WEB APPLICATION SECURITY TESTING REPORT

Presented By

NANDHITHA V N

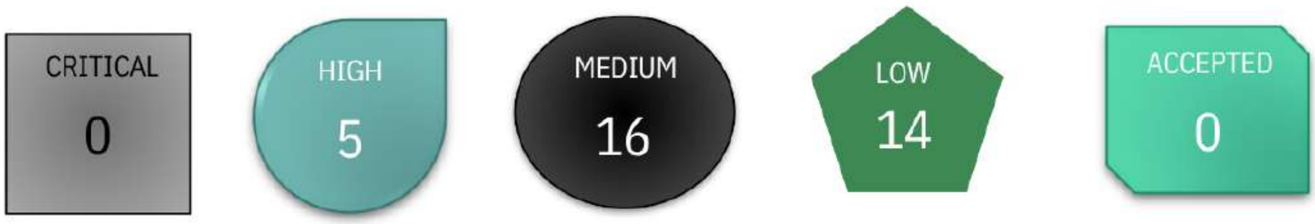
DATE : 05.10.2025

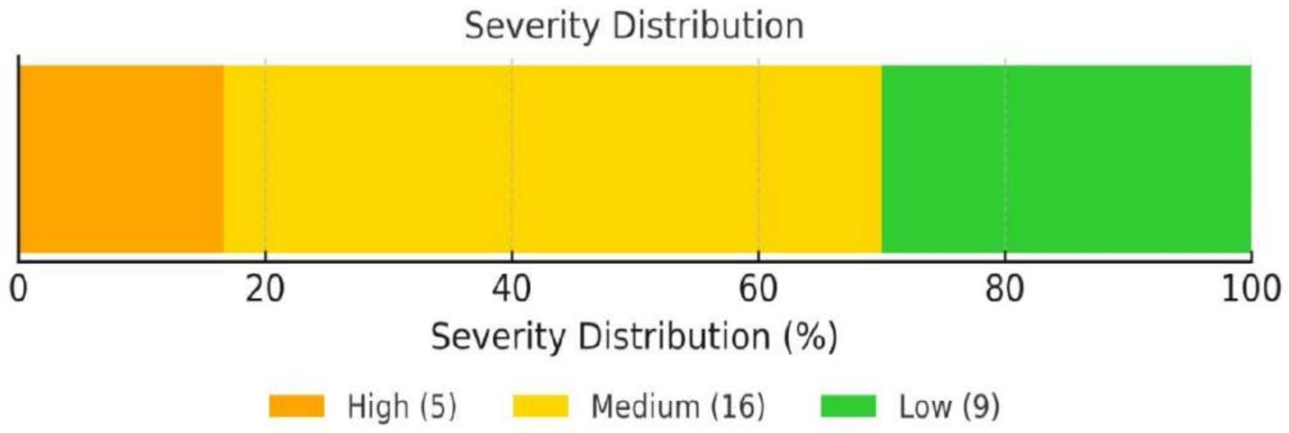
1.EXECUTIVESUMMARY:

Thisreport presents theresults of a vulnerability assessment and penetration testing (VAPT) exercise conducted in a controlled lab environment. The objective was to identify, exploit, and document web application vulnerabilities using both automated and manual techniques.

1.1 Total Vulnerabilities

Below are the total number of vulnerabilities were identified, spanning across categories such as SQL Injection, Cross-Site Scripting (XSS), CSRF, Command Injection, File Inclusion, and misconfigurations. The tools used include Hosted Scan for automated scanning, Burp Suite for manual testing, DVWA for simulation, and SQL Map for injection exploitation





