Mac Flooding Challenge

A MAC flooding is a technique employed to compromise the security of network switches. The attack works by forcing legitimate MAC table contents out of the switch and forcing a unicast flooding behavior potentially sending sensitive information to portions of the network where it is not normally intended to go.

The goal of this challenge is to intercept and decrypt messages exchanged between two hosts (from our GNS3 lab) performing a MAC Flooding attack. Let's call the first host `H1` and the second one `H2`.

In this scenario we have to act as the **Attacker**. More in detail, we have to build a Python3 script able to spoof packet using **MAC Flooding** techniques. In order to do that, we have to send many ARP messages onto a local area network using many different (random) mac address with the aim to fill the switch mac address table. In this way, the traffic meant for our victim (`H2`) will be sent in broadcast and thus the attacker will be able to intercept its messages.

Prerequisites

- Download and run the `ARPSpoofingChallengeSender.py` on host_1 `H1`
- Execute the script on `H1`

```
ARPSpoofingChallengeSender.py <HOST_1_SRC_IP> <HOST_2_DST_IP>
```

Hints:

- In order to send ARP messages onto local area network using Python3, install `scapy` python module
- Arp messages can be sent using the following function:
  - `sendp(Ether(src=<<MAC_ADDRESS>>,dst=<<MAC_ADDRESS>>)/ARP(op=2, psrc="<<IP_ADDRESS(Or subnet)>>", hwdst="<<BROADCAST_MAC_ADDRESS>>"), loop=1)`
- A random mac address can be generated using `RandMac()` function in scapy
- Data can be filtered using `tshark`
- Remember that data collected from `tshark` are in byte, thus they must be converted in `base64`
- Some data are in `plaintext`, some other are crypted. Decrypt data using discovered password and encryption algorithm
  `openssl enc -a <ALGORITHM> -k <KEY> -d -base64`