Man in the Middle with Scapy

We are interested on implementing a *man in the middle* attack by using Arp packets (layer 2) using Scapy API.
Every computer has an **arp table** to store the IP-MAC addresses map corresponding to the hosts that have communicated with. If no previous communication have been established before, the computer will send a broadcast message asking for the MAC corresponding to a particular IP address. However, an attacker can send a false response. Then, this table (ARP cache) can be spoofed, allowing an attacker to receive the traffic destined to someone else.
For instance, we can spoof the arp table in two devices, the gateway and other machine and insert us in the middle of the communication. In essence, **ARP spoofing** or ARP cache poisoning consist of an attacker sending a crafted ARP packet with spoofed address.

ARP packet crafting is needed to make **ARP spoofing**. In order to achieve this task we are going to use Scapy. The script will be developed on Python 3 with scapy library support.

```
sr1(ARP(op=1, dst=”192.168.1.1”), timeout=1)
```

Here we are sending a single ARP packet destined to the give address. This function send a **request** (op=1) packet to discover the MAC address corresponding to dst IP address.

This attack requires access to local network. Network hosts automatically cache any ARP replies received. Then, we are interested on creating a false entry on the victim's cache.

1. The first step will consist on scanning the network to know what hosts are alive.
2. Then, we chose the victim and get the corresponding MAC address.
3. The default gateway and victim arp table are spoofed.
4. We sniff the communication and get useful insights.
5. Finally we restore network (arp tables) as before the attack.

ARP_spoofing.py implements the described attack process.

`arp_poisong` function receives the mac and ip addresses of the victims. Then, we send ARP packets with `op=2` to write spoofed entries in the victims ARP tables and put our computer in the middle. An interesting attack is to choose the default gateway as victim_0 and see how the communication occurs.

In order to see the attack procedure and how packets are destined to the attacker's system we can use a sniffer like Wireshark. However, we can create in Scapy a customized sniffer.

```
sniff(prn=arp_display, filter=“arp”, store=0, count=10)
```

We are using our own function(arp_display) to manipulate the output of the packets, filter option specify what packets will be captured, store refers to putting the filter packets in a file(.pcap) and count will determine the max number of captured packets.