# Task-5 Capstone & Incident Response — Report

## Title & Cover

**Project Title:** **Student:** Jatin **Mentor / Organisation:** ApexPlanet Software Pvt. Ltd. **Dates:** **Environment:** Kali (attacker) IP: 192.168.56.4; Metasploitable (target) IP: 192.168.56.5; Other VMs:

## Executive Summary

“This capstone demonstrates a full offensive → defensive cycle on an isolated lab VM. We performed reconnaissance, controlled exploitation, simulated an SSH brute-force incident, detected and contained the attack, applied remediation and verified the result. The report includes findings, evidence and mitigation recommendations.”

## Scope & Rules of Engagement

* In-scope systems: Metasploitable2 (192.168.56.5), Kali (192.168.56.4), local lab network (host-only).
* Out-of-scope: Internet-facing assets, third-party systems.
* Approvals: Lab-only, isolated environment.

## Tools Used

List tools and versions (examples): - Kali Linux (version) - Nmap (7.x) - Metasploit Framework (msfconsole) - Hydra, John the Ripper - Wireshark / tcpdump - OpenVAS/Nessus (if used) - Burp Suite

## Methodology

Outline the phases followed: 1. Recon 2. Scanning 3. Exploitation 4. Post-Exploitation 5. Incident Response Simulation 6. Remediation / Hardening

## Implementation & Evidence

### DVWA:- 1.1 SQL Injection

Method: Used DVWA SQL Injection module at Low security. Payloads: ' OR '1'='1' --, UNION SELECT ... to extract users.

Finding: Database returned multiple user records and password hashes, proving SQL injection.

### 

### SQL Injection — Prepared Statements (PDO)

**PHP PDO prepared statement example:**  
$pdo = new PDO("mysql:host=127.0.0.1;dbname=dvwa","dvwa","p@ssw0rd");  
$stmt = $pdo->prepare("SELECT first\_name, last\_name FROM users WHERE user\_id = ?");  
$stmt->execute([$\_GET['id']]);  
while ($row = $stmt->fetch()) {  
 echo htmlspecialchars($row['first\_name'], ENT\_QUOTES, 'UTF-8') . ' ' . htmlspecialchars($row['last\_name'], ENT\_QUOTES, 'UTF-8');  
}

### 1.2 Cross-Site Scripting (XSS)

Types tested: Stored (High), Reflected (Medium), DOM (Low).

Payloads used (examples):