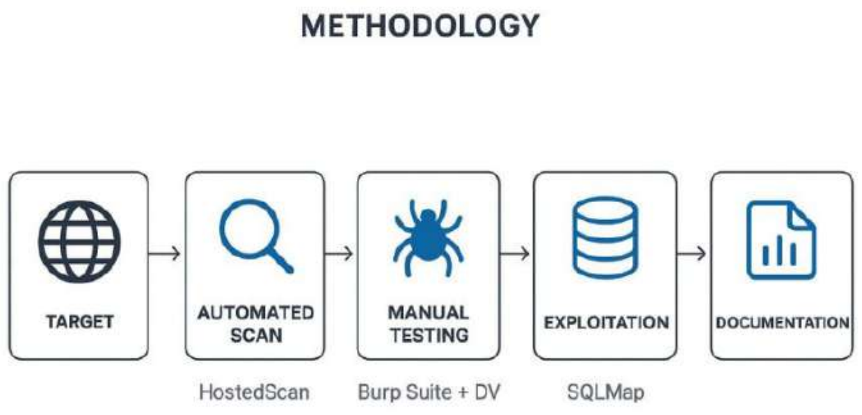
2.Methodology:

2.1 Tool Icons / Logos

1.HostedScan – magnifying glass or scanning icon
2.Burp Suite – spider or proxy icon
3.DVWA – target or web icon
4.SQLMap – database icon

2.2 Testing Workflow Diagram

Target → Automated Scan (HostedScan) → Manual Testing (Burp Suite + DVWA) → Exploitation (SQLMap) → Documentation



2.3 **Security Level Illustration**

DVWA's three levels

- LOW
- MEDIUM
- HIGH

2.4 **Manual vs Automated Testing Comparison**

APPROACH	TOOLS USED	PURPOSE
Automated scan	HostedScan	Surface-level findings
Manual Testing	Burp Suite, DVWA	Exploitation & Validation

3.HostedScan Results

Target: vulnweb.com

Scan Type: Passive Web Application Scan

Findings:

