhaus DROP Listed Traffic Inbound group 4
2022-09-29 13:08:20	⚠	2	TCP	Misc Attack	204.238.137.249	6885	172.18.20.240	80	1:2400031	ET DROP Spamhaus DROP Listed Traffic Inbound group 32
2022-09-29 13:08:20	⚠	2	TCP	Misc Attack	197.154.214.132	6680	172.18.20.240	80	1:2400023	ET DROP Spamhaus DROP Listed Traffic Inbound group 24

On peut voir les adresses IP sources usurpées, le *GID* et *SID* des règles qui ont déclenché ces alertes, la description par rapport à ces règles et les autres informations. Ces alertes sont créées par les règles qu'on a ajoutées depuis Snort lui-même (l'onglet *Global Settings*).

Maintenant on va créer notre propre règle dans la partie *custom.rules* :

```
alert tcp any any -> $HOME_NET 80 (flags: S; msg:"Possible TCP DoS"; flow: stateless; threshold: type both, track by_src, count 70, seconds 10; sid:10001;rev:1;)
```

Dans cette règle, tous les paquets SYN qui vont vers le réseau HOME_NET, si le nombre total de ces paquets qui viennent depuis une source précise (track by_src), dans une période de 10 secondes est supérieur à 70, il déclenche une alerte avec le message « Possible TCP DoS ».

Nous ajoutons cette règle dans le *custom.rules* et refaisons le teste. Nous vérifions l'onglet alerte et nous voyons qu'il n'y a aucune alerte. La raison pour laquelle Snort n'a pas capturé les attaques est que les paquets SYN viennent des adresses IP aléatoires usurpées.

0 Entries in Active Log

Date	Action	Pri	Proto	Class	Source IP	SPort	Destination IP	DPort	GID:SID	Description

Nous changeons le *track* et on le met *par destination* (au lieu de *par source*) avec l'option *track by_dst*.

```
alert tcp any any -> $HOME_NET 80 (flags: S; msg:"Possible TCP DoS"; flow: stateless; threshold: type both, track by_dst, count 70, seconds 10; sid:10001;rev:1;)
```

nous relançons l'attaque pour **30 secondes** et nous vérifions l'onglet alerte :

3 Entries in Active Log

Date	Action	Pri	Proto	Class	Source IP	SPort	Destination IP	DPort	GID:SID	Description
2022-09-29 13:47:03	⚠	0	TCP		222.23.130.219	31356	172.18.20.240	80	1:10001	Possible TCP DoS
2022-09-29 13:46:53	⚠	0	TCP		39.60.234.39	64771	172.18.20.240	80	1:10001	Possible TCP DoS
2022-09-29 13:46:43	⚠	0	TCP		110.39.61.65	1676	172.18.20.240	80	1:10001	Possible TCP DoS

Nous avons reçu trois alertes pour cette attaque de 30 secondes. Autrement dit, avec l'option threshold, nous avons reçu une alerte par chaque période de 10 secondes.