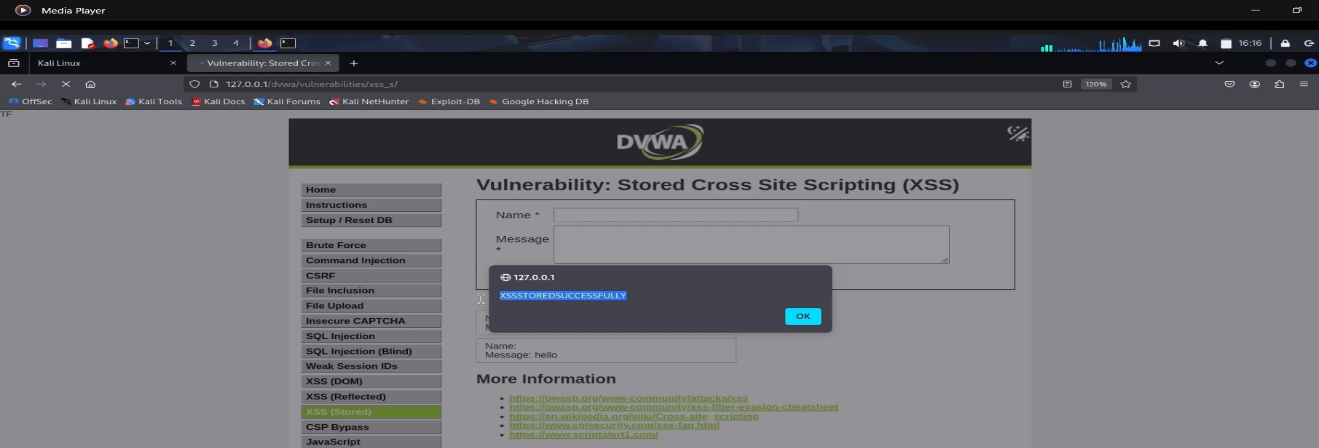
<script>alert("Stored XSS")</script>

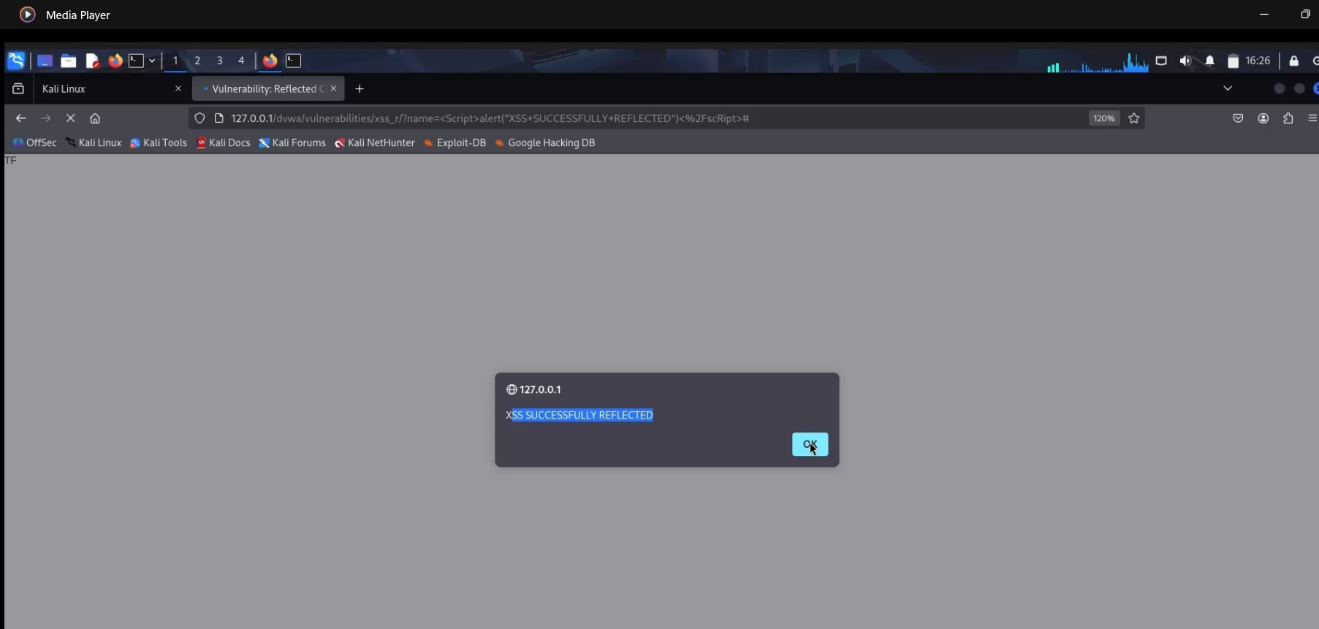
Findings: Stored payload persisted and executed for visitors; reflected payload executed on submission; DOM payload executed client-side.