1.Outdated JS Libraries: AngularJS, Bootstrap, jQuery, Sessvars

2.Missing Security Headers: CSP, X-Frame-Options, X-Content-Type-Options

3.Cookie Misconfigurations: No HttpOnly, No SameSite

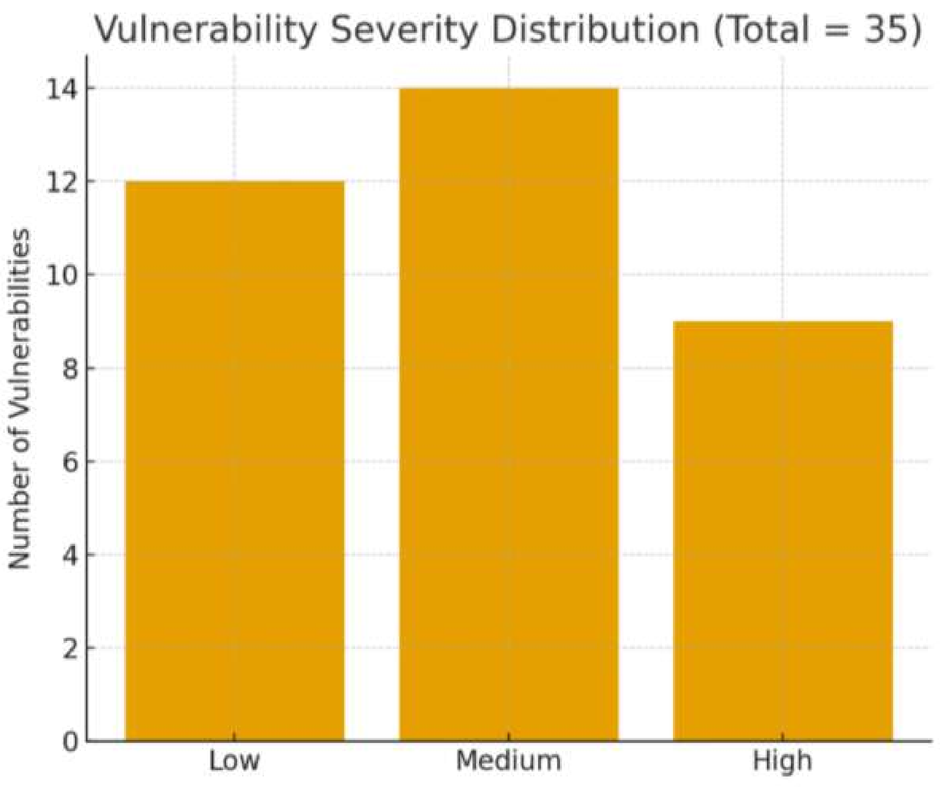
4.CSRF Token Absence

5.Cross-Domain Misconfiguration

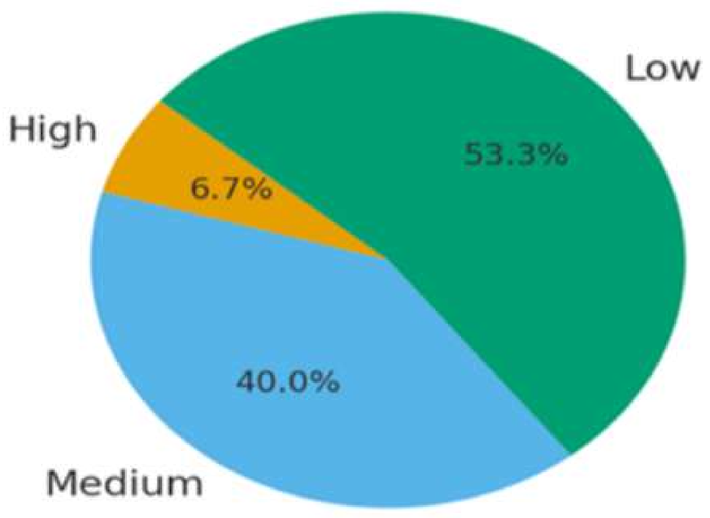
6.Server Info Disclosure

SEVERITY BREAKDOWN

Severity	Count
HIGH	5
MEDIUM	16
LOW	14
TOTAL	35



HostedScan Serverity Breakdown (pie chart)



4.Manual Exploitation: Injection Flaws

4.1 SQL Injection (SQLi)

- **Classic SQLi:** Bypassed authentication using ' OR '1'='1 payload.
- **Blind SQLi:** Extracted database information using time-based delays.
- **Union-based SQLi:** Retrieved sensitive user data from the database.
- **Tool:** SQLMap automated the data exfiltration process.

4.2 Command Injection

- **Exploitation:** Injected OS commands into a "Ping" utility input field.
- **Proof of Concept:**

  ```
  127.0.0.1 && cat /etc/passwd
  ```

- **Impact:** Successfully executed commands to list directories and read system files, demonstrating full server compromise potential.

5.Manual Exploitation: XSS & CSRF

5.1 Cross-Site Scripting (XSS)

- **Stored XSS:** Injected malicious scripts into a guestbook that executed for every visitor.
- **Reflected XSS:** Delivered malicious payloads via URL parameters.
- **DOM-based XSS:** Manipulated the client-side DOM environment.

5.2 Cross-Site Request Forgery (CSRF)

- **Vulnerability:** Password change functionality lacked CSRF tokens.

- **Exploitation:** Generated a malicious HTML page using Burp Suite that auto-submitted a password change request when visited by an authenticated user.
- **Impact:** Attacker can force users to change their password without consent

6.Other Critical Vulnerabilities

6.1 File Inclusion

- **Local File Inclusion (LFI):** Used path traversal to access sensitive server files.
- **Payload:** ?page=../../../../etc/passwd
- **Impact:** Unauthorized access to system files containing passwords.

6.2 Weak Session Management

- **Issue:** Session IDs were predictable (e.g., MD5 hash of incrementing integers).
- **Impact:** Allows session hijacking and account takeover.

