Capstone Project: Full VAPT Cycle

Activities

Simulate a complete Vulnerability Assessment and Penetration Testing (VAPT) cycle on an intentionally vulnerable application and document findings professionally.

Tools

- Kali Linux
- Metasploit
- OpenVAS
- Google Docs

Perform vulnerability scanning

Simulate exploitation in a lab environment

Document findings aligned with PTES methodology

Simulation: SQL Injection on DVWA

Target: DVWA (Damn Vulnerable Web Application – Lab)
Reference: TryHackMe learning path
Objective: Validate SQL Injection vulnerability using an automated testing tool.

Identified input parameters vulnerable to injection

Used an automated SQL injection testing tool to validate vulnerability

Confirmed database interaction due to improper input sanitization

Recorded results without modifying production data

SQL Injection testing was performed against DVWA in a controlled lab environment. Improper input validation allowed database interaction, confirming the presence of a SQL Injection vulnerability.

Timestamp	Target IP	Vulnerability	PTES Phase
2025-08-18 12:00:00	192.168.1.200	XSS	Exploitation

PTES Phase	Activity
Pre-engagement	Scope definition
Intelligence Gathering	Target identification
Vulnerability Analysis	OpenVAS scanning
Exploitation	SQL Injection / XSS validation
Post-Exploitation	Evidence & impact analysis
Reporting	Risk & remediation documentation

A full VAPT cycle was simulated against an intentionally vulnerable web application. Automated scanning identified vulnerabilities, which were validated through controlled exploitation. Findings were mapped to PTES phases and documented to demonstrate real-world penetration testing workflow.

Remediation

Implement **server-side input validation and sanitization** for all user inputs.

Use **parameterized queries / prepared statements** to prevent SQL injection.

Apply **output encoding** to mitigate XSS vulnerabilities.

Enforce **least-privilege access** for database and application accounts.

Update and patch all vulnerable components and frameworks.

Disable unnecessary services and harden configurations.

Verification

Perform a **full rescan using OpenVAS** after remediation.

Re-test previously vulnerable endpoints to confirm fixes.

Validate that no exploitable input points remain.

Reporting: 200-Word PTES Report

This assessment followed the Penetration Testing Execution Standard (PTES) to evaluate the security posture of an intentionally vulnerable web application in a controlled lab environment. The engagement began with scope definition and intelligence gathering to identify target assets. Vulnerability analysis was performed using OpenVAS, which identified multiple issues including cross-site scripting and improper input validation.

During the exploitation phase, controlled testing validated the presence of SQL injection vulnerabilities within the application. These vulnerabilities allowed unauthorized interaction with backend databases due to insufficient input sanitization. Exploitation was conducted strictly for validation purposes, without impacting system availability or integrity.

Post-exploitation activities focused on evidence collection and impact analysis. Findings confirmed that insecure coding practices and misconfigurations significantly increased the application's attack surface. Each vulnerability was mapped to the appropriate PTES phase and documented with supporting evidence.

Remediation recommendations were provided, emphasizing secure coding practices, input validation, and regular vulnerability scanning. A rescan was advised to confirm mitigation effectiveness. This assessment demonstrates the importance of combining automated scanning with manual validation to accurately assess risk and improve overall security posture.

Briefing: 100-Word Non-Technical Summary

A security assessment was conducted on a test application to identify weaknesses that could be exploited by attackers. Automated scans and controlled testing revealed issues such as insecure input handling, which could allow unauthorized access to sensitive data. These findings indicate a risk to data confidentiality and application integrity if left unresolved. Recommended actions include improving input validation, strengthening system configurations, and regularly scanning for vulnerabilities. Addressing these issues will reduce the likelihood of

attacks and improve overall system security. This assessment highlights the importance of proactive security testing to prevent real-world breaches.