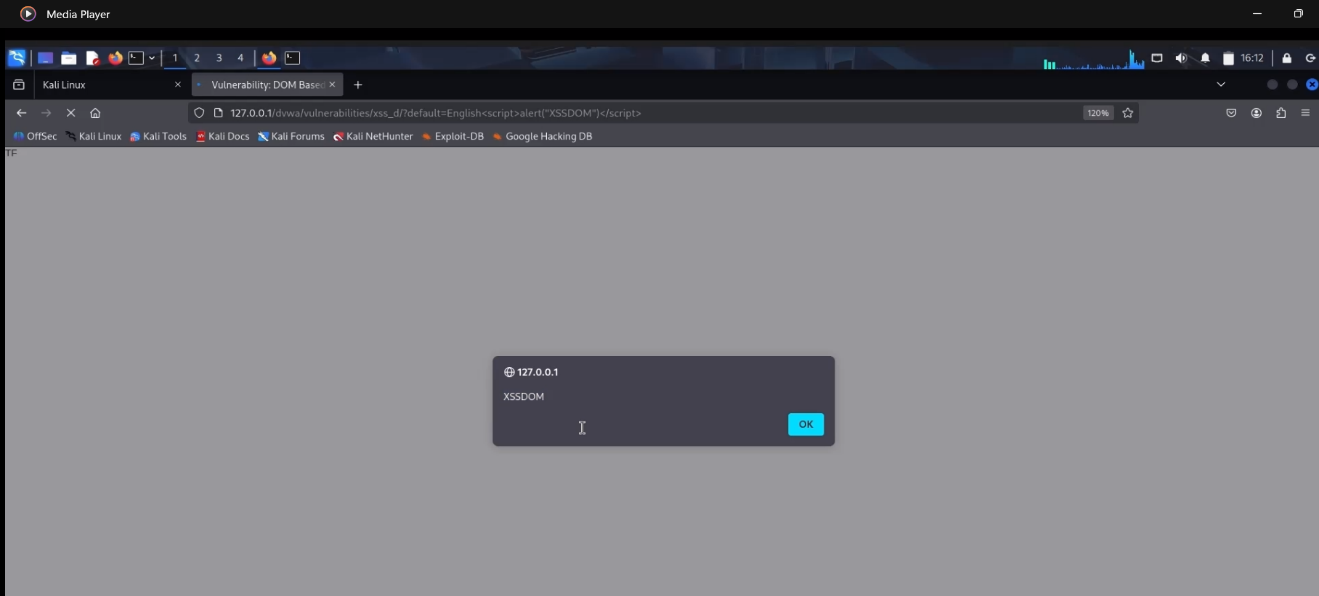
### XSS — Output Encoding

**PHP output encoding example:**  
echo htmlspecialchars($user\_input, ENT\_QUOTES, 'UTF-8');

Also implement Content-Security-Policy (CSP) headers.