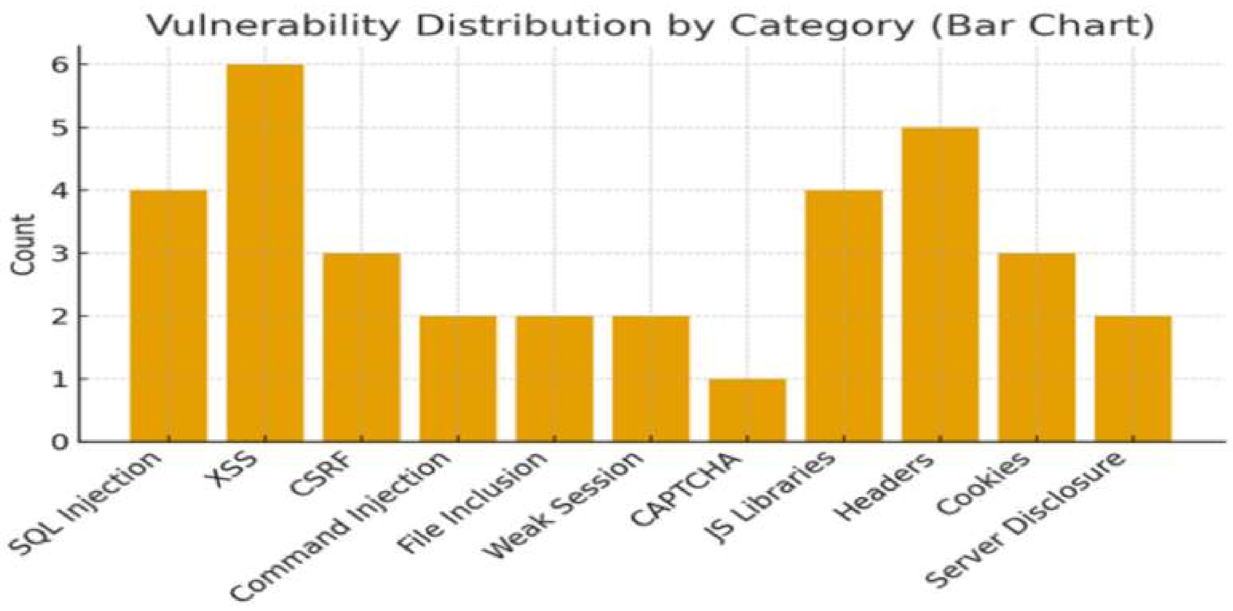
3. CAPTCHA Bypass

- **Vulnerability:** CAPTCHA validation was performed on the client-side.
- **Exploitation:** Bypassed by intercepting and modifying the response with Burp Repeater.

7.Vulnerability Summary

A consolidated view of all 35 identified vulnerabilities.