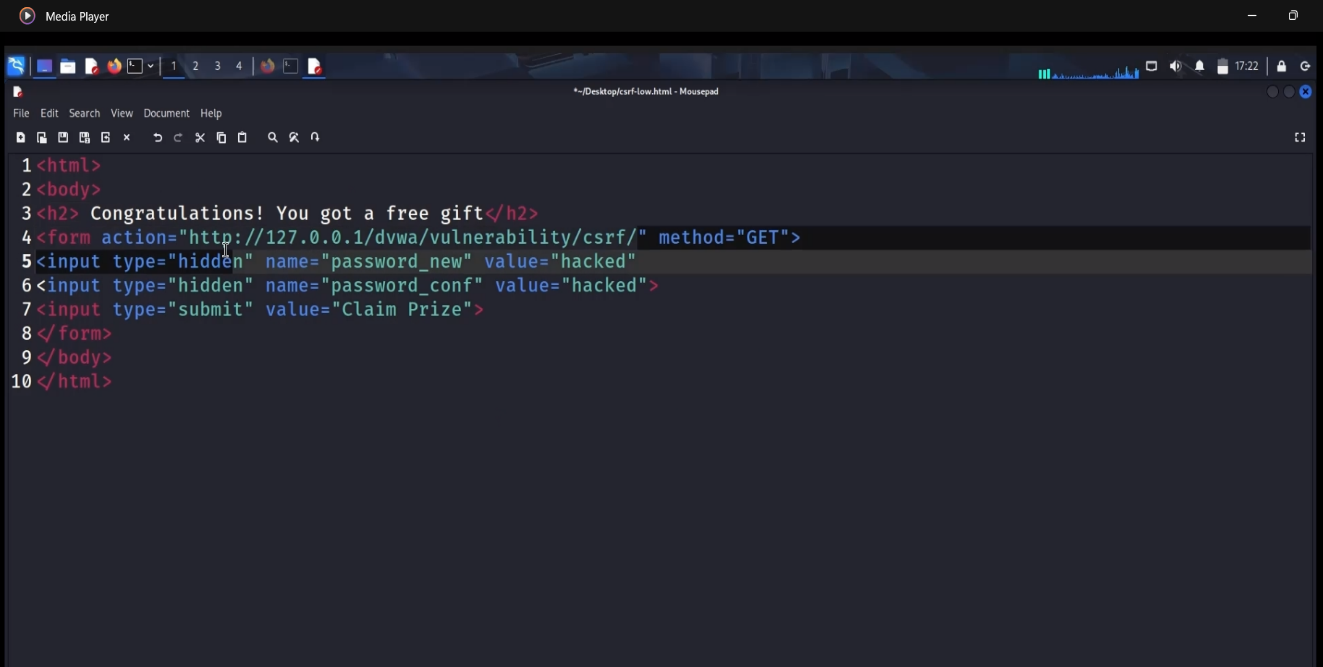


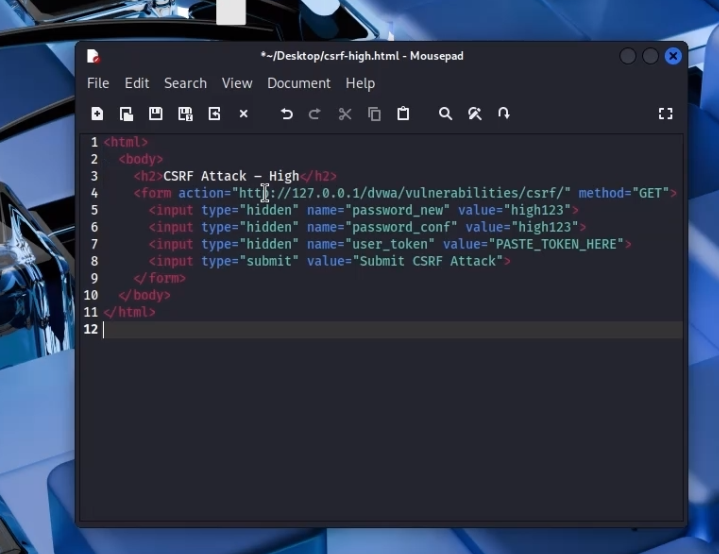


### 1.3 Cross-Site Request Forgery (CSRF)

Tested at Low, Medium, High security levels.

Low: No CSRF token; external form changed user password silently.

High: Token rotates and validated; attack blocked.



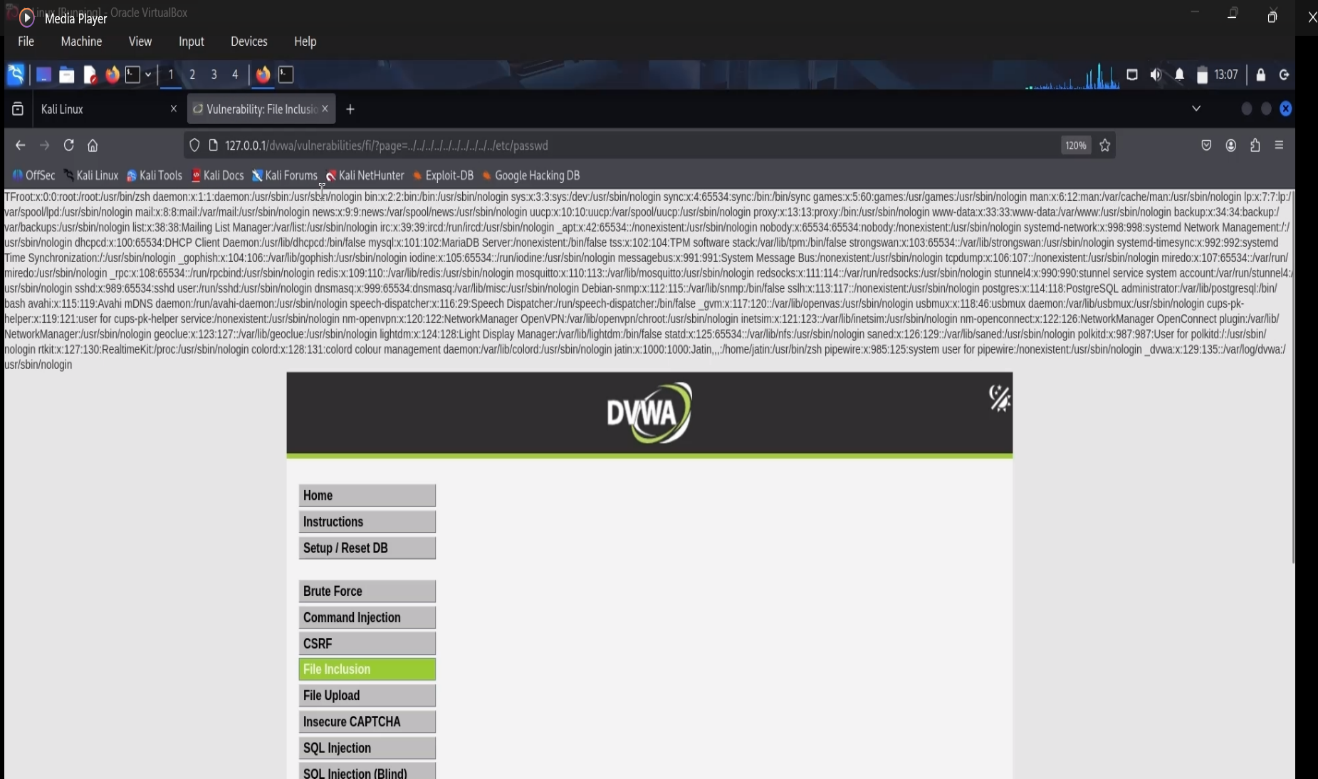
### CSRF — Token-based protection

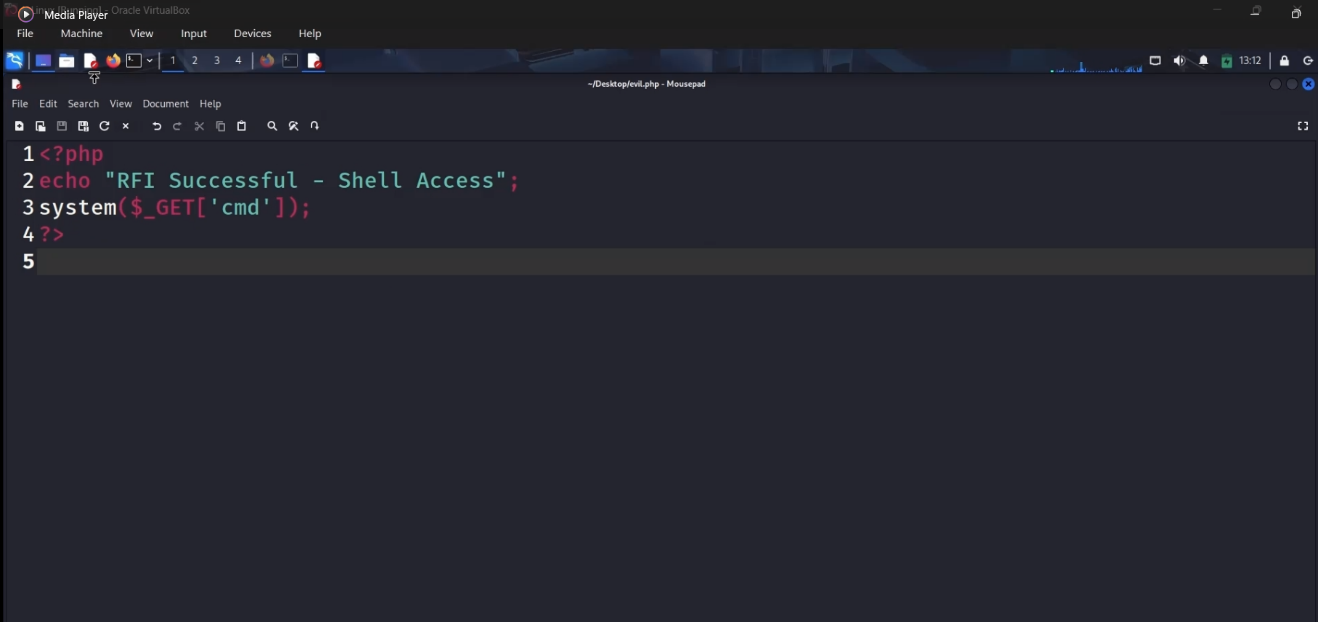
**CSRF token generation and validation (PHP):**  
// generate  
if (empty($\_SESSION['token'])) {  
 $\_SESSION['token'] = bin2hex(random\_bytes(32));  
}  
// form  
<input type="hidden" name="user\_token" value="<?php echo $\_SESSION['token']; ?>">  
// validation  
if (!hash\_equals($\_SESSION['token'], $\_REQUEST['user\_token'] ?? '')) {  
 die('Invalid CSRF token');  
}