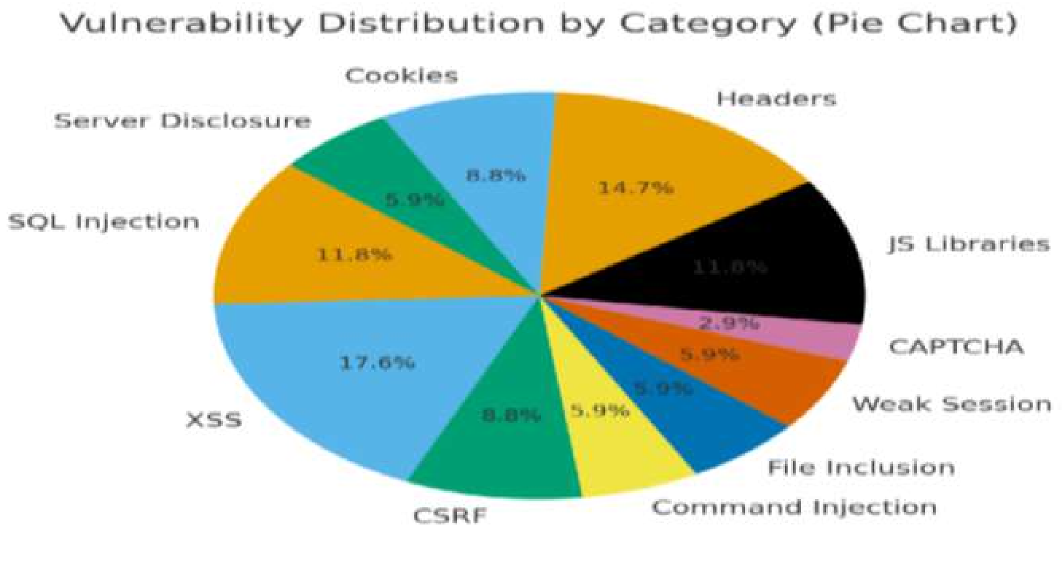
Category	Count	Severity	Tool Used
SQL Injection	4	High	SQLMap, Burp
XSS	6	Medium	DVWA, Burp
CSRF	3	Medium	Burp
Command Injection	2	High	DVWA
File Inclusion	2	High	DVWA
Weak Session	2	Medium	DVWA
CAPTCHA	1	Low	DVWA
JS Libraries	4	Medium	HostedScan
Headers	5	Medium	HostedScan
Cookies	3	Low	HostedScan
Server Disclosure	2	Low	HostedScan



8.Mapping to OWASP Top 10 2021

Our findings directly correlate with the industry-standard OWASP

OWASP Category	Related Findings from this Lab
A01:2021 - Broken Access Control	File Inclusion, Weak Session IDs
A02:2021 - Cryptographic Failures	Server Information Disclosure
A03:2021 - Injection	SQL Injection, Command Injection
A05:2021 - Security Misconfiguration	Missing Headers, CAPTCHA Misconfiguration
A06:2021 - Vulnerable Components	Outdated JS Libraries
A07:2021 - Identification & Auth Failures	CAPTCHA Bypass, Weak Sessions
A08:2021 - Software/Data Integrity	CSRF (Lack of Integrity Checks)



9.Actionable Recommendations: Technical

9.1 **Input Validation& Sanitization**

- Use **parameterized queries** or ORMs to prevent SQLi.
- Implement strict allow-lists for user input in command execution functions.

9.2 **Web Application Hardening**

- Implement and enforce **anti-CSRF tokens** on all state-changing requests
- Configure web server to set critical security headers: **Content-Security-Policy, X-Frame-Options, X-Content-Type-Options**.
- Secure cookies by adding HttpOnly and SameSite flags.

9.3 **Authentication & Session Management**

- Use secure, random number generators for **Session ID** creation.
- Enforce server-side **CAPTCHA validation** and rate-limiting on login attempts.

10. Recommendations: Process & Maintenance

10.1 Patch Management:

- Establish a formal process to regularly **update and patch** all third-party libraries and components.

10.2 Secure Development Lifecycle (SDL):

- Integrate security testing (SAST/DAST) early in the development cycle.
- Conduct **regular VAPT exercises**, especially after major releases.

10.3 Defense in Depth:

- Employ a **WAF (Web Application Firewall)** as a mitigating control to detect and block common attacks.

Conclusion

- ThisMiniVAPT Lab successfully identified**35 vulnerabilities** across various severitylevels.
- It provided hands-on experience withindustry-standard tools and techniques for both automated andmanual security testing.
- The findings underscore the prevalenceofOWASP Top 10 risks and the necessity of **proactive securitymeasures**.
- This exercise has significantly strengthenedpractical skills in ethical hacking and vulnerability analysis,forming a solid foundation for a career in cybersecurity.

"Security is not a product, but a process." –Bruce Schneier

Acknowledgments

I would like to express my sincere gratitude to:

- **Future intern** for providing the platform and resources necessary to explore the depths of cybersecurity.

Thank You,

NANDHITHA V N