1.4 File Inclusion (LFI / RFI)

LFI test payload: ?page=../../../../etc/passwd

Finding: /etc/passwd contents were shown (LFI confirmed).

RFI: Enabled allow\_url\_include temporarily in lab and included attacker evil.php which executed (whoami output). 



### File Inclusion — Whitelist & disable remote includes

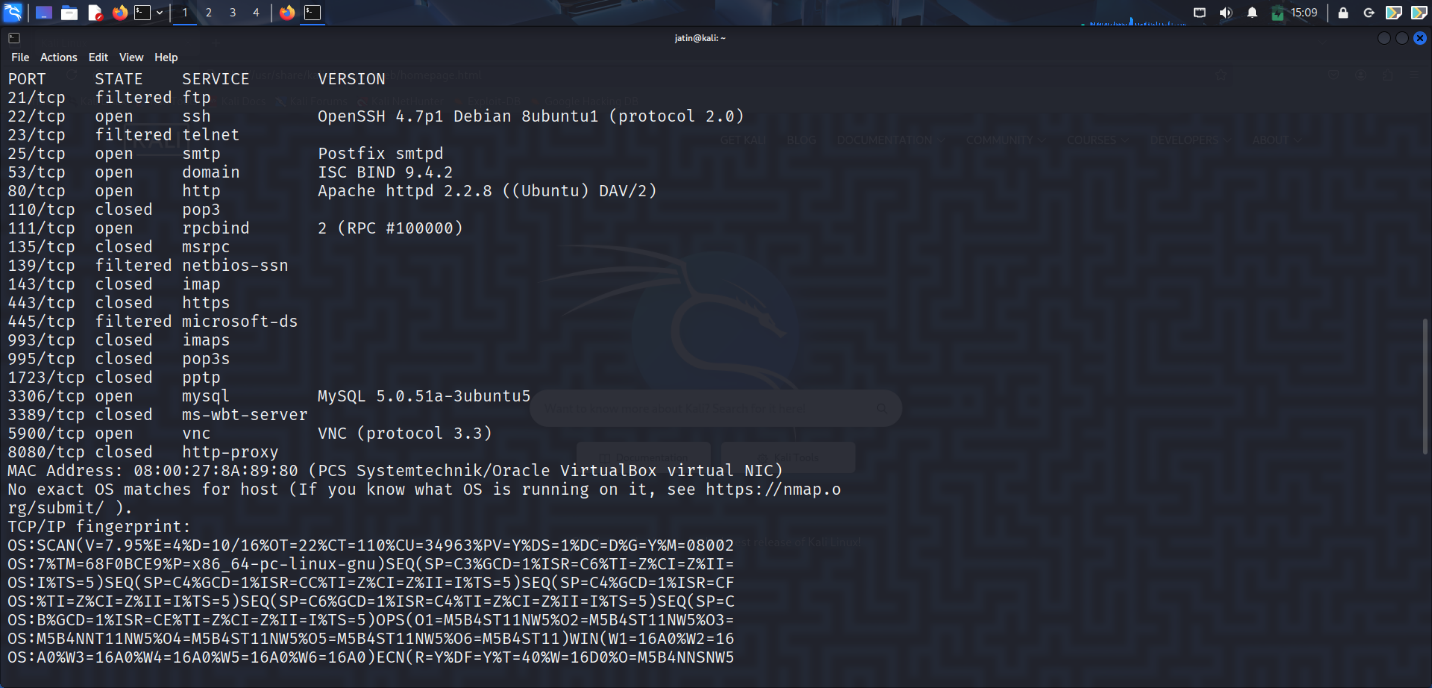
Disable remote includes: set allow\_url\_include = Off in php.ini (recommended).

**Whitelist include example:**  
$allowed = ['home.php','about.php'];  
$page = $\_GET['page'] ?? 'home.php';  
if (in\_array($page, $allowed, true)) {  
 include \_\_DIR\_\_ . '/pages/' . $page;  
} else {  
 echo "Access denied";  
}

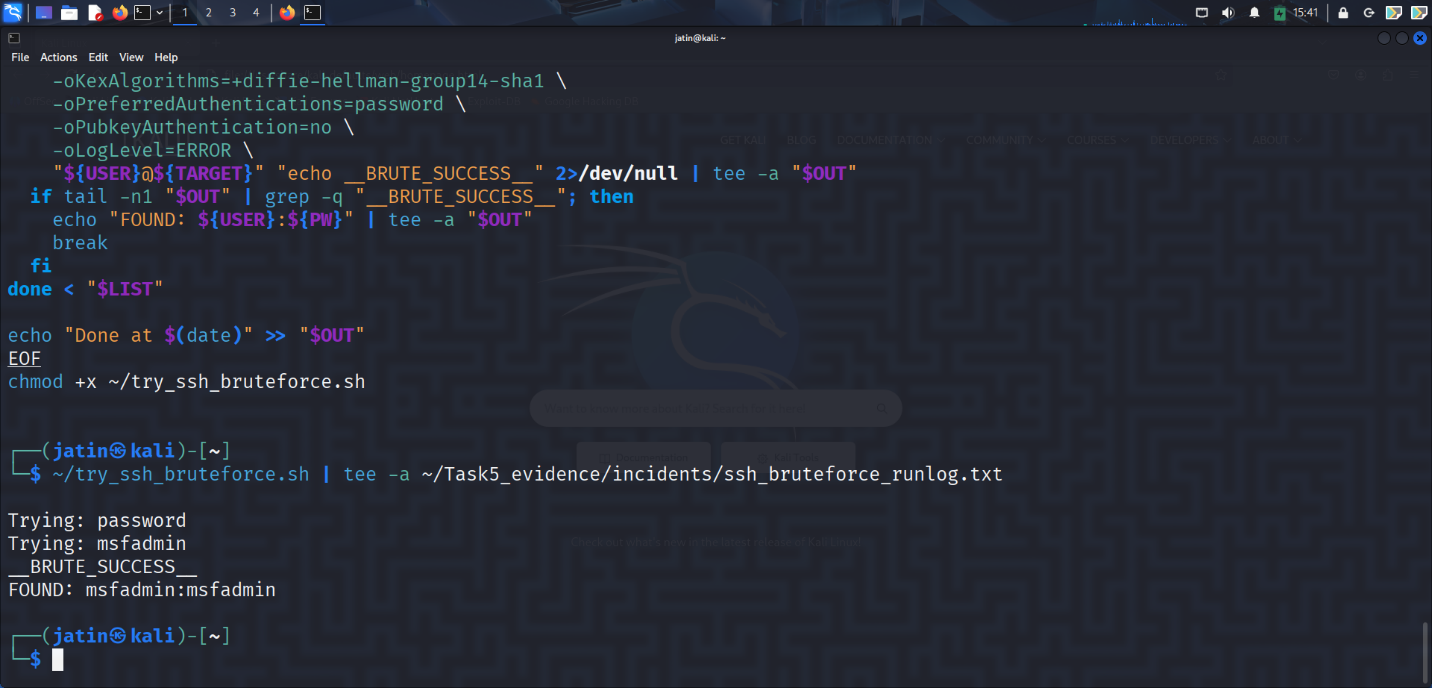
## 2. Reconnaisance

**A. Recon & Scanning** - Commands run (copy/paste):

whois example.com  
nmap -sS -sV -O --top-ports 100 192.168.56.4 -oN task5\_nmap\_top100.txt



**C. Incident Simulation & Detection** - Simulated incident (e.g., SSH brute force, SQLi, webshell).-

**D. Containment & Eradication** - Steps you took to contain (block IP, stop service) and eradicate (remove malware, patch). Commands + outputs. - Evidence placeholders: 

**E. Recovery & Lessons Learned** - Steps to recover services and verification scans. - Short lessons and follow-up recommendations.

## Findings & Risk Matrix

Table format for each issue: Vulnerability | Evidence | Severity | Impact | Recommendation

## Remediation Commands (copy-paste)

sudo apt update && sudo apt upgrade -y  
sudo ufw deny 21/tcp  
sudo systemctl stop vsftpd && sudo systemctl disable vsftpd  
# etc.

## Appendices

* Full Nmap output
* Full Metasploit console transcript
* Raw logs and pcaps

# Phase-2 Project Report

**1) Objectives:**

* Demonstrate a full offensive → defensive cycle on an isolated lab target.
* Identify and exploit a vulnerability (controlled), capture evidence, and simulate an incident.
* Detect the incident via log analysis and packet capture.
* Contain, eradicate and recover the target (block attacker, rotate creds, harden services).
* Produce a professional incident report with IOCs, remediation steps and verification scans.

**2) Scope:**

**In-scope**

* Lab VMs only: Metasploitable (target) 192.168.56.4, Kali (attacker) 192.168.56.3 .
* Tools and tests listed in the Tools section.
* Actions: Recon, scanning, controlled exploitation, brute-force simulation, log analysis, blocking, hardening, reporting.

**Out-of-scope**

* Any external/Internet systems or third-party services.
* Any destructive activity that would persist beyond the lab (no wiping, no propagation).
* Social engineering against real people.

**3) Tools:**

* Kali Linux (attacker): Nmap, Hydra, Metasploit (msfconsole), John, Netcat, tcpdump, Wireshark, ssh/sshpass, git.
* Target VM: Metasploitable2 (victim) with vulnerable services (vsftpd, ssh, etc.).
* Analysis: Wireshark, tcpdump, journalctl//var/log, strace (for dynamic malware demo).
* Reporting: LibreOffice / MS Word, Git & GitHub, simple image editor for screenshots.
* Optional: OpenVAS / Nessus or GVM for vulnerability scanning; Fail2Ban / UFW for remediation.

**4) Timeline:**

* **Day 1 (2–3 hours):** Snapshot lab, Recon & Scanning (Nmap), save outputs & screenshots.
* **Day 2 (2–3 hours):** Exploitation (Metasploit / Hydra), capture session evidence and logs.
* **Day 3 (1–2 hours):** Incident detection (log analysis, pcap), identify attacker IP & IOCs.
* **Day 4 (1–2 hours):** Containment & eradication (UFW/iptables block, change creds, remove artifacts).
* **Day 5 (1–2 hours):** Hardening steps (SSH config, disable services, sysctl), verification scans.
* **Day 6 (2–3 hours):** Final report assembly, export PDF, push to GitHub, upload video & submit links.

